UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN MATHÉMATIQUES ET INFORMATIQUE
APPLIQUÉES

PAR
MOHAMED RAFIK BITAM

LES SYSTÈMES APPLICATIFS TYPÉS : VERS UNE MISE EN ŒUVRE RÉELLE

DECEMBRE 2022

Résumé

L'être humain ne cesse de se surpasser et ce depuis la découverte du feu, de l'invention du téléphone et jusqu'au développement des micro-puces. De ce fait, de nos jours, nous vivons entourés de diverses technologies à la fois complexes et précises. Dans ce projet, notre intérêt est porté sur l'intelligence artificielle dans le domaine du traitement des langues naturelles, et plus précisément sur les systèmes applicatifs et la logique combinatoire. Nous y avons étudié de façon théorique la capacité de la logique combinatoire à avoir une puissance générative supérieure à la plupart des modèles formels de grammaires existants.

La logique combinatoire est une logique non-standard largement utilisée en traitement des langues naturelles que ce soit au niveau syntaxique, sémantique ou même cognitif pour représenter le sens de certains verbes par exemple. Malgré les résultats intéressants obtenus à la suite de l'utilisation de la logique combinatoire, cette dernière est souvent considérée comme étant hors contexte. Une question reste en suspens : quelle est la capacité générative de la logique combinatoire ?

Pour répondre à cette question nous avons analysé plusieurs exemples d'expressions combinatoires à distance. Nous en avons déduit l'algorithme qui nous a permis de mesurerer la capacité générative du modèle combinatoire. Nous avons implémenté les différents combinateurs ainsi que notre algorithme de capacité générative dans une application Web. Pour ce faire, nous-nous sommes orientés vers quatre langages et plateformes : Python, Flask, Javascript, Réact. Nous avons utilisé Python ainsi que Flask pour implémenter nos différents combinateurs et notre algorithme, tandis que JavaScript et Réact ont été utilisés pour développer notre interface graphique.

Les résultats obtenus au terme de notre travail de recherche sont concluants. Toutes les chaines de type $A^nB^nC^n$ ont été générées avec succès; permettant ainsi de prouver que la logique combinatoire est sensible au contexte (type 1) et qu'elle est bel et bien capable d'avoir une capacité générative supérieure à la plupart des modèles grammaticaux formels existant. Bien entendu ce travail ouvre la porte à un autre travail théorique essentiel qui portera sur la preuve mathématique de ce qui a été réalisé.

Abstract

The human being never ceases to surpass himself and this since his existence; from the discovery of fire to the invention of the telephone and until the development of microchips. As a result, today we live surrounded by various technologies that are both complex and precise. During this project our interest is focused on artificial intelligence in the field of natural language processing and more precisely on combinatory logic and application systems. We have studied theoretically the ability of combinatory logic to have a generative capacity superior to most existing grammars.

Combinatory logic is a non-standard logic widely used in natural language processing, whether at the syntactic, semantic or even cognitive level to represent the meaning of certain verbs for example. Despite the interesting results obtained from the use of combinatory logic, the latter is often considered as being out of context, a question remains: what is the generative capacity of combinatory logic?

To answer this question, we analyzed examples of combinatory expressions at a distance, which allowed us to derive our algorithm to calculate the generative capacity of the combinatory model. We implemented the different combinators as well as our generative capacity algorithm in a web application. To do so, we have oriented ourselves towards four languages and platforms: Python, Flask, Javascript, React. We used Python and Flask to implement our different combinators and our algorithm, while JavaScript and React were used to develop our graphical interface.

The results obtained during our research work were conclusive, all the chains of type $A^nB^nC^n$ were successfully generated; thus, allowing to prove that combinatory logic is context sensitive grammar (type 1) and that it is indeed capable of having a generative capacity superior to most of the existing grammars. Of course, this work opens the door to another essential theoretical work which will deal with the mathematical proof of what has been implemented.

Key words: combinatorial logic, application systems, grammars, generative capacity, chains.

“ C'est dans l'effort que l'on trouve la satisfaction et non dans la réussite. Un plein effort est une pleine victoire. “

Mahatma Gandhi

Remerciements

La réalisation de ce travail n'aurait été possible sans le concours de plusieurs personnes à qui je voudrais adresser mes remerciements.

Je souhaiterais tout d'abord adresser toute ma reconnaissance au directeur de mon mémoire, Monsieur Ismail Biskri, pour ses précieux conseils et son orientation ficelée tout au long de notre travail de recherche et qui ont contribué à enrichir ma réflexion.

Je tiens à remercier les professeurs de l'université de Trois Rivières pour l'enseignement de qualité qui a su m'apporter une profonde satisfaction intellectuelle

J'aimerais exprimer ma gratitude à mes parents, qui ont cru en moi, qui m'ont soutenu tout au long de mon parcours ;

A mes deux sœurs pour m'avoir épaulé moralement dans la construction de ce mémoire.

Mots clés :

- Combinateur,
- Grammaires catégorielles,
- Hiérarchie de Chomsky.
- La logique combinatoire,
- Opérande,
- Opérateur,
- Système applicatif.

Il est important de noter qu'afin d’élaborer notre travail; nous nous sommes appuyés sur des règles préexistantes et qui représentent les fondamentaux de notre projet. Nous avons aussi utilisé des termes techniques communs qui sont propres à notre domaine de recherche.

TABLE DES MATIERES

Chapitre 1 : Introduction 11

Chapitre 2 : Les systèmes applicatifs et la logique combinatoire 15

1. Introduction : 16

2. Systèmes applicatifs typés : 16

3. La logique combinatoire, son cadre historique et ses objectives : 18

4. Combinateurs logiques : 19

- **Le combinateur de composition B :** 19
- **Le combinateur de permutation C :** 19
- **Le combinateur identité I :** 20
- **Le combinateur de duplication W :** 20
- **Le combinateur d'effacement K :** 21
- **Le combinateur de distribution S :** 21
- **Le combinateur de coordination ϕ :** 22
- **Le combinateur de distribution ψ :** 22
- **Combinateurs complexes :** 22
- **Utilisation des puissances :** 23
- **Utilisation des distances :** 24

5. Théorème des combinateurs : 24

6. L'intérêt apporté par l'association de la logique combinatoire aux systèmes applicatifs : 26

Chapitre 3 : État de l'Art 27

1. Introduction : 28

2. Domaine d'application de la logique combinatoire : 28

3. Mises en œuvre fondées à base de la logique combinatoire : 29

3.1. Haskell: 29

3.2. Unlamda: 29

3.3. Lojban: 29

3.4. X-SAIGA: 30

4. Les grammaires catégorielles : 30

4.1. Calcul Lambek : 30

4.2. Grammaire applicative et universelle (GAU) : 32

4.3. Grammaire catégorielle combinatoire applicative : ... 33
Chapitre 4 : La hiérarchie de Chomsky ... 36
1. Introduction : ... 37
2. La hiérarchie de Chomsky : ... 37
3. Type 0 : grammaires générales : ... 38
4. Type 1 : grammaires contextuelles : ... 39
5. Type 2 : grammaires non contextuelles : ... 40
6. Type 3 : grammaires régulières : ... 41
Chapitre 5 : La capacité générative de la logique combinatoire ... 44
1. Introduction : ... 45
2. Exemples d'expressions combinatoires à distance : ... 45
3. Analyse des exemples : ... 46
4. Algorithme: ... 49
5. Commenter la capacité générative de notre travail ... 52
Chapitre 6 : Implémentation et résultats ... 55
1. Introduction : ... 56
2. Langages et plateformes utilisés : ... 56
3. Implémentation de combinateurs simples : ... 57
4. Implémentation de la fonction de simplification : ... 65
5. Implémentation d'un combinateur complexe : ... 65
6. Implémentation combinateurs de puissance : ... 66
7. Implémentation combinateurs de distance : ... 67
8. Capacite générative : ... 68
9. Interface graphique : ... 70
10. Résultats : ... 72
Chapitre 7 : Conclusion ... 74
Références : ... 76
Annexe: ... 79

LISTE DES FIGURES

Figure 1 Le modèle de la Grammaire Catégorielle Universelle de Shaumyan [11]..... 33
Figure 2 : Étapes d'une compilation..... 35
Figure 3: Hiérarchie de Chomsky..... 37
Figure 4: Exécution de l'expression générée..... 51
Figure 5: Exemple sur une expression N=5 52
Figure 6: Exemple sur une expression N=6 52
Figure 7: Exemple sur une expression N=7 52
Figure 8: Exemple sur une expression N=8 53
Figure 9: Exemple sur une expression N=9 53
Figure 10: Exemple sur une expression N=10..... 54
Figure 11 : La première partie de notre interface graphique 70
Figure 12: La deuxième partie de notre interface dédiée à la capacite générative. 71

LISTE DES TABLEAUX

Tableau 1 Les règles de la GCCA [5, 8]..... 34
Tableau 2 : série de résultats 72

Chapitre 1 : Introduction

La technologie a connu un changement radical au cours de ces dernières années. Que ce soit dans le domaine de l'informatique ou de l'intelligence artificielle; de nouvelles recherches et inventions plus avancées ont cédé leurs places à des machines qui offrent l'excellence et qui nous aident à imiter les connaissances humaines. Cependant, ces nouvelles inventions sont le résultat de nombreuses technologies qui fonctionnent dans l'ombre et qui permettent de fournir des services qui sont proches de nos connaissances humaines. Le système expert basé sur des règles est l'une de ces technologies, il est largement utilisé dans tous les secteurs pour développer des applications et des systèmes d'intelligence artificielle.

Un système basé sur des règles est un système qui applique des règles créées par l'humain. Il permet d'effectuer diverses tâches telles que : des diagnostics, résoudre un problème, stocker, trier, manipuler des données, interpréter ou déterminer un plan d'action dans une situation particulière. Ce faisant, il imite l'intelligence humaine.

Présente au cœur des processus automatisés, cette technologie aide à développer des systèmes et des applications basés sur la connaissance, c'est-à-dire des programmes et des logiciels intelligents capables de fournir une expertise spécialisée dans la résolution de problèmes dans un sujet spécifique en utilisant des connaissances spécifiques au domaine (par exemple : le traitement de langue naturel qui nous intéresse).

Dans ces systèmes, les connaissances sont codées sous la forme de faits, d'objectifs et de règles, et sont utilisées pour évaluer et manipuler les données. De plus, ceux-ci sont appliqués à des systèmes impliquant des règles créées ou organisées par l'homme qui peuvent être utilisées pour effectuer une analyse lexicale afin de compiler ou d'interpréter des programmes informatiques, ou dans le traitement du langage naturel, d'où l'appellation de systèmes applicatifs.

Par ailleurs, il existe dans la littérature scientifique plusieurs systèmes applicatifs typés, dont le lambda-calcul de Church et la logique combinatoire. La plupart des langages fonctionnels comme LISP utilisent le lambda-calcul comme base de leur modélisation.

Bien qu'utilisé dans certains langages fonctionnels comme Haskell, la logique combinatoire[1,2] est en particulier, utilisée dans des approches d'analyse syntaxique, sémantique et même cognitive des langages naturels[3]. D'un point de vue extensionnel ces deux systèmes sont considérés comme équivalents. Néanmoins, ils ne le sont pas d'un point de vue intentionnel[4]. Contrairement au lambda-calcul, la logique combinatoire n'utilise pas de variables. Elle utilise des opérateurs abstraits appelés combinateurs afin de composer ou de transformer des opérateurs pour obtenir des opérateurs plus complexes. Les combinateurs sont indépendants d'une interprétation restrictive à une utilisation spécifique. L'action des combinateurs est exprimée par une règle unique appelée règle de β-réduction ; elle définit l'équivalence entre l'expression logique sans combinateur et celle avec combinateur.

Dans de précédents travaux tels que ceux conduits par « Ismail Biskri et J.P Desclés » [8], les auteurs ont pu démontrer qu'il était possible d'adopter les systèmes applicatifs typés afin d'offrir un cadre général, flexible et adaptable pour créer des chaines de traitement dans le cadre de l'ingénierie des données textuelles. La capacité générative ainsi que la flexibilité des combinateurs, qui sont les opérateurs abstraits utilisés dans le modèle des systèmes applicatifs typés, ont permis de mettre en évidence plusieurs propriétés très prometteuses.

La logique combinatoire est souvent considérée comme étant hors contexte, le présent projet de recherche consiste à démontrer que la logique combinatoire et le modèle applicatif typé sont capables de générer une capacité générative supérieure à la plupart des grammaires qui existent, et ainsi prouver que c'est une logique qui est sensible au contexte. Il sera utile pour permettre « le proof of concept ».

Bien que le modèle applicatif typé ait été validé par plusieurs publications, sa mise en œuvre n'est pas facile et elle rencontrera plusieurs défis principalement techniques à surmonter.

Il est important de rappeler que le présent travail est un travail théorique qui ne sera pas présenté sous la forme d'un schéma classique : théorie, implémentation suivie d'une partie expérimentation. Sa présentation se déroulera comme suit :

Nous allons présenter dans le chapitre qui suit (deuxième chapitre) les systèmes applicatifs et la logique combinatoire, ainsi que l'intérêt de leur association.

Dans notre troisième chapitre nous présenterons une étude sur l'état de l'art de la logique combinatoire

Notre quatrième chapitre sera dédié à présenter la hiérarchie de Chomsky et les différents grammaires et langages.

Le cinquième chapitre quant à lui est consacré à démontrer que la logique combinatoire est capable d'avoir une capacité générative supérieure à la plupart des grammaires qui existent.

Notre sixième chapitre contiendra la présentation de la mise en œuvre de notre application.

Chapitre 2 : Les systèmes applicatifs et la logique combinatoire

1. Introduction :

Les systèmes applicatifs et la logique combinatoire représentent le noyau de notre projet, afin d'expliquer au mieux le contexte de notre travail; nous allons consacrer ce chapitre à la présentation de ces deux notions.

Pour se faire nous introduirons dans un premier temps les systèmes applicatifs, suivis d'une brève présentation de la logique combinatoire; de son cadre historique et de ses objectifs.

Dans un second temps nous présenterons quelques combinateurs élémentaires ainsi que leurs actions. Par la suite nous introduirons quelques combinateurs que nous utiliserons tout au long de notre travail; il s'agit des combinateurs complexes, de puissance et de distance.

Pour finir nous aborderons l'intérêt de l'association des systèmes applicatifs et de la logique combinatoire.

2. Systèmes applicatifs typés :

Globalement le langage applicatif est composé de trois classes d'objets :

- Les termes.
- Les propositions.
- Les opérateurs.

Les systèmes applicatifs postulent un modèle général dans lequel une opération de construction applique un opérateur à un opérande pour donner un résultat. Les expressions sont structurées par une simple juxtaposition de deux arguments, un opérateur suivi par son opérande. Si X est un opérateur et Y est son opérande, alors XY représente l'application de X à Y.

Les expressions (XY) Z et X (YZ) sont aussi des expressions applicatives. Dans (XY) Z, que nous pouvons également écrire XYZ, X est un opérateur appliqué à son

premier opérande Y pour construire l'opérateur complexe XY dont l'opérande est Z. Tandis que dans X (YZ), Y est un opérateur appliqué à son opérande Z pour construire YZ l'opérande de l'opérateur X. Notant que les expressions (XY) Z et X (YZ) ne sont pas des expressions équivalentes, car l'opération d'application n'est pas associative.

L'ensemble des expressions applicatives est récursif, il est construit selon les règles suivantes:

- Les propositions (p) et les termes (t) sont des types de base.
- Les opérateurs de base et les opérandes de base sont des expressions applicatives.
- Soit A et B des expressions applicatives, alors l'expression AB est applicative.

D'un autre côté, est afin de ne pas construire des expressions contradictoires, ou pour empêcher certaines raisonnements, Curry a développé en 1934 et 1936 la théorie des types, puis cette dernière a été développée une autre fois en 1970 par Morris et Milner dans le cadre des langages de programmation fonctionnelles. [7]

Selon Curry :

- Chaque terme possède un ensemble de types possibles, mais généralement un parmi ces types sera le plus général.
- Si un terme X possède un type β donc nous pouvons le noter de la façon suivante X : β
- Si un terme Y possède un type β→α et le terme X possède un type β alors (YX) aura le type α (YX : α).

L'ensemble des types d'un système applicatif est défini comme suit:[5-7]

- Les types de bases sont des types.
- Si α et β sont des types alors Fαβ est un type.
- F est un opérateur pour simplifier les types.
- Fαβ est un type fonctionnel qui représente le type de toutes les fonctions allant de α vers β.

La règle applicative suivante nous permettra la simplification des types obtenus :

$$\frac{F\alpha\beta \quad , \quad \alpha}{\beta}$$

- $F\alpha\beta$ c'est le type de l'opérateur.
- α c'est le type de l'opérande.
- β c'est le type de l'application de l'opérateur à l'opérande.

Dans la partie suivante nous présenterons le schéma de type de quelques combinateurs.

3. La logique combinatoire, son cadre historique et ses objectives :

La logique combinatoire fût son apparition en 1924 grâce à Schônfinkel qui a introduit la notion de combinateurs [8]. Six ans plus tard; en 1930, quelques résultats de Schônfinkel ont été retrouvés indépendamment par Curry (logicien américain) qui les développera par la suite. Les travaux de Curry ont permis à Kleene en 1934 de représenter les nombres entiers et leurs fonctions. En 1935, Rosser, à son tour a pu parfaire l'œuvre de Curry grâce aux résultats de Kleene. [9]

En 1958 Curry et Feys ont repris cette logique pour l'adapter et minimiser le nombre d'opérateurs nécessaires afin de pouvoir l'intégrer dans les calculs de prédicats. [10]

La logique combinatoire est aussi liée au λ-*calcul* de Church (1941), qui est une règle destinée à la définition de l'action d'un combinateur sur un argument. [8,11] Aujourd'hui, ces deux systèmes sont largement utilisés par les informaticiens pour analyser correctement les propriétés sémantiques des langages de programmation tels que Java et Python. [5, 8]

La logique combinatoire est introduite pour apporter des solutions logiques à certains paradoxes (comme la logique de Russell [8]). Un objectif très intéressant qui est d'éliminer le besoin de variables mathématiques (dans sa forme actuelle, nous l'appelons: un système d'application sans variables).

4. Combinateurs logiques :

Un combinateur est un opérateur abstrait qui « nous permet de construire à partir d'opérateurs élémentaires des opérateurs de plus en plus complexes » [5]. L'action d'un combinateur sur un argument est simplifiée par une règle appelée β-réduction. [4,3] Le but de cette règle est d'établir une relation entre une expression avec combinateurs et une expression équivalente sans combinateurs. [5]

La logique de combinaison est basée sur deux "opérations" de base; que nous appellerons aussi deux "combineurs" S et K et que nous définirons plus tard. [8,12][4, 8, 27]

Nous présenterons ci-dessous quelques combinateurs logiques [8,12-14] :

- **Le combinateur de composition B :**

B est un combinateur à trois arguments, donc B porte sur une suite comme :abc. Ce combinateur s'applique réellement sur le premier élément de la suite (a), il effectue donc une composition entre le deuxième élément (b) et le troisième élément(c) de la suite.

Il est défini par la règle β-réduction suivante :

$$B\ abc \rightarrow a(bc)$$

$$B\ xyz \rightarrow x(yz)$$

- **Le combinateur de permutation C :**

À son tour le combinateur C est aussi un combinateur à trois arguments, donc comme nous l'avions expliqué auparavant; le combinateur C porte sur une suite comme : 012.

Ce combinateur s'applique réellement sur le premier élément de la suite, il effectue donc une permutation entre le deuxième et le troisième élément de la suite. Il est défini par la règle β-réduction suivante :

$$C\ abc \rightarrow acb$$

$$C\ 012 \rightarrow 021$$

Le combinateur C est de souvent utilisé pour décrire les verbes symétriques (débattre, rencontrer, croiser, etc…). Par exemple nous voulons prouver que l'expression : Jean croise Sara, est équivalente à l'expression : Sara croise Jean. L'expression combinatoire équivalente à Jean croise Sara, est : croise Jean Sara.

- Croise= C croise
- Croise Jean Sara ↔ C croise Jean Sara
- C croise Jean Sara → croise Sara Jean

- **Le combinateur identité I :**

Le combinateur I est un combinateur à argument unique. Il effectue l'identification de l'élément unique de la suite. Autrement dit le combinateur I n'a pas d'effet sur notre suite. Il est défini par la règle β-réduction suivante :

$$I\ x \rightarrow x$$

- **Le combinateur de duplication W :**

C'est un combinateur qui effectue des répétitions possibles. Cette répétition sera appliquée sur le deuxième élément de la suite. Il est défini par la règle β-réduction suivante :

$$W\ xz \rightarrow xzz$$

$$W\ ab \rightarrow abb$$

Les combinateurs W sont souvent utilisés dans les cas des prédicats réfléchis tels que : se-préparer, se-raser, ...etc. L'exemple suivant nous permettra de mieux expliquer son fonctionnement :

Prenons un prédicat réfléchi tel que **se-prépare** et appliquons-le sur **David** : Le résultat sera : **se-prépare David**.

Posons **se- prépare =W prépare**, notre expression sera comme suit **:**

W prépare David → prépare David David.

Cela montre que ces deux propositions sont sémantiquement équivalentes.

- **Le combinateur d'effacement K :**

C'est un combinateur qui effectue des éliminations possibles. L'élimination sera sur le deuxième élément de la suite. Donc Il prend deux arguments et rend son premier argument en résultat. Il est défini par la règle β-réduction suivante :

$$K\ xz \rightarrow x$$

$$K\ ab \rightarrow a$$

- **Le combinateur de distribution S :**

S est le combinateur de distribution selon Curry. Il est aussi le combinateur de substitution selon Steedman. Ce combinateur agit sur deux opérateurs Y et Z respectivement unaire et binaire, en leur associant un combinateur SYZ. Il est défini par la règle β-réduction suivante :

$$SYZX \rightarrow YX(ZX)$$

- **Le combinateur de changement de type C* :**

Si nous voulons changer le statut d'un opérande pour en faire un opérateur, nous aurons besoin du combinateur C*. Ce combinateur est défini par la règle β-réduction suivante :

$$C^*YZ \rightarrow ZY$$

$$C^*XY \rightarrow YX$$

Donc si Z est un opérateur ayant un opérande Y, en appliquant le combinateur C*, Z devient l'opérande de C* Z.

- **Le combinateur de coordination ϕ :**

Ce combinateur doit son nom à son rôle dans l'étude de la coordination. Le combinateur ϕ est un combinateur à quatre arguments, il s'applique à une séquence comme : ABCz. L'action du combinateur de coordination est définie par la règle β-réduction suivante :

$$\phi\, ABCz \rightarrow\ A(Bz)(Cz)$$

- **Le combinateur de distribution ψ :**

Λe combinateur de distribution ψ est aussi un combinateur à quatre arguments, qui se porte sur une suite comme : ABCz. L'action du combinateur de distribution est définie par la règle β-réduction suivante :

$$\psi ABCz \rightarrow\ A(BC)(Bz)$$

- **Combinateurs complexes :**

Ces combinateurs sont les résultats de l'association de plusieurs combinateurs élémentaires, par exemple : BBC, SSK, BSK, BBS, …etc.

Leurs actions sont déterminées par l'application successive des combinateurs élémentaires qui les composent, en commençant par le combinateur le plus à gauche vers celui de droite.[11,13] Ci-dessous un simple exemple sur le déroulement des combinateurs complexes :

- SS(KI)yz
- Sy((KI)y) z
- yz((KIy)z)
- yz(Iz)
- yzz

- **Utilisation des puissances :**

Avec l'exemple des combinateurs complexes B^2, W^3, C^3 …etc, nous appellerons combinateur à puissance tout combinateur de type X^n ou n>1, si X est régulier nous avons donc : $X^1=X$, $X^2=BXX$, $X^3=(BXX)X$ …etc ce qui nous conduit à : $X^n=BXX^{n-1}$.

Lors de l'étude des combinateurs à puissance, nous avons rencontré deux types de cas :

1. Le combinateur à puissance X^n qui n'a pas besoin d'un nombre d'arguments supérieur à celui de X à l'exemple de : I^n, C^n et W^n. [12]

I	Ix→x	C^1	Cxyz→ xzy	W^1	Wxy→xyy
I^2	Ix→x	C^2	Cxzy→ xyz	W^2	Wxyy→xyyy
I^3	Ix→x	C^3	Cxyz→ xzy	W^3	Wxyyy→xyyyy

Nous remarquons concernant le combinateur d'identité I que quel que soit le nombre de n $I^n=I$. Concernant le combinateur C; nous remarquons que le résultat de C^3 est le même résultat que celui de C^1 et le résultat de C^2 est le même résultat si nous appliquons le combinateur I. Nous pouvons conclure que généralement : $C^n = I$ si le n est pair et $C^n = C$ si le n est impair.

Dans le cas du combinateur de duplication W, W^n écrit le deuxième argument de W et le répète n fois, il le reproduit donc n+1 fois.

2. Le combinateur à puissance X^n qui a besoin d'un nombre d'arguments supérieur à celui de X à l'exemple de : K^n et B^n [9].

K^n exige n+1 arguments et les supprime tous sauf le premier, le B^n exige n+2 arguments et les associe tous (par une paire de parenthèse) sauf le premier. Nous montrons un simple exemple ci-dessous:

- B^2abcd
- BBB^1abcd
- B(Ba)bcd
- (Ba)(bc)d
- a(bcd)

- **Utilisation des distances :**

Les combinateurs à distance représentent un autre cas particulier des combinateurs complexes. Ces combinateurs sont appelés ainsi, car leur action se fait à distance, ils sont définis par la règle suivante : $X_n=B^{n-1}X$. [9]

Ci-dessous nous montrons un exemple à propos de l'utilisation des distances :

- C_2abcd
- BCabcd
- C(ab)cd
- (ab)dc
- abdc

5. Théorème des combinateurs :

Selon Curry nous pouvons exprimer tous les combinateurs en fonction des combinateurs S et K. [5]

Afin d'expliquer au mieux ce théorème; nous allons présenter quelques exemples :

- Nous pouvons remplacer le combinateur I par le combinateur complexe SKK :

- Ix→x et I = SKK
- SKKx = Kx(Kx) = x (vrai)

- Dans le cas du combinateur B, il pourrait être remplacé par S(KS)K :

  B012 →**0(12)** et B=S(KS)K

  S(KS)K012 → (KS)0(K0)12

  →S(K0)12

  →K02(12)

  →**0(12)** (vrai)

- À son tour le combinateur W peut être remplacé par SS(K(SKK)) :

  Wfx→**fxx** et W=SS(K(SKK))

  SS(K(SKK))fx→Sf((K(SKK))f)x

  →fx(((K(SKK))f)x)

  →fx((SKK)x)

  →fxKxKx

  →**fxx** (vrai)

- Il en est de même pour le combinateur de permutation(C) ; qui peut être à son tour remplacé par S(BBS)(KK) :

  Cfxy→**fyx** et C=S(BBS)(KK)

  S(BBS)(KK)fxy→(BBS)f((KK)f)xy

  →B(Sf)((KK)f)xy

  →(Sf)(((KK)f)x)y

  →fy((((KK)f)x)y)

  →fyKxy

  →**fyx** (vrai)

  ffghx→f(gx)(hx) et f=B(BS)B

  B(BS)Bfghx→(BS)(Bf)ghx

  →S((Bf)g)hx

  →((Bf)g)x(hx)

  →f(gx)(hx)

6. L'intérêt apporté par l'association de la logique combinatoire aux systèmes applicatifs :

Les systèmes applicatifs permettent d'exprimer des fonctions et leurs séquences [15]. Ils sont utilisés pour décrire l'application continue de plusieurs fonctions dans une chaîne de traitements, par exemple : M est une fonction qui s'applique à deux entrées, N est une fonction qui s'applique à une seule entrée et « 0, 1 » des valeurs.

Si nous désirons prendre le résultat renvoyé par l'application de la fonction N à la valeur 0 comme une première entrée pour la fonction M, l'expression sera: M(N0)1. Bien que ce mode d'expression soit lisible par l'homme, l'inconvénient est que les noms de fonction et leurs entrées seront mélangés.

L'utilisation des combinateurs logiques nous donnera une solution définitive à cet inconvénient car elle nous permettra d'exprimer autrement cet enchainement de fonctions. Par exemple, en associant la logique combinatoire dans le cas de l'expression précédente M(N0)1; l'expression pourrait être écrite de la façon suivante: **BMN01**. [12] Notre nouvelle expression sera divisée en trois parties :

- ✓ Les combinateurs : dans notre cas c'est B.
- ✓ Les fonctions : M, N dans notre cas.
- ✓ Et enfin les valeurs : 0,1.

Le rôle des combinateurs et des fonctions consiste à présenter la chaine de traitements et l'ordre d'application, quant aux valeurs d'entrée; elles nous donneront l'ensemble des entrées attendues pour cette chaine. [12]

Après avoir présenté les systèmes applicatifs, la logique combinatoire et ses différents combinateurs, nous avons montré à la fin de notre chapitre l'intérêt apporté par l'association de la logique combinatoire aux systèmes applicatifs. Dans le chapitre suivant nous allons montrer quelques mises en œuvre à base de la logique combinatoire ainsi que son importance dans de différents domaines, et en particulier dans le domaine linguistique.

Chapitre 3 : État de l’Art

1. Introduction :

Dans ce chapitre une étude sur l'état de l'art de la logique combinatoire est présentée en trois parties:

La première est consacrée à présenter les différents domaines d'application de la logique combinatoire, suivie d'une deuxième partie consacrée aux mises en œuvre basées sur le concept de la logique combinatoire.

La dernière partie quant à elle est consacrée aux différentes grammaires catégorielles.

2. Domaine d'application de la logique combinatoire :

La logique combinatoire joue un rôle primordial dans les fondements des mathématiques, et dans la programmation fonctionnelle en informatique. Dans ce dernier exemple, cette logique est considérée comme un modèle simplifié de calcul; elle est utilisée dans la théorie de calculabilité et des preuves. D'un autre coté elle peut être considérée comme une variante du calcul λ, où un ensemble de combinateurs limités et de fonctions primitives remplacent les expressions λ. Bien qu'il soit facile de convertir des expressions lambda en expressions combinatoires, la réduction combinatoire est beaucoup plus simple que la réduction lambda.

La logique combinatoire était aussi largement utilisée dans la modélisation de quelques langages fonctionnels à l'exemple des langage Unlamda et Haskell que nous allons aborder ci-dessous. Nous présentons dans la section qui suit quelques mises en œuvre qui ont été fondées à base de la logique combinatoire.

3. Mises en œuvre fondées à base de la logique combinatoire :

3.1. Haskell:

Haskell est un langage de programmation informatique purement fonctionnel qui porte le nom du mathématicien **Haskell Brooks Curry**; dont les travaux sur la logique combinatoire ont posé les bases de fondement des langages fonctionnels. Ce langage est principalement basé sur le calcul lambda , d'où son logo officiel qui contient la lettre grecque lambda.[2,16]

Quelques nouveaux langages de programmation connus se sont également orientés vers Haskell. Par exemple le langage universel Python avait repris la notation lambda et la syntaxe de traitement des listes de Haskell. [2]

3.2. Unlamda:

Unlamda inventé par D.Madore en 1999, est un langage de programmation fondé sur les notions de la logique combinatoire en ce qui concerne l'élimination des variables. Il est principalement basé sur deux fonctions intégrées (S et K) et sur un opérateur nommé Apply.[17]

3.3. Lojban:

Lojban est une langue artificielle, elle n'a cependant pas été conçue principalement pour être une langue internationale, mais plutôt pour être un outil linguistique pour étudier et comprendre une langue. Ses applications linguistiques et informatiques la rendent unique parmi les langues internationales, c'est un langage humain capable de tout exprimer.

La grammaire de Lojban utilise des éléments tirés de la logique mathématique, par exemple : des structures de type prédicat. Bien qu'elle n'utilise pas directement la logique

combinatoire; elle peut donner des indices et des analogies, sur l'utilité de la logique combinatoire en linguistique.

3.4. X-SAIGA:

L'objectif du projet X-SAIGA est de créer des algorithmes et des implémentations qui permettent de construire des processeurs de langage (reconnaisseurs, analyseurs, interprètes, traducteurs, etc.) sous forme de spécifications exécutables, embarquées, modulaires et efficaces de grammaires. L'analyse syntaxique est effectuée avec un ensemble de combinateurs d'analyseurs syntaxiques en surmontant certaines limitations de longue date.

D'un autre coté la logique combinatoire est très importante dans le domaine linguistique, elle est utilisée dans les théories des différentes grammaires catégorielle, et dans de différents programmes de traitement de langue que nous allons expliquer par la suite.

4. Les grammaires catégorielles :

4.1. Calcul Lambek :

Dans le but de résoudre quelques problèmes syntaxiques, et afin de faciliter l'analyse des phrases, le chercheur canadien J.Lambek, dans ses publications (de 1958 et de 1961) qui entrent dans le cadre des grammaires catégorielles, a introduit de nouvelles règles et notations [1,8,18]. Le système introduit par Lambek se compose de :

1. Des types primitifs S et N.
2. Deux opérateurs de construction : / et \.
3. Un opérateur de concaténation représenté par le symbole (+) par exemple : Pour deux expressions, T1 de type A et T2 de types B respectifs, leur concaténation sera comme suit : T1 + T2 et sera de type A + B.

4. Une règle de réduction « → ». Par exemple A → B signifie que le type A se réduit au type B.

Les règles suivantes fondent la base du système de calcul de Lambek [1,8,19]:

1. La réflexivité: A → A
2. L'associativité: (A + B)+C → A+ (B + C)

   A+ (B +C) → (A+ B) +C
3. La transitivité : Si A → B et B → C Alors: A → C

   Si A +B → C Alors A → C / B

   Si A + B → C Alors A → C \ A

   Si A → C / B Alors A + B → C

   Si B → C \ A Alors A +B → C

Les relations entre types peuvent être faites par les théorèmes suivants :

- A → (A B) IB
- (C / B) + B → C
- B → C \ (C / B)
- (C / B) + (B / A) → (C / A)
- C / B → (C / A) / (B / A)
- (B \ A) / C → (B / A) \ C
- (A / B) / C → A / (C+ B)
- Si A → A' et B → B' Alors A B → A' B'
- Si A → A' et B → B' Alors A \ B' → A' / B

4.2. Grammaire applicative et universelle (GAU) :

Dans le but de mettre en œuvre une grammaire applicative universelle, Shaumyan propose une distinction entre deux niveaux de représentation et d'analyse d'une langue : le niveau phénotype et le niveau génotype. [8,20].

- **Le niveau phénotypique:** généralement connu sous le nom de « niveau de surface » et regroupe l'ensemble des représentations concaténées. Il permet d'évoquer tous les traits spécifiques d'une langue donnée à travers la description de l'ordre de ses mots, sa syntaxe ou encore tous ses traits morphologiques. Ce niveau est présenté sous forme d'expressions concaténées. [8,13].
- **Le niveau génotypique :** Ce niveau permet de retracer les invariants sémiotiques, principaux constituants des langages naturels. La description de ces invariants se fait sous forme d'opérations et de relations, contrairement au niveau phénotypique ce niveau est présenté sous forme d'un langage abstrait appelé langage génotypique qui a ses propres règles qui se basent sur **la logique combinatoire de Curry** et sur le principe d'opérateur/opérande [5, 10].

La figure ci-dessous nous permettra de faire une distinction entre les deux niveaux citée auparavant [21] :

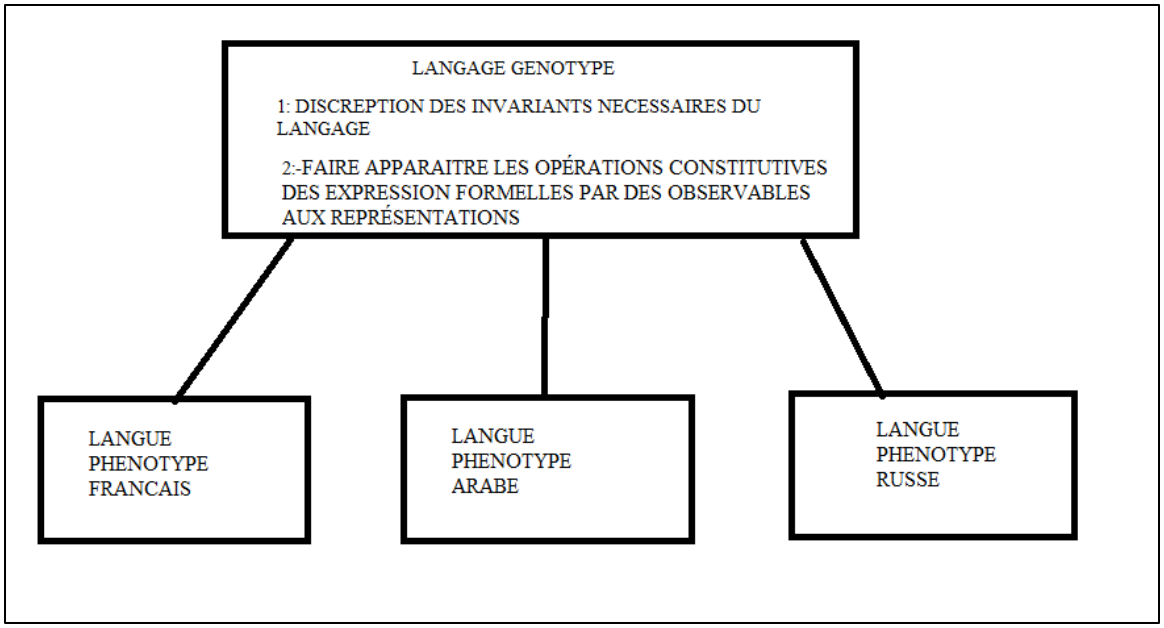


Figure 1 Le modèle de la Grammaire Catégorielle Universelle de Shaumyan [11]

4.3. Grammaire catégorielle combinatoire applicative :

La grammaire catégorielle combinatoire applicative (GCCA) consiste à relier les deux niveaux de la grammaire applicative universelle : phénotype et génotype, et ce par l'introduction d'un système formel.[1,8,11]

L'introduction des combinateurs logiques **S, B, C*** dans la séquence syntagmatique par les règles de la GCC, permettra le passage d'une structure concaténée à une structure applicative. [5]

Plus concrètement, la GCCA établit une association canonique entre : les règles catégorielles combinatoires de Steedman qui seront responsables de la vérification et de la correction syntaxique des énoncés, et les combinateurs logiques qui seront dans un premier temps responsable de la construction des expressions applicatives, et enfin responsables de la réduction des combinateurs. [8]

Le tableau suivant regroupe les règles de la GCCA; d'application, de changement de type et de composition :

Règle d'application	$[X/Y : u_1]\text{-}[Y : u_2]$ ----------> ; $[Y : u_1]\text{-}[X\backslash Y : u2]$ <---------- $[X : (u_1\ u_2)]$; $[X : (u_2\ u_1)]$
Règle de changement de type	$[X : u]$ ----------> **T** ; $[X : u]$ <---------- **T** $[Y/(Y\backslash X) : (\mathbf{C}_*, u)]$; $[Y\backslash(Y/X) : (\mathbf{C}_*\ u)]$ $[X : u]$ ----------> **Tx** ; $[X : u]$ <---------- **Tx** $[Y/(Y/X) : (\mathbf{C}_*, u)]$; $[Y\backslash(Y\backslash X) : (\mathbf{C}_*\ u)]$
Règle de composition	$[X/Y : u_1]\text{-}[Y/Z : u_2]$ ----------> **B** ; $[Y\backslash Z : u_1]\text{-}[X\backslash Y : u_2]$ <---------- **B** $[X/Z : (\mathbf{B}\ u_1\ u_2)]$; $[X\backslash Z : (\mathbf{B}\ u_2\ u_1)]$ $[X/Y : u_1]\text{-}[Y\backslash Z : u_2]$ ----------> **Bx** ; $[Y/Z : u_1]\text{-}[X\backslash Y : u_2]$ <---------- **Bx** $[X\backslash Z : (\mathbf{B}\ u_1\ u_2)]$; $[X/Z : (\mathbf{B}\ u_2\ u_1)]$

Tableau 1 Les règles de la GCCA [5, 8]

Les étapes du traitement proposé (fondé sur la GCCA) sont résumées dans la figure suivante :

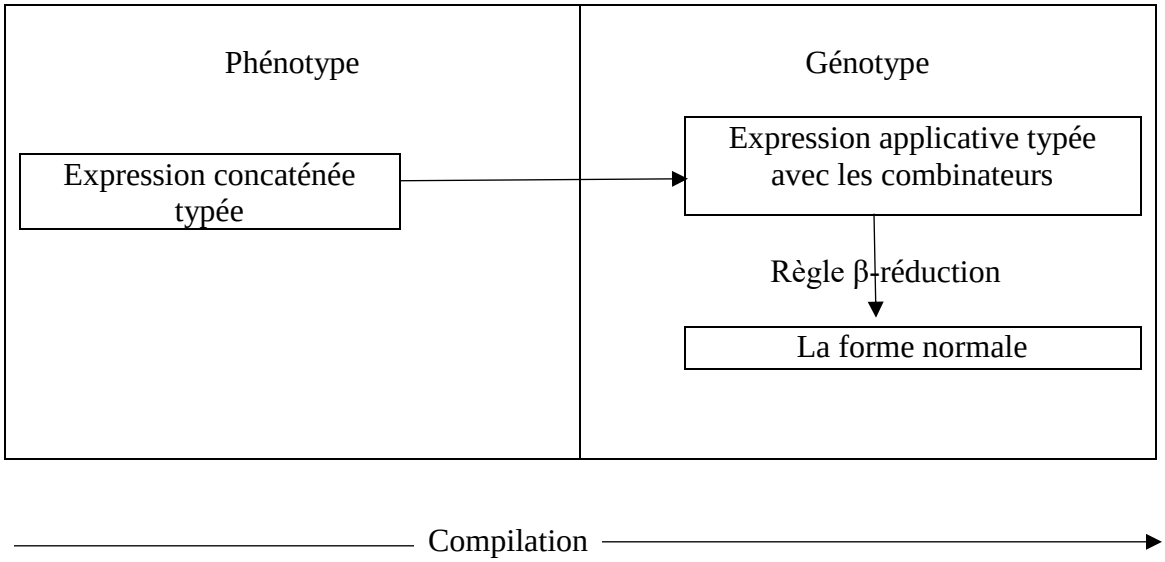


Figure 2 : Étapes d'une compilation

Autrement dit, selon le chercheur Ismail Biskri[8], un traitement complet fondé sur la GCCA s'effectue en deux étapes :

1. **La première étape : «**est consacrée à la vérification de la bonne connexion syntaxique et la construction de structures prédicatives avec des combinateurs introduits à certaines positions de la chaîne syntagmatique».[21]
2. **La deuxième étape** « consiste à utiliser les règles de β-réduction des combinateurs pour former une structure prédicative sous-jacente à l'expression phénotypique. L'expression obtenue est applicative et appartient au langage génotype ».

Si lors de ce chapitre nous avons pu traiter et cerner avec clarté le domaine de la logique combinatoire et son application, les deux chapitres suivants nous permettront de nous positionner dans ce domaine et de montrer l'originalité et le caractère novateur de notre travail.

Chapitre 4 : La hiérarchie de Chomsky

1. Introduction :

Nous présenterons dans ce chapitre la hiérarchie de Chomsky ainsi que les différents types de grammaires et de langages, nous expliquerons par la suite leur capacité générative.

2. La hiérarchie de Chomsky :

En 1956 le célèbre linguiste américain Noam Chomsky découvre une classification des langages décrite par les grammaires formelles et introduit une hiérarchie qui porte son nom : **la hiérarchie de Chomsky[22]**

Selon Chomsky, il existe 4 classes de langages de type 0 à type 3 hiérarchiquement imbriquées.

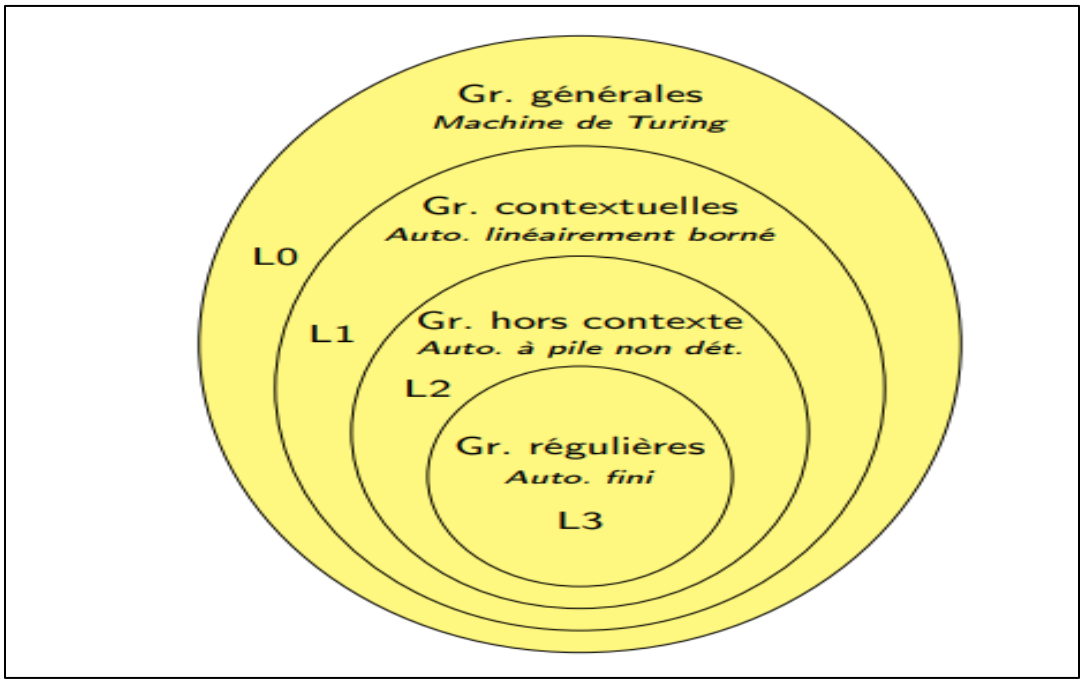


Figure 3: Hiérarchie de Chomsky

Les langages de type 0 sont les plus généraux et incluent donc les langages type 1 qui sont des langages **contextuels** qui incluent à leur tour les langages type 2 dits langages **hors-contexte** qui à leur tour incluent les langages type 3 dits **réguliers[23]**.

La classification des langages dépend des grammaires qui les génèrent. Ainsi, un l angage de type 1 est généré par une grammaire de type 1. Les grammaires sont classées selon la forme de leurs règles. [23,24]

Symboles terminaux et non-terminaux :

En informatique, les symboles utilisés dans les règles de production d'une grammaire formelle sont appelés Symbole terminaux et non-terminaux [25]

Les symboles non-terminaux se conforment au nom des règles de la grammaire et génèrent une séquence de terminaux et de non-terminaux, en outres les symboles terminaux appartiennent à la chaîne de caractères générés par la grammaire.

Pour présenter un nombre réel par exemple, nous proposons la grammaire suivante :

- <Entier> : = ['-'] <chiffre> {<chiffre>}
- <Réel> : = [','] <chiffre> {<chiffre>}
- <Chiffre> : = '-' | ',' | '9' | '8' | '7' | '6' | '5' | '4' | '3' | '2' | '1' | '0'

Dans la grammaire précédente les symboles : {-, ',', '9', '8', '7', '6', '5', '4', '3', '2', '1', '0' } sont des terminaux, tandis que <Entier>, <Réel>, <Chiffre> sont des symboles non-terminaux.

3. Type 0 : grammaires générales :

Ces grammaires génèrent une classe très grande de mots, plus exactement elles génèrent tous les langages qui peuvent être reconnus par une Machine de Turing. Ces langages sont également connus comme les langages récursivement énumérables. Dans un intervalle de temps fini nous saurons si la phrase appartient à cette grammaire, sinon la machine bouclera sans donner de réponse.

4. Type 1 : grammaires contextuelles :

Les grammaires contextuelles sont le niveau programmable le plus élevé, elles génèrent des langages contextuels. Ces grammaires sont définies par des règles du type :

$$\alpha A\beta \rightarrow \alpha\gamma\beta.$$

Ou le A est un non-terminal et les : **α**, **β** et **γ** appartiennent à une chaine de terminaux et de non terminaux. Tout simplement nous pouvons dire que le symbole non terminal **A** est remplacé par **γ** si nous avons les contextes **α** à gauche et **β** à droite[25,26].

Les grammaires de type 1 sont nommées contextuelles car le remplacement d'un élément non terminal dépend des éléments qui l'entourent (son contexte) [24, 25].

Exemple : Nous considérons la grammaire contextuelle suivante :

- S → xyz/xXyz
- Xy → yX
- Xz → Yyzz
- yY → Yy
- xY → xx/xxX

Cette grammaire est constituée de cinq règles. Dans chaque règle nous pouvons remplacer l'élément de gauche par l'élément de droite. Nous pouvons par exemple remplacer (Xy) par (yX).

Nous voulons savoir quel langage est généré par cette grammaire :

- S → xXyz
- → xyXz
- → xyYyzz
- → xYyyzz
- → xxXyyzz
- → xxyXyzz
- → xxyyXzz
- → xxyyYyzzz
- → xxyYyyzzz
- → xxYyyyzzz

- → xxxyyyxxx

A la première étape et selon notre grammaire nous avons remplacé « S » par «xXyz ». A la deuxième étape et selon notre grammaire nous avons remplacé x«**Xy** »z par «**yX»** et nous avons obtenu xyXz.A la troisième étape nous avons remplacé xy«**Xz** » par «**Yyzz»** et nous avons obtenu xyYyzz, ainsi de suite. A l'avant dernière étape et encore selon notre grammaire nous avons remplacé x«**xY**» yyyzzz par **«xx» »** et nous avons obtenu xxxyyyzzz qui sera notre résultat final.

Nous pouvons dès lors dire que cette grammaire engendre le langage non algébrique suivant : $\mathbf{a^n b^n c^n \mid n \geq 1}$

5. Type 2 : grammaires non contextuelles :

Ces grammaires génèrent les langages sans contexte. Ces grammaires sont définies par des règles du type : $\mathbf{A \rightarrow \gamma}$**.** Avec **A** un symbole non-terminal et **γ** une chaine composée de symboles terminaux et non terminaux.

Autrement dit, nous pouvons remplacer le non terminal **A** par **γ,** sans prendre en compte son contexte. Ces grammaires produisent avec précision des langages hors contexte dits aussi langages algébriques reconnus par « des automates à pile » [24, 25].
Voici un exemple d'une grammaire non contextuelle simple :

- $Z \rightarrow aZb$
- $Z \rightarrow \varepsilon$

Où ε correspond à la chaine vide.

Cette grammaire est constituée de 2 règles. Dans chacune des deux règles nous pouvons remplacer l'élément de gauche par l'élément de droite, par exemple nous pouvons remplacer (Z) par (aZb) selon la première règle ou bien par une chaine vide(ε) selon la deuxième règle.

Nous voulons savoir quel langage est généré par cette grammaire :

- $Z \rightarrow aZb$
- $\rightarrow aaZbb$
- $\rightarrow aabb$

- Au niveau de la première étape et selon notre grammaire nous avons remplacé «Z» par «aZb » ,
- A la deuxième étape et selon notre grammaire nous avons remplacé a«**Z** »b par «**aZb»** et nous avons obtenu aaZbb,
- A l'étape numéro trois nous avons remplacé aa«**Z** »bb par «**ε»** et nous avons obtenu aabb qui sera notre résultat final.

Nous pouvons dès lors dire que cette grammaire engendre le langage non rationnel suivant : $a^n b^n \mid n \geq 0$.

6. Type 3 : grammaires régulières :

Il faut distinguer deux cas équivalents des grammaires régulières :

- Les grammaires linéaires gauches. Dont les règles sont de la forme :
  - $A \rightarrow B\alpha$
  - $A \rightarrow \alpha$
  - $B \rightarrow \varepsilon$

  Le **A** et **B** sont des non terminaux et le : **α** est un terminal.

Cette grammaire est constituée de trois règles. Dans chaque règle nous pouvons remplacer l'élément de gauche par l'élément de droite, par exemple nous pouvons remplacer (A) par (Bα) selon la première règle ou bien par (α) selon la deuxième règle, tandis que le (B) pourrait être remplacé par une chaine vide (**ε**) selon la troisième règle.

- Les grammaires linéaires droites. Dont les règles sont de la forme :
  - $A \rightarrow \alpha B$
  - $A \rightarrow \alpha$
  - $A \rightarrow \varepsilon$

Cette grammaire est constituée de trois règles. Dans chaque règle nous pouvons remplacer l'élément de gauche par l'élément de droite, par exemple nous pouvons remplacer (A) par (αB) selon la première règle ou bien par (α) selon la deuxième règle, ou bien par une chaine vide (**ε**) selon la troisième règle.

Le **A** et **B** sont des non terminaux et le : **α** est un terminal.

Nous pouvons dès lors dire que le coté de gauche de chaque règle contient un seul symbole non-terminal, tandis que le membre de droite contient un symbole terminal et/ou un autre symbole non terminal.
Le symbole non terminal doit toujours être à droite du symbole terminal pour les grammaires régulières à droite, tandis que pour les grammaires régulières à gauche il doit se trouver à gauche [24, 25]. Un exemple de grammaire linéaire droite :

- $S \rightarrow \alpha S$
- $S \rightarrow \beta C$
- $S \rightarrow \beta$
- $C \rightarrow \gamma C$
- $C \rightarrow \gamma$

Cette grammaire est constituée de cinq règles. Dans chaque règle nous pouvons remplacer l'élément de gauche par l'élément de droite, par exemple nous pouvons remplacer (S) par (αS) selon la première règle ou bien par (βC) selon la deuxième règle, ou bien par (β) selon la troisième règle.

Nous voulons savoir quel langage est généré par cette grammaire :

- S→αS
- →ααS
- →ααβ

A la première étape et selon notre grammaire nous avons remplacé «**S**» par «αS » ,à la deuxième étape et selon notre grammaire nous avons remplacé α«**S**» par «**αS»** et nous avons obtenu **αα**S, à la troisième étape nous avons remplacé **αα«S»** par «β**»** et nous avons obtenu **αα**β qui sera notre résultat final.

Nous pouvons dès lors dire que cette grammaire engendre le langage suivant :

$a^n b^m \mid n \geq 1$.

Les modèles que nous venons d'expliquer se basent sur des règles de réécriture, dans le prochain chapitre nous allons passer à un modèle dans lequel l'opération fondamentale est l'opération de l'application d'un opérateur à son opérande.

Chapitre 5 : La capacité générative de la logique combinatoire

1. Introduction :

La logique combinatoire est souvent considérée dans la littérature comme étant hors contexte, nous démontrerons dans ce qui suit que c'est une logique qui est sensible au contexte. Nous pourrons ainsi déterminer sa capacité générative. Le but de ce chapitre est de montrer que la logique combinatoire est capable d'avoir une capacité générative supérieure à la plupart des grammaires qui existent.

2. Exemples d'expressions combinatoires à distance :

Dans cette partie nous présentons une série d'exemples d'expressions combinatoires ainsi que leurs résultats qui sont de type ($A^n B^n C^n$) et qui seront utiles pour la suite de notre travail :

- $W_2\ W_3\ W_4$ 0 1 2 >> 00 11 22>> $0^2 1^2 2^2$
- $W_5\ W_6\ W_7\ W_2\ W_4\ W_6$ 0 1 2 >> 000 111 222>>$0^3 1^3 2^3$
- $W_8\ W_9\ W_{10}\ W_5\ W_7\ W_9\ W_2\ W_5\ W_8$ 0 1 2 >> 0000 1111 2222>>$0^4 1^4 2^4$
- $W_{11}\ W_{12}\ W_{13}\ W_8\ W_{10}\ W_{12}\ W_5\ W_8\ W_{11}\ W_2\ W_6\ W_{10}$ 0 1 2 >> 00000 11111 22222>>$0^5 1^5 2^5$
- $W_{14}\ W_{15}\ W_{16}\ W_{11}\ W_{13}\ W_{15}\ W_8\ W_{11}\ W_{14}\ W_5\ W_9\ W_{13}\ W_2\ W_7\ W_{12}$ 0 1 2 >> 000000 111111 222222>>$0^6 1^6 2^6$
- $W_{17}\ W_{18}\ W_{19}\ W_{14}\ W_{16}\ W_{18}\ W_{11}\ W_{14}\ W_{17}\ W_8\ W_{12}\ W_{16}\ W_5\ W_{10}\ W_{15}\ W_2\ W_8\ W_{14}$ 0 1 2 >> 0000000 1111111 2222222>>$0^7 1^7 2^7$

Nous remarquons une régularité dans les expressions combinatoires qui nous permet de construire des expressions de type : $0^N 1^N 2^N$

Pour représenter cette régularité nous aurons besoin de déterminer quatre paramètres :M, N, J, I. Les paramètres M et N sont des paramètres fixes qui représentent respectivement : le nombre d'arguments à dupliquer et le nombre de duplications voulues. Tandis que les paramètres I et J sont des paramètres propres à notre algorithme.

Nous nous basons sur la régularité remarquée, de ce fait nous avons mis en place une formule basée sur ces quatre paramètres afin de calculer les indices de distance des combinateurs W ($W_{[(n-i)*m]-1+j}$)

Nous montrerons dans la partie suivante en se basant sur quelques exemples précédents comment cette formule nous permettra de calculer l'indice de distance de chaque combinateur, et nous montrerons comment nous obtiendrons chacun des combinateurs.

3. Analyse des exemples :

- $W_2\ W_3\ W_4$ 0 1 2 >> 00 11 22 >> $0^2 1^2 2^2$

  Dans cet exemple les valeurs de nos paramètres seront comme suit :

  1. Pour créer W_2 et en se basant sur la formule ($W_{[(n-i)*m]-1+j}$) les valeurs de nos paramètres seront : M=3, N=2, I=1, J=0.
  2. M=3, N=2, I=1, J=1 pour crée W_3 ($W_{[(2-1)*3]-1+1*1}$)
  3. M=3, N=2, I=1, J=2 pour crée W_4 ($W_{[(2-1)*3]-1+2*1}$)

Dans ce cas nous avons 3 arguments à dupliquer (0,1,2) M=3, et nous voulons les dupliquer chacun 2 fois : N=2. D'un autre côté le paramètres I est fixe dans ce cas (I=1), mais le paramètre J varie au cours des trois étapes (J=0…2).

- $W_5\ W_6\ W_7\ W_2\ W_4\ W_6$ 0 1 2 >> 000 111 222

  Dans cet exemple les valeurs de nos paramètres seront comme suit :

  1. M=3, N=3, I=1, J=0 pour crée W_5 en se basant sur la formule : ($W_{[(3-1)*3]-1+0*1}$)
  2. M=3, N=3, I=1, J=1 pour crée W_6 ($W_{[(3-1)*3]-1+1*1}$)
  3. M=3, N=3, I=1, J=2 pour crée W_7 ($W_{[(3-1)*3]-1+2*1}$)
  4. M=3, N=3, I=2, J=0 pour crée W_2 ($W_{[(3-2)*3]-1+0*2}$)
  5. M=3, N=3, I=2, J=1 pour crée W_4 ($W_{[(3-2)*3]-1+1*2}$)
  6. M=3, N=3, I=2, J=2 pour crée W_6 ($W_{[(3-2)*3]-1+2*2}$)

Dans cet exemple nous avons 3 arguments à dupliquer (0,1,2) M=3, nous voulons les dupliquer chacun 3 fois : N=3. Les arguments I et J varient de la manière suivante :

- A partir de l'étape 1 et jusqu'à l'étape 3 nous fixons I (I=1) et nous varions J (J=0…2)
- A partir de l'étape 4 et jusqu'à l'étape 6 nous incrémentons I (I=2) et nous le fixons et nous revarions J de la même manière que les étapes précédentes (J=0…2)

Dans cet exemple nous remarquons que nous avons eu besoin de 6 combinateurs de distance W contrairement à l'exemple précèdent lorsque nous avons créé juste 3 combinateurs, nous expliquerons ça plus tard.

- $W_8\ W_9\ W_{10}\ \ W_5\ W_7\ W_9\ \ W_2\ W_5\ W_8\ \ 0\ 1\ 2$ >> 0000 1111 2222

Dans cet exemple les valeurs de nos paramètres seront comme suit :

1. M=3, N=4, I=1, J=0 pour crée W_8 ($W_{[(4-1)*3]-1+0*1}$)
2. M=3, N=4, I=1, J=1 pour crée W_9 ($W_{[(4-1)*3]-1+1*1}$)
3. M=3, N=4, I=1, J=2 pour crée W_{10} ($W_{[(4-1)*3]-1+2*1}$)
4. M=3, N=4, I=2, J=0 pour crée W_5 ($W_{[(4-2)*3]-1+0*2}$)
5. M=3, N=4, I=2, J=1 pour crée W_7 ($W_{[(4-2)*3]-1+1*2}$)
6. M=3, N=4, I=2, J=2 pour crée W_9 ($W_{[(4-2)*3]-1+2*2}$)
7. M=3, N=4, I=3, J=0 pour crée W_2 ($W_{[(4-3)*3]-1+0*3}$)
8. M=3, N=4, I=3, J=1 pour crée W_5 ($W_{[(4-3)*3]-1+1*3}$)
9. M=3, N=4, I=3, J=2 pour crée W_8 ($W_{[(4-3)*3]-1+2*3}$)

Dans cet exemple nous avons 3 arguments à dupliquer (0,1,2) M=3, nous voulons les dupliquer chacun 4 fois : N=4.

Dans cet exemple les arguments I et J varient de la manière suivante :

- A partir de l'étape 1 et jusqu'à l'étape 3 nous fixons I (I=1) et nous varions J (J=0…2).
- A partir de l'étape 4 et jusqu'à l'étape 6 nous incrémentons I (I=2) et nous le fixons, nous varions J de la même manière que les étapes précédentes (J=0...2).
- A partir de l'étape 7 et jusqu'à l'étape 9 nous réincrémentons I (I=3) et nous le fixons, nous varions J de la même manière que les étapes précédentes (J=0…2).

Aussi dans cet exemple nous remarquons que nous avons eu besoin de 9 combinateurs de distance W contrairement aux deux exemples précédents, nous expliquerons cela plus tard.

- $W_{11}\ W_{12}\ W_{13}\ W_8\ W_{10}\ W_{12}\ W_5\ W_8\ W_{11}\ W_2\ W_6\ W_{10}$ 0 1 2 >> 00000 11111 22222

Dans cet exemple la valeur de nos paramètres sera comme suit :

1. M=3, N=5, I=1, J=0 pour crée W_{11} en suivant la formule :($W_{[(5-1)*3]-1+0*1}$)
2. M=3, N=5, I=1, J=1 pour crée W_{12} ($W_{[(5-1)*3]-1+1*1}$)
3. M=3, N=5, I=1, J=2 pour crée W_{13} ($W_{[(5-1)*3]-1+2*1}$)
4. M=3, N=5, I=2, J=0 pour crée W_8 ($W_{[(5-2)*3]-1+0*2}$)
5. M=3, N=5, I=2, J=1 pour crée W_{10} ($W_{[(5-2)*3]-1+1*2}$)
6. M=3, N=5, I=2, J=2 pour crée W_{12} ($W_{[(5-2)*3]-1+2*2}$)
7. M=3, N=5, I=3, J=0 pour crée W_5 ($W_{[(5-3)*3]-1+0*3}$)
8. M=3, N=5, I=3, J=1 pour crée W_8 ($W_{[(5-3)*3]-1+1*3}$)
9. M=3, N=5, I=3, J=2 pour crée W_{11} ($W_{[(5-3)*3]-1+2*3}$)
10. M=3, N=5, I=4, J=0 pour crée W_2 ($W_{[(5-4)*3]-1+0*4}$)
11. M=3, N=5, I=4, J=1 pour crée W_6 ($W_{[(5-4)*3]-1+1*4}$)
12. M=3, N=5, I=4, J=2 pour crée W_{10} ($W_{[(5-4)*3]-1+2*4}$)

Dans cet exemple nous avons 3 arguments à dupliquer (0,1,2) M=3, nous voulons les duplique chacun 5 fois : N=5. D'un autre côté dans cet exemple les arguments I et J varient de la manière suivante :

- A partir de l'étape 1 et jusqu'à l'étape 3 nous fixons I (I=1) et nous varions J (J=0…2)
- A partir de l'étape 4 et jusqu'à l'étape 6 nous incrémentons I et nous le fixons (I=2) et nous varions J de la même manière que les étapes précédentes (J=0…2).
- A partir de l'étape 7 et jusqu'à l'étape 9 nous réincrémentons I et nous le fixons (I=3) et nous varions J de la même manière que les étapes précédentes (J=0…2).

- A partir de l'étape 10 et jusqu'à l'étape 12 nous réincrémentons encore une fois I et nous le fixons (I=4) et nous revarions J de la même manière que les étapes précédentes (J=0…2).

Ce que nous constatons à partir des exemples analysés, c'est que le nombre des arguments (0,1,2) et le nombre de duplications (N) influent sur le nombre de combinateurs de distance W de tel sorte que le nombre de combinateurs W sera égal à : M*(N-1); Exemple : N=2 →3 combinateurs W

N=3 →6 combinateurs W

N=4 →9 combinateurs W

Quant à la complexité de notre algorithme elle sera égale à : M*N.

4. Algorithme:

```
     M = | {0, 1, 2} |     ;     0^N 1^N 2^N

     I: 1 ...  N – 1      ;     J: 0 ... M - 1

Pour n allant de 2 à l'infini

     chaine = ""

     pour i allant de 1 à n - 1 faire

          pour j allant de 0 à 2 faire

               concaténer chaine et "W [(n - i) * m] - 1 + j * i"

               j = j + 1

          fin pour

     fin pour

     concaténer chaine et " 0 1 2"

     afficher chaine

     n= n + 1

fin pour
```

Afin de bien expliquer notre algorithme, nous allons présenter un exemple de son déroulement :

Nous voulons générer une expression combinatoire qui nous permettra d'atteindre le résultat suivant : $0^4 1^4 2^4$ (000011112222).

Sur cet exemple les valeurs de M (nombre d'arguments) et de N (nombre de duplications) seront comme suit : M=3 N=4.

Initialement notre chaine sera vide.

1. Initialement aussi notre paramètre I prendra la valeur de 1. Nous exécutons la boucle qui est à l'intérieur :
    - J sera initialisé à 0, en vertu l'expression qui se situe dans la deuxième boucle ($W_{[(4-1)*3]-1+0*1}$) nous allons avoir $\mathbf{W_8}$
    - Nous incrémentons le J qui prendra la valeur de 1, et en vertu de la même expression, nous allons avoir $\mathbf{W_9}$.
    - Nous réincrémentons encore une fois le J qui prendra cette fois la valeur de 2, et en vertu de l'expression ($W_{[(4-1)*3]-1+2*1}$), nous allons avoir $\mathbf{W_{10}}$.

    $\mathbf{W_8W_9W_{10}}$ seront successivement les trois premiers combinateurs à concaténer dans notre chaine initialement vide.
2. En sortant de la boucle précédente, nous incrémentons la valeur de I (I=2) et nous exécuterons la même boucle:
    - J sera initialisé à 0, en vertu de l'expression qui se situe dans cette boucle ($W_{[(4-2)*3]-1+0*2}$) nous allons avoir $\mathbf{W_5}$.
    - Nous incrémentons le J qui prendra la valeur de 1, en vertu de la même expression précédente, nous allons avoir $\mathbf{W_7}$
    - Nous réincrémentons encore une fois le J qui prendra la valeur de 2, et en vertu de la même expression, nous allons avoir $\mathbf{W_9}$.

    $\mathbf{W_5W_7W_9}$ seront successivement les trois deuxièmes combinateurs à concaténer dans notre chaine qui sera comme suit $\mathbf{W_8W_9W_{10}\ W_5W_7W_9}$
3. En sortant de cette boucle nous réincrémentons la valeur de I (I=3) et nous exécuterons la boucle qui est à l'intérieur :

- J sera initialisé à 0, en vertu l'expression qui se situe dans la deuxième boucle ($W_{[(4-3)*3]-1+0*3}$) nous allons avoir **W_2.**
- Nous incrémentons le J qui prendra la valeur de 1, et en vertu l'expression précédente ($W_{[(4-3)*3]-1+1*3}$) nous allons avoir **W_5.**
- Nous réincrémentons encore une fois le J qui va avoir la valeur de 2, et en vertu l'expression ($W_{[(4-3)*3]-1+2*3}$) nous allons avoir **W_8**.

$W_2W_5W_8$ seront les trois combinateurs à concaténer dans notre chaine qui sera comme suit **$W_8W_9W_{10}$ $W_5W_7W_9$ $W_2W_5W_8$.**

4. Une fois le I=N (dans notre cas I=4) nous sortons des deux boucles, nous concaténerons notre chaine (**$W_8W_9W_{10}$ $W_5W_7W_9$ $W_2W_5W_8$**) et nos arguments initiaux (012), et nous afficherons notre résultat final qui sera l'expression combinatoire suivante : **$W_8W_9W_{10}$ $W_5W_7W_9$ $W_2W_5W_8$** 012

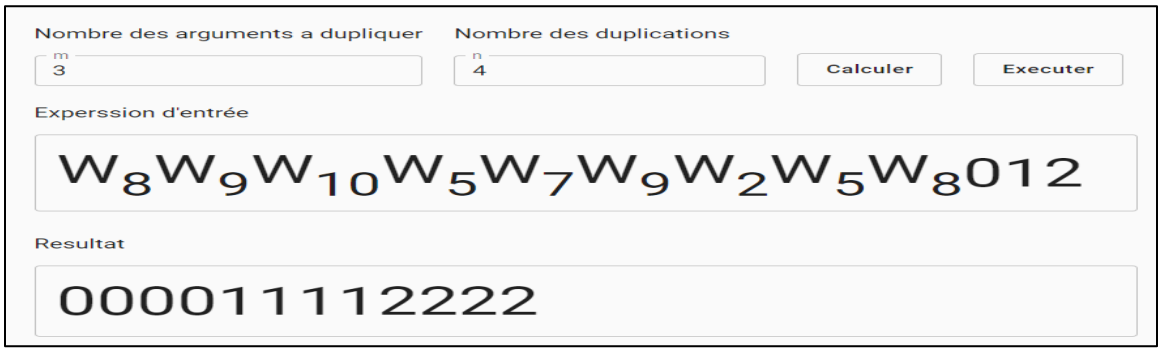


Figure 4: Exécution de l'expression générée

En exécutant l'expression combinatoire générée en appliquant les règles de β-réduction, nous constatons qu'effectivement nous avons bel et bien obtenu l'expression attendue, la figure ci-dessus le prouve.

5. Commenter la capacité générative de notre travail

Afin de s'assurer que notre algorithme fonctionne correctement; nous allons générer dans cette partie d'autres expressions combinatoires pour (N allant de 5 jusqu'à 10):

Figure 5: Exemple sur une expression N=5

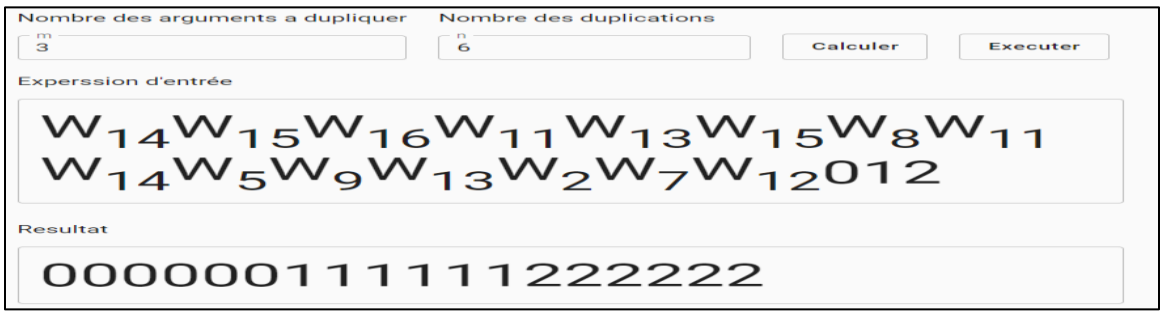


Figure 6: Exemple sur une expression N=6

Figure 7: Exemple sur une expression N=7

Figure 8: Exemple sur une expression N=8

Figure 9: Exemple sur une expression N=9

Figure 10: Exemple sur une expression N=10

En vertu de ce qui a été dit par rapport à la hiérarchie de Chomsky nous constatons qu'effectivement nous sommes capables de générer des expressions de type $\mathbf{0^n1^n2^n}$.

Nous pouvons affirmer dès lors que la logique combinatoire est une logique qui est sensible au contexte et qui est donc capable d'avoir une capacité générative supérieure à la plupart des grammaires qui existent.

L'implémentation des combinateurs utilisés dans notre thématique ainsi que nos résultats seront détaillées dans le chapitre suivant.

Chapitre 6 :
Implémentation et résultats

1. Introduction :

Nous présentons dans ce chapitre la mise en œuvre de notre application; nous commencerons par une présentation des langages et des plateformes que nous avons choisis; par la suite nous présenterons l'implémentation des différents combinateurs, ainsi que quelques interfaces illustrant les différentes options offertes par notre application.

Une série d'exemples : pour une expression à trois arguments 0,1,2 et pour **n** allant de 2 jusqu'à 100 est présentée en annexe.

2. Langages et plateformes utilisés :

Nous avons implémenté les différents combinateurs ainsi que notre algorithme de capacité générative dans une application Web. Pour le faire, nous-nous sommes orientés vers quatre langages et plateformes : Python, Flask, Javascript, Réact.

Nous avons utilisé Python ainsi que Flask pour implémenter nos différents combinateurs et notre algorithme, tandis que JavaScript et Réact ont été utilisés pour développer notre interface graphique.

Une brève description des quatre langages et plateformes utilisés sera présentée dans la section suivante.

Python : est l'un des langages de programmation les plus populaires, largement utilisé pour le développement de serveurs web, de logiciels, etc. Très simple à utiliser (a cause de sa syntaxe facile comparé à d'autres langages de programmation) et peut-être aussi utilisé par des différents paradigmes de programmation (oriente-objets, fonctionnel, etc.)

Flask : est un framework Web Python petit et léger qui fournit des outils et des fonctionnalités utiles qui facilitent la création d'applications Web en Python. Il offre de la flexibilité aux développeurs et constitue un cadre plus accessible pour les nouveaux développeurs, car il permet de créer rapidement une application Web à l'aide d'un seul fichier Python. Flask est également extensible et ne force pas une structure de répertoires particulière et ne nécessite pas de code passe-partout compliqué avant de commencer.

JavaScript : est un langage de programmation informatique dynamique. Il est léger et le plus souvent utilisé dans le cadre de pages Web, dont les implémentations permettent au script côté client d'interagir avec l'utilisateur et de créer des pages dynamiques. C'est un langage de programmation interprété avec des capacités orientées objet.

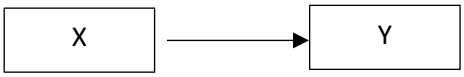
React : React est une bibliothèque JavaScript permettant de créer des applications modernes. React est utilisé pour gérer la couche de vue et peut être utilisé pour le développement d'applications Web et mobiles, Ce sont les fonctionnalités qui rendent la bibliothèque React si bonne et puissante et ce qui distingue également React des autres frameworks et bibliothèques

3. Implémentation de combinateurs simples :

Comme nous l'avons expliqué auparavant, chaque combinateur logique possède un nombre minimal d'arguments qui lui est propre.

- **Combinateur d'identité I:**

L'implémentation du combinateur d'identité a été réalisée en représentant les arguments sous forme d'une liste d'éléments, chaque élément de cette liste est considéré comme un argument indépendant. Dans notre exemple :X et Y sont deux arguments indépendants.



L'entrée de cette fonction est un nombre variable d'arguments qui est représenté par (*args), la sortie est également un nombre variable d'arguments selon la règle de réduction précédente.

- **Combinateur de composition B :**

Selon la règle de composition, le nombre minimal d'arguments à passer à cette fonction est de 3 par exemple : BXYZ.

Chaque argument correspond à ce qui suit : le premier argument (X dans notre exemple) c'est le premier opérateur, le deuxième argument (Y) c'est le deuxième opérateur et le troisième argument (Z) c'est l'opérande.

L'entrée de cette fonction est identique à l'entrée de la fonction précédente d'identité qui est présentée par un nombre variable d'arguments (*args). Dans notre exemple: X, Y et Z sont trois arguments indépendants.

Tandis que la sortie de cette fonction sera une liste d'arguments (au minimum deux) dans laquelle le premier élément de cette liste sera le premier opérateur (X dans notre exemple) et le deuxième élément sera une liste qui contient le deuxième opérateur et l'opérande combinés((YZ) dans notre exemple).

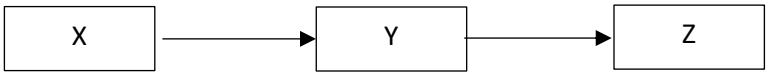
Dans le cas où le nombre d'arguments de cette fonction sera supérieur à 3, nous prendrons toujours en considération les trois premiers arguments les plus à gauche pour appliquer la règle de la composition tandis que les autres arguments seront retournés tel qu'ils sont dans une liste. Par exemple, si nous exécutons cette composition (B,"x", "y", "z", "w") le résultat sera ["x",["y", "z"], "w"].

Dans le cas où le nombre d'arguments est inférieur à 3 un message d'erreur sera affiché.

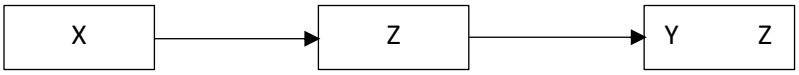
- **Le combinateur de substitution S :**

La règle de substitution est tout à fait identique à la règle de composition précédente, elle demande aussi le même nombre minimal d'arguments qui est : 3 arguments, par exemple SXYZ. Chaque argument correspond à ce qui suit : le premier argument (X dans notre exemple) c'est le premier opérateur, le deuxième argument (Y) c'est le deuxième opérateur et le troisième argument (Z) c'est l'opérande.

L'entrée de cette fonction est également un nombre variable d'arguments (*args).



La sortie à son tour est aussi une liste d'arguments (au minimum trois) dans laquelle le premier élément sera le premier opérateur (X dans notre exemple), et le deuxième élément sera l'opérande (Z dans notre exemple), tandis que le troisième élément sera une liste qui contient le deuxième opérateur (Y dans notre exemple) et de l'opérande(Z) combinés ensemble.



Dans le cas où le nombre d'arguments de cette fonction sera supérieur à 3, cette fonction prendra toujours en considération les 3 premiers arguments les plus à gauche pour appliquer la règle de distribution tandis que le reste d'arguments sera retourné tel qu'il est dans une liste. Par exemple, si nous exécutons (S,"X", "Y", "Z", "W") le résultat sera ["X", "Z",["Y", "Z"], "W"],

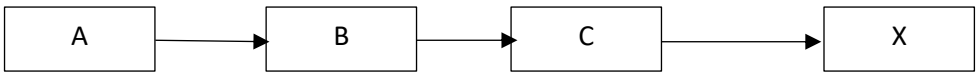
Dans le cas où le nombre d'arguments est inférieur à 3 un message d'erreur sera affiché.

- **Le combinateur de coordination ϕ :**

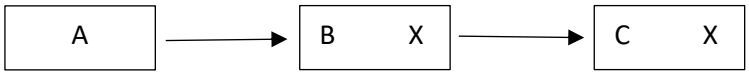
Selon la règle de coordination, le nombre minimum d'arguments à passer à cette fonction est de 4 : trois opérateurs est un opérande, par exemple : ϕ ABCX

Les trois premiers arguments à gauche présentent nos trois opérateurs, consécutivement : opérateur 1, opérateur 2 et opérateur 3, le quatrième argument sera notre opérande.

L'entrée de cette fonction est identique aux entrées des fonctions précédentes qui sont présentées par un nombre variable d'arguments (*args).



Tandis que la sortie de cette fonction sera une liste d'arguments (au minimum trois éléments) dans laquelle le premier élément de cette liste sera le premier opérateur (A dans notre exemple) et le deuxième élément sera une liste qui contient le deuxième opérateur et l'opérande combinés((BX) dans notre exemple), alors que le troisième élément sera une liste aussi qui contient le troisième opérateur combiné avec l'opérande ((CX) dans notre exemple).



Dans le cas où le nombre d'arguments de cette fonction sera supérieur à 4, cette fonction prendra toujours en considération les 4 premiers arguments les plus à gauche pour appliquer la règle de la coordination tandis que le reste d'arguments sera retourné tel qu'il est dans une liste. Par exemple, si nous exécutons (ϕ,"A", "B", "C", "X","Y") le résultat sera ["A",["B","X"], ["C", "X"], "Y"].

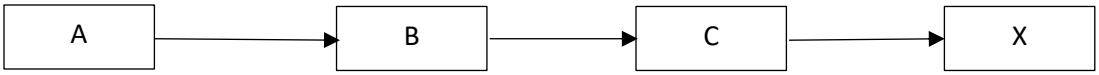
Dans le cas où le nombre d'arguments est inférieur à 4 un message d'erreur sera affiché.

- **Le combinateur de distribution Ψ :**

La règle de distribution a aussi besoin de quatre arguments aux minimums pour passer : trois opérateurs est un opérande, par exemple : Ψ ABCX

Les trois premiers arguments à gauche présentent nos trois opérateurs, consécutivement : opérateur 1, opérateur 2 et opérateur 3, le quatrième argument sera notre opérande.

L'entrée de cette fonction est identique aux entrées des fonctions précédentes qui sont présentées par un nombre variable d'arguments (*args).



Tandis que la sortie de cette fonction sera une liste d'arguments (au minimum trois éléments) dans laquelle le premier élément de cette liste sera le premier opérateur (A dans notre exemple) et le deuxième élément sera une liste qui contient le deuxième opérateur et troisième opérateur combinés((BC) dans notre exemple), alors que le troisième élément sera une liste aussi qui contient le deuxième opérateur combine avec l'opérande ((BX)dans notre exemple).

Dans le cas où le nombre d'arguments de cette fonction sera supérieur à 4, cette fonction prendra toujours en considération les 4 premiers arguments les plus à gauche pour appliquer la règle de la coordination tandis que le reste d'arguments sera retourné tel qu'il est dans une liste. Par exemple, si nous exécutons (Ψ,"A", "B", "C", "X","Y") le résultat sera ["A",["B","C"], ["B", "X"], "Y"].

Dans le cas où le nombre d'arguments est inférieur à 4 un message d'erreur sera affiché.

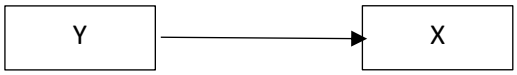
- **Le combinateur de changement de type C* :**

La règle de changement de type a besoin de deux arguments au minimum pour passer : un opérateur et un opérande, par exemple : C*YZ

Le premier argument à gauche sera notre opérateur tandis que le deuxième argument sera notre opérande.

Comme toutes les fonctions précédentes l'entrée de cette fonction sera présentée par un nombre variable d'arguments (*args).

Quant à la sortie de cette fonction elle sera une liste d'arguments (au minimum deux éléments) dans laquelle le premier élément sera notre opérande (Y dans notre exemple) et le deuxième élément sera notre opérateur (X dans notre exemple).



Dans le cas où le nombre d'arguments de cette fonction sera supérieur à 2, cette fonction prendra toujours en considération les 2 premiers arguments les plus à gauche pour appliquer la règle de changement de type tandis que le reste d'arguments sera retourné tel qu'il est dans une liste. Par exemple, si nous exécutons (C*,"A", "B", "C", "D",) le résultat sera ["B","A","C", "D"].

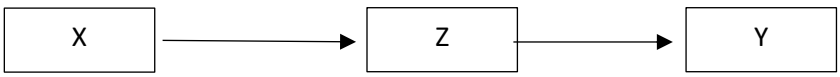
Dans le cas où le nombre d'arguments est inférieur à 2 un message d'erreur sera affiché.

- **Le combinateur de permutation C:**

À son tour le combinateur C est aussi un combinateur à trois arguments, donc la règle de permutation a besoin de trois arguments au minimum pour passer : deux opérateurs et un opérande par exemple : CXYZ

Il en est de même pour cette règle; son entrée sera également un nombre variable d'arguments (*args).

Sa sortie sera une liste d'arguments (au minimum trois) dans laquelle le premier élément sera le premier opérateur (X dans notre exemple), et le deuxième élément sera l'opérande (Z dans notre exemple), tandis que le troisième élément sera le deuxième opérateur (Y dans notre exemple).



Dans le cas où le nombre d'arguments de cette fonction sera supérieur à 3, cette fonction prendra toujours en considération les 3 premiers arguments les plus à gauche pour appliquer la règle de la permutation tandis que le reste d'arguments sera retourné tel qu'il est dans une liste. Par exemple, si nous exécutons (C,"x", "y", "z", "w") le résultat sera ["x", "z", "y", "w"],

Dans le cas où le nombre d'arguments est inférieur à 3 un message d'erreur sera affiché.

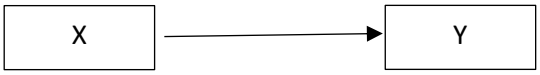
- **Le combinateur de duplication W :**

L'action du combinateur de duplication est définie par la règle β-réduction suivante :

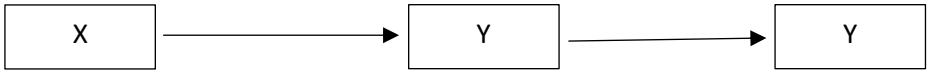
$$WXY \rightarrow XYY$$

La règle de duplication a besoin seulement de deux arguments au minimum pour passer : un opérateur et un opérande.

Le premier argument à gauche sera notre opérateur tandis que le deuxième argument sera notre opérande. A son tour la fonction de duplication sera présentée par un nombre variable d'arguments (*args).



Tandis que la sortie de cette fonction sera une liste d'arguments (au minimum trois éléments) dans laquelle le premier élément de cette liste sera notre opérateur (X dans notre exemple) et le deuxième élément sera notre opérateur (Y dans notre exemple), le troisième élément sera encore une fois notre opérande (Y dans notre exemple).



Dans le cas où le nombre d'arguments de cette fonction sera supérieur à 2, cette fonction prendra toujours en considération les deux premiers arguments les plus à gauche

pour appliquer la règle de changement de type tandis que le reste d'arguments sera retourné tel qu'il est dans une liste. Par exemple, si nous exécutons (W,"A", "B", "C", "D",) le résultat sera ["A","B","B", "C", "D"].

Dans le cas où le nombre d'arguments est inférieur à 2 un message d'erreur sera affiché.

- **Le combinateur d'effacement K :**

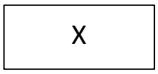
L'action du combinateur d'effacement est définie par la règle β-réduction suivante :

$$KXY \rightarrow X$$

La règle d'effacement a aussi besoin de deux arguments au minimum pour passer : un opérateur et un opérande.

Le premier argument à gauche sera notre opérateur tandis que le deuxième argument sera notre opérande. A son tour la fonction d'effacement sera présentée par un nombre variable d'arguments (*args).

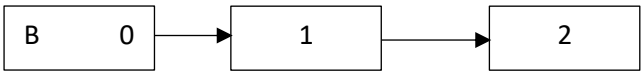
Tandis que la sortie de cette fonction sera une liste d'arguments qui contient notre opérateur seulement (X dans notre cas).



Dans le cas où le nombre d'arguments de cette fonction sera supérieur à 2, cette fonction prendra toujours en considération les 2 premiers arguments les plus à gauche pour appliquer la règle d'effacement tandis que le reste d'arguments sera retourné tel qu'il est dans une liste. Par exemple, si nous exécutons (K,"A", "B", "C", "D",) le résultat sera ["A", "C", "D"]. Dans le cas où le nombre d'arguments est inférieur à 2 un message d'erreur sera affiché.

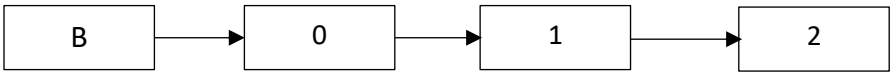
4. Implémentation de la fonction de simplification :

Cette fonction est chargée d'assurer l'exécution de la fonction **Combinateur Complexe**. Dans le cas où le premier élément à gauche de notre liste soit lui-même une liste nous devons éclater cette liste pour pouvoir l'exécuter. Par exemple : étant donné l'expression combinatoire suivante : (B 0)12. Dans ce cas notre entrée sera une liste qui contient trois éléments comme suit :



Sans l'utilisation de la fonction Simplification l'exécution s'arrêtera puisque notre premier élément est une liste ["B", "0"], pour résoudre ce problème nous appelons la fonction Simplification pour extraire ces éléments, ainsi notre nouvelle entrée sera :

["B", "0", "1","2"] :

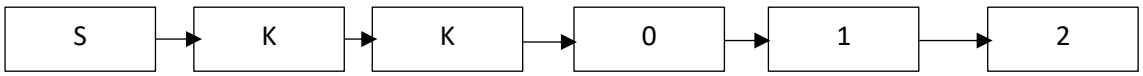


Maintenant nous pouvons exécuter cette expression combinatoire et notre sortie sera comme suit :

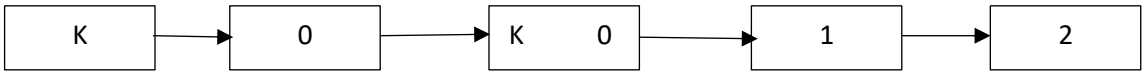
5. Implémentation d'un combinateur complexe :

L'exécution du combinateur complexe se fait par la mise en correspondance de la liste d'éléments que nous avons passée en entrée, nous nous intéressons toujours au premier élément à gauche. Nous prenons l'exemple d'un combinateur complexe SKK012, et nous allons présenter son exécution.

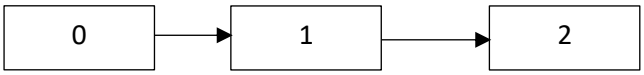
Dans ce cas notre entrée sera une liste qui contient six éléments, elle est représentée comme suit:



Nous commençons la simplification de notre expression, en commençant toujours par le premier élément à gauche (combinateur S dans notre exemple):



Nous obtenons une nouvelle liste qui va être exécutée encore une fois à son tour en commençant par le premier élément à gauche (combinateur K dans notre ce cas). La nouvelle liste obtenue sera notre résultat final:



6. Implémentation combinateurs de puissance :

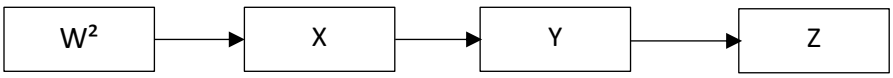
L'action du combinateur de puissance est définie par la règle β-réduction suivante :

$X^n = BXX^{n-1}$

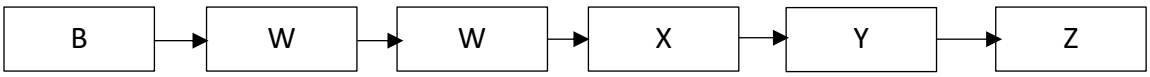
Nous allons expliquer ci-dessous un exemple d'exécution d'une expression combinatoire qui contient un combinateur de puissance, nous prenons l'exemple suivant : W^2XYZ.

$W^2XYZ \rightarrow BWWXYZ \rightarrow W(WX)YZ \rightarrow (WX)YYZ \rightarrow XYYYZ$.

Dans ce cas notre entrée sera une liste qui contient quatre éléments, elle est représentée comme suit:



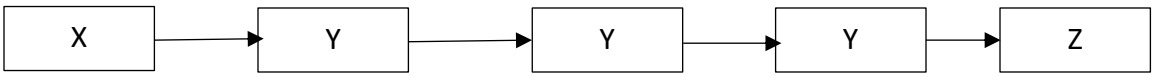
En appliquant la règle de puissance nous allons avoir une nouvelle forme de notre liste d'entrée :



Maintenant nous pouvons commencer à exécuter notre expression, en commençant toujours par le premier élément à gauche :

En arrivant à cette étape, notre premier élément sera composé de deux arguments, donc nous devrions l'éclater en utilisant la fonction de simplification présentée auparavant :

Nous continuerons notre exécution pour avoir notre sortie qui sera une liste à son tour :



7. Implémentation combinateurs de distance :

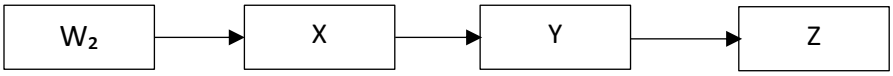
L'action du combinateur de distance est définie par la règle β-réduction suivante :

$X_i=B^{i-1}X$

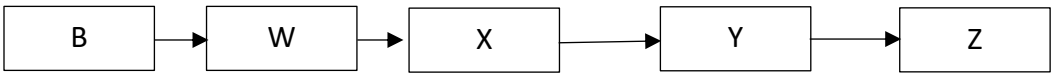
Nous allons expliquer ci-dessous à travers un exemple l'exécution d'une expression combinatoire qui contient un combinateur de distance, nous prenons l'exemple suivant : W_2XYZ.

$W_2XYZ \rightarrow BWXYZ \rightarrow W(XY)Z \rightarrow (XY)ZZ.$

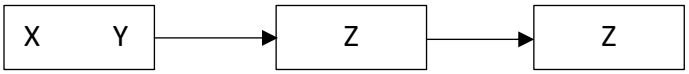
Dans ce cas notre entrée sera une liste qui contient quatre éléments, elle est représentée comme suit:



En appliquant la règle de distance nous allons avoir une nouvelle forme de notre liste d'entrée qui sera comme suit :



Maintenant nous pouvons commencer à exécuter notre expression, en commençant toujours par le premier élément le plus à gauche (le combinateur B)



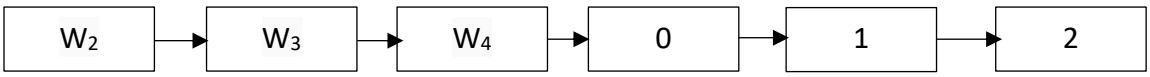
8. Capacite générative :

Cette fonction est responsable de calculer la capacité générative, la façon dont nous avons implémenté cette fonction est très simple :

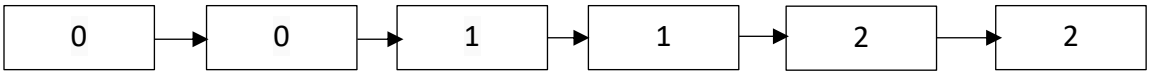
Cette fonction prend deux entrées : M et N, le M sera le nombre d'arguments voulus qui seront utilisés dans l'expression combinatoire, tandis que le N sera le nombre de duplications voulues pour chacun des arguments.

Par exemple : M=| {0, 1, 2} | cela veut dire : M= 3 et nous choisissons N=2.

La sortie de cette fonction sera une expression combinatoire qui sera représentée à son tour par une liste d'éléments.



L’exécution de cette expression combinatoire nous permettra d’obtenir un résultat final représenté sous forme d’une liste aussi :



9. Interface graphique :

Sur notre interface principale, nous retrouvons deux parties distinctes : la première permet d'effectuer toutes les opérations combinatoires possibles, tandis que la deuxième nous permettra de gérer la capacite générative dont nous avons parlé précédemment.

La première partie de notre interface est présentée ci- dessous :

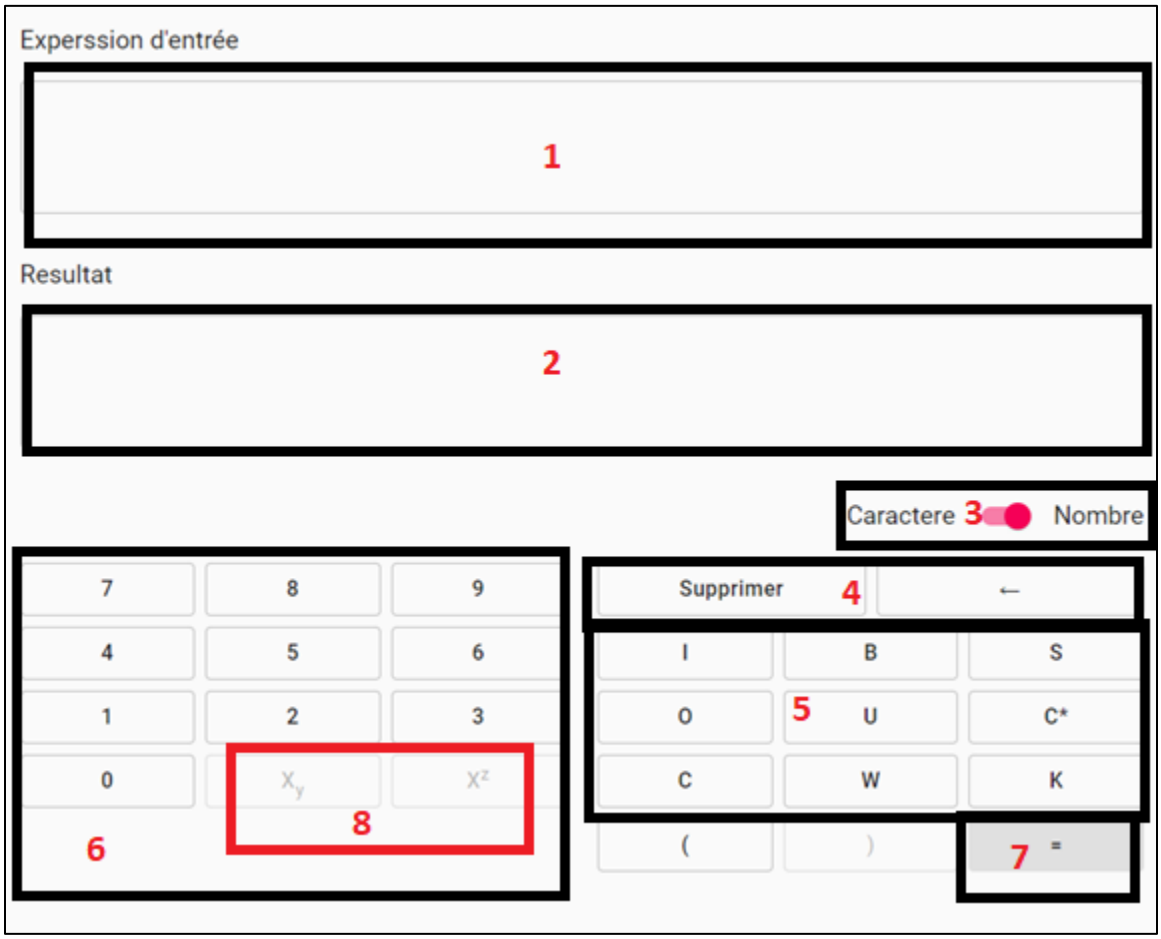


Figure 11 : La première partie de notre interface graphique

Dans la figure 11 nous avons présenté la première partie de notre interface qui nous permettra de faire entrer une quelconque expression combinatoire et de l'exécuter. Pour mieux expliquer notre première interface, la figure 11 a été divisée en 8 cadres dont l'utilité de chacun sera expliquée par la suite.

Le cadre 1 : dans ce cadre nous entrons notre expression combinatoire d'entrée.

Le cadre 2 : c'est ici que notre résultat sera affiché.

Le cadre 3 : il a un rapport direct avec le cadre 6; il nous permettra de choisir le type d'arguments que nous voulons faire entrer (de type caractère ou bien de type entiers).

Le cadre 4 : contient deux boutons; un pour la suppression complète de notre expression d'entrée et un autre pour une suppression par élément.

Le cadre 5 : contient neufs boutons dédiés aux différents combinateurs que nous avons présentés auparavant.

La cadre 6 : contient 10 boutons que nous allons utiliser pour entrer nos arguments.

Le cadre 7 : contient le bouton d'exécution (=) qui nous permettra d'exécuter notre expression.

Le cadre 8 : contient deux boutons; le premier permet l'utilisation de la distance de combinateurs, et le deuxième permet l'utilisation de la puissance.

La deuxième partie de notre interface est présentée dans la figure ci-dessous :

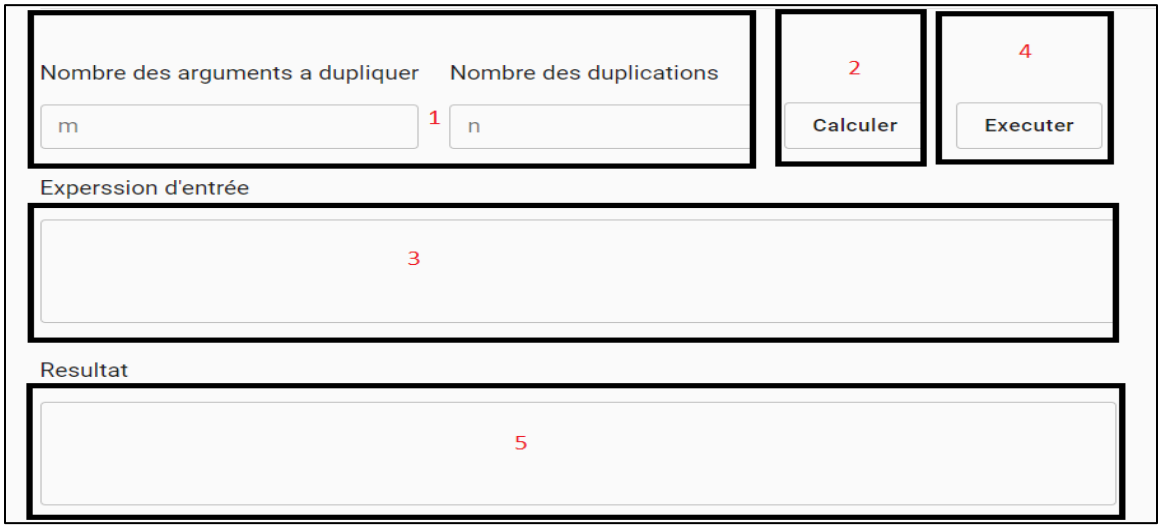


Figure 12: La deuxième partie de notre interface dédiée à la capacite générative.

Pour mieux expliquer notre deuxième interface, la figure 12 a été divisée en 5 cadres, nous expliquons ci-après l'utilité de chacun.

- Le cadre 1 : c'est là où nous entrons le nombre d'arguments « m » et le nombre de duplications voulues « n ».
- Le cadre 2 : contient un bouton qui permet d'exécuter l'algorithme de la capacité générative.
- Le cadre 3 : c'est là où nous allons recevoir l'expression combinatoire calculée.
- Le cadre 4 : contient un bouton qui nous permet d'exécuter l'expression combinatoire obtenue.
- Le cadre 5 : à son tour nous affiche le résultat d'exécution de l'expression combinatoire obtenue (les arguments dupliqués (n fois)).

10.Résultats :

Nous allons présenter dans le tableau qui suit une série de résultats pour une expression combinatoire de trois arguments (0 1 2) pour n allant de 2 à 11.

N	**Expressions et résultats**
2	$W_2W_3W_4012 \rightarrow 001122$
3	$W_5W_6W_7W_2W_4W_6012 \rightarrow 000111222$
4	$W_8W_9W_{10}W_5W_7W_9W_2W_5W_8012 \rightarrow 000011112222$
5	$W_{11}W_{12}W_{13}W_8W_{10}W_{12}W_5W_8W_{11}W_2W_6W_{10}012 \rightarrow 000001111122222$
6	$W_{14}W_{15}W_{16}W_{11}W_{13}W_{15}W_8W_{11}W_{14}W_5W_9W_{13}W_2W_7W_{12}012 \rightarrow 000000111111222222$
7	$W_{17}W_{18}W_{19}W_{14}W_{16}W_{18}W_{11}W_{14}W_{17}W_8W_{12}W_{16}W_5W_{10}W_{15}W_2W_8W_{14}012 \rightarrow 00000001111111222222222$
8	$W_{20}W_{21}W_{22}W_{17}W_{19}W_{21}W_{14}W_{17}W_{20}W_{11}W_{15}W_{19}W_8W_{13}W_{18}W_5W_{11}W_{17}W_2W_9W_{16}012 \rightarrow 000000001111111122222222$
9	$W_{23}W_{24}W_{25}W_{20}W_{22}W_{24}W_{17}W_{20}W_{23}W_{14}W_{18}W_{22}W_{11}W_{16}W_{21}W_8W_{14}W_{20}W_5W_{12}W_{19}W_2W_{10}W_{18}012 \rightarrow 000000000111111111222222222$
10	$W_{26}W_{27}W_{28}W_{23}W_{25}W_{27}W_{20}W_{23}W_{26}W_{17}W_{21}W_{25}W_{14}W_{19}W_{24}W_{11}W_{17}W_{23}W_8W_{15}W_{22}W_5W_{13}W_{21}W_2W_{11}W_{20}012 \rightarrow$ 0000000000111111111122222222222
11	$W_{29}W_{30}W_{31}W_{26}W_{28}W_{30}W_{23}W_{26}W_{29}W_{20}W_{24}W_{28}W_{17}W_{22}W_{27}W_{14}W_{20}W_{26}W_{11}W_{18}W_{25}W_8W_{16}W_{24}W_5W_{14}W_{23}W_2W_{12}W_{22}012$ $\rightarrow$0000000000011111111111222222222222

Tableau 2 : série de résultats

Nos résultats prouvent que nous avons réussi toutes nos chaines de types $A^nB^nC^n$. Vous trouverez en annexe une série d'exemples : pour une expression à trois arguments 0,1,2 et pour **n** allant de 2 jusqu'à 100.

Chapitre 7 : Conclusion

Dans le cadre de ce travail, nous nous sommes intéressés aux systèmes applicatifs et à la logique combinatoire dans le domaine du traitement du langage. Notre objectif consistait à démontrer que la logique combinatoire est une logique sensible au contexte et quelle est capable d'avoir une capacité générative supérieure à la plupart des grammaires qui existent.

Il a fallu dans un premier temps présenter les systèmes applicatifs, la logique combinatoire, ainsi que l'intérêt de leur association. Tout cela a été suivi d'une étude de l'état de l'art sur la grammaire catégorielle et d'une présentation de la hiérarchie de Chomsky ainsi que des différents types de langages et de grammaires.

Au moyen de cette présentation il convenait alors d'analyser des exemples d'expressions combinatoires à distance. Cette analyse nous a permis de déduire notre algorithme, qui à son tour nous a permis de calculer la capacité générative du modèle combinatoire, et ainsi prouver que la logique combinatoire est une logique sensible au contexte.

Ce travail de mémoire se voulait principalement théorique, mais dans cette perspective, il serait pertinent de procéder à une démonstration mathématique qui devrait faire l'objet de futurs travaux.

Références :

[1]. Sammoud Mohamed Sadok. (2010). Applications linguistiques multilingues: apport des grammaires catégorielles et de la logique combinatoire. In: Université du Québec à Trois-Rivières.

[2]. www.ionos.fr. Available from: https://www.ionos.fr/digitalguide/sites-internet/developpement-web/langage-haskell/

[3]. Gayton Frédéric. (2004). Vers une nouvelle ingénierie de l'information. In: Université du Québec à Trois-Rivières.

[4]. Joly Adam. (2009). Du document textuel à la carte sémantique fonctionnelle. In: Université du Québec à Trois-Rivières.

[5]. Hindley Roger and Jonathan Seldin. (1986). Introduction to Combinators and λ-Calculus: Cambridge University Press.

[6]. Robinson John Alan. (1979). Logic: Form and Function: The Mechanization of Deductive Reasoning.

[7]. Bellot Patrick, Jean-Philippe Cottin, and Jean-François Monin. (1995). Développement et validation de logiciels. Méthodes formelles. Techniques de l'Ingénieur, no H. **2**: p. 550.

[8]. Biskri Ismaïl and Desclés Jean-Pierre. (1996). Du phénotype au génotype: la grammaire catégorielle combinatoire applicative. TALN (Traitement Automatique des Langues Naturelles): p. 87-96.

[9]. Schönfinkel Moses. (1924). Über die Bausteine der mathematischen Logik. Mathematische annalen. **92**(3): p. 305-316.

[10]. Curry Haskell Brooks. (1958). Combinatory logic. Vol. 1.: North-Holland Amsterdam.

[11]. Hassani Mohamed. (2016). Vers un modèle formel applicatif pour une nouvelle ingénierie de la langue et de l'information. In: Université du Québec à Trois-Rivières.

[12]. Ginisti Jean-Pierre. (1988). Présentation de la logique combinatoire en vue de ses applications. Mathématiques et Sciences humaines. **103**: p. 45-66.

[13]. Desclés Jean-Pierre and Biskri Ismail. (1995). Logique combinatoire et linguistique: grammaire catégorielle combinatoire applicative. Mathématiques et sciences humaines. **132**: p. 39-68.
[14]. Hindley Roger, Lercher Bruce, and Seldin Jonathan. (1972). Introduction to combinatory logic. Vol. 7: CUP Archive.
[15]. Anastacio Marie. (2015). Construction de chaînes de traitement pour l'analyse de texte: une approche combinatoire typée. In: Université du Québec à Trois-Rivières.
[16]. Hughes John (1989). Why Functional Programming Matters. In The Computer Journal, Vol. 32, No. 2, pp. 98 - 107
[17]. Types de grammaires. (2021). Available from: https://complex-systems-ai.com/theorie-des-langages/types-de-grammaires/
[18]. Retoré Christian. (2000). Systèmes déductifs et traitement des langues: un panorama des grammaires catégorielles. In: INRIA.
[19]. Lecomte Alain . (2021). Grammaire et Théorie de la Preuve : une Introduction. In Traitement Automatique des Langues, volume 37, numéro 2, pp. 1—38.
[20]. Segond Frédérique and Chanod Jean-Pierre. (1988). Grammaire applicative: traitement informatique de la composante morpho-syntaxique. Mathématiques et Sciences humaines. **103**: p. 23-43.
[21]. Desclés Jean-Pierre. (2003). La Grammaire Applicative et Cognitive construit-elle des représentations universelles? Linx. Revue des linguistes de l'université Paris X Nanterre, (48): p. 139-160.
[22]. Biskri Ismaïl. (2009). Applications linguistiques multilingues destinées au WEB: Apport des Grammaires Catégorielles. Jean Pierre Des clés, Florence Le Priol.(Eds.), Dans Annotations automatiques et recherche d'informations, Presses Hermès, Paris.
[23]. Biskri Ismaïl, Desclés Jean Pierre, and Jouis Christophe. (1997). La Grammaire Catégorielle Combinatoire Applicative appliquée au français.

Lexicomatique et Dictionnairiques. Actes des Vème Journée Scientifique du réseau thématique LTT" Lexicologie Terminologie, Traduction", Tunis, 1997: p. 25-27.

[24]. Turing Alan, Girard Jean-Yves (1995). La machine de Turing. Éditions du Seuil, France.

[25]. Chomsky Noam. (1956). Three models for the description of language;. In *IRE Transactions on Information Theory*, n^{o} 2, 1956, p. 113–124

Annexe:

N	Expressions et résultats
2	$W_2W_3W_4012 \rightarrow 001122$
3	$W_5W_6W_7W_2W_4W_6012 \rightarrow 000111222$
4	$W_8W_9W_{10}W_5W_7W_9W_2W_5W_8012 \rightarrow 000011112222$
5	$W_{11}W_{12}W_{13}W_8W_{10}W_{12}W_5W_8W_{11}W_2W_6W_{10}012 \rightarrow 000001111122222$
6	$W_{14}W_{15}W_{16}W_{11}W_{13}W_{15}W_8W_{11}W_{14}W_5W_9W_{13}W_2W_7W_{12}012 \rightarrow 000000111111222222$
7	$W_{17}W_{18}W_{19}W_{14}W_{16}W_{18}W_{11}W_{14}W_{17}W_8W_{12}W_{16}W_5W_{10}W_{15}W_2W_8W_{14}012 \rightarrow 00000001111111222222$
8	$W_{20}W_{21}W_{22}W_{17}W_{19}W_{21}W_{14}W_{17}W_{20}W_{11}W_{15}W_{19}W_8W_{13}W_{18}W_5W_{11}W_{17}W_2W_9W_{16}012 \rightarrow 00000000111111111122222222$
9	$W_{23}W_{24}W_{25}W_{20}W_{22}W_{24}W_{17}W_{20}W_{23}W_{14}W_{18}W_{22}W_{11}W_{16}W_{21}W_8W_{14}W_{20}W_5W_{12}W_{19}W_2W_{10}W_{18}012 \rightarrow 000000000111111111222222222$
10	$W_{26}W_{27}W_{28}W_{23}W_{25}W_{27}W_{20}W_{23}W_{26}W_{17}W_{21}W_{25}W_{14}W_{19}W_{24}W_{11}W_{17}W_{23}W_8W_{15}W_{22}W_5W_{13}W_{21}W_2W_{11}W_{20}012 \rightarrow 0000000000111111111122222222222$
11	$W_{29}W_{30}W_{31}W_{26}W_{28}W_{30}W_{23}W_{26}W_{29}W_{20}W_{24}W_{28}W_{17}W_{22}W_{27}W_{14}W_{20}W_{26}W_{11}W_{18}W_{25}W_8W_{16}W_{24}W_5W_{14}W_{23}W_2W_{12}W_{22}012 \rightarrow 00000000000111111111112222222222$
12	$W_{32}W_{33}W_{34}W_{29}W_{31}W_{33}W_{26}W_{29}W_{32}W_{23}W_{27}W_{31}W_{20}W_{25}W_{30}W_{17}W_{23}W_{29}W_{14}W_{21}W_{28}W_{11}W_{19}W_{27}W_8W_{17}W_{26}W_5W_{15}W_{25}W_2W_{13}W_{24}012 \rightarrow 000000000000111111111112222222222$
13	$W_{35}W_{36}W_{37}W_{32}W_{34}W_{36}W_{29}W_{32}W_{35}W_{26}W_{30}W_{34}W_{23}W_{28}W_{33}W_{20}W_{26}W_{32}W_{17}W_{24}W_{31}W_{14}W_{22}W_{30}W_{11}W_{20}W_{29}W_8W_{18}W_{28}W_5W_{16}W_{27}W_2W_{14}W_{26}012 \rightarrow 0000000000000111111111111122222222222$
14	$W_{38}W_{39}W_{40}W_{35}W_{37}W_{39}W_{32}W_{35}W_{38}W_{29}W_{33}W_{37}W_{26}W_{31}W_{36}W_{23}W_{29}W_{35}W_{20}W_{27}W_{34}W_{17}W_{25}W_{33}W_{14}W_{23}W_{32}W_{11}W_{21}W_{31}W_8W_{19}W_{30}W_5W_{17}W_{29}W_2W_{15}W_{28}012 \rightarrow 00000000000000111111111111122222222222222$
15	$W_{41}W_{42}W_{43}W_{38}W_{40}W_{42}W_{35}W_{38}W_{41}W_{32}W_{36}W_{40}W_{29}W_{34}W_{39}W_{26}W_{32}W_{38}W_{23}W_{30}W_{37}W_{20}W_{28}W_{36}W_{17}W_{26}W_{35}W_{14}W_{24}W_{34}W_{11}W_{22}W_{33}W_8W_{20}W_{32}W_5W_{18}W_{31}W_2W_{16}W_{30}012 \rightarrow 000000000000000111111111111111122222222222222$
16	$W_{44}W_{45}W_{46}W_{41}W_{43}W_{45}W_{38}W_{41}W_{44}W_{35}W_{39}W_{43}W_{32}W_{37}W_{42}W_{29}W_{35}W_{41}W_{26}W_{33}W_{40}W_{23}W_{31}W_{39}W_{20}W_{29}W_{38}W_{17}W_{27}W_{37}W_{14}W_{25}W_{36}W_{11}W_{23}W_{35}W_8W_{21}W_{34}W_5W_{19}W_{33}W_2W_{17}W_{32}012 \rightarrow 000000000000000011111111111111111222222222222222$
17	$W_{47}W_{48}W_{49}W_{44}W_{46}W_{48}W_{41}W_{44}W_{47}W_{38}W_{42}W_{46}W_{35}W_{40}W_{45}W_{32}W_{38}W_{44}W_{29}W_{36}W_{43}W_{26}W_{34}W_{42}W_{23}W_{32}W_{41}W_{20}W_{30}W_{40}W_{17}W_{28}W_{39}W_{14}W_{26}W_{38}W_{11}W_{24}W_{37}W_8W_{22}W_{36}W_5W_{20}W_{35}W_2W_{18}W_{34}012 \rightarrow 000000000000000000111111111111111112222222222222222$
18	$W_{50}W_{51}W_{52}W_{47}W_{49}W_{51}W_{44}W_{47}W_{50}W_{41}W_{45}W_{49}W_{38}W_{43}W_{48}W_{35}W_{41}W_{47}W_{32}W_{39}W_{46}W_{29}W_{37}W_{45}W_{26}W_{35}W_{44}W_{23}W_{33}W_{43}W_{20}W_{31}W_{42}W_{17}W_{29}W_{41}W_{14}W_{27}W_{40}W_{11}W_{25}W_{39}W_8W_{23}W_{38}W_5W_{21}W_{37}W_2W_{19}W_{36}012 \rightarrow 0000000000000000001111111111111111112222222222222222$
19	$W_{53}W_{54}W_{55}W_{50}W_{52}W_{54}W_{47}W_{50}W_{53}W_{44}W_{48}W_{52}W_{41}W_{46}W_{51}W_{38}W_{44}W_{50}W_{35}W_{42}W_{49}W_{32}W_{40}W_{48}W_{29}W_{38}W_{47}W_{26}W_{36}W_{46}W_{23}W_{34}W_{45}W_{20}W_{32}W_{44}W_{17}W_{30}W_{43}W_{14}W_{28}W_{42}W_{11}W_{26}W_{41}W_8W_{24}W_{40}W_5W_{22}W_{39}W_2W_{20}W_{38}012 \rightarrow 0000000000000000000011111111111111111112222222222222222222$
20	$W_{56}W_{57}W_{58}W_{53}W_{55}W_{57}W_{50}W_{53}W_{56}W_{47}W_{51}W_{55}W_{44}W_{49}W_{54}W_{41}W_{47}W_{53}W_{38}W_{45}W_{52}W_{35}W_{43}W_{51}W_{32}W_{41}W_{50}W_{29}W_{39}W_{49}W_{26}W_{37}W_{48}W_{23}W_{35}W_{47}W_{20}W_{33}W_{46}W_{17}W_{31}W_{45}W_{14}W_{29}W_{44}W_{11}W_{27}W_{43}W_8W_{25}W_{42}W_5W_{23}W_{41}W_2W_{21}W_{40}012 \rightarrow 0000000000000000000001111111111111111111122222222222222222222$
21	$W_{59}W_{60}W_{61}W_{56}W_{58}W_{60}W_{53}W_{56}W_{59}W_{50}W_{54}W_{58}W_{47}W_{52}W_{57}W_{44}W_{50}W_{56}W_{41}W_{48}W_{55}W_{38}W_{46}W_{54}W_{35}W_{44}W_{53}W_{32}W_{42}W_{52}W_{29}W_{40}W_{51}W_{26}W_{38}W_{50}W_{23}W_{36}W_{49}W_{20}W_{34}W_{48}W_{17}W_{32}W_{47}W_{14}W_{30}W_{46}W_{11}W_{28}W_{45}W_8W_{26}W_{44}W_5W_{24}W_{43}W_2W_{22}W_{42}012 \rightarrow 0000000000000000000000111111111111111111111222222222222222222222$
22	$W_{62}W_{63}W_{64}W_{59}W_{61}W_{63}W_{56}W_{59}W_{62}W_{53}W_{57}W_{61}W_{50}W_{55}W_{60}W_{47}W_{53}W_{59}W_{44}W_{51}W_{58}W_{41}W_{49}W_{57}W_{38}W_{47}W_{56}W_{35}W_{45}W_{55}W_{32}W_{43}W_{54}W_{29}W_{41}W_{53}W_{26}W_{39}W_{52}W_{23}W_{37}W_{51}W_{20}W_{35}W_{50}W_{17}W_{33}W_{49}W_{14}W_{31}W_{48}W_{11}W_{29}W_{47}W_8W_{27}W_{46}W_5W_{25}W_{45}W_2W_{23}W_{44}012 \rightarrow 0000000000000000000000011111111111111111111222222222222222222222$

23	$W_{65}W_{66}W_{67}W_{62}W_{64}W_{66}W_{59}W_{62}W_{65}W_{56}W_{60}W_{64}W_{53}W_{58}W_{63}W_{50}W_{56}W_{62}W_{47}W_{54}W_{61}W_{44}W_{52}W_{60}W_{41}$ $W_{50}W_{59}W_{38}W_{48}W_{58}W_{35}W_{46}W_{57}W_{32}W_{44}W_{56}W_{29}W_{42}W_{55}W_{26}W_{40}W_{54}W_{23}W_{38}W_{53}W_{20}W_{36}W_{52}W_{17}W_{34}$ $W_{51}W_{14}W_{32}W_{50}W_{11}W_{30}W_{49}W_{8}W_{28}W_{48}W_{5}W_{26}W_{47}W_{2}W_{24}W_{46}012\rightarrow$ 000000000000000000000000111111111111111111111111222222222222222222222222
24	$W_{68}W_{69}W_{70}W_{65}W_{67}W_{69}W_{62}W_{65}W_{68}W_{59}W_{63}W_{67}W_{56}W_{61}W_{66}W_{53}W_{59}W_{65}W_{50}W_{57}W_{64}W_{47}W_{55}W_{63}W_{44}$ $W_{53}W_{62}W_{41}W_{51}W_{61}W_{38}W_{49}W_{60}W_{35}W_{47}W_{59}W_{32}W_{45}W_{58}W_{29}W_{43}W_{57}W_{26}W_{41}W_{56}W_{23}W_{39}W_{55}W_{20}W_{37}$ $W_{54}W_{17}W_{35}W_{53}W_{14}W_{33}W_{52}W_{11}W_{31}W_{51}W_{8}W_{29}W_{50}W_{5}W_{27}W_{49}W_{2}W_{25}W_{48}012\rightarrow$ 000000000000000000000000000000111111111111111111111111111111222222222222222222222222
25	$W_{71}W_{72}W_{73}W_{68}W_{70}W_{72}W_{65}W_{68}W_{71}W_{62}W_{66}W_{70}W_{59}W_{64}W_{69}W_{56}W_{62}W_{68}W_{53}W_{60}W_{67}W_{50}W_{58}W_{66}W_{47}$ $W_{56}W_{65}W_{44}W_{54}W_{64}W_{41}W_{52}W_{63}W_{38}W_{50}W_{62}W_{35}W_{48}W_{61}W_{32}W_{46}W_{60}W_{29}W_{44}W_{59}W_{26}W_{42}W_{58}W_{23}W_{40}$ $W_{57}W_{20}W_{38}W_{56}W_{17}W_{36}W_{55}W_{14}W_{34}W_{54}W_{11}W_{32}W_{53}W_{8}W_{30}W_{52}W_{5}W_{28}W_{51}W_{2}W_{26}W_{50}012\rightarrow$ 000000000000000000000000000000111111111111111111111111111111222222222222222222222222
26	$W_{74}W_{75}W_{76}W_{71}W_{73}W_{75}W_{68}W_{71}W_{74}W_{65}W_{69}W_{73}W_{62}W_{67}W_{72}W_{59}W_{65}W_{71}W_{56}W_{63}W_{70}W_{53}W_{61}W_{69}W_{50}$ $W_{59}W_{68}W_{47}W_{57}W_{67}W_{44}W_{55}W_{66}W_{41}W_{53}W_{65}W_{38}W_{51}W_{64}W_{35}W_{49}W_{63}W_{32}W_{47}W_{62}W_{29}W_{45}W_{61}W_{26}W_{43}$ $W_{60}W_{23}W_{41}W_{59}W_{20}W_{39}W_{58}W_{17}W_{37}W_{57}W_{14}W_{35}W_{56}W_{11}W_{33}W_{55}W_{8}W_{31}W_{54}W_{5}W_{29}W_{53}W_{2}W_{27}W_{52}012$ $\rightarrow$ 000000000000000000000000000000111111111111111111111111111111222222222222222222222222
27	$W_{77}W_{78}W_{79}W_{74}W_{76}W_{78}W_{71}W_{74}W_{77}W_{68}W_{72}W_{76}W_{65}W_{70}W_{75}W_{62}W_{68}W_{74}W_{59}W_{66}W_{73}W_{56}W_{64}W_{72}W_{53}$ $W_{62}W_{71}W_{50}W_{60}W_{70}W_{47}W_{58}W_{69}W_{44}W_{56}W_{68}W_{41}W_{54}W_{67}W_{38}W_{52}W_{66}W_{35}W_{50}W_{65}W_{32}W_{48}W_{64}W_{29}W_{46}$ $W_{63}W_{26}W_{44}W_{62}W_{23}W_{42}W_{61}W_{20}W_{40}W_{60}W_{17}W_{38}W_{59}W_{14}W_{36}W_{58}W_{11}W_{34}W_{57}W_{8}W_{32}W_{56}W_{5}W_{30}W_{55}W_{2}$ $W_{28}W_{54}012\rightarrow$ 000000000000000000000000000000111111111111111111111111111111222222222222222222222222222222222222
28	$W_{80}W_{81}W_{82}W_{77}W_{79}W_{81}W_{74}W_{77}W_{80}W_{71}W_{75}W_{79}W_{68}W_{73}W_{78}W_{65}W_{71}W_{77}W_{62}W_{69}W_{76}W_{59}W_{67}W_{75}W_{56}$ $W_{65}W_{74}W_{53}W_{63}W_{73}W_{50}W_{61}W_{72}W_{47}W_{59}W_{71}W_{44}W_{57}W_{70}W_{41}W_{55}W_{69}W_{38}W_{53}W_{68}W_{35}W_{51}W_{67}W_{32}W_{49}$ $W_{66}W_{29}W_{47}W_{65}W_{26}W_{45}W_{64}W_{23}W_{43}W_{63}W_{20}W_{41}W_{62}W_{17}W_{39}W_{61}W_{14}W_{37}W_{60}W_{11}W_{35}W_{59}W_{8}W_{33}W_{58}W$ $_{5}W_{31}W_{57}W_{2}W_{29}W_{56}012\rightarrow$ 000000000000000000000000000000111111111111111111111111111111222222222222222222222222222222222222
29	$W_{83}W_{84}W_{85}W_{80}W_{82}W_{84}W_{77}W_{80}W_{83}W_{74}W_{78}W_{82}W_{71}W_{76}W_{81}W_{68}W_{74}W_{80}W_{65}W_{72}W_{79}W_{62}W_{70}W_{78}W_{59}$ $W_{68}W_{77}W_{56}W_{66}W_{76}W_{53}W_{64}W_{75}W_{50}W_{62}W_{74}W_{47}W_{60}W_{73}W_{44}W_{58}W_{72}W_{41}W_{56}W_{71}W_{38}W_{54}W_{70}W_{35}W_{52}$ $W_{69}W_{32}W_{50}W_{68}W_{29}W_{48}W_{67}W_{26}W_{46}W_{66}W_{23}W_{44}W_{65}W_{20}W_{42}W_{64}W_{17}W_{40}W_{63}W_{14}W_{38}W_{62}W_{11}W_{36}W_{61}$ $W_{8}W_{34}W_{60}W_{5}W_{32}W_{59}W_{2}W_{30}W_{58}012\rightarrow$ 000000000000000000000000000000111111111111111111111111111111222222222222222222222222222222222222
30	$W_{86}W_{87}W_{88}W_{83}W_{85}W_{87}W_{80}W_{83}W_{86}W_{77}W_{81}W_{85}W_{74}W_{79}W_{84}W_{71}W_{77}W_{83}W_{68}W_{75}W_{82}W_{65}W_{73}W_{81}W_{62}$ $W_{71}W_{80}W_{59}W_{69}W_{79}W_{56}W_{67}W_{78}W_{53}W_{65}W_{77}W_{50}W_{63}W_{76}W_{47}W_{61}W_{75}W_{44}W_{59}W_{74}W_{41}W_{57}W_{73}W_{38}W_{55}$ $W_{72}W_{35}W_{53}W_{71}W_{32}W_{51}W_{70}W_{29}W_{49}W_{69}W_{26}W_{47}W_{68}W_{23}W_{45}W_{67}W_{20}W_{43}W_{66}W_{17}W_{41}W_{65}W_{14}W_{39}W_{64}$ $W_{11}W_{37}W_{63}W_{8}W_{35}W_{62}W_{5}W_{33}W_{61}W_{2}W_{31}W_{60}012\rightarrow$ 000000000000000000000000000000111111111111111111111111111111222222222222222222222222222222222222
31	$W_{89}W_{90}W_{91}W_{86}W_{88}W_{90}W_{83}W_{86}W_{89}W_{80}W_{84}W_{88}W_{77}W_{82}W_{87}W_{74}W_{80}W_{86}W_{71}W_{78}W_{85}W_{68}W_{76}W_{84}W_{65}$ $W_{74}W_{83}W_{62}W_{72}W_{82}W_{59}W_{70}W_{81}W_{56}W_{68}W_{80}W_{53}W_{66}W_{79}W_{50}W_{64}W_{78}W_{47}W_{62}W_{77}W_{44}W_{60}W_{76}W_{41}W_{58}$ $W_{75}W_{38}W_{56}W_{74}W_{35}W_{54}W_{73}W_{32}W_{52}W_{72}W_{29}W_{50}W_{71}W_{26}W_{48}W_{70}W_{23}W_{46}W_{69}W_{20}W_{44}W_{68}W_{17}W_{42}W_{67}$ $W_{14}W_{40}W_{66}W_{11}W_{38}W_{65}W_{8}W_{36}W_{64}W_{5}W_{34}W_{63}W_{2}W_{32}W_{62}012\rightarrow$ 000000000000000000000000000000011111111111111111111111111111222222222222222222222222222222222222
32	$W_{92}W_{93}W_{94}W_{89}W_{91}W_{93}W_{86}W_{89}W_{92}W_{83}W_{87}W_{91}W_{80}W_{85}W_{90}W_{77}W_{83}W_{89}W_{74}W_{81}W_{88}W_{71}W_{79}W_{87}W_{68}$ $W_{77}W_{86}W_{65}W_{75}W_{85}W_{62}W_{73}W_{84}W_{59}W_{71}W_{83}W_{56}W_{69}W_{82}W_{53}W_{67}W_{81}W_{50}W_{65}W_{80}W_{47}W_{63}W_{79}W_{44}W_{61}$ $W_{78}W_{41}W_{59}W_{77}W_{38}W_{57}W_{76}W_{35}W_{55}W_{75}W_{32}W_{53}W_{74}W_{29}W_{51}W_{73}W_{26}W_{49}W_{72}W_{23}W_{47}W_{71}W_{20}W_{45}W_{70}$ $W_{17}W_{43}W_{69}W_{14}W_{41}W_{68}W_{11}W_{39}W_{67}W_{8}W_{37}W_{66}W_{5}W_{35}W_{65}W_{2}W_{33}W_{64}012\rightarrow$ 000000000000000000000000000000011111111111111111111111111111222222222222222222222222222222222222
33	$W_{95}W_{96}W_{97}W_{92}W_{94}W_{96}W_{89}W_{92}W_{95}W_{86}W_{90}W_{94}W_{83}W_{88}W_{93}W_{80}W_{86}W_{92}W_{77}W_{84}W_{91}W_{74}W_{82}W_{90}W_{71}$ $W_{80}W_{89}W_{68}W_{78}W_{88}W_{65}W_{76}W_{87}W_{62}W_{74}W_{86}W_{59}W_{72}W_{85}W_{56}W_{70}W_{84}W_{53}W_{68}W_{83}W_{50}W_{66}W_{82}W_{47}W_{64}$ $W_{81}W_{44}W_{62}W_{80}W_{41}W_{60}W_{79}W_{38}W_{58}W_{78}W_{35}W_{56}W_{77}W_{32}W_{54}W_{76}W_{29}W_{52}W_{75}W_{26}W_{50}W_{74}W_{23}W_{48}W_{73}$ $W_{20}W_{46}W_{72}W_{17}W_{44}W_{71}W_{14}W_{42}W_{70}W_{11}W_{40}W_{69}W_{8}W_{38}W_{68}W_{5}W_{36}W_{67}W_{2}W_{34}W_{66}012\rightarrow$ 000000000000000000000000000000111111111111111111111111111111222222222222222222222222222222222222
34	$W_{98}W_{99}W_{100}W_{95}W_{97}W_{99}W_{92}W_{95}W_{98}W_{89}W_{93}W_{97}W_{86}W_{91}W_{96}W_{83}W_{89}W_{95}W_{80}W_{87}W_{94}W_{77}W_{85}W_{93}W_{74}$ $W_{83}W_{92}W_{71}W_{81}W_{91}W_{68}W_{79}W_{90}W_{65}W_{77}W_{89}W_{62}W_{75}W_{88}W_{59}W_{73}W_{87}W_{56}W_{71}W_{86}W_{53}W_{69}W_{85}W_{50}W_{67}$ $W_{84}W_{47}W_{65}W_{83}W_{44}W_{63}W_{82}W_{41}W_{61}W_{81}W_{38}W_{59}W_{80}W_{35}W_{57}W_{79}W_{32}W_{55}W_{78}W_{29}W_{53}W_{77}W_{26}W_{51}W_{76}$

	$W_{23}W_{49}W_{75}W_{20}W_{47}W_{74}W_{17}W_{45}W_{73}W_{14}W_{43}W_{72}W_{11}W_{41}W_{71}W_{8}W_{39}W_{70}W_{5}W_{37}W_{69}W_{2}W_{35}W_{68}012\rightarrow0$ 000000000000000000000000000000001111111111111111111111111111111122222222222222 2222222222222222222
35	$W_{101}W_{102}W_{103}W_{98}W_{100}W_{102}W_{95}W_{98}W_{101}W_{92}W_{96}W_{100}W_{89}W_{94}W_{99}W_{86}W_{92}W_{98}W_{83}W_{90}W_{97}W_{80}W_{88}W_{96}$ $W_{77}W_{86}W_{95}W_{74}W_{84}W_{94}W_{71}W_{82}W_{93}W_{68}W_{80}W_{92}W_{65}W_{78}W_{91}W_{62}W_{76}W_{90}W_{59}W_{74}W_{89}W_{56}W_{72}W_{88}W_{53}$ $W_{70}W_{87}W_{50}W_{68}W_{86}W_{47}W_{66}W_{85}W_{44}W_{64}W_{84}W_{41}W_{62}W_{83}W_{38}W_{60}W_{82}W_{35}W_{58}W_{81}W_{32}W_{56}W_{80}W_{29}W_{54}$ $W_{79}W_{26}W_{52}W_{78}W_{23}W_{50}W_{77}W_{20}W_{48}W_{76}W_{17}W_{46}W_{75}W_{14}W_{44}W_{74}W_{11}W_{42}W_{73}W_{8}W_{40}W_{72}W_{5}W_{38}W_{71}W_{2}$ $W_{36}W_{70}012\rightarrow$ 00000000000000000000000000000000001111111111111111111111111111111111 12222222222222222222222222222222222
36	$W_{104}W_{105}W_{106}W_{101}W_{103}W_{105}W_{98}W_{101}W_{104}W_{95}W_{99}W_{103}W_{92}W_{97}W_{102}W_{89}W_{95}W_{101}W_{86}W_{93}W_{100}W_{83}W_{91}$ $W_{99}W_{80}W_{89}W_{98}W_{77}W_{87}W_{97}W_{74}W_{85}W_{96}W_{71}W_{83}W_{95}W_{68}W_{81}W_{94}W_{65}W_{79}W_{93}W_{62}W_{77}W_{92}W_{59}W_{75}W_{91}$ $W_{56}W_{73}W_{90}W_{53}W_{71}W_{89}W_{50}W_{69}W_{88}W_{47}W_{67}W_{87}W_{44}W_{65}W_{86}W_{41}W_{63}W_{85}W_{38}W_{61}W_{84}W_{35}W_{59}W_{83}W_{32}$ $W_{57}W_{82}W_{29}W_{55}W_{81}W_{26}W_{53}W_{80}W_{23}W_{51}W_{79}W_{20}W_{49}W_{78}W_{17}W_{47}W_{77}W_{14}W_{45}W_{76}W_{11}W_{43}W_{75}W_{8}W_{41}W$ $_{74}W_{5}W_{39}W_{73}W_{2}W_{37}W_{72}012\rightarrow$ 00000000000000000000000000000000000001111111111111111111 11111111111111222222222222222222222222222222222222
37	$W_{107}W_{108}W_{109}W_{104}W_{106}W_{108}W_{101}W_{104}W_{107}W_{98}W_{102}W_{106}W_{95}W_{100}W_{105}W_{92}W_{98}W_{104}W_{89}W_{96}W_{103}W_{86}$ $W_{94}W_{102}W_{83}W_{92}W_{101}W_{80}W_{90}W_{100}W_{77}W_{88}W_{99}W_{74}W_{86}W_{98}W_{71}W_{84}W_{97}W_{68}W_{82}W_{96}W_{65}W_{80}W_{95}W_{62}W_{7}$ $_{8}W_{94}W_{59}W_{76}W_{93}W_{56}W_{74}W_{92}W_{53}W_{72}W_{91}W_{50}W_{70}W_{90}W_{47}W_{68}W_{89}W_{44}W_{66}W_{88}W_{41}W_{64}W_{87}W_{38}W_{62}W_{86}$ $W_{35}W_{60}W_{85}W_{32}W_{58}W_{84}W_{29}W_{56}W_{83}W_{26}W_{54}W_{82}W_{23}W_{52}W_{81}W_{20}W_{50}W_{80}W_{17}W_{48}W_{79}W_{14}W_{46}W_{78}W_{11}$ $W_{44}W_{77}W_{8}W_{42}W_{76}W_{5}W_{40}W_{75}W_{2}W_{38}W_{74}012\rightarrow$ 00000000000000000000000000000000000000111111 11111111111111111111111111111122222222222222222222222222222222222222
38	$W_{110}W_{111}W_{112}W_{107}W_{109}W_{111}W_{104}W_{107}W_{110}W_{101}W_{105}W_{109}W_{98}W_{103}W_{108}W_{95}W_{101}W_{107}W_{92}W_{99}W_{106}W_{89}$ $W_{97}W_{105}W_{86}W_{95}W_{104}W_{83}W_{93}W_{103}W_{80}W_{91}W_{102}W_{77}W_{89}W_{101}W_{74}W_{87}W_{100}W_{71}W_{85}W_{99}W_{68}W_{83}W_{98}W_{65}$ $W_{81}W_{97}W_{62}W_{79}W_{96}W_{59}W_{77}W_{95}W_{56}W_{75}W_{94}W_{53}W_{73}W_{93}W_{50}W_{71}W_{92}W_{47}W_{69}W_{91}W_{44}W_{67}W_{90}W_{41}W_{65}$ $W_{89}W_{38}W_{63}W_{88}W_{35}W_{61}W_{87}W_{32}W_{59}W_{86}W_{29}W_{57}W_{85}W_{26}W_{55}W_{84}W_{23}W_{53}W_{83}W_{20}W_{51}W_{82}W_{17}W_{49}W_{81}$ $W_{14}W_{47}W_{80}W_{11}W_{45}W_{79}W_{8}W_{43}W_{78}W_{5}W_{41}W_{77}W_{2}W_{39}W_{76}012\rightarrow$ 00000000000000000000000000000000 00000000111111111111111111111111111111111111122222222222222222222222222222222222 222
39	$W_{113}W_{114}W_{115}W_{110}W_{112}W_{114}W_{107}W_{110}W_{113}W_{104}W_{108}W_{112}W_{101}W_{106}W_{111}W_{98}W_{104}W_{110}W_{95}W_{102}W_{109}W$ $_{92}W_{100}W_{108}W_{89}W_{98}W_{107}W_{86}W_{96}W_{106}W_{83}W_{94}W_{105}W_{80}W_{92}W_{104}W_{77}W_{90}W_{103}W_{74}W_{88}W_{102}W_{71}W_{86}W_{101}$ $W_{68}W_{84}W_{100}W_{65}W_{82}W_{99}W_{62}W_{80}W_{98}W_{59}W_{78}W_{97}W_{56}W_{76}W_{96}W_{53}W_{74}W_{95}W_{50}W_{72}W_{94}W_{47}W_{70}W_{93}W_{44}$ $W_{68}W_{92}W_{41}W_{66}W_{91}W_{38}W_{64}W_{90}W_{35}W_{62}W_{89}W_{32}W_{60}W_{88}W_{29}W_{58}W_{87}W_{26}W_{56}W_{86}W_{23}W_{54}W_{85}W_{20}W_{52}$ $W_{84}W_{17}W_{50}W_{83}W_{14}W_{48}W_{82}W_{11}W_{46}W_{81}W_{8}W_{44}W_{80}W_{5}W_{42}W_{79}W_{2}W_{40}W_{78}012\rightarrow$ 00000000000000000 00000000000000000000001111111111111111111111111111111111111122222222222222222222 222222222222222222
40	$W_{116}W_{117}W_{118}W_{113}W_{115}W_{117}W_{110}W_{113}W_{116}W_{107}W_{111}W_{115}W_{104}W_{109}W_{114}W_{101}W_{107}W_{113}W_{98}W_{105}W_{112}$ $W_{95}W_{103}W_{111}W_{92}W_{101}W_{110}W_{89}W_{99}W_{109}W_{86}W_{97}W_{108}W_{83}W_{95}W_{107}W_{80}W_{93}W_{106}W_{77}W_{91}W_{105}W_{74}W_{89}W$ $_{104}W_{71}W_{87}W_{103}W_{68}W_{85}W_{102}W_{65}W_{83}W_{101}W_{62}W_{81}W_{100}W_{59}W_{79}W_{99}W_{56}W_{77}W_{98}W_{53}W_{75}W_{97}W_{50}W_{73}W_{96}$ $W_{47}W_{71}W_{95}W_{44}W_{69}W_{94}W_{41}W_{67}W_{93}W_{38}W_{65}W_{92}W_{35}W_{63}W_{91}W_{32}W_{61}W_{90}W_{29}W_{59}W_{89}W_{26}W_{57}W_{88}W_{23}$ $W_{55}W_{87}W_{20}W_{53}W_{86}W_{17}W_{51}W_{85}W_{14}W_{49}W_{84}W_{11}W_{47}W_{83}W_{8}W_{45}W_{82}W_{5}W_{43}W_{81}W_{2}W_{41}W_{80}012\rightarrow0000$ 000000000000000000000000000000000000001111111111111111111111111111111111111122222 22222222222222222222222222222222
41	$W_{119}W_{120}W_{121}W_{116}W_{118}W_{120}W_{113}W_{116}W_{119}W_{110}W_{114}W_{118}W_{107}W_{112}W_{117}W_{104}W_{110}W_{116}W_{101}W_{108}W_{115}$ $W_{98}W_{106}W_{114}W_{95}W_{104}W_{113}W_{92}W_{102}W_{112}W_{89}W_{100}W_{111}W_{86}W_{98}W_{110}W_{83}W_{96}W_{109}W_{80}W_{94}W_{108}W_{77}W_{92}$ $W_{107}W_{74}W_{90}W_{106}W_{71}W_{88}W_{105}W_{68}W_{86}W_{104}W_{65}W_{84}W_{103}W_{62}W_{82}W_{102}W_{59}W_{80}W_{101}W_{56}W_{78}W_{100}W_{53}W_{7}$ $_{6}W_{99}W_{50}W_{74}W_{98}W_{47}W_{72}W_{97}W_{44}W_{70}W_{96}W_{41}W_{68}W_{95}W_{38}W_{66}W_{94}W_{35}W_{64}W_{93}W_{32}W_{62}W_{92}W_{29}W_{60}W_{91}$ $W_{26}W_{58}W_{90}W_{23}W_{56}W_{89}W_{20}W_{54}W_{88}W_{17}W_{52}W_{87}W_{14}W_{50}W_{86}W_{11}W_{48}W_{85}W_{8}W_{46}W_{84}W_{5}W_{44}W_{83}W_{2}W_{42}$ $W_{82}012\rightarrow$ 00000000000000000000000000000000000000001111111111111111111111111111111 11111111122222222222222222222222222222222222222222
42	$W_{122}W_{123}W_{124}W_{119}W_{121}W_{123}W_{116}W_{119}W_{122}W_{113}W_{117}W_{121}W_{110}W_{115}W_{120}W_{107}W_{113}W_{119}W_{104}W_{111}W_{118}$ $W_{101}W_{109}W_{117}W_{98}W_{107}W_{116}W_{95}W_{105}W_{115}W_{92}W_{103}W_{114}W_{89}W_{101}W_{113}W_{86}W_{99}W_{112}W_{83}W_{97}W_{111}W_{80}W_{9}$ $_{5}W_{110}W_{77}W_{93}W_{109}W_{74}W_{91}W_{108}W_{71}W_{89}W_{107}W_{68}W_{87}W_{106}W_{65}W_{85}W_{105}W_{62}W_{83}W_{104}W_{59}W_{81}W_{103}W_{56}W$ $_{79}W_{102}W_{53}W_{77}W_{101}W_{50}W_{75}W_{100}W_{47}W_{73}W_{99}W_{44}W_{71}W_{98}W_{41}W_{69}W_{97}W_{38}W_{67}W_{96}W_{35}W_{65}W_{95}W_{32}W_{63}$ $W_{94}W_{29}W_{61}W_{93}W_{26}W_{59}W_{92}W_{23}W_{57}W_{91}W_{20}W_{55}W_{90}W_{17}W_{53}W_{89}W_{14}W_{51}W_{88}W_{11}W_{49}W_{87}W_{8}W_{47}W_{86}W$

	$_{5}W_{45}W_{85}W_{2}W_{43}W_{84}$012→000000000000000000000000000000000000000000111111111111111111111111111111111111111111222222222222222222222222222222222222222222
43	$W_{125}W_{126}W_{127}W_{122}W_{124}W_{126}W_{119}W_{122}W_{125}W_{116}W_{120}W_{124}W_{113}W_{118}W_{123}W_{110}W_{116}W_{122}W_{107}W_{114}W_{121}W_{104}W_{112}W_{120}W_{101}W_{110}W_{119}W_{98}W_{108}W_{118}W_{95}W_{106}W_{117}W_{92}W_{104}W_{116}W_{89}W_{102}W_{115}W_{86}W_{100}W_{114}W_{83}W_{98}W_{113}W_{80}W_{96}W_{112}W_{77}W_{94}W_{111}W_{74}W_{92}W_{110}W_{71}W_{90}W_{109}W_{68}W_{88}W_{108}W_{65}W_{86}W_{107}W_{62}W_{84}W_{106}W_{59}W_{82}W_{105}W_{56}W_{80}W_{104}W_{53}W_{78}W_{103}W_{50}W_{76}W_{102}W_{47}W_{74}W_{101}W_{44}W_{72}W_{100}W_{41}W_{70}W_{99}W_{38}W_{68}W_{98}W_{35}W_{66}W_{97}W_{32}W_{64}W_{96}W_{29}W_{62}W_{95}W_{26}W_{60}W_{94}W_{23}W_{58}W_{93}W_{20}W_{56}W_{92}W_{17}W_{54}W_{91}W_{14}W_{52}W_{90}W_{11}W_{50}W_{89}W_{8}W_{48}W_{88}W_{5}W_{46}W_{87}W_{2}W_{44}W_{86}$012→0000000000000000000000000000000000000000000111111111111111111111111111111111111111111122222222222222222222222222222222222222222222
44	$W_{128}W_{129}W_{130}W_{125}W_{127}W_{129}W_{122}W_{125}W_{128}W_{119}W_{123}W_{127}W_{116}W_{121}W_{126}W_{113}W_{119}W_{125}W_{110}W_{117}W_{124}W_{107}W_{115}W_{123}W_{104}W_{113}W_{122}W_{101}W_{111}W_{121}W_{98}W_{109}W_{120}W_{95}W_{107}W_{119}W_{92}W_{105}W_{118}W_{89}W_{103}W_{117}W_{86}W_{101}W_{116}W_{83}W_{99}W_{115}W_{80}W_{97}W_{114}W_{77}W_{95}W_{113}W_{74}W_{93}W_{112}W_{71}W_{91}W_{111}W_{68}W_{89}W_{110}W_{65}W_{87}W_{109}W_{62}W_{85}W_{108}W_{59}W_{83}W_{107}W_{56}W_{81}W_{106}W_{53}W_{79}W_{105}W_{50}W_{77}W_{104}W_{47}W_{75}W_{103}W_{44}W_{73}W_{102}W_{41}W_{71}W_{101}W_{38}W_{69}W_{100}W_{35}W_{67}W_{99}W_{32}W_{65}W_{98}W_{29}W_{63}W_{97}W_{26}W_{61}W_{96}W_{23}W_{59}W_{95}W_{20}W_{57}W_{94}W_{17}W_{55}W_{93}W_{14}W_{53}W_{92}W_{11}W_{51}W_{91}W_{8}W_{49}W_{90}W_{5}W_{47}W_{89}W_{2}W_{45}W_{88}$012→00000000000000000000000000000000000000000000111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222
45	$W_{131}W_{132}W_{133}W_{128}W_{130}W_{132}W_{125}W_{128}W_{131}W_{122}W_{126}W_{130}W_{119}W_{124}W_{129}W_{116}W_{122}W_{128}W_{113}W_{120}W_{127}W_{110}W_{118}W_{126}W_{107}W_{116}W_{125}W_{104}W_{114}W_{124}W_{101}W_{112}W_{123}W_{98}W_{110}W_{122}W_{95}W_{108}W_{121}W_{92}W_{106}W_{120}W_{89}W_{104}W_{119}W_{86}W_{102}W_{118}W_{83}W_{100}W_{117}W_{80}W_{98}W_{116}W_{77}W_{96}W_{115}W_{74}W_{94}W_{114}W_{71}W_{92}W_{113}W_{68}W_{90}W_{112}W_{65}W_{88}W_{111}W_{62}W_{86}W_{110}W_{59}W_{84}W_{109}W_{56}W_{82}W_{108}W_{53}W_{80}W_{107}W_{50}W_{78}W_{106}W_{47}W_{76}W_{105}W_{44}W_{74}W_{104}W_{41}W_{72}W_{103}W_{38}W_{70}W_{102}W_{35}W_{68}W_{101}W_{32}W_{66}W_{100}W_{29}W_{64}W_{99}W_{26}W_{62}W_{98}W_{23}W_{60}W_{97}W_{20}W_{58}W_{96}W_{17}W_{56}W_{95}W_{14}W_{54}W_{94}W_{11}W_{52}W_{93}W_{8}W_{50}W_{92}W_{5}W_{48}W_{91}W_{2}W_{46}W_{90}$012→000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222
46	$W_{134}W_{135}W_{136}W_{131}W_{133}W_{135}W_{128}W_{131}W_{134}W_{125}W_{129}W_{133}W_{122}W_{127}W_{132}W_{119}W_{125}W_{131}W_{116}W_{123}W_{130}W_{113}W_{121}W_{129}W_{110}W_{119}W_{128}W_{107}W_{117}W_{127}W_{104}W_{115}W_{126}W_{101}W_{113}W_{125}W_{98}W_{111}W_{124}W_{95}W_{109}W_{123}W_{92}W_{107}W_{122}W_{89}W_{105}W_{121}W_{86}W_{103}W_{120}W_{83}W_{101}W_{119}W_{80}W_{99}W_{118}W_{77}W_{97}W_{117}W_{74}W_{95}W_{116}W_{71}W_{93}W_{115}W_{68}W_{91}W_{114}W_{65}W_{89}W_{113}W_{62}W_{87}W_{112}W_{59}W_{85}W_{111}W_{56}W_{83}W_{110}W_{53}W_{81}W_{109}W_{50}W_{79}W_{108}W_{47}W_{77}W_{107}W_{44}W_{75}W_{106}W_{41}W_{73}W_{105}W_{38}W_{71}W_{104}W_{35}W_{69}W_{103}W_{32}W_{67}W_{102}W_{29}W_{65}W_{101}W_{26}W_{63}W_{100}W_{23}W_{61}W_{99}W_{20}W_{59}W_{98}W_{17}W_{57}W_{97}W_{14}W_{55}W_{96}W_{11}W_{53}W_{95}W_{8}W_{51}W_{94}W_{5}W_{49}W_{93}W_{2}W_{47}W_{92}$012→000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222
47	$W_{137}W_{138}W_{139}W_{134}W_{136}W_{138}W_{131}W_{134}W_{137}W_{128}W_{132}W_{136}W_{125}W_{130}W_{135}W_{122}W_{128}W_{134}W_{119}W_{126}W_{133}W_{116}W_{124}W_{132}W_{113}W_{122}W_{131}W_{110}W_{120}W_{130}W_{107}W_{118}W_{129}W_{104}W_{116}W_{128}W_{101}W_{114}W_{127}W_{98}W_{112}W_{126}W_{95}W_{110}W_{125}W_{92}W_{108}W_{124}W_{89}W_{106}W_{123}W_{86}W_{104}W_{122}W_{83}W_{102}W_{121}W_{80}W_{100}W_{120}W_{77}W_{98}W_{119}W_{74}W_{96}W_{118}W_{71}W_{94}W_{117}W_{68}W_{92}W_{116}W_{65}W_{90}W_{115}W_{62}W_{88}W_{114}W_{59}W_{86}W_{113}W_{56}W_{84}W_{112}W_{53}W_{82}W_{111}W_{50}W_{80}W_{110}W_{47}W_{78}W_{109}W_{44}W_{76}W_{108}W_{41}W_{74}W_{107}W_{38}W_{72}W_{106}W_{35}W_{70}W_{105}W_{32}W_{68}W_{104}W_{29}W_{66}W_{103}W_{26}W_{64}W_{102}W_{23}W_{62}W_{101}W_{20}W_{60}W_{100}W_{17}W_{58}W_{99}W_{14}W_{56}W_{98}W_{11}W_{54}W_{97}W_{8}W_{52}W_{96}W_{5}W_{50}W_{95}W_{2}W_{48}W_{94}$012→0000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222
48	$W_{140}W_{141}W_{142}W_{137}W_{139}W_{141}W_{134}W_{137}W_{140}W_{131}W_{135}W_{139}W_{128}W_{133}W_{138}W_{125}W_{131}W_{137}W_{122}W_{129}W_{136}W_{119}W_{127}W_{135}W_{116}W_{125}W_{134}W_{113}W_{123}W_{133}W_{110}W_{121}W_{132}W_{107}W_{119}W_{131}W_{104}W_{117}W_{130}W_{101}W_{115}W_{129}W_{98}W_{113}W_{128}W_{95}W_{111}W_{127}W_{92}W_{109}W_{126}W_{89}W_{107}W_{125}W_{86}W_{105}W_{124}W_{83}W_{103}W_{123}W_{80}W_{101}W_{122}W_{77}W_{99}W_{121}W_{74}W_{97}W_{120}W_{71}W_{95}W_{119}W_{68}W_{93}W_{118}W_{65}W_{91}W_{117}W_{62}W_{89}W_{116}W_{59}W_{87}W_{115}W_{56}W_{85}W_{114}W_{53}W_{83}W_{113}W_{50}W_{81}W_{112}W_{47}W_{79}W_{111}W_{44}W_{77}W_{110}W_{41}W_{75}W_{109}W_{38}W_{73}W_{108}W_{35}W_{71}W_{107}W_{32}W_{69}W_{106}W_{29}W_{67}W_{105}W_{26}W_{65}W_{104}W_{23}W_{63}W_{103}W_{20}W_{61}W_{102}W_{17}W_{59}W_{101}W_{14}W_{57}W_{100}W_{11}W_{55}W_{99}W_{8}W_{53}W_{98}W_{5}W_{51}W_{97}W_{2}W_{49}W_{96}$012→000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222
49	$W_{143}W_{144}W_{145}W_{140}W_{142}W_{144}W_{137}W_{140}W_{143}W_{134}W_{138}W_{142}W_{131}W_{136}W_{141}W_{128}W_{134}W_{140}W_{125}W_{132}W_{139}W_{122}W_{130}W_{138}W_{119}W_{128}W_{137}W_{116}W_{126}W_{136}W_{113}W_{124}W_{135}W_{110}W_{122}W_{134}W_{107}W_{120}W_{133}W_{104}W_{118}W_{132}W_{101}W_{116}W_{131}W_{98}W_{114}W_{130}W_{95}W_{112}W_{129}W_{92}W_{110}W_{128}W_{89}W_{108}W_{127}W_{86}W_{106}W_{126}W_{83}W_{104}W_{125}W_{80}W_{102}W_{124}W_{77}W_{100}W_{123}W_{74}W_{98}W_{122}W_{71}W_{96}W_{121}W_{68}W_{94}W_{120}W_{65}W_{92}W_{119}W_{62}W_{90}W_{118}W_{59}W_{88}W_{117}$

	$W_{56}W_{86}W_{116}W_{53}W_{84}W_{115}W_{50}W_{82}W_{114}W_{47}W_{80}W_{113}W_{44}W_{78}W_{112}W_{41}W_{76}W_{111}W_{38}W_{74}W_{110}W_{35}W_{72}W_{109}$ $W_{32}W_{70}W_{108}W_{29}W_{68}W_{107}W_{26}W_{66}W_{106}W_{23}W_{64}W_{105}W_{20}W_{62}W_{104}W_{17}W_{60}W_{103}W_{14}W_{58}W_{102}W_{11}W_{56}W_{101}$ $W_{8}W_{54}W_{100}W_{5}W_{52}W_{99}W_{2}W_{50}W_{98}$012→000000000000000000000000000000000000000000000000 0011111111111111111111111111111111111111111111111111222222222222222222222222222222222 2222222222222222222
50	$W_{146}W_{147}W_{148}W_{143}W_{145}W_{147}W_{140}W_{143}W_{146}W_{137}W_{141}W_{145}W_{134}W_{139}W_{144}W_{131}W_{137}W_{143}W_{128}W_{135}W_{142}$ $W_{125}W_{133}W_{141}W_{122}W_{131}W_{140}W_{119}W_{129}W_{139}W_{116}W_{127}W_{138}W_{113}W_{125}W_{137}W_{110}W_{123}W_{136}W_{107}W_{121}W_{135}$ $W_{104}W_{119}W_{134}W_{101}W_{117}W_{133}W_{98}W_{115}W_{132}W_{95}W_{113}W_{131}W_{92}W_{111}W_{130}W_{89}W_{109}W_{129}W_{86}W_{107}W_{128}W_{83}$ $W_{105}W_{127}W_{80}W_{103}W_{126}W_{77}W_{101}W_{125}W_{74}W_{99}W_{124}W_{71}W_{97}W_{123}W_{68}W_{95}W_{122}W_{65}W_{93}W_{121}W_{62}W_{91}W_{120}$ $W_{59}W_{89}W_{119}W_{56}W_{87}W_{118}W_{53}W_{85}W_{117}W_{50}W_{83}W_{116}W_{47}W_{81}W_{115}W_{44}W_{79}W_{114}W_{41}W_{77}W_{113}W_{38}W_{75}W_{112}$ $W_{35}W_{73}W_{111}W_{32}W_{71}W_{110}W_{29}W_{69}W_{109}W_{26}W_{67}W_{108}W_{23}W_{65}W_{107}W_{20}W_{63}W_{106}W_{17}W_{61}W_{105}W_{14}W_{59}W_{104}$ $W_{11}W_{57}W_{103}W_{8}W_{55}W_{102}W_{5}W_{53}W_{101}W_{2}W_{51}W_{100}$012→0000000000000000000000000000000000000 000000000000001111111111111111111111111111111111111111111111111112222222222222222222 22222222222222222222222222222222
51	$W_{149}W_{150}W_{151}W_{146}W_{148}W_{150}W_{143}W_{146}W_{149}W_{140}W_{144}W_{148}W_{137}W_{142}W_{147}W_{134}W_{140}W_{146}W_{131}W_{138}W_{145}$ $W_{128}W_{136}W_{144}W_{125}W_{134}W_{143}W_{122}W_{132}W_{142}W_{119}W_{130}W_{141}W_{116}W_{128}W_{140}W_{113}W_{126}W_{139}W_{110}W_{124}W_{138}$ $W_{107}W_{122}W_{137}W_{104}W_{120}W_{136}W_{101}W_{118}W_{135}W_{98}W_{116}W_{134}W_{95}W_{114}W_{133}W_{92}W_{112}W_{132}W_{89}W_{110}W_{131}W_{86}$ $W_{108}W_{130}W_{83}W_{106}W_{129}W_{80}W_{104}W_{128}W_{77}W_{102}W_{127}W_{74}W_{100}W_{126}W_{71}W_{98}W_{125}W_{68}W_{96}W_{124}W_{65}W_{94}W_{123}$ $W_{62}W_{92}W_{122}W_{59}W_{90}W_{121}W_{56}W_{88}W_{120}W_{53}W_{86}W_{119}W_{50}W_{84}W_{118}W_{47}W_{82}W_{117}W_{44}W_{80}W_{116}W_{41}W_{78}W_{115}$ $W_{38}W_{76}W_{114}W_{35}W_{74}W_{113}W_{32}W_{72}W_{112}W_{29}W_{70}W_{111}W_{26}W_{68}W_{110}W_{23}W_{66}W_{109}W_{20}W_{64}W_{108}W_{17}W_{62}W_{107}$ $W_{14}W_{60}W_{106}W_{11}W_{58}W_{105}W_{8}W_{56}W_{104}W_{5}W_{54}W_{103}W_{2}W_{52}W_{102}$012→00000000000000000000000000 0000000000000000000000000011111111111111111111111111111111111111111111111111112222 222222222222222222222222222222222222222222222222
52	$W_{152}W_{153}W_{154}W_{149}W_{151}W_{153}W_{146}W_{149}W_{152}W_{143}W_{147}W_{151}W_{140}W_{145}W_{150}W_{137}W_{143}W_{149}W_{134}W_{141}W_{148}$ $W_{131}W_{139}W_{147}W_{128}W_{137}W_{146}W_{125}W_{135}W_{145}W_{122}W_{133}W_{144}W_{119}W_{131}W_{143}W_{116}W_{129}W_{142}W_{113}W_{127}W_{141}$ $W_{110}W_{125}W_{140}W_{107}W_{123}W_{139}W_{104}W_{121}W_{138}W_{101}W_{119}W_{137}W_{98}W_{117}W_{136}W_{95}W_{115}W_{135}W_{92}W_{113}W_{134}W_{89}$ $W_{111}W_{133}W_{86}W_{109}W_{132}W_{83}W_{107}W_{131}W_{80}W_{105}W_{130}W_{77}W_{103}W_{129}W_{74}W_{101}W_{128}W_{71}W_{99}W_{127}W_{68}W_{97}W_{126}$ $W_{65}W_{95}W_{125}W_{62}W_{93}W_{124}W_{59}W_{91}W_{123}W_{56}W_{89}W_{122}W_{53}W_{87}W_{121}W_{50}W_{85}W_{120}W_{47}W_{83}W_{119}W_{44}W_{81}$ $W_{118}W_{41}W_{79}W_{117}W_{38}W_{77}W_{116}W_{35}W_{75}W_{115}W_{32}W_{73}W_{114}W_{29}W_{71}W_{113}W_{26}W_{69}W_{112}W_{23}W_{67}W_{111}W_{20}W_{65}$ $W_{110}W_{17}W_{63}W_{109}W_{14}W_{61}W_{108}W_{11}W_{59}W_{107}W_{8}W_{57}W_{106}W_{5}W_{55}W_{105}W_{2}W_{53}W_{104}$012→000000000000 00000000000000000000000000000000000000000111111111111111111111111111111111111111111 1111111111122222222222222222222222222222222222222222222222222222
53	$W_{155}W_{156}W_{157}W_{152}W_{154}W_{156}W_{149}W_{152}W_{155}W_{146}W_{150}W_{154}W_{143}W_{148}W_{153}W_{140}W_{146}W_{152}W_{137}W_{144}W_{151}$ $W_{134}W_{142}W_{150}W_{131}W_{140}W_{149}W_{128}W_{138}W_{148}W_{125}W_{136}W_{147}W_{122}W_{134}W_{146}W_{119}W_{132}W_{145}W_{116}W_{130}W_{144}$ $W_{113}W_{128}W_{143}W_{110}W_{126}W_{142}W_{107}W_{124}W_{141}W_{104}W_{122}W_{140}W_{101}W_{120}W_{139}W_{98}W_{118}W_{138}W_{95}W_{116}W_{137}W_{92}$ $W_{114}W_{136}W_{89}W_{112}W_{135}W_{86}W_{110}W_{134}W_{83}W_{108}W_{133}W_{80}W_{106}W_{132}W_{77}W_{104}W_{131}W_{74}W_{102}W_{130}W_{71}W_{100}$ $W_{129}W_{68}W_{98}W_{128}W_{65}W_{96}W_{127}W_{62}W_{94}W_{126}W_{59}W_{92}W_{125}W_{56}W_{90}W_{124}W_{53}W_{88}W_{123}W_{50}W_{86}W_{122}W_{47}W_{84}$ $W_{121}W_{44}W_{82}W_{120}W_{41}W_{80}W_{119}W_{38}W_{78}W_{118}W_{35}W_{76}W_{117}W_{32}W_{74}W_{116}W_{29}W_{72}W_{115}W_{26}W_{70}W_{114}W_{23}W_{68}$ $W_{113}W_{20}W_{66}W_{112}W_{17}W_{64}W_{111}W_{14}W_{62}W_{110}W_{11}W_{60}W_{109}W_{8}W_{58}W_{108}W_{5}W_{56}W_{107}W_{2}W_{54}W_{106}$012→0 00000000000000000000000000000000000000000000000000000111111111111111111111111111111 111111111111111111111111222222222222222222222222222222222222222222222222222222
54	$W_{158}W_{159}W_{160}W_{155}W_{157}W_{159}W_{152}W_{155}W_{158}W_{149}W_{153}W_{157}W_{146}W_{151}W_{156}W_{143}W_{149}W_{155}W_{140}W_{147}W_{154}$ $W_{137}W_{145}W_{153}W_{134}W_{143}W_{152}W_{131}W_{141}W_{151}W_{128}W_{139}W_{150}W_{125}W_{137}W_{149}W_{122}W_{135}W_{148}W_{119}W_{133}W_{147}$ $W_{116}W_{131}W_{146}W_{113}W_{129}W_{145}W_{110}W_{127}W_{144}W_{107}W_{125}W_{143}W_{104}W_{123}W_{142}W_{101}W_{121}W_{141}W_{98}W_{119}W_{140}$ $W_{95}W_{117}W_{139}W_{92}W_{115}W_{138}W_{89}W_{113}W_{137}W_{86}W_{111}W_{136}W_{83}W_{109}W_{135}W_{80}W_{107}W_{134}W_{77}W_{105}W_{133}W_{74}W_{103}$ $W_{132}W_{71}W_{101}W_{131}W_{68}W_{99}W_{130}W_{65}W_{97}W_{129}W_{62}W_{95}W_{128}W_{59}W_{93}W_{127}W_{56}W_{91}W_{126}W_{53}W_{89}W_{125}W_{50}$ $W_{87}W_{124}W_{47}W_{85}W_{123}W_{44}W_{83}W_{122}W_{41}W_{81}W_{121}W_{38}W_{79}W_{120}W_{35}W_{77}W_{119}W_{32}W_{75}W_{118}W_{29}W_{73}W_{117}W_{26}$ $W_{71}W_{116}W_{23}W_{69}W_{115}W_{20}W_{67}W_{114}W_{17}W_{65}W_{113}W_{14}W_{63}W_{112}W_{11}W_{61}W_{111}W_{8}W_{59}W_{110}W_{5}W_{57}W_{109}W_{2}$ $W_{55}W_{108}$012→0000000000000000000000000000000000000000000000000000000111111111111111 11111111111111111111111111111111111111112222222222222222222222222222222222222222222 222222222222
55	$W_{161}W_{162}W_{163}W_{158}W_{160}W_{162}W_{155}W_{158}W_{161}W_{152}W_{156}W_{160}W_{149}W_{154}W_{159}W_{146}W_{152}W_{158}W_{143}W_{150}W_{157}$ $W_{140}W_{148}W_{156}W_{137}W_{146}W_{155}W_{134}W_{144}W_{154}W_{131}W_{142}W_{153}W_{128}W_{140}W_{152}W_{125}W_{138}W_{151}W_{122}W_{136}W_{150}$ $W_{119}W_{134}W_{149}W_{116}W_{132}W_{148}W_{113}W_{130}W_{147}W_{110}W_{128}W_{146}W_{107}W_{126}W_{145}W_{104}W_{124}W_{144}W_{101}W_{122}W_{143}$ $W_{98}W_{120}W_{142}W_{95}W_{118}W_{141}W_{92}W_{116}W_{140}W_{89}W_{114}W_{139}W_{86}W_{112}W_{138}W_{83}W_{110}W_{137}W_{80}W_{108}W_{136}W_{77}W_{106}$ $W_{135}W_{74}W_{104}W_{134}W_{71}W_{102}W_{133}W_{68}W_{100}W_{132}W_{65}W_{98}W_{131}W_{62}W_{96}W_{130}W_{59}W_{94}W_{129}W_{56}W_{92}W_{128}W$

	$_{53}W_{90}W_{127}W_{50}W_{88}W_{126}W_{47}W_{86}W_{125}W_{44}W_{84}W_{124}W_{41}W_{82}W_{123}W_{38}W_{80}W_{122}W_{35}W_{78}W_{121}W_{32}W_{76}W_{120}$ $W_{29}W_{74}W_{119}W_{26}W_{72}W_{118}W_{23}W_{70}W_{117}W_{20}W_{68}W_{116}W_{17}W_{66}W_{115}W_{14}W_{64}W_{114}W_{11}W_{62}W_{113}W_{8}W_{60}W_{112}$ $W_{5}W_{58}W_{111}W_{2}W_{56}W_{110}012\rightarrow$ 00000000000000000000000000000000000000000000000000000000000011 11111111111111111111111111111111111111111111111111111111111122222222222222222222222222222 22222222222222222222222222
56	$W_{164}W_{165}W_{166}W_{161}W_{163}W_{165}W_{158}W_{161}W_{164}W_{155}W_{159}W_{163}W_{152}W_{157}W_{162}W_{149}W_{155}W_{161}W_{146}W_{153}W_{160}$ $W_{143}W_{151}W_{159}W_{140}W_{149}W_{158}W_{137}W_{147}W_{157}W_{134}W_{145}W_{156}W_{131}W_{143}W_{155}W_{128}W_{141}W_{154}W_{125}W_{139}W_{153}$ $W_{122}W_{137}W_{152}W_{119}W_{135}W_{151}W_{116}W_{133}W_{150}W_{113}W_{131}W_{149}W_{110}W_{129}W_{148}W_{107}W_{127}W_{147}W_{104}W_{125}W_{146}$ $W_{101}W_{123}W_{145}W_{98}W_{121}W_{144}W_{95}W_{119}W_{143}W_{92}W_{117}W_{142}W_{89}W_{115}W_{141}W_{86}W_{113}W_{140}W_{83}W_{111}W_{139}W_{80}$ $W_{109}W_{138}W_{77}W_{107}W_{137}W_{74}W_{105}W_{136}W_{71}W_{103}W_{135}W_{68}W_{101}W_{134}W_{65}W_{99}W_{133}W_{62}W_{97}W_{132}W_{59}W_{95}W_{13}$ $_{1}W_{56}W_{93}W_{130}W_{53}W_{91}W_{129}W_{50}W_{89}W_{128}W_{47}W_{87}W_{127}W_{44}W_{85}W_{126}W_{41}W_{83}W_{125}W_{38}W_{81}W_{124}W_{35}W_{79}W_{1}$ $_{23}W_{32}W_{77}W_{122}W_{29}W_{75}W_{121}W_{26}W_{73}W_{120}W_{23}W_{71}W_{119}W_{20}W_{69}W_{118}W_{17}W_{67}W_{117}W_{14}W_{65}W_{116}W_{11}W_{63}W$ $_{115}W_{8}W_{61}W_{114}W_{5}W_{59}W_{113}W_{2}W_{57}W_{112}012\rightarrow$ 00000000000000000000000000000000000000000000000000000000000000 00000000000111111111111111111111111111111111111111111111111111111111122222222222222 22222222222222222222222222222222222222222222
57	$W_{167}W_{168}W_{169}W_{164}W_{166}W_{168}W_{161}W_{164}W_{167}W_{158}W_{162}W_{166}W_{155}W_{160}W_{165}W_{152}W_{158}W_{164}W_{149}W_{156}W_{163}$ $W_{146}W_{154}W_{162}W_{143}W_{152}W_{161}W_{140}W_{150}W_{160}W_{137}W_{148}W_{159}W_{134}W_{146}W_{158}W_{131}W_{144}W_{157}W_{128}W_{142}W_{156}$ $W_{125}W_{140}W_{155}W_{122}W_{138}W_{154}W_{119}W_{136}W_{153}W_{116}W_{134}W_{152}W_{113}W_{132}W_{151}W_{110}W_{130}W_{150}W_{107}W_{128}W_{149}$ $W_{104}W_{126}W_{148}W_{101}W_{124}W_{147}W_{98}W_{122}W_{146}W_{95}W_{120}W_{145}W_{92}W_{118}W_{144}W_{89}W_{116}W_{143}W_{86}W_{114}W_{142}W_{83}$ $W_{112}W_{141}W_{80}W_{110}W_{140}W_{77}W_{108}W_{139}W_{74}W_{106}W_{138}W_{71}W_{104}W_{137}W_{68}W_{102}W_{136}W_{65}W_{100}W_{135}W_{62}W_{98}W$ $_{134}W_{59}W_{96}W_{133}W_{56}W_{94}W_{132}W_{53}W_{92}W_{131}W_{50}W_{90}W_{130}W_{47}W_{88}W_{129}W_{44}W_{86}W_{128}W_{41}W_{84}W_{127}W_{38}W_{82}$ $W_{126}W_{35}W_{80}W_{125}W_{32}W_{78}W_{124}W_{29}W_{76}W_{123}W_{26}W_{74}W_{122}W_{23}W_{72}W_{121}W_{20}W_{70}W_{120}W_{17}W_{68}W_{119}W_{14}W_{6}$ $_{6}W_{118}W_{11}W_{64}W_{117}W_{8}W_{62}W_{116}W_{5}W_{60}W_{115}W_{2}W_{58}W_{114}012\rightarrow$ 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000 00000000000000000000000000011111111111111111111111111111111111111111111111111111111111 12222222222222222222222222222222222222222222222222222222222222222222
58	$W_{170}W_{171}W_{172}W_{167}W_{169}W_{171}W_{164}W_{167}W_{170}W_{161}W_{165}W_{169}W_{158}W_{163}W_{168}W_{155}W_{161}W_{167}W_{152}W_{159}W_{166}$ $W_{149}W_{157}W_{165}W_{146}W_{155}W_{164}W_{143}W_{153}W_{163}W_{140}W_{151}W_{162}W_{137}W_{149}W_{161}W_{134}W_{147}W_{160}W_{131}W_{145}W_{159}$ $W_{128}W_{143}W_{158}W_{125}W_{141}W_{157}W_{122}W_{139}W_{156}W_{119}W_{137}W_{155}W_{116}W_{135}W_{154}W_{113}W_{133}W_{153}W_{110}W_{131}W_{152}$ $W_{107}W_{129}W_{151}W_{104}W_{127}W_{150}W_{101}W_{125}W_{149}W_{98}W_{123}W_{148}W_{95}W_{121}W_{147}W_{92}W_{119}W_{146}W_{89}W_{117}W_{145}W_{86}$ $W_{115}W_{144}W_{83}W_{113}W_{143}W_{80}W_{111}W_{142}W_{77}W_{109}W_{141}W_{74}W_{107}W_{140}W_{71}W_{105}W_{139}W_{68}W_{103}W_{138}W_{65}W_{101}$ $W_{137}W_{62}W_{99}W_{136}W_{59}W_{97}W_{135}W_{56}W_{95}W_{134}W_{53}W_{93}W_{133}W_{50}W_{91}W_{132}W_{47}W_{89}W_{131}W_{44}W_{87}W_{130}W_{41}W_{8}$ $_{5}W_{129}W_{38}W_{83}W_{128}W_{35}W_{81}W_{127}W_{32}W_{79}W_{126}W_{29}W_{77}W_{125}W_{26}W_{75}W_{124}W_{23}W_{73}W_{123}W_{20}W_{71}W_{122}W_{17}W$ $_{69}W_{121}W_{14}W_{67}W_{120}W_{11}W_{65}W_{119}W_{8}W_{63}W_{118}W_{5}W_{61}W_{117}W_{2}W_{59}W_{116}012\rightarrow$ 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000 00000000000000000000000000000000000000000000011111111111111111111111111111111111111111 11111111111111222222222222222222222222222222222222222222222222222222222222222222222222
59	$W_{173}W_{174}W_{175}W_{170}W_{172}W_{174}W_{167}W_{170}W_{173}W_{164}W_{168}W_{172}W_{161}W_{166}W_{171}W_{158}W_{164}W_{170}W_{155}W_{162}W_{169}$ $W_{152}W_{160}W_{168}W_{149}W_{158}W_{167}W_{146}W_{156}W_{166}W_{143}W_{154}W_{165}W_{140}W_{152}W_{164}W_{137}W_{150}W_{163}W_{134}W_{148}W_{162}$ $W_{131}W_{146}W_{161}W_{128}W_{144}W_{160}W_{125}W_{142}W_{159}W_{122}W_{140}W_{158}W_{119}W_{138}W_{157}W_{116}W_{136}W_{156}W_{113}W_{134}W_{155}$ $W_{110}W_{132}W_{154}W_{107}W_{130}W_{153}W_{104}W_{128}W_{152}W_{101}W_{126}W_{151}W_{98}W_{124}W_{150}W_{95}W_{122}W_{149}W_{92}W_{120}W_{148}W_{8}$ $_{9}W_{118}W_{147}W_{86}W_{116}W_{146}W_{83}W_{114}W_{145}W_{80}W_{112}W_{144}W_{77}W_{110}W_{143}W_{74}W_{108}W_{142}W_{71}W_{106}W_{141}W_{68}W_{104}$ $W_{140}W_{65}W_{102}W_{139}W_{62}W_{100}W_{138}W_{59}W_{98}W_{137}W_{56}W_{96}W_{136}W_{53}W_{94}W_{135}W_{50}W_{92}W_{134}W_{47}W_{90}W_{133}W_{44}$ $W_{88}W_{132}W_{41}W_{86}W_{131}W_{38}W_{84}W_{130}W_{35}W_{82}W_{129}W_{32}W_{80}W_{128}W_{29}W_{78}W_{127}W_{26}W_{76}W_{126}W_{23}W_{74}W_{125}W_{2}$ $_{0}W_{72}W_{124}W_{17}W_{70}W_{123}W_{14}W_{68}W_{122}W_{11}W_{66}W_{121}W_{8}W_{64}W_{120}W_{5}W_{62}W_{119}W_{2}W_{60}W_{118}012\rightarrow$ 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000 00000000000000000000000000000000000000000000000000011111111111111111111111111111111111 11111111111111111111111111112222222222222222222222222222222222222222222222222222222222 222222
60	$W_{176}W_{177}W_{178}W_{173}W_{175}W_{177}W_{170}W_{173}W_{176}W_{167}W_{171}W_{175}W_{164}W_{169}W_{174}W_{161}W_{167}W_{173}W_{158}W_{165}W_{172}$ $W_{155}W_{163}W_{171}W_{152}W_{161}W_{170}W_{149}W_{159}W_{169}W_{146}W_{157}W_{168}W_{143}W_{155}W_{167}W_{140}W_{153}W_{166}W_{137}W_{151}W_{165}$ $W_{134}W_{149}W_{164}W_{131}W_{147}W_{163}W_{128}W_{145}W_{162}W_{125}W_{143}W_{161}W_{122}W_{141}W_{160}W_{119}W_{139}W_{159}W_{116}W_{137}W_{158}$ $W_{113}W_{135}W_{157}W_{110}W_{133}W_{156}W_{107}W_{131}W_{155}W_{104}W_{129}W_{154}W_{101}W_{127}W_{153}W_{98}W_{125}W_{152}W_{95}W_{123}W_{151}W$ $_{92}W_{121}W_{150}W_{89}W_{119}W_{149}W_{86}W_{117}W_{148}W_{83}W_{115}W_{147}W_{80}W_{113}W_{146}W_{77}W_{111}W_{145}W_{74}W_{109}W_{144}W_{71}W_{107}$ $W_{143}W_{68}W_{105}W_{142}W_{65}W_{103}W_{141}W_{62}W_{101}W_{140}W_{59}W_{99}W_{139}W_{56}W_{97}W_{138}W_{53}W_{95}W_{137}W_{50}W_{93}W_{136}W_{47}$ $W_{91}W_{135}W_{44}W_{89}W_{134}W_{41}W_{87}W_{133}W_{38}W_{85}W_{132}W_{35}W_{83}W_{131}W_{32}W_{81}W_{130}W_{29}W_{79}W_{129}W_{26}W_{77}W_{128}W_{2}$ $_{3}W_{75}W_{127}W_{20}W_{73}W_{126}W_{17}W_{71}W_{125}W_{14}W_{69}W_{124}W_{11}W_{67}W_{123}W_{8}W_{65}W_{122}W_{5}W_{63}W_{121}W_{2}W_{61}W_{120}012$ $\rightarrow$ 00000000000000000000000000000000000000000000000000000000000000000000000000000000000001111111111111111111

	11111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222
61	$W_{179}W_{180}W_{181}W_{176}W_{178}W_{180}W_{173}W_{176}W_{179}W_{170}W_{174}W_{178}W_{167}W_{172}W_{177}W_{164}W_{170}W_{176}W_{161}W_{168}W_{175}W_{158}W_{166}W_{174}W_{155}W_{164}W_{173}W_{152}W_{162}W_{172}W_{149}W_{160}W_{171}W_{146}W_{158}W_{170}W_{143}W_{156}W_{169}W_{140}W_{154}W_{168}W_{137}W_{152}W_{167}W_{134}W_{150}W_{166}W_{131}W_{148}W_{165}W_{128}W_{146}W_{164}W_{125}W_{144}W_{163}W_{122}W_{142}W_{162}W_{119}W_{140}W_{161}W_{116}W_{138}W_{160}W_{113}W_{136}W_{159}W_{110}W_{134}W_{158}W_{107}W_{132}W_{157}W_{104}W_{130}W_{156}W_{101}W_{128}W_{155}W_{98}W_{126}W_{154}W_{95}W_{124}W_{153}W_{92}W_{122}W_{152}W_{89}W_{120}W_{151}W_{86}W_{118}W_{150}W_{83}W_{116}W_{149}W_{80}W_{114}W_{148}W_{77}W_{112}W_{147}W_{74}W_{110}W_{146}W_{71}W_{108}W_{145}W_{68}W_{106}W_{144}W_{65}W_{104}W_{143}W_{62}W_{102}W_{142}W_{59}W_{100}W_{141}W_{56}W_{98}W_{140}W_{53}W_{96}W_{139}W_{50}W_{94}W_{138}W_{47}W_{92}W_{137}W_{44}W_{90}W_{136}W_{41}W_{88}W_{135}W_{38}W_{86}W_{134}W_{35}W_{84}W_{133}W_{32}W_{82}W_{132}W_{29}W_{80}W_{131}W_{26}W_{78}W_{130}W_{23}W_{76}W_{129}W_{20}W_{74}W_{128}W_{17}W_{72}W_{127}W_{14}W_{70}W_{126}W_{11}W_{68}W_{125}W_{8}W_{66}W_{124}W_{5}W_{64}W_{123}W_{2}W_{62}W_{122}$012→0000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222
62	$W_{182}W_{183}W_{184}W_{179}W_{181}W_{183}W_{176}W_{179}W_{182}W_{173}W_{177}W_{181}W_{170}W_{175}W_{180}W_{167}W_{173}W_{179}W_{164}W_{171}W_{178}W_{161}W_{169}W_{177}W_{158}W_{167}W_{176}W_{155}W_{165}W_{175}W_{152}W_{163}W_{174}W_{149}W_{161}W_{173}W_{146}W_{159}W_{172}W_{143}W_{157}W_{171}W_{140}W_{155}W_{170}W_{137}W_{153}W_{169}W_{134}W_{151}W_{168}W_{131}W_{149}W_{167}W_{128}W_{147}W_{166}W_{125}W_{145}W_{165}W_{122}W_{143}W_{164}W_{119}W_{141}W_{163}W_{116}W_{139}W_{162}W_{113}W_{137}W_{161}W_{110}W_{135}W_{160}W_{107}W_{133}W_{159}W_{104}W_{131}W_{158}W_{101}W_{129}W_{157}W_{98}W_{127}W_{156}W_{95}W_{125}W_{155}W_{92}W_{123}W_{154}W_{89}W_{121}W_{153}W_{86}W_{119}W_{152}W_{83}W_{117}W_{151}W_{80}W_{115}W_{150}W_{77}W_{113}W_{149}W_{74}W_{111}W_{148}W_{71}W_{109}W_{147}W_{68}W_{107}W_{146}W_{65}W_{105}W_{145}W_{62}W_{103}W_{144}W_{59}W_{101}W_{143}W_{56}W_{99}W_{142}W_{53}W_{97}W_{141}W_{50}W_{95}W_{140}W_{47}W_{93}W_{139}W_{44}W_{91}W_{138}W_{41}W_{89}W_{137}W_{38}W_{87}W_{136}W_{35}W_{85}W_{135}W_{32}W_{83}W_{134}W_{29}W_{81}W_{133}W_{26}W_{79}W_{132}W_{23}W_{77}W_{131}W_{20}W_{75}W_{130}W_{17}W_{73}W_{129}W_{14}W_{71}W_{128}W_{11}W_{69}W_{127}W_{8}W_{67}W_{126}W_{5}W_{65}W_{125}W_{2}W_{63}W_{124}$012→00000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222
63	$W_{185}W_{186}W_{187}W_{182}W_{184}W_{186}W_{179}W_{182}W_{185}W_{176}W_{180}W_{184}W_{173}W_{178}W_{183}W_{170}W_{176}W_{182}W_{167}W_{174}W_{181}W_{164}W_{172}W_{180}W_{161}W_{170}W_{179}W_{158}W_{168}W_{178}W_{155}W_{166}W_{177}W_{152}W_{164}W_{176}W_{149}W_{162}W_{175}W_{146}W_{160}W_{174}W_{143}W_{158}W_{173}W_{140}W_{156}W_{172}W_{137}W_{154}W_{171}W_{134}W_{152}W_{170}W_{131}W_{150}W_{169}W_{128}W_{148}W_{168}W_{125}W_{146}W_{167}W_{122}W_{144}W_{166}W_{119}W_{142}W_{165}W_{116}W_{140}W_{164}W_{113}W_{138}W_{163}W_{110}W_{136}W_{162}W_{107}W_{134}W_{161}W_{104}W_{132}W_{160}W_{101}W_{130}W_{159}W_{98}W_{128}W_{158}W_{95}W_{126}W_{157}W_{92}W_{124}W_{156}W_{89}W_{122}W_{155}W_{86}W_{120}W_{154}W_{83}W_{118}W_{153}W_{80}W_{116}W_{152}W_{77}W_{114}W_{151}W_{74}W_{112}W_{150}W_{71}W_{110}W_{149}W_{68}W_{108}W_{148}W_{65}W_{106}W_{147}W_{62}W_{104}W_{146}W_{59}W_{102}W_{145}W_{56}W_{100}W_{144}W_{53}W_{98}W_{143}W_{50}W_{96}W_{142}W_{47}W_{94}W_{141}W_{44}W_{92}W_{140}W_{41}W_{90}W_{139}W_{38}W_{88}W_{138}W_{35}W_{86}W_{137}W_{32}W_{84}W_{136}W_{29}W_{82}W_{135}W_{26}W_{80}W_{134}W_{23}W_{78}W_{133}W_{20}W_{76}W_{132}W_{17}W_{74}W_{131}W_{14}W_{72}W_{130}W_{11}W_{70}W_{129}W_{8}W_{68}W_{128}W_{5}W_{66}W_{127}W_{2}W_{64}W_{126}$012→000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222
64	$W_{188}W_{189}W_{190}W_{185}W_{187}W_{189}W_{182}W_{185}W_{188}W_{179}W_{183}W_{187}W_{176}W_{181}W_{186}W_{173}W_{179}W_{185}W_{170}W_{177}W_{184}W_{167}W_{175}W_{183}W_{164}W_{173}W_{182}W_{161}W_{171}W_{181}W_{158}W_{169}W_{180}W_{155}W_{167}W_{179}W_{152}W_{165}W_{178}W_{149}W_{163}W_{177}W_{146}W_{161}W_{176}W_{143}W_{159}W_{175}W_{140}W_{157}W_{174}W_{137}W_{155}W_{173}W_{134}W_{153}W_{172}W_{131}W_{151}W_{171}W_{128}W_{149}W_{170}W_{125}W_{147}W_{169}W_{122}W_{145}W_{168}W_{119}W_{143}W_{167}W_{116}W_{141}W_{166}W_{113}W_{139}W_{165}W_{110}W_{137}W_{164}W_{107}W_{135}W_{163}W_{104}W_{133}W_{162}W_{101}W_{131}W_{161}W_{98}W_{129}W_{160}W_{95}W_{127}W_{159}W_{92}W_{125}W_{158}W_{89}W_{123}W_{157}W_{86}W_{121}W_{156}W_{83}W_{119}W_{155}W_{80}W_{117}W_{154}W_{77}W_{115}W_{153}W_{74}W_{113}W_{152}W_{71}W_{111}W_{151}W_{68}W_{109}W_{150}W_{65}W_{107}W_{149}W_{62}W_{105}W_{148}W_{59}W_{103}W_{147}W_{56}W_{101}W_{146}W_{53}W_{99}W_{145}W_{50}W_{97}W_{144}W_{47}W_{95}W_{143}W_{44}W_{93}W_{142}W_{41}W_{91}W_{141}W_{38}W_{89}W_{140}W_{35}W_{87}W_{139}W_{32}W_{85}W_{138}W_{29}W_{83}W_{137}W_{26}W_{81}W_{136}W_{23}W_{79}W_{135}W_{20}W_{77}W_{134}W_{17}W_{75}W_{133}W_{14}W_{73}W_{132}W_{11}W_{71}W_{131}W_{8}W_{69}W_{130}W_{5}W_{67}W_{129}W_{2}W_{65}W_{128}$012→000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222
65	$W_{191}W_{192}W_{193}W_{188}W_{190}W_{192}W_{185}W_{188}W_{191}W_{182}W_{186}W_{190}W_{179}W_{184}W_{189}W_{176}W_{182}W_{188}W_{173}W_{180}W_{187}W_{170}W_{178}W_{186}W_{167}W_{176}W_{185}W_{164}W_{174}W_{184}W_{161}W_{172}W_{183}W_{158}W_{170}W_{182}W_{155}W_{168}W_{181}W_{152}W_{166}W_{180}W_{149}W_{164}W_{179}W_{146}W_{162}W_{178}W_{143}W_{160}W_{177}W_{140}W_{158}W_{176}W_{137}W_{156}W_{175}W_{134}W_{154}W_{174}W_{131}W_{152}W_{173}W_{128}W_{150}W_{172}W_{125}W_{148}W_{171}W_{122}W_{146}W_{170}W_{119}W_{144}W_{169}W_{116}W_{142}W_{168}W_{113}W_{140}W_{167}W_{110}W_{138}W_{166}W_{107}W_{136}W_{165}W_{104}W_{134}W_{164}W_{101}W_{132}W_{163}W_{98}W_{130}W_{162}W_{95}W_{128}W_{161}W_{92}W_{126}W_{160}W_{89}W_{124}W_{159}W_{86}W_{122}W_{158}W_{83}W_{120}W_{157}W_{80}W_{118}W_{156}W_{77}W_{116}W_{155}W_{74}W_{114}W_{154}W_{71}W_{112}W_{153}W_{68}W_{110}W_{152}W_{65}W_{108}W_{151}W_{62}W_{106}W_{150}W_{59}W_{104}W_{149}W_{56}W_{102}W_{148}W_{53}W_{100}W_{147}W_{50}W_{98}W_{146}W_{47}W_{96}W_{145}W_{44}W_{94}W_{144}W_{41}W_{92}W_{143}W_{38}W_{90}W_{142}W_{35}W_{88}W_{141}W_{32}W_{86}W_{140}W_{29}W_{84}W_{139}W_{26}W_{82}W_{138}W_{23}W_{80}W_{137}W_{20}W_{78}W_{136}W_{17}W_{76}W_{135}W_{14}W_{74}W_{134}W_{11}W_{72}W_{133}W_{8}W_{70}W_{132}W_{5}W_{68}W_{131}W_{2}W_{66}W_{130}$012→0000000000000000000

	0000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222
66	$W_{194}W_{195}W_{196}W_{191}W_{193}W_{195}W_{188}W_{191}W_{194}W_{185}W_{189}W_{193}W_{182}W_{187}W_{192}W_{179}W_{185}W_{191}W_{176}W_{183}W_{190}W_{173}W_{181}W_{189}W_{170}W_{179}W_{188}W_{167}W_{177}W_{187}W_{164}W_{175}W_{186}W_{161}W_{173}W_{185}W_{158}W_{171}W_{184}W_{155}W_{169}W_{183}W_{152}W_{167}W_{182}W_{149}W_{165}W_{181}W_{146}W_{163}W_{180}W_{143}W_{161}W_{179}W_{140}W_{159}W_{178}W_{137}W_{157}W_{177}W_{134}W_{155}W_{176}W_{131}W_{153}W_{175}W_{128}W_{151}W_{174}W_{125}W_{149}W_{173}W_{122}W_{147}W_{172}W_{119}W_{145}W_{171}W_{116}W_{143}W_{170}W_{113}W_{141}W_{169}W_{110}W_{139}W_{168}W_{107}W_{137}W_{167}W_{104}W_{135}W_{166}W_{101}W_{133}W_{165}W_{98}W_{131}W_{164}W_{95}W_{129}W_{163}W_{92}W_{127}W_{162}W_{89}W_{125}W_{161}W_{86}W_{123}W_{160}W_{83}W_{121}W_{159}W_{80}W_{119}W_{158}W_{77}W_{117}W_{157}W_{74}W_{115}W_{156}W_{71}W_{113}W_{155}W_{68}W_{111}W_{154}W_{65}W_{109}W_{153}W_{62}W_{107}W_{152}W_{59}W_{105}W_{151}W_{56}W_{103}W_{150}W_{53}W_{101}W_{149}W_{50}W_{99}W_{148}W_{47}W_{97}W_{147}W_{44}W_{95}W_{146}W_{41}W_{93}W_{145}W_{38}W_{91}W_{144}W_{35}W_{89}W_{143}W_{32}W_{87}W_{142}W_{29}W_{85}W_{141}W_{26}W_{83}W_{140}W_{23}W_{81}W_{139}W_{20}W_{79}W_{138}W_{17}W_{77}W_{137}W_{14}W_{75}W_{136}W_{11}W_{73}W_{135}W_{8}W_{71}W_{134}W_{5}W_{69}W_{133}W_{2}W_{67}W_{132}$012→000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222
67	$W_{197}W_{198}W_{199}W_{194}W_{196}W_{198}W_{191}W_{194}W_{197}W_{188}W_{192}W_{196}W_{185}W_{190}W_{195}W_{182}W_{188}W_{194}W_{179}W_{186}W_{193}W_{176}W_{184}W_{192}W_{173}W_{182}W_{191}W_{170}W_{180}W_{190}W_{167}W_{178}W_{189}W_{164}W_{176}W_{188}W_{161}W_{174}W_{187}W_{158}W_{172}W_{186}W_{155}W_{170}W_{185}W_{152}W_{168}W_{184}W_{149}W_{166}W_{183}W_{146}W_{164}W_{182}W_{143}W_{162}W_{181}W_{140}W_{160}W_{180}W_{137}W_{158}W_{179}W_{134}W_{156}W_{178}W_{131}W_{154}W_{177}W_{128}W_{152}W_{176}W_{125}W_{150}W_{175}W_{122}W_{148}W_{174}W_{119}W_{146}W_{173}W_{116}W_{144}W_{172}W_{113}W_{142}W_{171}W_{110}W_{140}W_{170}W_{107}W_{138}W_{169}W_{104}W_{136}W_{168}W_{101}W_{134}W_{167}W_{98}W_{132}W_{166}W_{95}W_{130}W_{165}W_{92}W_{128}W_{164}W_{89}W_{126}W_{163}W_{86}W_{124}W_{162}W_{83}W_{122}W_{161}W_{80}W_{120}W_{160}W_{77}W_{118}W_{159}W_{74}W_{116}W_{158}W_{71}W_{114}W_{157}W_{68}W_{112}W_{156}W_{65}W_{110}W_{155}W_{62}W_{108}W_{154}W_{59}W_{106}W_{153}W_{56}W_{104}W_{152}W_{53}W_{102}W_{151}W_{50}W_{100}W_{150}W_{47}W_{98}W_{149}W_{44}W_{96}W_{148}W_{41}W_{94}W_{147}W_{38}W_{92}W_{146}W_{35}W_{90}W_{145}W_{32}W_{88}W_{144}W_{29}W_{86}W_{143}W_{26}W_{84}W_{142}W_{23}W_{82}W_{141}W_{20}W_{80}W_{140}W_{17}W_{78}W_{139}W_{14}W_{76}W_{138}W_{11}W_{74}W_{137}W_{8}W_{72}W_{136}W_{5}W_{70}W_{135}W_{2}W_{68}W_{134}$012→000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222
68	$W_{200}W_{201}W_{202}W_{197}W_{199}W_{201}W_{194}W_{197}W_{200}W_{191}W_{195}W_{199}W_{188}W_{193}W_{198}W_{185}W_{191}W_{197}W_{182}W_{189}W_{196}W_{179}W_{187}W_{195}W_{176}W_{185}W_{194}W_{173}W_{183}W_{193}W_{170}W_{181}W_{192}W_{167}W_{179}W_{191}W_{164}W_{177}W_{190}W_{161}W_{175}W_{189}W_{158}W_{173}W_{188}W_{155}W_{171}W_{187}W_{152}W_{169}W_{186}W_{149}W_{167}W_{185}W_{146}W_{165}W_{184}W_{143}W_{163}W_{183}W_{140}W_{161}W_{182}W_{137}W_{159}W_{181}W_{134}W_{157}W_{180}W_{131}W_{155}W_{179}W_{128}W_{153}W_{178}W_{125}W_{151}W_{177}W_{122}W_{149}W_{176}W_{119}W_{147}W_{175}W_{116}W_{145}W_{174}W_{113}W_{143}W_{173}W_{110}W_{141}W_{172}W_{107}W_{139}W_{171}W_{104}W_{137}W_{170}W_{101}W_{135}W_{169}W_{98}W_{133}W_{168}W_{95}W_{131}W_{167}W_{92}W_{129}W_{166}W_{89}W_{127}W_{165}W_{86}W_{125}W_{164}W_{83}W_{123}W_{163}W_{80}W_{121}W_{162}W_{77}W_{119}W_{161}W_{74}W_{117}W_{160}W_{71}W_{115}W_{159}W_{68}W_{113}W_{158}W_{65}W_{111}W_{157}W_{62}W_{109}W_{156}W_{59}W_{107}W_{155}W_{56}W_{105}W_{154}W_{53}W_{103}W_{153}W_{50}W_{101}W_{152}W_{47}W_{99}W_{151}W_{44}W_{97}W_{150}W_{41}W_{95}W_{149}W_{38}W_{93}W_{148}W_{35}W_{91}W_{147}W_{32}W_{89}W_{146}W_{29}W_{87}W_{145}W_{26}W_{85}W_{144}W_{23}W_{83}W_{143}W_{20}W_{81}W_{142}W_{17}W_{79}W_{141}W_{14}W_{77}W_{140}W_{11}W_{75}W_{139}W_{8}W_{73}W_{138}W_{5}W_{71}W_{137}W_{2}W_{69}W_{136}$012→000000000000000000000000000000000000000000000000000000000000000000001111111111111111111111111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222222222222222222222222
69	$W_{203}W_{204}W_{205}W_{200}W_{202}W_{204}W_{197}W_{200}W_{203}W_{194}W_{198}W_{202}W_{191}W_{196}W_{201}W_{188}W_{194}W_{200}W_{185}W_{192}W_{199}W_{182}W_{190}W_{198}W_{179}W_{188}W_{197}W_{176}W_{186}W_{196}W_{173}W_{184}W_{195}W_{170}W_{182}W_{194}W_{167}W_{180}W_{193}W_{164}W_{178}W_{192}W_{161}W_{176}W_{191}W_{158}W_{174}W_{190}W_{155}W_{172}W_{189}W_{152}W_{170}W_{188}W_{149}W_{168}W_{187}W_{146}W_{166}W_{186}W_{143}W_{164}W_{185}W_{140}W_{162}W_{184}W_{137}W_{160}W_{183}W_{134}W_{158}W_{182}W_{131}W_{156}W_{181}W_{128}W_{154}W_{180}W_{125}W_{152}W_{179}W_{122}W_{150}W_{178}W_{119}W_{148}W_{177}W_{116}W_{146}W_{176}W_{113}W_{144}W_{175}W_{110}W_{142}W_{174}W_{107}W_{140}W_{173}W_{104}W_{138}W_{172}W_{101}W_{136}W_{171}W_{98}W_{134}W_{170}W_{95}W_{132}W_{169}W_{92}W_{130}W_{168}W_{89}W_{128}W_{167}W_{86}W_{126}W_{166}W_{83}W_{124}W_{165}W_{80}W_{122}W_{164}W_{77}W_{120}W_{163}W_{74}W_{118}W_{162}W_{71}W_{116}W_{161}W_{68}W_{114}W_{160}W_{65}W_{112}W_{159}W_{62}W_{110}W_{158}W_{59}W_{108}W_{157}W_{56}W_{106}W_{156}W_{53}W_{104}W_{155}W_{50}W_{102}W_{154}W_{47}W_{100}W_{153}W_{44}W_{98}W_{152}W_{41}W_{96}W_{151}W_{38}W_{94}W_{150}W_{35}W_{92}W_{149}W_{32}W_{90}W_{148}W_{29}W_{88}W_{147}W_{26}W_{86}W_{146}W_{23}W_{84}W_{145}W_{20}W_{82}W_{144}W_{17}W_{80}W_{143}W_{14}W_{78}W_{142}W_{11}W_{76}W_{141}W_{8}W_{74}W_{140}W_{5}W_{72}W_{139}W_{2}W_{70}W_{138}$012→000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222
70	$W_{206}W_{207}W_{208}W_{203}W_{205}W_{207}W_{200}W_{203}W_{206}W_{197}W_{201}W_{205}W_{194}W_{199}W_{204}W_{191}W_{197}W_{203}W_{188}W_{195}W_{202}W_{185}W_{193}W_{201}W_{182}W_{191}W_{200}W_{179}W_{189}W_{199}W_{176}W_{187}W_{198}W_{173}W_{185}W_{197}W_{170}W_{183}W_{196}W_{167}W_{181}W_{195}W_{164}W_{179}W_{194}W_{161}W_{177}W_{193}W_{158}W_{175}W_{192}W_{155}W_{173}W_{191}W_{152}W_{171}W_{190}W_{149}W_{169}W_{189}W_{146}W_{167}W_{188}W_{143}W_{165}W_{187}W_{140}W_{163}W_{186}W_{137}W_{161}W_{185}W_{134}W_{159}W_{184}W_{131}W_{157}W_{183}W_{128}W_{155}W_{182}W_{125}W_{153}W_{181}W_{122}W_{151}W_{180}W_{119}W_{149}W_{179}W_{116}W_{147}W_{178}W_{113}W_{145}W_{177}W_{110}W_{143}W_{176}W_{107}W_{141}W_{175}W_{104}W_{139}W_{174}$

	$W_{101}W_{137}W_{173}W_{98}W_{135}W_{172}W_{95}W_{133}W_{171}W_{92}W_{131}W_{170}W_{89}W_{129}W_{169}W_{86}W_{127}W_{168}W_{83}W_{125}W_{167}W_{80}W_{123}W_{166}W_{77}W_{121}W_{165}W_{74}W_{119}W_{164}W_{71}W_{117}W_{163}W_{68}W_{115}W_{162}W_{65}W_{113}W_{161}W_{62}W_{111}W_{160}W_{59}W_{109}W_{159}W_{56}W_{107}W_{158}W_{53}W_{105}W_{157}W_{50}W_{103}W_{156}W_{47}W_{101}W_{155}W_{44}W_{99}W_{154}W_{41}W_{97}W_{153}W_{38}W_{95}W_{152}W_{35}W_{93}W_{151}W_{32}W_{91}W_{150}W_{29}W_{89}W_{149}W_{26}W_{87}W_{148}W_{23}W_{85}W_{147}W_{20}W_{83}W_{146}W_{17}W_{81}W_{145}W_{14}W_{79}W_{144}W_{11}W_{77}W_{143}W_{8}W_{75}W_{142}W_{5}W_{73}W_{141}W_{2}W_{71}W_{140}$012→0000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222222222222222222222222222
71	$W_{209}W_{210}W_{211}W_{206}W_{208}W_{210}W_{203}W_{206}W_{209}W_{200}W_{204}W_{208}W_{197}W_{202}W_{207}W_{194}W_{200}W_{206}W_{191}W_{198}W_{205}W_{188}W_{196}W_{204}W_{185}W_{194}W_{203}W_{182}W_{192}W_{202}W_{179}W_{190}W_{201}W_{176}W_{188}W_{200}W_{173}W_{186}W_{199}W_{170}W_{184}W_{198}W_{167}W_{182}W_{197}W_{164}W_{180}W_{196}W_{161}W_{178}W_{195}W_{158}W_{176}W_{194}W_{155}W_{174}W_{193}W_{152}W_{172}W_{192}W_{149}W_{170}W_{191}W_{146}W_{168}W_{190}W_{143}W_{166}W_{189}W_{140}W_{164}W_{188}W_{137}W_{162}W_{187}W_{134}W_{160}W_{186}W_{131}W_{158}W_{185}W_{128}W_{156}W_{184}W_{125}W_{154}W_{183}W_{122}W_{152}W_{182}W_{119}W_{150}W_{181}W_{116}W_{148}W_{180}W_{113}W_{146}W_{179}W_{110}W_{144}W_{178}W_{107}W_{142}W_{177}W_{104}W_{140}W_{176}W_{101}W_{138}W_{175}W_{98}W_{136}W_{174}W_{95}W_{134}W_{173}W_{92}W_{132}W_{172}W_{89}W_{130}W_{171}W_{86}W_{128}W_{170}W_{83}W_{126}W_{169}W_{80}W_{124}W_{168}W_{77}W_{122}W_{167}W_{74}W_{120}W_{166}W_{71}W_{118}W_{165}W_{68}W_{116}W_{164}W_{65}W_{114}W_{163}W_{62}W_{112}W_{162}W_{59}W_{110}W_{161}W_{56}W_{108}W_{160}W_{53}W_{106}W_{159}W_{50}W_{104}W_{158}W_{47}W_{102}W_{157}W_{44}W_{100}W_{156}W_{41}W_{98}W_{155}W_{38}W_{96}W_{154}W_{35}W_{94}W_{153}W_{32}W_{92}W_{152}W_{29}W_{90}W_{151}W_{26}W_{88}W_{150}W_{23}W_{86}W_{149}W_{20}W_{84}W_{148}W_{17}W_{82}W_{147}W_{14}W_{80}W_{146}W_{11}W_{78}W_{145}W_{8}W_{76}W_{144}W_{5}W_{74}W_{143}W_{2}W_{72}W_{142}$012→00000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222
72	$W_{212}W_{213}W_{214}W_{209}W_{211}W_{213}W_{206}W_{209}W_{212}W_{203}W_{207}W_{211}W_{200}W_{205}W_{210}W_{197}W_{203}W_{209}W_{194}W_{201}W_{208}W_{191}W_{199}W_{207}W_{188}W_{197}W_{206}W_{185}W_{195}W_{205}W_{182}W_{193}W_{204}W_{179}W_{191}W_{203}W_{176}W_{189}W_{202}W_{173}W_{187}W_{201}W_{170}W_{185}W_{200}W_{167}W_{183}W_{199}W_{164}W_{181}W_{198}W_{161}W_{179}W_{197}W_{158}W_{177}W_{196}W_{155}W_{175}W_{195}W_{152}W_{173}W_{194}W_{149}W_{171}W_{193}W_{146}W_{169}W_{192}W_{143}W_{167}W_{191}W_{140}W_{165}W_{190}W_{137}W_{163}W_{189}W_{134}W_{161}W_{188}W_{131}W_{159}W_{187}W_{128}W_{157}W_{186}W_{125}W_{155}W_{185}W_{122}W_{153}W_{184}W_{119}W_{151}W_{183}W_{116}W_{149}W_{182}W_{113}W_{147}W_{181}W_{110}W_{145}W_{180}W_{107}W_{143}W_{179}W_{104}W_{141}W_{178}W_{101}W_{139}W_{177}W_{98}W_{137}W_{176}W_{95}W_{135}W_{175}W_{92}W_{133}W_{174}W_{89}W_{131}W_{173}W_{86}W_{129}W_{172}W_{83}W_{127}W_{171}W_{80}W_{125}W_{170}W_{77}W_{123}W_{169}W_{74}W_{121}W_{168}W_{71}W_{119}W_{167}W_{68}W_{117}W_{166}W_{65}W_{115}W_{165}W_{62}W_{113}W_{164}W_{59}W_{111}W_{163}W_{56}W_{109}W_{162}W_{53}W_{107}W_{161}W_{50}W_{105}W_{160}W_{47}W_{103}W_{159}W_{44}W_{101}W_{158}W_{41}W_{99}W_{157}W_{38}W_{97}W_{156}W_{35}W_{95}W_{155}W_{32}W_{93}W_{154}W_{29}W_{91}W_{153}W_{26}W_{89}W_{152}W_{23}W_{87}W_{151}W_{20}W_{85}W_{150}W_{17}W_{83}W_{149}W_{14}W_{81}W_{148}W_{11}W_{79}W_{147}W_{8}W_{77}W_{146}W_{5}W_{75}W_{145}W_{2}W_{73}W_{144}$012→000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222
73	$W_{215}W_{216}W_{217}W_{212}W_{214}W_{216}W_{209}W_{212}W_{215}W_{206}W_{210}W_{214}W_{203}W_{208}W_{213}W_{200}W_{206}W_{212}W_{197}W_{204}W_{211}W_{194}W_{202}W_{210}W_{191}W_{200}W_{209}W_{188}W_{198}W_{208}W_{185}W_{196}W_{207}W_{182}W_{194}W_{206}W_{179}W_{192}W_{205}W_{176}W_{190}W_{204}W_{173}W_{188}W_{203}W_{170}W_{186}W_{202}W_{167}W_{184}W_{201}W_{164}W_{182}W_{200}W_{161}W_{180}W_{199}W_{158}W_{178}W_{198}W_{155}W_{176}W_{197}W_{152}W_{174}W_{196}W_{149}W_{172}W_{195}W_{146}W_{170}W_{194}W_{143}W_{168}W_{193}W_{140}W_{166}W_{192}W_{137}W_{164}W_{191}W_{134}W_{162}W_{190}W_{131}W_{160}W_{189}W_{128}W_{158}W_{188}W_{125}W_{156}W_{187}W_{122}W_{154}W_{186}W_{119}W_{152}W_{185}W_{116}W_{150}W_{184}W_{113}W_{148}W_{183}W_{110}W_{146}W_{182}W_{107}W_{144}W_{181}W_{104}W_{142}W_{180}W_{101}W_{140}W_{179}W_{98}W_{138}W_{178}W_{95}W_{136}W_{177}W_{92}W_{134}W_{176}W_{89}W_{132}W_{175}W_{86}W_{130}W_{174}W_{83}W_{128}W_{173}W_{80}W_{126}W_{172}W_{77}W_{124}W_{171}W_{74}W_{122}W_{170}W_{71}W_{120}W_{169}W_{68}W_{118}W_{168}W_{65}W_{116}W_{167}W_{62}W_{114}W_{166}W_{59}W_{112}W_{165}W_{56}W_{110}W_{164}W_{53}W_{108}W_{163}W_{50}W_{106}W_{162}W_{47}W_{104}W_{161}W_{44}W_{102}W_{160}W_{41}W_{100}W_{159}W_{38}W_{98}W_{158}W_{35}W_{96}W_{157}W_{32}W_{94}W_{156}W_{29}W_{92}W_{155}W_{26}W_{90}W_{154}W_{23}W_{88}W_{153}W_{20}W_{86}W_{152}W_{17}W_{84}W_{151}W_{14}W_{82}W_{150}W_{11}W_{80}W_{149}W_{8}W_{78}W_{148}W_{5}W_{76}W_{147}W_{2}W_{74}W_{146}$012→000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222
74	$W_{218}W_{219}W_{220}W_{215}W_{217}W_{219}W_{212}W_{215}W_{218}W_{209}W_{213}W_{217}W_{206}W_{211}W_{216}W_{203}W_{209}W_{215}W_{200}W_{207}W_{214}W_{197}W_{205}W_{213}W_{194}W_{203}W_{212}W_{191}W_{201}W_{211}W_{188}W_{199}W_{210}W_{185}W_{197}W_{209}W_{182}W_{195}W_{208}W_{179}W_{193}W_{207}W_{176}W_{191}W_{206}W_{173}W_{189}W_{205}W_{170}W_{187}W_{204}W_{167}W_{185}W_{203}W_{164}W_{183}W_{202}W_{161}W_{181}W_{201}W_{158}W_{179}W_{200}W_{155}W_{177}W_{199}W_{152}W_{175}W_{198}W_{149}W_{173}W_{197}W_{146}W_{171}W_{196}W_{143}W_{169}W_{195}W_{140}W_{167}W_{194}W_{137}W_{165}W_{193}W_{134}W_{163}W_{192}W_{131}W_{161}W_{191}W_{128}W_{159}W_{190}W_{125}W_{157}W_{189}W_{122}W_{155}W_{188}W_{119}W_{153}W_{187}W_{116}W_{151}W_{186}W_{113}W_{149}W_{185}W_{110}W_{147}W_{184}W_{107}W_{145}W_{183}W_{104}W_{143}W_{182}W_{101}W_{141}W_{181}W_{98}W_{139}W_{180}W_{95}W_{137}W_{179}W_{92}W_{135}W_{178}W_{89}W_{133}W_{177}W_{86}W_{131}W_{176}W_{83}W_{129}W_{175}W_{80}W_{127}W_{174}W_{77}W_{125}W_{173}W_{74}W_{123}W_{172}W_{71}W_{121}W_{171}W_{68}W_{119}W_{170}W_{65}W_{117}W_{169}W_{62}W_{115}W_{168}W_{59}W_{113}W_{167}W_{56}W_{111}W_{166}W_{53}W_{109}W_{165}W_{50}W_{107}W_{164}W_{47}W_{105}W_{163}W_{44}W_{103}W_{162}W_{41}W_{101}W_{161}W_{38}W_{99}W_{160}W_{35}W_{97}W_{159}W_{32}W_{95}W_{158}W_{29}W_{93}W_{157}W_{26}W_{91}$

	$W_{156}W_{23}W_{89}W_{155}W_{20}W_{87}W_{154}W_{17}W_{85}W_{153}W_{14}W_{83}W_{152}W_{11}W_{81}W_{151}W_{8}W_{79}W_{150}W_{5}W_{77}W_{149}W_{2}W_{75}W_{148}$012→000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222
75	$W_{221}W_{222}W_{223}W_{218}W_{220}W_{222}W_{215}W_{218}W_{221}W_{212}W_{216}W_{220}W_{209}W_{214}W_{219}W_{206}W_{212}W_{218}W_{203}W_{210}W_{217}W_{200}W_{208}W_{216}W_{197}W_{206}W_{215}W_{194}W_{204}W_{214}W_{191}W_{202}W_{213}W_{188}W_{200}W_{212}W_{185}W_{198}W_{211}W_{182}W_{196}W_{210}W_{179}W_{194}W_{209}W_{176}W_{192}W_{208}W_{173}W_{190}W_{207}W_{170}W_{188}W_{206}W_{167}W_{186}W_{205}W_{164}W_{184}W_{204}W_{161}W_{182}W_{203}W_{158}W_{180}W_{202}W_{155}W_{178}W_{201}W_{152}W_{176}W_{200}W_{149}W_{174}W_{199}W_{146}W_{172}W_{198}W_{143}W_{170}W_{197}W_{140}W_{168}W_{196}W_{137}W_{166}W_{195}W_{134}W_{164}W_{194}W_{131}W_{162}W_{193}W_{128}W_{160}W_{192}W_{125}W_{158}W_{191}W_{122}W_{156}W_{190}W_{119}W_{154}W_{189}W_{116}W_{152}W_{188}W_{113}W_{150}W_{187}W_{110}W_{148}W_{186}W_{107}W_{146}W_{185}W_{104}W_{144}W_{184}W_{101}W_{142}W_{183}W_{98}W_{140}W_{182}W_{95}W_{138}W_{181}W_{92}W_{136}W_{180}W_{89}W_{134}W_{179}W_{86}W_{132}W_{178}W_{83}W_{130}W_{177}W_{80}W_{128}W_{176}W_{77}W_{126}W_{175}W_{74}W_{124}W_{174}W_{71}W_{122}W_{173}W_{68}W_{120}W_{172}W_{65}W_{118}W_{171}W_{62}W_{116}W_{170}W_{59}W_{114}W_{169}W_{56}W_{112}W_{168}W_{53}W_{110}W_{167}W_{50}W_{108}W_{166}W_{47}W_{106}W_{165}W_{44}W_{104}W_{164}W_{41}W_{102}W_{163}W_{38}W_{100}W_{162}W_{35}W_{98}W_{161}W_{32}W_{96}W_{160}W_{29}W_{94}W_{159}W_{26}W_{92}W_{158}W_{23}W_{90}W_{157}W_{20}W_{88}W_{156}W_{17}W_{86}W_{155}W_{14}W_{84}W_{154}W_{11}W_{82}W_{153}W_{8}W_{80}W_{152}W_{5}W_{78}W_{151}W_{2}W_{76}W_{150}$012→000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222222
76	$W_{224}W_{225}W_{226}W_{221}W_{223}W_{225}W_{218}W_{221}W_{224}W_{215}W_{219}W_{223}W_{212}W_{217}W_{222}W_{209}W_{215}W_{221}W_{206}W_{213}W_{220}W_{203}W_{211}W_{219}W_{200}W_{209}W_{218}W_{197}W_{207}W_{217}W_{194}W_{205}W_{216}W_{191}W_{203}W_{215}W_{188}W_{201}W_{214}W_{185}W_{199}W_{213}W_{182}W_{197}W_{212}W_{179}W_{195}W_{211}W_{176}W_{193}W_{210}W_{173}W_{191}W_{209}W_{170}W_{189}W_{208}W_{167}W_{187}W_{207}W_{164}W_{185}W_{206}W_{161}W_{183}W_{205}W_{158}W_{181}W_{204}W_{155}W_{179}W_{203}W_{152}W_{177}W_{202}W_{149}W_{175}W_{201}W_{146}W_{173}W_{200}W_{143}W_{171}W_{199}W_{140}W_{169}W_{198}W_{137}W_{167}W_{197}W_{134}W_{165}W_{196}W_{131}W_{163}W_{195}W_{128}W_{161}W_{194}W_{125}W_{159}W_{193}W_{122}W_{157}W_{192}W_{119}W_{155}W_{191}W_{116}W_{153}W_{190}W_{113}W_{151}W_{189}W_{110}W_{149}W_{188}W_{107}W_{147}W_{187}W_{104}W_{145}W_{186}W_{101}W_{143}W_{185}W_{98}W_{141}W_{184}W_{95}W_{139}W_{183}W_{92}W_{137}W_{182}W_{89}W_{135}W_{181}W_{86}W_{133}W_{180}W_{83}W_{131}W_{179}W_{80}W_{129}W_{178}W_{77}W_{127}W_{177}W_{74}W_{125}W_{176}W_{71}W_{123}W_{175}W_{68}W_{121}W_{174}W_{65}W_{119}W_{173}W_{62}W_{117}W_{172}W_{59}W_{115}W_{171}W_{56}W_{113}W_{170}W_{53}W_{111}W_{169}W_{50}W_{109}W_{168}W_{47}W_{107}W_{167}W_{44}W_{105}W_{166}W_{41}W_{103}W_{165}W_{38}W_{101}W_{164}W_{35}W_{99}W_{163}W_{32}W_{97}W_{162}W_{29}W_{95}W_{161}W_{26}W_{93}W_{160}W_{23}W_{91}W_{159}W_{20}W_{89}W_{158}W_{17}W_{87}W_{157}W_{14}W_{85}W_{156}W_{11}W_{83}W_{155}W_{8}W_{81}W_{154}W_{5}W_{79}W_{153}W_{2}W_{77}W_{152}$012→000000000000000000000000000000000000000000000000000000000000000000000000000001111111111111111111111111111111111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222222222222222222222222222222222
77	$W_{227}W_{228}W_{229}W_{224}W_{226}W_{228}W_{221}W_{224}W_{227}W_{218}W_{222}W_{226}W_{215}W_{220}W_{225}W_{212}W_{218}W_{224}W_{209}W_{216}W_{223}W_{206}W_{214}W_{222}W_{203}W_{212}W_{221}W_{200}W_{210}W_{220}W_{197}W_{208}W_{219}W_{194}W_{206}W_{218}W_{191}W_{204}W_{217}W_{188}W_{202}W_{216}W_{185}W_{200}W_{215}W_{182}W_{198}W_{214}W_{179}W_{196}W_{213}W_{176}W_{194}W_{212}W_{173}W_{192}W_{211}W_{170}W_{190}W_{210}W_{167}W_{188}W_{209}W_{164}W_{186}W_{208}W_{161}W_{184}W_{207}W_{158}W_{182}W_{206}W_{155}W_{180}W_{205}W_{152}W_{178}W_{204}W_{149}W_{176}W_{203}W_{146}W_{174}W_{202}W_{143}W_{172}W_{201}W_{140}W_{170}W_{200}W_{137}W_{168}W_{199}W_{134}W_{166}W_{198}W_{131}W_{164}W_{197}W_{128}W_{162}W_{196}W_{125}W_{160}W_{195}W_{122}W_{158}W_{194}W_{119}W_{156}W_{193}W_{116}W_{154}W_{192}W_{113}W_{152}W_{191}W_{110}W_{150}W_{190}W_{107}W_{148}W_{189}W_{104}W_{146}W_{188}W_{101}W_{144}W_{187}W_{98}W_{142}W_{186}W_{95}W_{140}W_{185}W_{92}W_{138}W_{184}W_{89}W_{136}W_{183}W_{86}W_{134}W_{182}W_{83}W_{132}W_{181}W_{80}W_{130}W_{180}W_{77}W_{128}W_{179}W_{74}W_{126}W_{178}W_{71}W_{124}W_{177}W_{68}W_{122}W_{176}W_{65}W_{120}W_{175}W_{62}W_{118}W_{174}W_{59}W_{116}W_{173}W_{56}W_{114}W_{172}W_{53}W_{112}W_{171}W_{50}W_{110}W_{170}W_{47}W_{108}W_{169}W_{44}W_{106}W_{168}W_{41}W_{104}W_{167}W_{38}W_{102}W_{166}W_{35}W_{100}W_{165}W_{32}W_{98}W_{164}W_{29}W_{96}W_{163}W_{26}W_{94}W_{162}W_{23}W_{92}W_{161}W_{20}W_{90}W_{160}W_{17}W_{88}W_{159}W_{14}W_{86}W_{158}W_{11}W_{84}W_{157}W_{8}W_{82}W_{156}W_{5}W_{80}W_{155}W_{2}W_{78}W_{154}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222
78	$W_{230}W_{231}W_{232}W_{227}W_{229}W_{231}W_{224}W_{227}W_{230}W_{221}W_{225}W_{229}W_{218}W_{223}W_{228}W_{215}W_{221}W_{227}W_{212}W_{219}W_{226}W_{209}W_{217}W_{225}W_{206}W_{215}W_{224}W_{203}W_{213}W_{223}W_{200}W_{211}W_{222}W_{197}W_{209}W_{221}W_{194}W_{207}W_{220}W_{191}W_{205}W_{219}W_{188}W_{203}W_{218}W_{185}W_{201}W_{217}W_{182}W_{199}W_{216}W_{179}W_{197}W_{215}W_{176}W_{195}W_{214}W_{173}W_{193}W_{213}W_{170}W_{191}W_{212}W_{167}W_{189}W_{211}W_{164}W_{187}W_{210}W_{161}W_{185}W_{209}W_{158}W_{183}W_{208}W_{155}W_{181}W_{207}W_{152}W_{179}W_{206}W_{149}W_{177}W_{205}W_{146}W_{175}W_{204}W_{143}W_{173}W_{203}W_{140}W_{171}W_{202}W_{137}W_{169}W_{201}W_{134}W_{167}W_{200}W_{131}W_{165}W_{199}W_{128}W_{163}W_{198}W_{125}W_{161}W_{197}W_{122}W_{159}W_{196}W_{119}W_{157}W_{195}W_{116}W_{155}W_{194}W_{113}W_{153}W_{193}W_{110}W_{151}W_{192}W_{107}W_{149}W_{191}W_{104}W_{147}W_{190}W_{101}W_{145}W_{189}W_{98}W_{143}W_{188}W_{95}W_{141}W_{187}W_{92}W_{139}W_{186}W_{89}W_{137}W_{185}W_{86}W_{135}W_{184}W_{83}W_{133}W_{183}W_{80}W_{131}W_{182}W_{77}W_{129}W_{181}W_{74}W_{127}W_{180}W_{71}W_{125}W_{179}W_{68}W_{123}W_{178}W_{65}W_{121}W_{177}W_{62}W_{119}W_{176}W_{59}W_{117}W_{175}W_{56}W_{115}W_{174}W_{53}W_{113}W_{173}W_{50}W_{111}W_{172}W_{47}W_{109}W_{171}W_{44}W_{107}W_{170}W_{41}W_{105}W_{169}W_{38}W_{103}W_{168}W_{35}W_{101}W_{167}W_{32}W_{99}W_{166}W_{29}W_{97}W_{165}W_{26}W_{95}W_{164}W_{23}W_{93}W_{163}W_{20}W_{91}W_{162}W_{17}W_{89}W$

	$_{161}W_{14}W_{87}W_{160}W_{11}W_{85}W_{159}W_{8}W_{83}W_{158}W_{5}W_{81}W_{157}W_{2}W_{79}W_{156}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222
79	$W_{233}W_{234}W_{235}W_{230}W_{232}W_{234}W_{227}W_{230}W_{233}W_{224}W_{228}W_{232}W_{221}W_{226}W_{231}W_{218}W_{224}W_{230}W_{215}W_{222}W_{229}W_{212}W_{220}W_{228}W_{209}W_{218}W_{227}W_{206}W_{216}W_{226}W_{203}W_{214}W_{225}W_{200}W_{212}W_{224}W_{197}W_{210}W_{223}W_{194}W_{208}W_{222}W_{191}W_{206}W_{221}W_{188}W_{204}W_{220}W_{185}W_{202}W_{219}W_{182}W_{200}W_{218}W_{179}W_{198}W_{217}W_{176}W_{196}W_{216}W_{173}W_{194}W_{215}W_{170}W_{192}W_{214}W_{167}W_{190}W_{213}W_{164}W_{188}W_{212}W_{161}W_{186}W_{211}W_{158}W_{184}W_{210}W_{155}W_{182}W_{209}W_{152}W_{180}W_{208}W_{149}W_{178}W_{207}W_{146}W_{176}W_{206}W_{143}W_{174}W_{205}W_{140}W_{172}W_{204}W_{137}W_{170}W_{203}W_{134}W_{168}W_{202}W_{131}W_{166}W_{201}W_{128}W_{164}W_{200}W_{125}W_{162}W_{199}W_{122}W_{160}W_{198}W_{119}W_{158}W_{197}W_{116}W_{156}W_{196}W_{113}W_{154}W_{195}W_{110}W_{152}W_{194}W_{107}W_{150}W_{193}W_{104}W_{148}W_{192}W_{101}W_{146}W_{191}W_{98}W_{144}W_{190}W_{95}W_{142}W_{189}W_{92}W_{140}W_{188}W_{89}W_{138}W_{187}W_{86}W_{136}W_{186}W_{83}W_{134}W_{185}W_{80}W_{132}W_{184}W_{77}W_{130}W_{183}W_{74}W_{128}W_{182}W_{71}W_{126}W_{181}W_{68}W_{124}W_{180}W_{65}W_{122}W_{179}W_{62}W_{120}W_{178}W_{59}W_{118}W_{177}W_{56}W_{116}W_{176}W_{53}W_{114}W_{175}W_{50}W_{112}W_{174}W_{47}W_{110}W_{173}W_{44}W_{108}W_{172}W_{41}W_{106}W_{171}W_{38}W_{104}W_{170}W_{35}W_{102}W_{169}W_{32}W_{100}W_{168}W_{29}W_{98}W_{167}W_{26}W_{96}W_{166}W_{23}W_{94}W_{165}W_{20}W_{92}W_{164}W_{17}W_{90}W_{163}W_{14}W_{88}W_{162}W_{11}W_{86}W_{161}W_{8}W_{84}W_{160}W_{5}W_{82}W_{159}W_{2}W_{80}W_{158}$012→0000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222222222222222222222222222222222222
80	$W_{236}W_{237}W_{238}W_{233}W_{235}W_{237}W_{230}W_{233}W_{236}W_{227}W_{231}W_{235}W_{224}W_{229}W_{234}W_{221}W_{227}W_{233}W_{218}W_{225}W_{232}W_{215}W_{223}W_{231}W_{212}W_{221}W_{230}W_{209}W_{219}W_{229}W_{206}W_{217}W_{228}W_{203}W_{215}W_{227}W_{200}W_{213}W_{226}W_{197}W_{211}W_{225}W_{194}W_{209}W_{224}W_{191}W_{207}W_{223}W_{188}W_{205}W_{222}W_{185}W_{203}W_{221}W_{182}W_{201}W_{220}W_{179}W_{199}W_{219}W_{176}W_{197}W_{218}W_{173}W_{195}W_{217}W_{170}W_{193}W_{216}W_{167}W_{191}W_{215}W_{164}W_{189}W_{214}W_{161}W_{187}W_{213}W_{158}W_{185}W_{212}W_{155}W_{183}W_{211}W_{152}W_{181}W_{210}W_{149}W_{179}W_{209}W_{146}W_{177}W_{208}W_{143}W_{175}W_{207}W_{140}W_{173}W_{206}W_{137}W_{171}W_{205}W_{134}W_{169}W_{204}W_{131}W_{167}W_{203}W_{128}W_{165}W_{202}W_{125}W_{163}W_{201}W_{122}W_{161}W_{200}W_{119}W_{159}W_{199}W_{116}W_{157}W_{198}W_{113}W_{155}W_{197}W_{110}W_{153}W_{196}W_{107}W_{151}W_{195}W_{104}W_{149}W_{194}W_{101}W_{147}W_{193}W_{98}W_{145}W_{192}W_{95}W_{143}W_{191}W_{92}W_{141}W_{190}W_{89}W_{139}W_{189}W_{86}W_{137}W_{188}W_{83}W_{135}W_{187}W_{80}W_{133}W_{186}W_{77}W_{131}W_{185}W_{74}W_{129}W_{184}W_{71}W_{127}W_{183}W_{68}W_{125}W_{182}W_{65}W_{123}W_{181}W_{62}W_{121}W_{180}W_{59}W_{119}W_{179}W_{56}W_{117}W_{178}W_{53}W_{115}W_{177}W_{50}W_{113}W_{176}W_{47}W_{111}W_{175}W_{44}W_{109}W_{174}W_{41}W_{107}W_{173}W_{38}W_{105}W_{172}W_{35}W_{103}W_{171}W_{32}W_{101}W_{170}W_{29}W_{99}W_{169}W_{26}W_{97}W_{168}W_{23}W_{95}W_{167}W_{20}W_{93}W_{166}W_{17}W_{91}W_{165}W_{14}W_{89}W_{164}W_{11}W_{87}W_{163}W_{8}W_{85}W_{162}W_{5}W_{83}W_{161}W_{2}W_{81}W_{160}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000001111111111111111111111111111111111111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222222222222222222222222222222222222
81	$W_{239}W_{240}W_{241}W_{236}W_{238}W_{240}W_{233}W_{236}W_{239}W_{230}W_{234}W_{238}W_{227}W_{232}W_{237}W_{224}W_{230}W_{236}W_{221}W_{228}W_{235}W_{218}W_{226}W_{234}W_{215}W_{224}W_{233}W_{212}W_{222}W_{232}W_{209}W_{220}W_{231}W_{206}W_{218}W_{230}W_{203}W_{216}W_{229}W_{200}W_{214}W_{228}W_{197}W_{212}W_{227}W_{194}W_{210}W_{226}W_{191}W_{208}W_{225}W_{188}W_{206}W_{224}W_{185}W_{204}W_{223}W_{182}W_{202}W_{222}W_{179}W_{200}W_{221}W_{176}W_{198}W_{220}W_{173}W_{196}W_{219}W_{170}W_{194}W_{218}W_{167}W_{192}W_{217}W_{164}W_{190}W_{216}W_{161}W_{188}W_{215}W_{158}W_{186}W_{214}W_{155}W_{184}W_{213}W_{152}W_{182}W_{212}W_{149}W_{180}W_{211}W_{146}W_{178}W_{210}W_{143}W_{176}W_{209}W_{140}W_{174}W_{208}W_{137}W_{172}W_{207}W_{134}W_{170}W_{206}W_{131}W_{168}W_{205}W_{128}W_{166}W_{204}W_{125}W_{164}W_{203}W_{122}W_{162}W_{202}W_{119}W_{160}W_{201}W_{116}W_{158}W_{200}W_{113}W_{156}W_{199}W_{110}W_{154}W_{198}W_{107}W_{152}W_{197}W_{104}W_{150}W_{196}W_{101}W_{148}W_{195}W_{98}W_{146}W_{194}W_{95}W_{144}W_{193}W_{92}W_{142}W_{192}W_{89}W_{140}W_{191}W_{86}W_{138}W_{190}W_{83}W_{136}W_{189}W_{80}W_{134}W_{188}W_{77}W_{132}W_{187}W_{74}W_{130}W_{186}W_{71}W_{128}W_{185}W_{68}W_{126}W_{184}W_{65}W_{124}W_{183}W_{62}W_{122}W_{182}W_{59}W_{120}W_{181}W_{56}W_{118}W_{180}W_{53}W_{116}W_{179}W_{50}W_{114}W_{178}W_{47}W_{112}W_{177}W_{44}W_{110}W_{176}W_{41}W_{108}W_{175}W_{38}W_{106}W_{174}W_{35}W_{104}W_{173}W_{32}W_{102}W_{172}W_{29}W_{100}W_{171}W_{26}W_{98}W_{170}W_{23}W_{96}W_{169}W_{20}W_{94}W_{168}W_{17}W_{92}W_{167}W_{14}W_{90}W_{166}W_{11}W_{88}W_{165}W_{8}W_{86}W_{164}W_{5}W_{84}W_{163}W_{2}W_{82}W_{162}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222222
82	$W_{242}W_{243}W_{244}W_{239}W_{241}W_{243}W_{236}W_{239}W_{242}W_{233}W_{237}W_{241}W_{230}W_{235}W_{240}W_{227}W_{233}W_{239}W_{224}W_{231}W_{238}W_{221}W_{229}W_{237}W_{218}W_{227}W_{236}W_{215}W_{225}W_{235}W_{212}W_{223}W_{234}W_{209}W_{221}W_{233}W_{206}W_{219}W_{232}W_{203}W_{217}W_{231}W_{200}W_{215}W_{230}W_{197}W_{213}W_{229}W_{194}W_{211}W_{228}W_{191}W_{209}W_{227}W_{188}W_{207}W_{226}W_{185}W_{205}W_{225}W_{182}W_{203}W_{224}W_{179}W_{201}W_{223}W_{176}W_{199}W_{222}W_{173}W_{197}W_{221}W_{170}W_{195}W_{220}W_{167}W_{193}W_{219}W_{164}W_{191}W_{218}W_{161}W_{189}W_{217}W_{158}W_{187}W_{216}W_{155}W_{185}W_{215}W_{152}W_{183}W_{214}W_{149}W_{181}W_{213}W_{146}W_{179}W_{212}W_{143}W_{177}W_{211}W_{140}W_{175}W_{210}W_{137}W_{173}W_{209}W_{134}W_{171}W_{208}W_{131}W_{169}W_{207}W_{128}W_{167}W_{206}W_{125}W_{165}W_{205}W_{122}W_{163}W_{204}W_{119}W_{161}W_{203}W_{116}W_{159}W_{202}W_{113}W_{157}W_{201}W_{110}W_{155}W_{200}W_{107}W_{153}W_{199}W_{104}W_{151}W_{198}W_{101}W_{149}W_{197}W_{98}W_{147}W_{196}W_{95}W_{145}W_{195}W_{92}W_{143}W_{194}W_{89}W_{141}W_{193}W_{86}W_{139}W_{192}W_{83}W_{137}W_{191}W_{80}W_{135}W_{190}W_{77}W_{133}W_{189}W_{74}W_{131}W_{188}W_{71}W_{129}W_{187}W_{68}W_{127}W_{186}W_{65}W_{125}W_{185}W_{62}W_{123}W_{184}W_{59}W_{121}W_{183}W_{56}W_{119}W_{182}W_{53}W_{117}W_{18}$

	$_{1}W_{50}W_{115}W_{180}W_{47}W_{113}W_{179}W_{44}W_{111}W_{178}W_{41}W_{109}W_{177}W_{38}W_{107}W_{176}W_{35}W_{105}W_{175}W_{32}W_{103}W_{174}W_{29}W_{101}W_{173}W_{26}W_{99}W_{172}W_{23}W_{97}W_{171}W_{20}W_{95}W_{170}W_{17}W_{93}W_{169}W_{14}W_{91}W_{168}W_{11}W_{89}W_{167}W_{8}W_{87}W_{166}W_{5}W_{85}W_{165}W_{2}W_{83}W_{164}$012→0000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222222222222222222222222222222222222222
83	$W_{245}W_{246}W_{247}W_{242}W_{244}W_{246}W_{239}W_{242}W_{245}W_{236}W_{240}W_{244}W_{233}W_{238}W_{243}W_{230}W_{236}W_{242}W_{227}W_{234}W_{241}W_{224}W_{232}W_{240}W_{221}W_{230}W_{239}W_{218}W_{228}W_{238}W_{215}W_{226}W_{237}W_{212}W_{224}W_{236}W_{209}W_{222}W_{235}W_{206}W_{220}W_{234}W_{203}W_{218}W_{233}W_{200}W_{216}W_{232}W_{197}W_{214}W_{231}W_{194}W_{212}W_{230}W_{191}W_{210}W_{229}W_{188}W_{208}W_{228}W_{185}W_{206}W_{227}W_{182}W_{204}W_{226}W_{179}W_{202}W_{225}W_{176}W_{200}W_{224}W_{173}W_{198}W_{223}W_{170}W_{196}W_{222}W_{167}W_{194}W_{221}W_{164}W_{192}W_{220}W_{161}W_{190}W_{219}W_{158}W_{188}W_{218}W_{155}W_{186}W_{217}W_{152}W_{184}W_{216}W_{149}W_{182}W_{215}W_{146}W_{180}W_{214}W_{143}W_{178}W_{213}W_{140}W_{176}W_{212}W_{137}W_{174}W_{211}W_{134}W_{172}W_{210}W_{131}W_{170}W_{209}W_{128}W_{168}W_{208}W_{125}W_{166}W_{207}W_{122}W_{164}W_{206}W_{119}W_{162}W_{205}W_{116}W_{160}W_{204}W_{113}W_{158}W_{203}W_{110}W_{156}W_{202}W_{107}W_{154}W_{201}W_{104}W_{152}W_{200}W_{101}W_{150}W_{199}W_{98}W_{148}W_{198}W_{95}W_{146}W_{197}W_{92}W_{144}W_{196}W_{89}W_{142}W_{195}W_{86}W_{140}W_{194}W_{83}W_{138}W_{193}W_{80}W_{136}W_{192}W_{77}W_{134}W_{191}W_{74}W_{132}W_{190}W_{71}W_{130}W_{189}W_{68}W_{128}W_{188}W_{65}W_{126}W_{187}W_{62}W_{124}W_{186}W_{59}W_{122}W_{185}W_{56}W_{120}W_{184}W_{53}W_{118}W_{183}W_{50}W_{116}W_{182}W_{47}W_{114}W_{181}W_{44}W_{112}W_{180}W_{41}W_{110}W_{179}W_{38}W_{108}W_{178}W_{35}W_{106}W_{177}W_{32}W_{104}W_{176}W_{29}W_{102}W_{175}W_{26}W_{100}W_{174}W_{23}W_{98}W_{173}W_{20}W_{96}W_{172}W_{17}W_{94}W_{171}W_{14}W_{92}W_{170}W_{11}W_{90}W_{169}W_{8}W_{88}W_{168}W_{5}W_{86}W_{167}W_{2}W_{84}W_{166}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000001111111111111111111111111111111111111111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222222222222222222222222222222222222222
84	$W_{248}W_{249}W_{250}W_{245}W_{247}W_{249}W_{242}W_{245}W_{248}W_{239}W_{243}W_{247}W_{236}W_{241}W_{246}W_{233}W_{239}W_{245}W_{230}W_{237}W_{244}W_{227}W_{235}W_{243}W_{224}W_{233}W_{242}W_{221}W_{231}W_{241}W_{218}W_{229}W_{240}W_{215}W_{227}W_{239}W_{212}W_{225}W_{238}W_{209}W_{223}W_{237}W_{206}W_{221}W_{236}W_{203}W_{219}W_{235}W_{200}W_{217}W_{234}W_{197}W_{215}W_{233}W_{194}W_{213}W_{232}W_{191}W_{211}W_{231}W_{188}W_{209}W_{230}W_{185}W_{207}W_{229}W_{182}W_{205}W_{228}W_{179}W_{203}W_{227}W_{176}W_{201}W_{226}W_{173}W_{199}W_{225}W_{170}W_{197}W_{224}W_{167}W_{195}W_{223}W_{164}W_{193}W_{222}W_{161}W_{191}W_{221}W_{158}W_{189}W_{220}W_{155}W_{187}W_{219}W_{152}W_{185}W_{218}W_{149}W_{183}W_{217}W_{146}W_{181}W_{216}W_{143}W_{179}W_{215}W_{140}W_{177}W_{214}W_{137}W_{175}W_{213}W_{134}W_{173}W_{212}W_{131}W_{171}W_{211}W_{128}W_{169}W_{210}W_{125}W_{167}W_{209}W_{122}W_{165}W_{208}W_{119}W_{163}W_{207}W_{116}W_{161}W_{206}W_{113}W_{159}W_{205}W_{110}W_{157}W_{204}W_{107}W_{155}W_{203}W_{104}W_{153}W_{202}W_{101}W_{151}W_{201}W_{98}W_{149}W_{200}W_{95}W_{147}W_{199}W_{92}W_{145}W_{198}W_{89}W_{143}W_{197}W_{86}W_{141}W_{196}W_{83}W_{139}W_{195}W_{80}W_{137}W_{194}W_{77}W_{135}W_{193}W_{74}W_{133}W_{192}W_{71}W_{131}W_{191}W_{68}W_{129}W_{190}W_{65}W_{127}W_{189}W_{62}W_{125}W_{188}W_{59}W_{123}W_{187}W_{56}W_{121}W_{186}W_{53}W_{119}W_{185}W_{50}W_{117}W_{184}W_{47}W_{115}W_{183}W_{44}W_{113}W_{182}W_{41}W_{111}W_{181}W_{38}W_{109}W_{180}W_{35}W_{107}W_{179}W_{32}W_{105}W_{178}W_{29}W_{103}W_{177}W_{26}W_{101}W_{176}W_{23}W_{99}W_{175}W_{20}W_{97}W_{174}W_{17}W_{95}W_{173}W_{14}W_{93}W_{172}W_{11}W_{91}W_{171}W_{8}W_{89}W_{170}W_{5}W_{87}W_{169}W_{2}W_{85}W_{168}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222222222
85	$W_{251}W_{252}W_{253}W_{248}W_{250}W_{252}W_{245}W_{248}W_{251}W_{242}W_{246}W_{250}W_{239}W_{244}W_{249}W_{236}W_{242}W_{248}W_{233}W_{240}W_{247}W_{230}W_{238}W_{246}W_{227}W_{236}W_{245}W_{224}W_{234}W_{244}W_{221}W_{232}W_{243}W_{218}W_{230}W_{242}W_{215}W_{228}W_{241}W_{212}W_{226}W_{240}W_{209}W_{224}W_{239}W_{206}W_{222}W_{238}W_{203}W_{220}W_{237}W_{200}W_{218}W_{236}W_{197}W_{216}W_{235}W_{194}W_{214}W_{234}W_{191}W_{212}W_{233}W_{188}W_{210}W_{232}W_{185}W_{208}W_{231}W_{182}W_{206}W_{230}W_{179}W_{204}W_{229}W_{176}W_{202}W_{228}W_{173}W_{200}W_{227}W_{170}W_{198}W_{226}W_{167}W_{196}W_{225}W_{164}W_{194}W_{224}W_{161}W_{192}W_{223}W_{158}W_{190}W_{222}W_{155}W_{188}W_{221}W_{152}W_{186}W_{220}W_{149}W_{184}W_{219}W_{146}W_{182}W_{218}W_{143}W_{180}W_{217}W_{140}W_{178}W_{216}W_{137}W_{176}W_{215}W_{134}W_{174}W_{214}W_{131}W_{172}W_{213}W_{128}W_{170}W_{212}W_{125}W_{168}W_{211}W_{122}W_{166}W_{210}W_{119}W_{164}W_{209}W_{116}W_{162}W_{208}W_{113}W_{160}W_{207}W_{110}W_{158}W_{206}W_{107}W_{156}W_{205}W_{104}W_{154}W_{204}W_{101}W_{152}W_{203}W_{98}W_{150}W_{202}W_{95}W_{148}W_{201}W_{92}W_{146}W_{200}W_{89}W_{144}W_{199}W_{86}W_{142}W_{198}W_{83}W_{140}W_{197}W_{80}W_{138}W_{196}W_{77}W_{136}W_{195}W_{74}W_{134}W_{194}W_{71}W_{132}W_{193}W_{68}W_{130}W_{192}W_{65}W_{128}W_{191}W_{62}W_{126}W_{190}W_{59}W_{124}W_{189}W_{56}W_{122}W_{188}W_{53}W_{120}W_{187}W_{50}W_{118}W_{186}W_{47}W_{116}W_{185}W_{44}W_{114}W_{184}W_{41}W_{112}W_{183}W_{38}W_{110}W_{182}W_{35}W_{108}W_{181}W_{32}W_{106}W_{180}W_{29}W_{104}W_{179}W_{26}W_{102}W_{178}W_{23}W_{100}W_{177}W_{20}W_{98}W_{176}W_{17}W_{96}W_{175}W_{14}W_{94}W_{174}W_{11}W_{92}W_{173}W_{8}W_{90}W_{172}W_{5}W_{88}W_{171}W_{2}W_{86}W_{170}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222222222222222
86	$W_{254}W_{255}W_{256}W_{251}W_{253}W_{255}W_{248}W_{251}W_{254}W_{245}W_{249}W_{253}W_{242}W_{247}W_{252}W_{239}W_{245}W_{251}W_{236}W_{243}W_{250}W_{233}W_{241}W_{249}W_{230}W_{239}W_{248}W_{227}W_{237}W_{247}W_{224}W_{235}W_{246}W_{221}W_{233}W_{245}W_{218}W_{231}W_{244}W_{215}W_{229}W_{243}W_{212}W_{227}W_{242}W_{209}W_{225}W_{241}W_{206}W_{223}W_{240}W_{203}W_{221}W_{239}W_{200}W_{219}W_{238}W_{197}W_{217}W_{237}W_{194}W_{215}W_{236}W_{191}W_{213}W_{235}W_{188}W_{211}W_{234}W_{185}W_{209}W_{233}W_{182}W_{207}W_{232}W_{179}W_{205}W_{231}W_{176}W_{203}W_{230}W_{173}W_{201}W_{229}W_{170}W_{199}W_{228}W_{167}W_{197}W_{227}W_{164}W_{195}W_{226}W_{161}W_{193}W_{225}W_{158}W_{191}W_{224}W_{155}W_{189}W_{223}W_{152}W_{187}W_{222}$

	$W_{149}W_{185}W_{221}W_{146}W_{183}W_{220}W_{143}W_{181}W_{219}W_{140}W_{179}W_{218}W_{137}W_{177}W_{217}W_{134}W_{175}W_{216}W_{131}W_{173}W_{215}W_{128}W_{171}W_{214}W_{125}W_{169}W_{213}W_{122}W_{167}W_{212}W_{119}W_{165}W_{211}W_{116}W_{163}W_{210}W_{113}W_{161}W_{209}W_{110}W_{159}W_{208}W_{107}W_{157}W_{207}W_{104}W_{155}W_{206}W_{101}W_{153}W_{205}W_{98}W_{151}W_{204}W_{95}W_{149}W_{203}W_{92}W_{147}W_{202}W_{89}W_{145}W_{201}W_{86}W_{143}W_{200}W_{83}W_{141}W_{199}W_{80}W_{139}W_{198}W_{77}W_{137}W_{197}W_{74}W_{135}W_{196}W_{71}W_{133}W_{195}W_{68}W_{131}W_{194}W_{65}W_{129}W_{193}W_{62}W_{127}W_{192}W_{59}W_{125}W_{191}W_{56}W_{123}W_{190}W_{53}W_{121}W_{189}W_{50}W_{119}W_{188}W_{47}W_{117}W_{187}W_{44}W_{115}W_{186}W_{41}W_{113}W_{185}W_{38}W_{111}W_{184}W_{35}W_{109}W_{183}W_{32}W_{107}W_{182}W_{29}W_{105}W_{181}W_{26}W_{103}W_{180}W_{23}W_{101}W_{179}W_{20}W_{99}W_{178}W_{17}W_{97}W_{177}W_{14}W_{95}W_{176}W_{11}W_{93}W_{175}W_{8}W_{91}W_{174}W_{5}W_{89}W_{173}W_{2}W_{87}W_{172}012\rightarrow$ 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
87	$W_{257}W_{258}W_{259}W_{254}W_{256}W_{258}W_{251}W_{254}W_{257}W_{248}W_{252}W_{256}W_{245}W_{250}W_{255}W_{242}W_{248}W_{254}W_{239}W_{246}W_{253}W_{236}W_{244}W_{252}W_{233}W_{242}W_{251}W_{230}W_{240}W_{250}W_{227}W_{238}W_{249}W_{224}W_{236}W_{248}W_{221}W_{234}W_{247}W_{218}W_{232}W_{246}W_{215}W_{230}W_{245}W_{212}W_{228}W_{244}W_{209}W_{226}W_{243}W_{206}W_{224}W_{242}W_{203}W_{222}W_{241}W_{200}W_{220}W_{240}W_{197}W_{218}W_{239}W_{194}W_{216}W_{238}W_{191}W_{214}W_{237}W_{188}W_{212}W_{236}W_{185}W_{210}W_{235}W_{182}W_{208}W_{234}W_{179}W_{206}W_{233}W_{176}W_{204}W_{232}W_{173}W_{202}W_{231}W_{170}W_{200}W_{230}W_{167}W_{198}W_{229}W_{164}W_{196}W_{228}W_{161}W_{194}W_{227}W_{158}W_{192}W_{226}W_{155}W_{190}W_{225}W_{152}W_{188}W_{224}W_{149}W_{186}W_{223}W_{146}W_{184}W_{222}W_{143}W_{182}W_{221}W_{140}W_{180}W_{220}W_{137}W_{178}W_{219}W_{134}W_{176}W_{218}W_{131}W_{174}W_{217}W_{128}W_{172}W_{216}W_{125}W_{170}W_{215}W_{122}W_{168}W_{214}W_{119}W_{166}W_{213}W_{116}W_{164}W_{212}W_{113}W_{162}W_{211}W_{110}W_{160}W_{210}W_{107}W_{158}W_{209}W_{104}W_{156}W_{208}W_{101}W_{154}W_{207}W_{98}W_{152}W_{206}W_{95}W_{150}W_{205}W_{92}W_{148}W_{204}W_{89}W_{146}W_{203}W_{86}W_{144}W_{202}W_{83}W_{142}W_{201}W_{80}W_{140}W_{200}W_{77}W_{138}W_{199}W_{74}W_{136}W_{198}W_{71}W_{134}W_{197}W_{68}W_{132}W_{196}W_{65}W_{130}W_{195}W_{62}W_{128}W_{194}W_{59}W_{126}W_{193}W_{56}W_{124}W_{192}W_{53}W_{122}W_{191}W_{50}W_{120}W_{190}W_{47}W_{118}W_{189}W_{44}W_{116}W_{188}W_{41}W_{114}W_{187}W_{38}W_{112}W_{186}W_{35}W_{110}W_{185}W_{32}W_{108}W_{184}W_{29}W_{106}W_{183}W_{26}W_{104}W_{182}W_{23}W_{102}W_{181}W_{20}W_{100}W_{180}W_{17}W_{98}W_{179}W_{14}W_{96}W_{178}W_{11}W_{94}W_{177}W_{8}W_{92}W_{176}W_{5}W_{90}W_{175}W_{2}W_{88}W_{174}012\rightarrow$ 00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
88	$W_{260}W_{261}W_{262}W_{257}W_{259}W_{261}W_{254}W_{257}W_{260}W_{251}W_{255}W_{259}W_{248}W_{253}W_{258}W_{245}W_{251}W_{257}W_{242}W_{249}W_{256}W_{239}W_{247}W_{255}W_{236}W_{245}W_{254}W_{233}W_{243}W_{253}W_{230}W_{241}W_{252}W_{227}W_{239}W_{251}W_{224}W_{237}W_{250}W_{221}W_{235}W_{249}W_{218}W_{233}W_{248}W_{215}W_{231}W_{247}W_{212}W_{229}W_{246}W_{209}W_{227}W_{245}W_{206}W_{225}W_{244}W_{203}W_{223}W_{243}W_{200}W_{221}W_{242}W_{197}W_{219}W_{241}W_{194}W_{217}W_{240}W_{191}W_{215}W_{239}W_{188}W_{213}W_{238}W_{185}W_{211}W_{237}W_{182}W_{209}W_{236}W_{179}W_{207}W_{235}W_{176}W_{205}W_{234}W_{173}W_{203}W_{233}W_{170}W_{201}W_{232}W_{167}W_{199}W_{231}W_{164}W_{197}W_{230}W_{161}W_{195}W_{229}W_{158}W_{193}W_{228}W_{155}W_{191}W_{227}W_{152}W_{189}W_{226}W_{149}W_{187}W_{225}W_{146}W_{185}W_{224}W_{143}W_{183}W_{223}W_{140}W_{181}W_{222}W_{137}W_{179}W_{221}W_{134}W_{177}W_{220}W_{131}W_{175}W_{219}W_{128}W_{173}W_{218}W_{125}W_{171}W_{217}W_{122}W_{169}W_{216}W_{119}W_{167}W_{215}W_{116}W_{165}W_{214}W_{113}W_{163}W_{213}W_{110}W_{161}W_{212}W_{107}W_{159}W_{211}W_{104}W_{157}W_{210}W_{101}W_{155}W_{209}W_{98}W_{153}W_{208}W_{95}W_{151}W_{207}W_{92}W_{149}W_{206}W_{89}W_{147}W_{205}W_{86}W_{145}W_{204}W_{83}W_{143}W_{203}W_{80}W_{141}W_{202}W_{77}W_{139}W_{201}W_{74}W_{137}W_{200}W_{71}W_{135}W_{199}W_{68}W_{133}W_{198}W_{65}W_{131}W_{197}W_{62}W_{129}W_{196}W_{59}W_{127}W_{195}W_{56}W_{125}W_{194}W_{53}W_{123}W_{193}W_{50}W_{121}W_{192}W_{47}W_{119}W_{191}W_{44}W_{117}W_{190}W_{41}W_{115}W_{189}W_{38}W_{113}W_{188}W_{35}W_{111}W_{187}W_{32}W_{109}W_{186}W_{29}W_{107}W_{185}W_{26}W_{105}W_{184}W_{23}W_{103}W_{183}W_{20}W_{101}W_{182}W_{17}W_{99}W_{181}W_{14}W_{97}W_{180}W_{11}W_{95}W_{179}W_{8}W_{93}W_{178}W_{5}W_{91}W_{177}W_{2}W_{89}W_{176}012\rightarrow$ 0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222222222222222222222222222222222222222222
89	$W_{263}W_{264}W_{265}W_{260}W_{262}W_{264}W_{257}W_{260}W_{263}W_{254}W_{258}W_{262}W_{251}W_{256}W_{261}W_{248}W_{254}W_{260}W_{245}W_{252}W_{259}W_{242}W_{250}W_{258}W_{239}W_{248}W_{257}W_{236}W_{246}W_{256}W_{233}W_{244}W_{255}W_{230}W_{242}W_{254}W_{227}W_{240}W_{253}W_{224}W_{238}W_{252}W_{221}W_{236}W_{251}W_{218}W_{234}W_{250}W_{215}W_{232}W_{249}W_{212}W_{230}W_{248}W_{209}W_{228}W_{247}W_{206}W_{226}W_{246}W_{203}W_{224}W_{245}W_{200}W_{222}W_{244}W_{197}W_{220}W_{243}W_{194}W_{218}W_{242}W_{191}W_{216}W_{241}W_{188}W_{214}W_{240}W_{185}W_{212}W_{239}W_{182}W_{210}W_{238}W_{179}W_{208}W_{237}W_{176}W_{206}W_{236}W_{173}W_{204}W_{235}W_{170}W_{202}W_{234}W_{167}W_{200}W_{233}W_{164}W_{198}W_{232}W_{161}W_{196}W_{231}W_{158}W_{194}W_{230}W_{155}W_{192}W_{229}W_{152}W_{190}W_{228}W_{149}W_{188}W_{227}W_{146}W_{186}W_{226}W_{143}W_{184}W_{225}W_{140}W_{182}W_{224}W_{137}W_{180}W_{223}W_{134}W_{178}W_{222}W_{131}W_{176}W_{221}W_{128}W_{174}W_{220}W_{125}W_{172}W_{219}W_{122}W_{170}W_{218}W_{119}W_{168}W_{217}W_{116}W_{166}W_{216}W_{113}W_{164}W_{215}W_{110}W_{162}W_{214}W_{107}W_{160}W_{213}W_{104}W_{158}W_{212}W_{101}W_{156}W_{211}W_{98}W_{154}W_{210}W_{95}W_{152}W_{209}W_{92}W_{150}W_{208}W_{89}W_{148}W_{207}W_{86}W_{146}W_{206}W_{83}W_{144}W_{205}W_{80}W_{142}W_{204}W_{77}W_{140}W_{203}W_{74}W_{138}W_{202}W_{71}W_{136}W_{201}W_{68}W_{134}W_{200}W_{65}W_{132}W_{199}W_{62}W_{130}W_{198}W_{59}W_{128}W_{197}W_{56}W_{126}W_{196}W_{53}W_{124}W_{195}W_{50}W_{122}W_{194}W_{47}W_{120}W_{193}W_{44}W_{118}W_{192}W_{41}W_{116}W_{191}W_{38}W_{114}W_{190}W_{35}W_{112}W_{189}W_{32}W_{110}W_{188}W_{29}W_{108}W_{187}W_{26}W_{106}W_{186}W_{23}W_{104}W_{185}W_{20}W_{102}W_{184}W_{17}W_{100}W_{183}W_{14}W_{98}W_{182}W_{11}W_{96}W_{181}W_{8}W_{94}W_{180}W_{5}W_{92}W_{179}W_{2}W_{90}W_{178}012\rightarrow$ 0000000000000000000000000000000000000000000000000000000000

	0000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
90	$W_{266}W_{267}W_{268}W_{263}W_{265}W_{267}W_{260}W_{263}W_{266}W_{257}W_{261}W_{265}W_{254}W_{259}W_{264}W_{251}W_{257}W_{263}W_{248}W_{255}W_{262}W_{245}W_{253}W_{261}W_{242}W_{251}W_{260}W_{239}W_{249}W_{259}W_{236}W_{247}W_{258}W_{233}W_{245}W_{257}W_{230}W_{243}W_{256}W_{227}W_{241}W_{255}W_{224}W_{239}W_{254}W_{221}W_{237}W_{253}W_{218}W_{235}W_{252}W_{215}W_{233}W_{251}W_{212}W_{231}W_{250}W_{209}W_{229}W_{249}W_{206}W_{227}W_{248}W_{203}W_{225}W_{247}W_{200}W_{223}W_{246}W_{197}W_{221}W_{245}W_{194}W_{219}W_{244}W_{191}W_{217}W_{243}W_{188}W_{215}W_{242}W_{185}W_{213}W_{241}W_{182}W_{211}W_{240}W_{179}W_{209}W_{239}W_{176}W_{207}W_{238}W_{173}W_{205}W_{237}W_{170}W_{203}W_{236}W_{167}W_{201}W_{235}W_{164}W_{199}W_{234}W_{161}W_{197}W_{233}W_{158}W_{195}W_{232}W_{155}W_{193}W_{231}W_{152}W_{191}W_{230}W_{149}W_{189}W_{229}W_{146}W_{187}W_{228}W_{143}W_{185}W_{227}W_{140}W_{183}W_{226}W_{137}W_{181}W_{225}W_{134}W_{179}W_{224}W_{131}W_{177}W_{223}W_{128}W_{175}W_{222}W_{125}W_{173}W_{221}W_{122}W_{171}W_{220}W_{119}W_{169}W_{219}W_{116}W_{167}W_{218}W_{113}W_{165}W_{217}W_{110}W_{163}W_{216}W_{107}W_{161}W_{215}W_{104}W_{159}W_{214}W_{101}W_{157}W_{213}W_{98}W_{155}W_{212}W_{95}W_{153}W_{211}W_{92}W_{151}W_{210}W_{89}W_{149}W_{209}W_{86}W_{147}W_{208}W_{83}W_{145}W_{207}W_{80}W_{143}W_{206}W_{77}W_{141}W_{205}W_{74}W_{139}W_{204}W_{71}W_{137}W_{203}W_{68}W_{135}W_{202}W_{65}W_{133}W_{201}W_{62}W_{131}W_{200}W_{59}W_{129}W_{199}W_{56}W_{127}W_{198}W_{53}W_{125}W_{197}W_{50}W_{123}W_{196}W_{47}W_{121}W_{195}W_{44}W_{119}W_{194}W_{41}W_{117}W_{193}W_{38}W_{115}W_{192}W_{35}W_{113}W_{191}W_{32}W_{111}W_{190}W_{29}W_{109}W_{189}W_{26}W_{107}W_{188}W_{23}W_{105}W_{187}W_{20}W_{103}W_{186}W_{17}W_{101}W_{185}W_{14}W_{99}W_{184}W_{11}W_{97}W_{183}W_{8}W_{95}W_{182}W_{5}W_{93}W_{181}W_{2}W_{91}W_{180}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
91	$W_{269}W_{270}W_{271}W_{266}W_{268}W_{270}W_{263}W_{266}W_{269}W_{260}W_{264}W_{268}W_{257}W_{262}W_{267}W_{254}W_{260}W_{266}W_{251}W_{258}W_{265}W_{248}W_{256}W_{264}W_{245}W_{254}W_{263}W_{242}W_{252}W_{262}W_{239}W_{250}W_{261}W_{236}W_{248}W_{260}W_{233}W_{246}W_{259}W_{230}W_{244}W_{258}W_{227}W_{242}W_{257}W_{224}W_{240}W_{256}W_{221}W_{238}W_{255}W_{218}W_{236}W_{254}W_{215}W_{234}W_{253}W_{212}W_{232}W_{252}W_{209}W_{230}W_{251}W_{206}W_{228}W_{250}W_{203}W_{226}W_{249}W_{200}W_{224}W_{248}W_{197}W_{222}W_{247}W_{194}W_{220}W_{246}W_{191}W_{218}W_{245}W_{188}W_{216}W_{244}W_{185}W_{214}W_{243}W_{182}W_{212}W_{242}W_{179}W_{210}W_{241}W_{176}W_{208}W_{240}W_{173}W_{206}W_{239}W_{170}W_{204}W_{238}W_{167}W_{202}W_{237}W_{164}W_{200}W_{236}W_{161}W_{198}W_{235}W_{158}W_{196}W_{234}W_{155}W_{194}W_{233}W_{152}W_{192}W_{232}W_{149}W_{190}W_{231}W_{146}W_{188}W_{230}W_{143}W_{186}W_{229}W_{140}W_{184}W_{228}W_{137}W_{182}W_{227}W_{134}W_{180}W_{226}W_{131}W_{178}W_{225}W_{128}W_{176}W_{224}W_{125}W_{174}W_{223}W_{122}W_{172}W_{222}W_{119}W_{170}W_{221}W_{116}W_{168}W_{220}W_{113}W_{166}W_{219}W_{110}W_{164}W_{218}W_{107}W_{162}W_{217}W_{104}W_{160}W_{216}W_{101}W_{158}W_{215}W_{98}W_{156}W_{214}W_{95}W_{154}W_{213}W_{92}W_{152}W_{212}W_{89}W_{150}W_{211}W_{86}W_{148}W_{210}W_{83}W_{146}W_{209}W_{80}W_{144}W_{208}W_{77}W_{142}W_{207}W_{74}W_{140}W_{206}W_{71}W_{138}W_{205}W_{68}W_{136}W_{204}W_{65}W_{134}W_{203}W_{62}W_{132}W_{202}W_{59}W_{130}W_{201}W_{56}W_{128}W_{200}W_{53}W_{126}W_{199}W_{50}W_{124}W_{198}W_{47}W_{122}W_{197}W_{44}W_{120}W_{196}W_{41}W_{118}W_{195}W_{38}W_{116}W_{194}W_{35}W_{114}W_{193}W_{32}W_{112}W_{192}W_{29}W_{110}W_{191}W_{26}W_{108}W_{190}W_{23}W_{106}W_{189}W_{20}W_{104}W_{188}W_{17}W_{102}W_{187}W_{14}W_{100}W_{186}W_{11}W_{98}W_{185}W_{8}W_{96}W_{184}W_{5}W_{94}W_{183}W_{2}W_{92}W_{182}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
92	$W_{272}W_{273}W_{274}W_{269}W_{271}W_{273}W_{266}W_{269}W_{272}W_{263}W_{267}W_{271}W_{260}W_{265}W_{270}W_{257}W_{263}W_{269}W_{254}W_{261}W_{268}W_{251}W_{259}W_{267}W_{248}W_{257}W_{266}W_{245}W_{255}W_{265}W_{242}W_{253}W_{264}W_{239}W_{251}W_{263}W_{236}W_{249}W_{262}W_{233}W_{247}W_{261}W_{230}W_{245}W_{260}W_{227}W_{243}W_{259}W_{224}W_{241}W_{258}W_{221}W_{239}W_{257}W_{218}W_{237}W_{256}W_{215}W_{235}W_{255}W_{212}W_{233}W_{254}W_{209}W_{231}W_{253}W_{206}W_{229}W_{252}W_{203}W_{227}W_{251}W_{200}W_{225}W_{250}W_{197}W_{223}W_{249}W_{194}W_{221}W_{248}W_{191}W_{219}W_{247}W_{188}W_{217}W_{246}W_{185}W_{215}W_{245}W_{182}W_{213}W_{244}W_{179}W_{211}W_{243}W_{176}W_{209}W_{242}W_{173}W_{207}W_{241}W_{170}W_{205}W_{240}W_{167}W_{203}W_{239}W_{164}W_{201}W_{238}W_{161}W_{199}W_{237}W_{158}W_{197}W_{236}W_{155}W_{195}W_{235}W_{152}W_{193}W_{234}W_{149}W_{191}W_{233}W_{146}W_{189}W_{232}W_{143}W_{187}W_{231}W_{140}W_{185}W_{230}W_{137}W_{183}W_{229}W_{134}W_{181}W_{228}W_{131}W_{179}W_{227}W_{128}W_{177}W_{226}W_{125}W_{175}W_{225}W_{122}W_{173}W_{224}W_{119}W_{171}W_{223}W_{116}W_{169}W_{222}W_{113}W_{167}W_{221}W_{110}W_{165}W_{220}W_{107}W_{163}W_{219}W_{104}W_{161}W_{218}W_{101}W_{159}W_{217}W_{98}W_{157}W_{216}W_{95}W_{155}W_{215}W_{92}W_{153}W_{214}W_{89}W_{151}W_{213}W_{86}W_{149}W_{212}W_{83}W_{147}W_{211}W_{80}W_{145}W_{210}W_{77}W_{143}W_{209}W_{74}W_{141}W_{208}W_{71}W_{139}W_{207}W_{68}W_{137}W_{206}W_{65}W_{135}W_{205}W_{62}W_{133}W_{204}W_{59}W_{131}W_{203}W_{56}W_{129}W_{202}W_{53}W_{127}W_{201}W_{50}W_{125}W_{200}W_{47}W_{123}W_{199}W_{44}W_{121}W_{198}W_{41}W_{119}W_{197}W_{38}W_{117}W_{196}W_{35}W_{115}W_{195}W_{32}W_{113}W_{194}W_{29}W_{111}W_{193}W_{26}W_{109}W_{192}W_{23}W_{107}W_{191}W_{20}W_{105}W_{190}W_{17}W_{103}W_{189}W_{14}W_{101}W_{188}W_{11}W_{99}W_{187}W_{8}W_{97}W_{186}W_{5}W_{95}W_{185}W_{2}W_{93}W_{184}$012→00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
93	$W_{275}W_{276}W_{277}W_{272}W_{274}W_{276}W_{269}W_{272}W_{275}W_{266}W_{270}W_{274}W_{263}W_{268}W_{273}W_{260}W_{266}W_{272}W_{257}W_{264}W_{271}W_{254}W_{262}W_{270}W_{251}W_{260}W_{269}W_{248}W_{258}W_{268}W_{245}W_{256}W_{267}W_{242}W_{254}W_{266}W_{239}W_{252}W_{265}W_{236}W_{250}W_{264}W_{233}W_{248}W_{263}W_{230}W_{246}W_{262}W_{227}W_{244}W_{261}W_{224}W_{242}W_{260}W_{221}W_{240}W_{259}W_{218}W_{238}W_{258}W_{215}W_{236}W_{257}W_{212}W_{234}W_{256}W_{209}W_{232}W_{255}W_{206}W_{230}W_{254}W_{203}W_{228}W_{253}W_{200}W_{226}W_{252}W_{197}W_{224}W_{251}W_{194}W_{222}W_{250}$

	$W_{191}W_{220}W_{249}W_{188}W_{218}W_{248}W_{185}W_{216}W_{247}W_{182}W_{214}W_{246}W_{179}W_{212}W_{245}W_{176}W_{210}W_{244}W_{173}W_{208}W_{243}W_{170}W_{206}W_{242}W_{167}W_{204}W_{241}W_{164}W_{202}W_{240}W_{161}W_{200}W_{239}W_{158}W_{198}W_{238}W_{155}W_{196}W_{237}W_{152}W_{194}W_{236}W_{149}W_{192}W_{235}W_{146}W_{190}W_{234}W_{143}W_{188}W_{233}W_{140}W_{186}W_{232}W_{137}W_{184}W_{231}W_{134}W_{182}W_{230}W_{131}W_{180}W_{229}W_{128}W_{178}W_{228}W_{125}W_{176}W_{227}W_{122}W_{174}W_{226}W_{119}W_{172}W_{225}W_{116}W_{170}W_{224}W_{113}W_{168}W_{223}W_{110}W_{166}W_{222}W_{107}W_{164}W_{221}W_{104}W_{162}W_{220}W_{101}W_{160}W_{219}W_{98}W_{158}W_{218}W_{95}W_{156}W_{217}W_{92}W_{154}W_{216}W_{89}W_{152}W_{215}W_{86}W_{150}W_{214}W_{83}W_{148}W_{213}W_{80}W_{146}W_{212}W_{77}W_{144}W_{211}W_{74}W_{142}W_{210}W_{71}W_{140}W_{209}W_{68}W_{138}W_{208}W_{65}W_{136}W_{207}W_{62}W_{134}W_{206}W_{59}W_{132}W_{205}W_{56}W_{130}W_{204}W_{53}W_{128}W_{203}W_{50}W_{126}W_{202}W_{47}W_{124}W_{201}W_{44}W_{122}W_{200}W_{41}W_{120}W_{199}W_{38}W_{118}W_{198}W_{35}W_{116}W_{197}W_{32}W_{114}W_{196}W_{29}W_{112}W_{195}W_{26}W_{110}W_{194}W_{23}W_{108}W_{193}W_{20}W_{106}W_{192}W_{17}W_{104}W_{191}W_{14}W_{102}W_{190}W_{11}W_{100}W_{189}W_{8}W_{98}W_{188}W_{5}W_{96}W_{187}W_{2}W_{94}W_{186}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000001111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
94	$W_{278}W_{279}W_{280}W_{275}W_{277}W_{279}W_{272}W_{275}W_{278}W_{269}W_{273}W_{277}W_{266}W_{271}W_{276}W_{263}W_{269}W_{275}W_{260}W_{267}W_{274}W_{257}W_{265}W_{273}W_{254}W_{263}W_{272}W_{251}W_{261}W_{271}W_{248}W_{259}W_{270}W_{245}W_{257}W_{269}W_{242}W_{255}W_{268}W_{239}W_{253}W_{267}W_{236}W_{251}W_{266}W_{233}W_{249}W_{265}W_{230}W_{247}W_{264}W_{227}W_{245}W_{263}W_{224}W_{243}W_{262}W_{221}W_{241}W_{261}W_{218}W_{239}W_{260}W_{215}W_{237}W_{259}W_{212}W_{235}W_{258}W_{209}W_{233}W_{257}W_{206}W_{231}W_{256}W_{203}W_{229}W_{255}W_{200}W_{227}W_{254}W_{197}W_{225}W_{253}W_{194}W_{223}W_{252}W_{191}W_{221}W_{251}W_{188}W_{219}W_{250}W_{185}W_{217}W_{249}W_{182}W_{215}W_{248}W_{179}W_{213}W_{247}W_{176}W_{211}W_{246}W_{173}W_{209}W_{245}W_{170}W_{207}W_{244}W_{167}W_{205}W_{243}W_{164}W_{203}W_{242}W_{161}W_{201}W_{241}W_{158}W_{199}W_{240}W_{155}W_{197}W_{239}W_{152}W_{195}W_{238}W_{149}W_{193}W_{237}W_{146}W_{191}W_{236}W_{143}W_{189}W_{235}W_{140}W_{187}W_{234}W_{137}W_{185}W_{233}W_{134}W_{183}W_{232}W_{131}W_{181}W_{231}W_{128}W_{179}W_{230}W_{125}W_{177}W_{229}W_{122}W_{175}W_{228}W_{119}W_{173}W_{227}W_{116}W_{171}W_{226}W_{113}W_{169}W_{225}W_{110}W_{167}W_{224}W_{107}W_{165}W_{223}W_{104}W_{163}W_{222}W_{101}W_{161}W_{221}W_{98}W_{159}W_{220}W_{95}W_{157}W_{219}W_{92}W_{155}W_{218}W_{89}W_{153}W_{217}W_{86}W_{151}W_{216}W_{83}W_{149}W_{215}W_{80}W_{147}W_{214}W_{77}W_{145}W_{213}W_{74}W_{143}W_{212}W_{71}W_{141}W_{211}W_{68}W_{139}W_{210}W_{65}W_{137}W_{209}W_{62}W_{135}W_{208}W_{59}W_{133}W_{207}W_{56}W_{131}W_{206}W_{53}W_{129}W_{205}W_{50}W_{127}W_{204}W_{47}W_{125}W_{203}W_{44}W_{123}W_{202}W_{41}W_{121}W_{201}W_{38}W_{119}W_{200}W_{35}W_{117}W_{199}W_{32}W_{115}W_{198}W_{29}W_{113}W_{197}W_{26}W_{111}W_{196}W_{23}W_{109}W_{195}W_{20}W_{107}W_{194}W_{17}W_{105}W_{193}W_{14}W_{103}W_{192}W_{11}W_{101}W_{191}W_{8}W_{99}W_{190}W_{5}W_{97}W_{189}W_{2}W_{95}W_{188}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
95	$W_{281}W_{282}W_{283}W_{278}W_{280}W_{282}W_{275}W_{278}W_{281}W_{272}W_{276}W_{280}W_{269}W_{274}W_{279}W_{266}W_{272}W_{278}W_{263}W_{270}W_{277}W_{260}W_{268}W_{276}W_{257}W_{266}W_{275}W_{254}W_{264}W_{274}W_{251}W_{262}W_{273}W_{248}W_{260}W_{272}W_{245}W_{258}W_{271}W_{242}W_{256}W_{270}W_{239}W_{254}W_{269}W_{236}W_{252}W_{268}W_{233}W_{250}W_{267}W_{230}W_{248}W_{266}W_{227}W_{246}W_{265}W_{224}W_{244}W_{264}W_{221}W_{242}W_{263}W_{218}W_{240}W_{262}W_{215}W_{238}W_{261}W_{212}W_{236}W_{260}W_{209}W_{234}W_{259}W_{206}W_{232}W_{258}W_{203}W_{230}W_{257}W_{200}W_{228}W_{256}W_{197}W_{226}W_{255}W_{194}W_{224}W_{254}W_{191}W_{222}W_{253}W_{188}W_{220}W_{252}W_{185}W_{218}W_{251}W_{182}W_{216}W_{250}W_{179}W_{214}W_{249}W_{176}W_{212}W_{248}W_{173}W_{210}W_{247}W_{170}W_{208}W_{246}W_{167}W_{206}W_{245}W_{164}W_{204}W_{244}W_{161}W_{202}W_{243}W_{158}W_{200}W_{242}W_{155}W_{198}W_{241}W_{152}W_{196}W_{240}W_{149}W_{194}W_{239}W_{146}W_{192}W_{238}W_{143}W_{190}W_{237}W_{140}W_{188}W_{236}W_{137}W_{186}W_{235}W_{134}W_{184}W_{234}W_{131}W_{182}W_{233}W_{128}W_{180}W_{232}W_{125}W_{178}W_{231}W_{122}W_{176}W_{230}W_{119}W_{174}W_{229}W_{116}W_{172}W_{228}W_{113}W_{170}W_{227}W_{110}W_{168}W_{226}W_{107}W_{166}W_{225}W_{104}W_{164}W_{224}W_{101}W_{162}W_{223}W_{98}W_{160}W_{222}W_{95}W_{158}W_{221}W_{92}W_{156}W_{220}W_{89}W_{154}W_{219}W_{86}W_{152}W_{218}W_{83}W_{150}W_{217}W_{80}W_{148}W_{216}W_{77}W_{146}W_{215}W_{74}W_{144}W_{214}W_{71}W_{142}W_{213}W_{68}W_{140}W_{212}W_{65}W_{138}W_{211}W_{62}W_{136}W_{210}W_{59}W_{134}W_{209}W_{56}W_{132}W_{208}W_{53}W_{130}W_{207}W_{50}W_{128}W_{206}W_{47}W_{126}W_{205}W_{44}W_{124}W_{204}W_{41}W_{122}W_{203}W_{38}W_{120}W_{202}W_{35}W_{118}W_{201}W_{32}W_{116}W_{200}W_{29}W_{114}W_{199}W_{26}W_{112}W_{198}W_{23}W_{110}W_{197}W_{20}W_{108}W_{196}W_{17}W_{106}W_{195}W_{14}W_{104}W_{194}W_{11}W_{102}W_{193}W_{8}W_{100}W_{192}W_{5}W_{98}W_{191}W_{2}W_{96}W_{190}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
96	$W_{284}W_{285}W_{286}W_{281}W_{283}W_{285}W_{278}W_{281}W_{284}W_{275}W_{279}W_{283}W_{272}W_{277}W_{282}W_{269}W_{275}W_{281}W_{266}W_{273}W_{280}W_{263}W_{271}W_{279}W_{260}W_{269}W_{278}W_{257}W_{267}W_{277}W_{254}W_{265}W_{276}W_{251}W_{263}W_{275}W_{248}W_{261}W_{274}W_{245}W_{259}W_{273}W_{242}W_{257}W_{272}W_{239}W_{255}W_{271}W_{236}W_{253}W_{270}W_{233}W_{251}W_{269}W_{230}W_{249}W_{268}W_{227}W_{247}W_{267}W_{224}W_{245}W_{266}W_{221}W_{243}W_{265}W_{218}W_{241}W_{264}W_{215}W_{239}W_{263}W_{212}W_{237}W_{262}W_{209}W_{235}W_{261}W_{206}W_{233}W_{260}W_{203}W_{231}W_{259}W_{200}W_{229}W_{258}W_{197}W_{227}W_{257}W_{194}W_{225}W_{256}W_{191}W_{223}W_{255}W_{188}W_{221}W_{254}W_{185}W_{219}W_{253}W_{182}W_{217}W_{252}W_{179}W_{215}W_{251}W_{176}W_{213}W_{250}W_{173}W_{211}W_{249}W_{170}W_{209}W_{248}W_{167}W_{207}W_{247}W_{164}W_{205}W_{246}W_{161}W_{203}W_{245}W_{158}W_{201}W_{244}W_{155}W_{199}W_{243}W_{152}W_{197}W_{242}W_{149}W_{195}W_{241}W_{146}W_{193}W_{240}W_{143}W_{191}W_{239}W_{140}W_{189}W_{238}W_{137}W_{187}W_{237}W_{134}W_{185}W_{236}W_{131}W_{183}W_{235}W_{128}W_{181}W_{234}W_{125}W_{179}W_{233}W_{122}W_{177}W_{232}W_{119}W_{175}W_{231}W_{116}W_{173}W_{230}W_{113}W_{171}W_{229}W_{110}W_{169}W_{228}W_{107}W_{167}W_{227}W_{104}W_{165}W_{226}W_{101}W_{163}W_{225}W_{98}W_{161}W_{224}$

	$W_{95}W_{159}W_{223}W_{92}W_{157}W_{222}W_{89}W_{155}W_{221}W_{86}W_{153}W_{220}W_{83}W_{151}W_{219}W_{80}W_{149}W_{218}W_{77}W_{147}W_{217}W_{74}W_{145}W_{216}W_{71}W_{143}W_{215}W_{68}W_{141}W_{214}W_{65}W_{139}W_{213}W_{62}W_{137}W_{212}W_{59}W_{135}W_{211}W_{56}W_{133}W_{210}W_{53}W_{131}W_{209}W_{50}W_{129}W_{208}W_{47}W_{127}W_{207}W_{44}W_{125}W_{206}W_{41}W_{123}W_{205}W_{38}W_{121}W_{204}W_{35}W_{119}W_{203}W_{32}W_{117}W_{202}W_{29}W_{115}W_{201}W_{26}W_{113}W_{200}W_{23}W_{111}W_{199}W_{20}W_{109}W_{198}W_{17}W_{107}W_{197}W_{14}W_{105}W_{196}W_{11}W_{103}W_{195}W_{8}W_{101}W_{194}W_{5}W_{99}W_{193}W_{2}W_{97}W_{192}$012→0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111122222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
97	$W_{287}W_{288}W_{289}W_{284}W_{286}W_{288}W_{281}W_{284}W_{287}W_{278}W_{282}W_{286}W_{275}W_{280}W_{285}W_{272}W_{278}W_{284}W_{269}W_{276}W_{283}W_{266}W_{274}W_{282}W_{263}W_{272}W_{281}W_{260}W_{270}W_{280}W_{257}W_{268}W_{279}W_{254}W_{266}W_{278}W_{251}W_{264}W_{277}W_{248}W_{262}W_{276}W_{245}W_{260}W_{275}W_{242}W_{258}W_{274}W_{239}W_{256}W_{273}W_{236}W_{254}W_{272}W_{233}W_{252}W_{271}W_{230}W_{250}W_{270}W_{227}W_{248}W_{269}W_{224}W_{246}W_{268}W_{221}W_{244}W_{267}W_{218}W_{242}W_{266}W_{215}W_{240}W_{265}W_{212}W_{238}W_{264}W_{209}W_{236}W_{263}W_{206}W_{234}W_{262}W_{203}W_{232}W_{261}W_{200}W_{230}W_{260}W_{197}W_{228}W_{259}W_{194}W_{226}W_{258}W_{191}W_{224}W_{257}W_{188}W_{222}W_{256}W_{185}W_{220}W_{255}W_{182}W_{218}W_{254}W_{179}W_{216}W_{253}W_{176}W_{214}W_{252}W_{173}W_{212}W_{251}W_{170}W_{210}W_{250}W_{167}W_{208}W_{249}W_{164}W_{206}W_{248}W_{161}W_{204}W_{247}W_{158}W_{202}W_{246}W_{155}W_{200}W_{245}W_{152}W_{198}W_{244}W_{149}W_{196}W_{243}W_{146}W_{194}W_{242}W_{143}W_{192}W_{241}W_{140}W_{190}W_{240}W_{137}W_{188}W_{239}W_{134}W_{186}W_{238}W_{131}W_{184}W_{237}W_{128}W_{182}W_{236}W_{125}W_{180}W_{235}W_{122}W_{178}W_{234}W_{119}W_{176}W_{233}W_{116}W_{174}W_{232}W_{113}W_{172}W_{231}W_{110}W_{170}W_{230}W_{107}W_{168}W_{229}W_{104}W_{166}W_{228}W_{101}W_{164}W_{227}W_{98}W_{162}W_{226}W_{95}W_{160}W_{225}W_{92}W_{158}W_{224}W_{89}W_{156}W_{223}W_{86}W_{154}W_{222}W_{83}W_{152}W_{221}W_{80}W_{150}W_{220}W_{77}W_{148}W_{219}W_{74}W_{146}W_{218}W_{71}W_{144}W_{217}W_{68}W_{142}W_{216}W_{65}W_{140}W_{215}W_{62}W_{138}W_{214}W_{59}W_{136}W_{213}W_{56}W_{134}W_{212}W_{53}W_{132}W_{211}W_{50}W_{130}W_{210}W_{47}W_{128}W_{209}W_{44}W_{126}W_{208}W_{41}W_{124}W_{207}W_{38}W_{122}W_{206}W_{35}W_{120}W_{205}W_{32}W_{118}W_{204}W_{29}W_{116}W_{203}W_{26}W_{114}W_{202}W_{23}W_{112}W_{201}W_{20}W_{110}W_{200}W_{17}W_{108}W_{199}W_{14}W_{106}W_{198}W_{11}W_{104}W_{197}W_{8}W_{102}W_{196}W_{5}W_{100}W_{195}W_{2}W_{98}W_{194}$012→00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
98	$W_{290}W_{291}W_{292}W_{287}W_{289}W_{291}W_{284}W_{287}W_{290}W_{281}W_{285}W_{289}W_{278}W_{283}W_{288}W_{275}W_{281}W_{287}W_{272}W_{279}W_{286}W_{269}W_{277}W_{285}W_{266}W_{275}W_{284}W_{263}W_{273}W_{283}W_{260}W_{271}W_{282}W_{257}W_{269}W_{281}W_{254}W_{267}W_{280}W_{251}W_{265}W_{279}W_{248}W_{263}W_{278}W_{245}W_{261}W_{277}W_{242}W_{259}W_{276}W_{239}W_{257}W_{275}W_{236}W_{255}W_{274}W_{233}W_{253}W_{273}W_{230}W_{251}W_{272}W_{227}W_{249}W_{271}W_{224}W_{247}W_{270}W_{221}W_{245}W_{269}W_{218}W_{243}W_{268}W_{215}W_{241}W_{267}W_{212}W_{239}W_{266}W_{209}W_{237}W_{265}W_{206}W_{235}W_{264}W_{203}W_{233}W_{263}W_{200}W_{231}W_{262}W_{197}W_{229}W_{261}W_{194}W_{227}W_{260}W_{191}W_{225}W_{259}W_{188}W_{223}W_{258}W_{185}W_{221}W_{257}W_{182}W_{219}W_{256}W_{179}W_{217}W_{255}W_{176}W_{215}W_{254}W_{173}W_{213}W_{253}W_{170}W_{211}W_{252}W_{167}W_{209}W_{251}W_{164}W_{207}W_{250}W_{161}W_{205}W_{249}W_{158}W_{203}W_{248}W_{155}W_{201}W_{247}W_{152}W_{199}W_{246}W_{149}W_{197}W_{245}W_{146}W_{195}W_{244}W_{143}W_{193}W_{243}W_{140}W_{191}W_{242}W_{137}W_{189}W_{241}W_{134}W_{187}W_{240}W_{131}W_{185}W_{239}W_{128}W_{183}W_{238}W_{125}W_{181}W_{237}W_{122}W_{179}W_{236}W_{119}W_{177}W_{235}W_{116}W_{175}W_{234}W_{113}W_{173}W_{233}W_{110}W_{171}W_{232}W_{107}W_{169}W_{231}W_{104}W_{167}W_{230}W_{101}W_{165}W_{229}W_{98}W_{163}W_{228}W_{95}W_{161}W_{227}W_{92}W_{159}W_{226}W_{89}W_{157}W_{225}W_{86}W_{155}W_{224}W_{83}W_{153}W_{223}W_{80}W_{151}W_{222}W_{77}W_{149}W_{221}W_{74}W_{147}W_{220}W_{71}W_{145}W_{219}W_{68}W_{143}W_{218}W_{65}W_{141}W_{217}W_{62}W_{139}W_{216}W_{59}W_{137}W_{215}W_{56}W_{135}W_{214}W_{53}W_{133}W_{213}W_{50}W_{131}W_{212}W_{47}W_{129}W_{211}W_{44}W_{127}W_{210}W_{41}W_{125}W_{209}W_{38}W_{123}W_{208}W_{35}W_{121}W_{207}W_{32}W_{119}W_{206}W_{29}W_{117}W_{205}W_{26}W_{115}W_{204}W_{23}W_{113}W_{203}W_{20}W_{111}W_{202}W_{17}W_{109}W_{201}W_{14}W_{107}W_{200}W_{11}W_{105}W_{199}W_{8}W_{103}W_{198}W_{5}W_{101}W_{197}W_{2}W_{99}W_{196}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
99	$W_{293}W_{294}W_{295}W_{290}W_{292}W_{294}W_{287}W_{290}W_{293}W_{284}W_{288}W_{292}W_{281}W_{286}W_{291}W_{278}W_{284}W_{290}W_{275}W_{282}W_{289}W_{272}W_{280}W_{288}W_{269}W_{278}W_{287}W_{266}W_{276}W_{286}W_{263}W_{274}W_{285}W_{260}W_{272}W_{284}W_{257}W_{270}W_{283}W_{254}W_{268}W_{282}W_{251}W_{266}W_{281}W_{248}W_{264}W_{280}W_{245}W_{262}W_{279}W_{242}W_{260}W_{278}W_{239}W_{258}W_{277}W_{236}W_{256}W_{276}W_{233}W_{254}W_{275}W_{230}W_{252}W_{274}W_{227}W_{250}W_{273}W_{224}W_{248}W_{272}W_{221}W_{246}W_{271}W_{218}W_{244}W_{270}W_{215}W_{242}W_{269}W_{212}W_{240}W_{268}W_{209}W_{238}W_{267}W_{206}W_{236}W_{266}W_{203}W_{234}W_{265}W_{200}W_{232}W_{264}W_{197}W_{230}W_{263}W_{194}W_{228}W_{262}W_{191}W_{226}W_{261}W_{188}W_{224}W_{260}W_{185}W_{222}W_{259}W_{182}W_{220}W_{258}W_{179}W_{218}W_{257}W_{176}W_{216}W_{256}W_{173}W_{214}W_{255}W_{170}W_{212}W_{254}W_{167}W_{210}W_{253}W_{164}W_{208}W_{252}W_{161}W_{206}W_{251}W_{158}W_{204}W_{250}W_{155}W_{202}W_{249}W_{152}W_{200}W_{248}W_{149}W_{198}W_{247}W_{146}W_{196}W_{246}W_{143}W_{194}W_{245}W_{140}W_{192}W_{244}W_{137}W_{190}W_{243}W_{134}W_{188}W_{242}W_{131}W_{186}W_{241}W_{128}W_{184}W_{240}W_{125}W_{182}W_{239}W_{122}W_{180}W_{238}W_{119}W_{178}W_{237}W_{116}W_{176}W_{236}W_{113}W_{174}W_{235}W_{110}W_{172}W_{234}W_{107}W_{170}W_{233}W_{104}W_{168}W_{232}W_{101}W_{166}W_{231}W_{98}W_{164}W_{230}W_{95}W_{162}W_{229}W_{92}W_{160}W_{228}W_{89}W_{158}W_{227}W_{86}W_{156}W_{226}W_{83}W_{154}W_{225}W_{80}W_{152}W_{224}W_{77}W_{150}W_{223}W_{74}W_{148}W_{222}W_{71}W_{146}W_{221}W_{68}W_{144}W_{220}W_{65}W_{142}W_{219}W_{62}W_{140}W_{218}W_{59}W_{138}W_{217}W_{56}W_{136}W_{216}W_{53}W_{134}W_{215}W_{50}W_{132}W_{214}W_{47}W_{130}W_{213}W_{44}W_{128}W_{212}W_{41}W_{126}W_{211}$

	$W_{38}W_{124}W_{210}W_{35}W_{122}W_{209}W_{32}W_{120}W_{208}W_{29}W_{118}W_{207}W_{26}W_{116}W_{206}W_{23}W_{114}W_{205}W_{20}W_{112}W_{204}W_{17}W_{110}W_{203}W_{14}W_{108}W_{202}W_{11}W_{106}W_{201}W_{8}W_{104}W_{200}W_{5}W_{102}W_{199}W_{2}W_{100}W_{198}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222
100	$W_{296}W_{297}W_{298}W_{293}W_{295}W_{297}W_{290}W_{293}W_{296}W_{287}W_{291}W_{295}W_{284}W_{289}W_{294}W_{281}W_{287}W_{293}W_{278}W_{285}W_{292}W_{275}W_{283}W_{291}W_{272}W_{281}W_{290}W_{269}W_{279}W_{289}W_{266}W_{277}W_{288}W_{263}W_{275}W_{287}W_{260}W_{273}W_{286}W_{257}W_{271}W_{285}W_{254}W_{269}W_{284}W_{251}W_{267}W_{283}W_{248}W_{265}W_{282}W_{245}W_{263}W_{281}W_{242}W_{261}W_{280}W_{239}W_{259}W_{279}W_{236}W_{257}W_{278}W_{233}W_{255}W_{277}W_{230}W_{253}W_{276}W_{227}W_{251}W_{275}W_{224}W_{249}W_{274}W_{221}W_{247}W_{273}W_{218}W_{245}W_{272}W_{215}W_{243}W_{271}W_{212}W_{241}W_{270}W_{209}W_{239}W_{269}W_{206}W_{237}W_{268}W_{203}W_{235}W_{267}W_{200}W_{233}W_{266}W_{197}W_{231}W_{265}W_{194}W_{229}W_{264}W_{191}W_{227}W_{263}W_{188}W_{225}W_{262}W_{185}W_{223}W_{261}W_{182}W_{221}W_{260}W_{179}W_{219}W_{259}W_{176}W_{217}W_{258}W_{173}W_{215}W_{257}W_{170}W_{213}W_{256}W_{167}W_{211}W_{255}W_{164}W_{209}W_{254}W_{161}W_{207}W_{253}W_{158}W_{205}W_{252}W_{155}W_{203}W_{251}W_{152}W_{201}W_{250}W_{149}W_{199}W_{249}W_{146}W_{197}W_{248}W_{143}W_{195}W_{247}W_{140}W_{193}W_{246}W_{137}W_{191}W_{245}W_{134}W_{189}W_{244}W_{131}W_{187}W_{243}W_{128}W_{185}W_{242}W_{125}W_{183}W_{241}W_{122}W_{181}W_{240}W_{119}W_{179}W_{239}W_{116}W_{177}W_{238}W_{113}W_{175}W_{237}W_{110}W_{173}W_{236}W_{107}W_{171}W_{235}W_{104}W_{169}W_{234}W_{101}W_{167}W_{233}W_{98}W_{165}W_{232}W_{95}W_{163}W_{231}W_{92}W_{161}W_{230}W_{89}W_{159}W_{229}W_{86}W_{157}W_{228}W_{83}W_{155}W_{227}W_{80}W_{153}W_{226}W_{77}W_{151}W_{225}W_{74}W_{149}W_{224}W_{71}W_{147}W_{223}W_{68}W_{145}W_{222}W_{65}W_{143}W_{221}W_{62}W_{141}W_{220}W_{59}W_{139}W_{219}W_{56}W_{137}W_{218}W_{53}W_{135}W_{217}W_{50}W_{133}W_{216}W_{47}W_{131}W_{215}W_{44}W_{129}W_{214}W_{41}W_{127}W_{213}W_{38}W_{125}W_{212}W_{35}W_{123}W_{211}W_{32}W_{121}W_{210}W_{29}W_{119}W_{209}W_{26}W_{117}W_{208}W_{23}W_{115}W_{207}W_{20}W_{113}W_{206}W_{17}W_{111}W_{205}W_{14}W_{109}W_{204}W_{11}W_{107}W_{203}W_{8}W_{105}W_{202}W_{5}W_{103}W_{201}W_{2}W_{101}W_{200}$012→000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000011111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111112222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222222