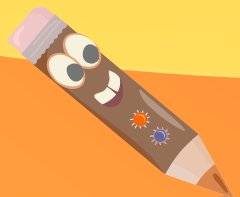
INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA

Exercices et solutions 2019
Années HarmoS 5/6

https://www.castor-informatique.ch/

Éditeurs :
Gabriel Parriaux, Jean-Philippe Pellet, Elsa Pellet, Christian Datzko, Susanne Datzko, Juraj Hromkovič, Regula Lacher



01010011010101100100100101000001001011010101001101010011010010010100010100101010100110101001100100101001001010000100001

SSIE

www.svia-ssie-ssii.ch
schweizerischervereinfürinformatikinderausbildung//sociétésuissepourl'informatiquedansl'enseignement//societàsvizzeraperl'informaticanell'insegnamento

Ont collaboré au Castor Informatique 2019

Christian Datzko, Susanne Datzko, Olivier Ens, Hanspeter Erni, Nora A. Escherle, Martin Guggisberg, Saskia Howald, Lucio Negrini, Gabriel Parriaux, Elsa Pellet, Jean-Philippe Pellet, Beat Trachsler.

Nous adressons nos remerciements à :
Juraj Hromkovič, Michelle Barnett, Michael Barot, Anna Laura John, Dennis Komm, Regula Lacher, Jacqueline Staub, Nicole Trachsler : ETHZ
Gabriel Thullen : Collège des Colombières
Valentina Dagienė : Bebras.org
Wolfgang Pohl, Hannes Endreß, Ulrich Kiesmüller, Kirsten Schlüter, Michael Weigend : Bundesweite Informatikwettbewerbe (BWINF), Allemagne
Chris Roffey : University of Oxford, Royaume-Uni
Carlo Bellettini, Violetta Lonati, Mattia Monga, Anna Morpurgo : ALaDDIn, Università degli Studi di Milano, Italie
Gerald Futschek, Wilfried Baumann, Florentina Voboril : Oesterreichische Computer Gesellschaft, Austria
Zsuzsa Pluhár : ELTE Informatikai Kar, Hongrie
Eljakim Schrijvers, Justina Dauksaite, Arne Heijenga, Dave Oostendorp, Andrea Schrijvers, Kyra Willekes, Saskia Zweerts : Cuttle.org, Pays-Bas
Christoph Frei : Chragokyberneticks (Logo Castor Informatique Suisse)
Andrea Leu, Maggie Winter, Brigitte Manz-Brunner : Senarclens Leu + Partner

La version allemande des exercices a également été utilisée en Allemagne et en Autriche.
L'adaptation française a été réalisée par Elsa Pellet et la version italienne par Veronica Ostini.

INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA

Le Castor Informatique 2019 a été réalisé par la Société Suisse de l'Informatique dans l'Enseignement SSIE et soutenu par la Fondation Hasler.

HASLERSTIFTUNG

Tous les liens ont été vérifiés le 1er novembre 2019. Ce cahier d'exercice a été produit le 2 janvier 2020 avec avec le logiciel de mise en page LaTeX.

Préambule

Très bien établi dans différents pays européens depuis plusieurs années, le concours « Castor Informatique » a pour but d'éveiller l'intérêt des enfants et des jeunes pour l'informatique. En Suisse, le concours est organisé en allemand, en français et en italien par la SSIE, la Société Suisse pour l'Informatique dans l'Enseignement, et soutenu par la Fondation Hasler dans le cadre du programme d'encouragement « FIT in IT ».
Le Castor Informatique est le partenaire suisse du concours « Bebras International Contest on Informatics and Computer Fluency » (`https://www.bebras.org/`), initié en Lituanie.
Le concours a été organisé pour la première fois en Suisse en 2010. Le Petit Castor (années HarmoS 5 et 6) a été organisé pour la première fois en 2012.
Le Castor Informatique vise à motiver les élèves à apprendre l'informatique. Il souhaite lever les réticences et susciter l'intérêt quant à l'enseignement de l'informatique à l'école. Le concours ne suppose aucun prérequis quant à l'utilisation des ordinateurs, sauf de savoir naviguer sur Internet, car le concours s'effectue en ligne. Pour répondre, il faut structurer sa pensée, faire preuve de logique mais aussi de fantaisie. Les exercices sont expressément conçus pour développer un intérêt durable pour l'informatique, au-delà de la durée du concours.
Le concours Castor Informatique 2019 a été fait pour cinq tranches d'âge, basées sur les années scolaires :

- Années HarmoS 5 et 6 (Petit Castor)
- Années HarmoS 7 et 8
- Années HarmoS 9 et 10
- Années HarmoS 11 et 12
- Années HarmoS 13 à 15

Les élèves des années HarmoS 5 et 6 avaient 9 exercices à résoudre : 3 faciles, 3 moyens, 3 difficiles. Les élèves des années HarmoS 7 et 8 avaient, quant à eux, 12 exercices à résoudre (4 de chaque niveau de difficulté). Finalement, chaque autre tranche d'âge devait résoudre 15 exercices (5 de chaque niveau de difficulté).
Chaque réponse correcte donnait des points, chaque réponse fausse réduisait le total des points. Ne pas répondre à une question n'avait aucune incidence sur le nombre de points. Le nombre de points de chaque exercice était fixé en fonction du degré de difficulté :

	Facile	Moyen	Difficile
Réponse correcte	6 points	9 points	12 points
Réponse fausse	−2 points	−3 points	−4 points

Utilisé au niveau international, ce système de distribution des points est conçu pour limiter le succès en cas de réponses données au hasard.
Chaque participant·e obtenait initialement 45 points (ou 27 pour la tranche d'âge « Petit Castor », et 36 pour les années HarmoS 7 et 8).
Le nombre de points maximal était ainsi de 180 (ou 108 pour la tranche d'âge « Petit Castor », et 144 pour les années HarmoS 7 et 8). Le nombre de points minimal était zéro.
Les réponses de nombreux exercices étaient affichées dans un ordre établi au hasard. Certains exercices ont été traités par plusieurs tranches d'âge.

Pour de plus amples informations :

SVIA-SSIE-SSII Société Suisse de l'Informatique dans l'Enseignement
Castor Informatique
Gabriel Parriaux

https://www.castor-informatique.ch/fr/kontaktieren/
https://www.castor-informatique.ch/
 https://www.facebook.com/informatikbiberch

Table des matières

1. Piscine !

C'est l'été et Anita, qui a douze ans, aimerait aller nager à la piscine. Elle prend Jean avec, son petit frère de six ans.
À l'entrée de la piscine est écrite la règle suivante :

— Âge minimal : 8 ans ; enfants de moins de 8 ans seulement accompagnés d'une personne de plus de 10 ans.

Qui a le droit d'entrer dans la piscine ?

A) Anita et Hans.

B) Anita, mais pas Jean.

C) Pas Anita, mais Jean.

D) Ni Anita ni Jean.

Solution

La règle a deux sens :

1. Toutes les personnes qui ont 8 ans ou plus ont le droit d'entrer dans la piscine. Comme Anita a plus de 8 ans, elle a le droit d'entrer.
2. Les personnes qui ont moins de 8 ans peuvent entrer dans la piscine si elles sont accompagnées d'une personne qui a plus de 10 ans. Comme Jean est accompagné d'Anita et qu'Anita a plus de 10 ans, Jean a aussi le droit d'entrer.

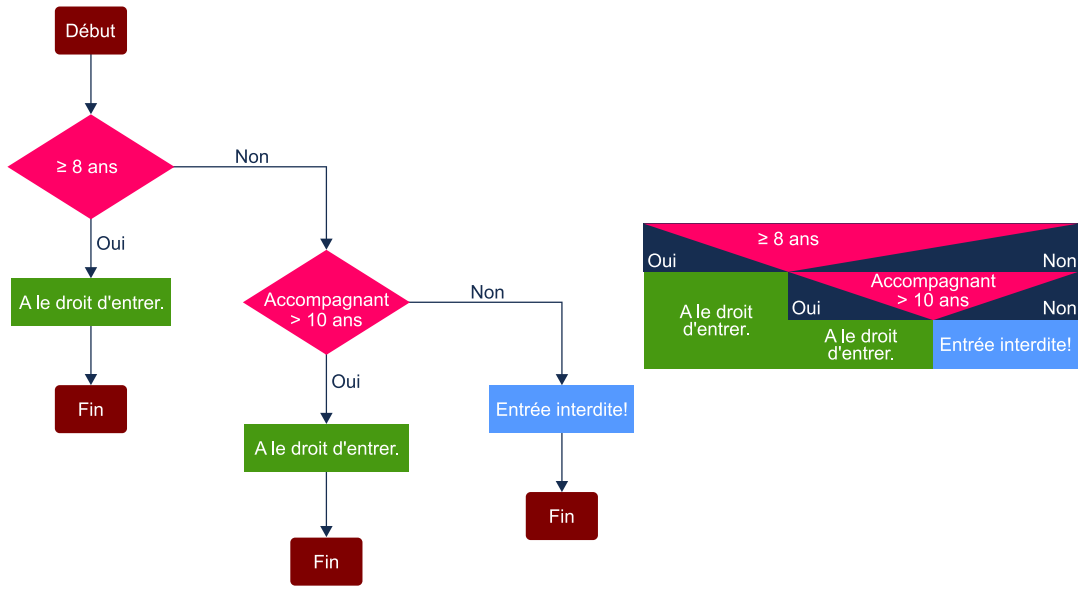
La bonne réponse est donc A) Anita et Jean.

C'est de l'informatique !

La règle de la piscine pose des *conditions* qui déterminent si quelque chose est permis ou interdit. Dans ce cas, deux conditions sont posées qui déterminent si une personne a le droit d'entrer ou pas. *Si* la condition est remplie, *alors* la personne peut entrer dans la piscine. On aurait donc aussi pu formuler la règle comme cela :

Si la personne a 8 ans ou plus :
Alors elle a le droit d'entrer dans la piscine
Sinon : Si la personne est accompagnée d'une personne de plus de 10 ans :
Alors elle a le droit d'entrer dans la piscine
Sinon elle n'a pas le droit d'entrer dans la piscine

On peut aussi représenter la règle par un *organigramme de programmation* (à gauche ; aussi appelé *ordinogramme*, *logigramme* ou encore *algorigramme*) ou alors par un *structogramme* (à droite) :



De tels processus de décisions sont appelés *branchements* en informatique. Ils y sont très souvent utilisés.

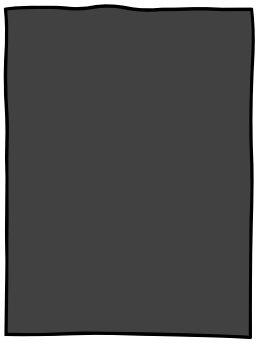
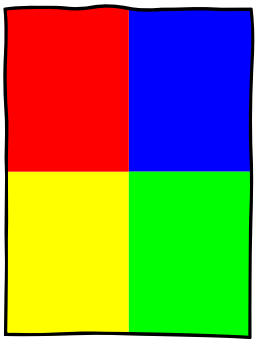
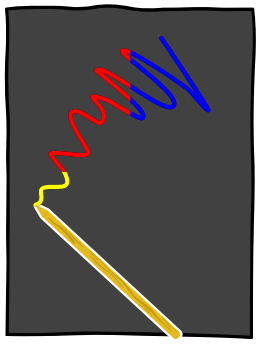
Mots clés et sites web

Branchement, organigramme, structogramme
— `https://fr.wikipedia.org/wiki/Instruction_conditionnelle_(programmation)`

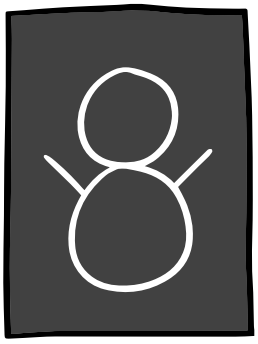
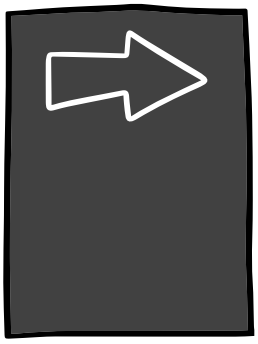
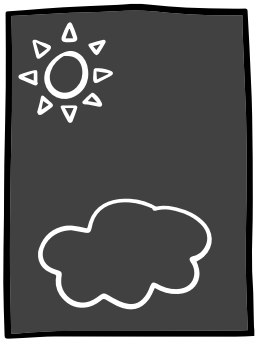
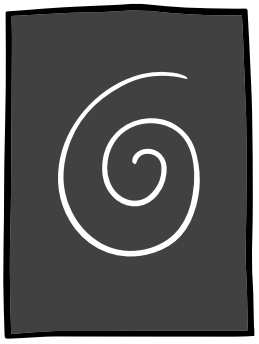
— `https://fr.wikipedia.org/wiki/Organigramme_de_programmation`
— `https://fr.wikipedia.org/wiki/Structogramme`

2. Papier à gratter

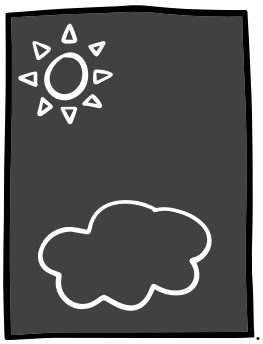
Tu peux faire des dessins colorés facilement avec du papier à gratter. Tu enlèves la couche du dessus avec un stylet en bois et la couche colorée du dessous apparaît.

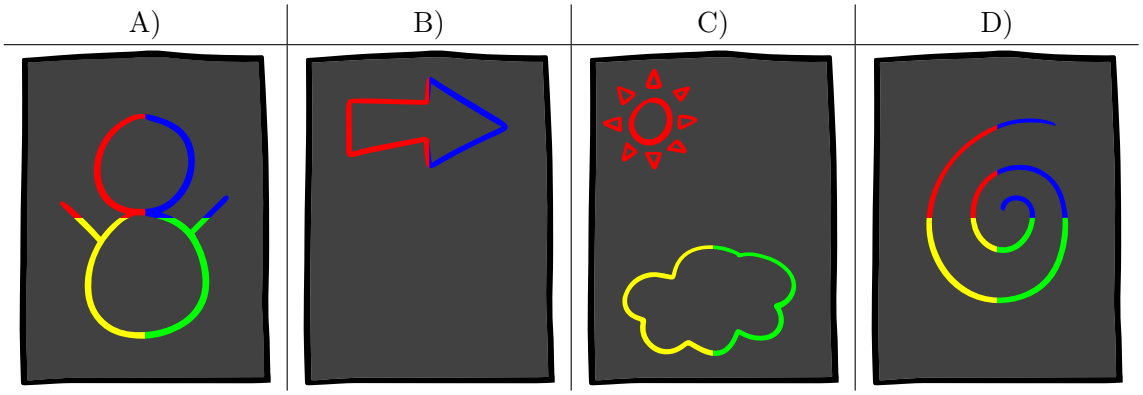
		
Au départ, le papier à gratter est recouvert d'une couche noire.	Derrière la couche noire se cachent ces quatre couleurs.	Une partie de la couche noire a été grattée avec le stylet en bois. Tu peux y voir les couleurs cachées en dessous.

En dessinant laquelle de ces quatre images vois-tu apparaître exactement trois couleurs ?

A)	B)	C)	D)
			

Solution

La bonne réponse est C)  .
Les couleurs suivantes apparaissent lorsque l'on dessine les quatre images :



La bonne réponse est donc C) : les couleurs rouge, jaune et verte apparaissent. La quatrième couleur bleue n'apparaît pas, car le quart de feuille en haut à droite n'est pas utilisé.
Les quatre couleurs apparaissent sur les images des réponses A) et D) et seulement les deux couleurs rouge et bleue sur l'image de la réponse B).

C'est de l'informatique !

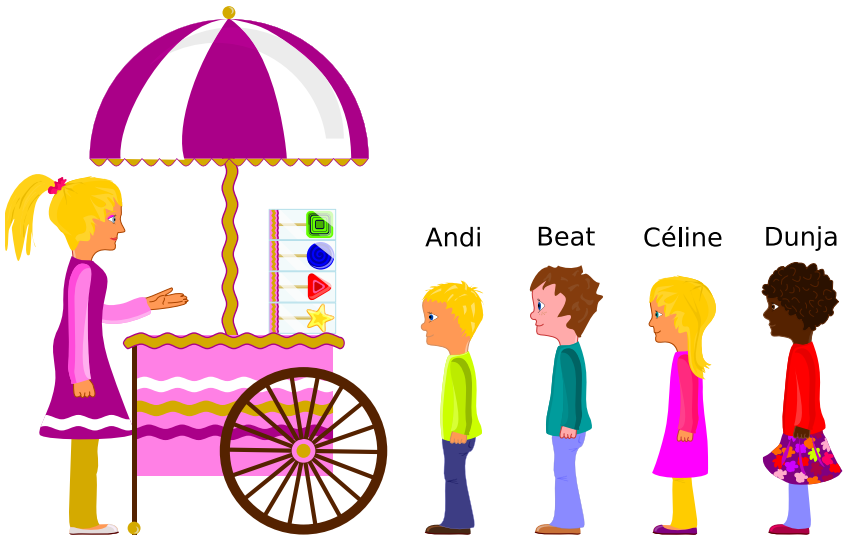
Lorsque l'on enlève la couche supérieure du papier à gratter, cette couche devient *transparente*, à cet endroit, on peut voir à travers le *calque* et donc voir les couleurs du dessous. Beaucoup de logiciels de retouche d'image utilisent des calques qui sont transparents à certains endroits. La plupart du temps, on les utilise dans l'autre sens : on a par exemple une photo en arrière-plan et on y ajoute un calque avec du texte. Ce calque est alors transparent partout sauf là où le texte est écrit. On pourrait bien sûr aussi écrire le texte directement sur l'image, mais en utilisant plusieurs calques, on peut ensuite modifier un seul calque alors que tous les autres calques restent pareils.
Dans cet exercice, on doit s'imaginer la couche du dessous pendant que l'on rend la couche du dessus transparente. C'est beaucoup plus simple si l'on sépare l'image en images plus petites. On peut ensuite regarder pour chaque quart de feuille si la couche du dessus a un endroit transparent ou pas. On sait ensuite si la couleur correspondante apparaît dans l'image complète ou pas. Ce processus s'appelle *décomposition*, il est souvent utilisé en informatique.

Mots clés et sites web

Retouche d'image, calque, décomposition
- `https://fr.wikipedia.org/wiki/Calque_(infographie)`
- `https://en.wikipedia.org/wiki/Decomposition_(computer_science)`

3. Kiosque

Andi, Beat, Céline et Dunja sont dans la file d'attente au kiosque. La vendeuse a une pile de sucettes devant elle. Elle vend toujours la sucette du haut de la pile. Andi reçoit la sucette rectangulaire verte  , car il est en première place dans la file d'attente et reçoit donc la sucette tout en haut de la pile.



Qui reçoit la sucette triangulaire rouge  *?*

A) Andi

B) Beat

C) Céline

D) Dunja

Solution

C'est le tour d'Andi en premier : il reçoit la sucette rectangulaire verte comme déjà décrit dans l'exercice.
Une fois que la sucette rectangulaire verte est vendue, la sucette bleue et ronde est tout en haut. Beat la reçoit comme il est en deuxième place dans la file d'attente.
Après la sucette bleue et ronde, c'est la sucette triangulaire rouge qui est tout en haut. Céline la reçoit comme elle est en troisième place dans la file d'attente. La bonne réponse est donc C) Céline.
Dunja ne repart pas non plus les mains vides : pour elle, il reste la sucette jaune en forme d'étoile.

C'est de l'informatique !

Andi, Beat, Céline et Dunja attendent dans une *file*. Si Eddy voulait se mettre dans la file, il devrait se mettre *à l'arrière* après Dunja. Cependant, c'est le tour d'Andi en premier qui est sur le *devant* de la file.
Les sucettes sont mises sur une *pile*. Si la vendeuse voulait vendre une sucette de plus, elle la mettrait *sur le haut* de la pile. Elle vend aussi toujours la sucette qu'elle prend du *haut* de la pile.
Une *file* (*queue* en anglais) peut ajouter des éléments à l'arrière (*enfiler*, *enqueue* en anglais) et en enlever devant (*défiler*, *dequeue* en anglais). Elle fonctionne donc d'après le principe que l'élément ajouté en premier est enlevé en premier (*premier entré, premier sorti* ; *first in, first out* ou *FIFO* en anglais). Une *pile* (*stack* en anglais) peut ajouter des éléments en haut (*empiler*, *push* en anglais) et en enlever du haut (*dépiler*, *pop* en anglais). Elle fonctionne donc selon le principe que le dernier élément ajouté est enlevé en premier (*dernier arrivé, premier sorti* ; *last in, first out* ou *LIFO* en anglais). Les files et les piles proposent souvent d'autres fonctions pour afficher le prochain élément à enlever (*front* et *top*, respectivement, en anglais) ou pour voir si la file ou la pile est vide (*empty* en anglais).
Ces deux structures sont souvent utilisées avec les ordinateurs parce qu'elles sont très simples. Les piles étaient par exemple déjà utilisées en 1945 par Konrad Zuse dans l'un des premiers ordinateurs du monde.

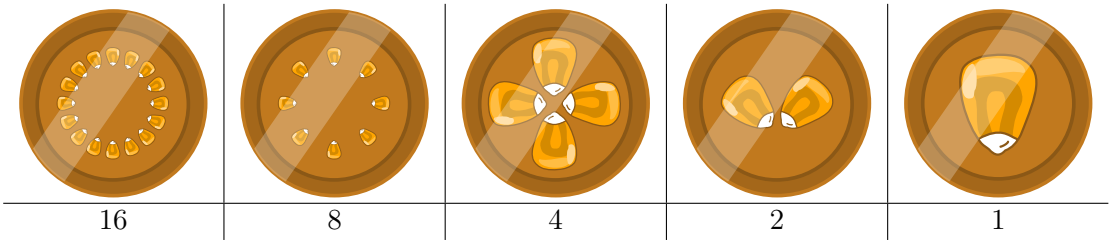
Mots clés et sites web

File, pile

- `https://fr.wikipedia.org/wiki/File_(structure_de_donn%C3%A9es)`
- `https://fr.wikipedia.org/wiki/Pile_(informatique)`
- `https://fr.wikipedia.org/wiki/Zuse_4`

4. Beavercoins

Au pays des castors, on utilise le « beavercoin » comme monnaie. Les pièces ont les valeurs suivantes :



Les castors n'aiment pas devoir porter beaucoup de pièces avec eux et paient donc avec le moins de pièces possible.
Quelles pièces utiliserais-tu pour payer 13 *beavercoins en utilisant le moins de pièces possible ?*

Solution

La meilleure et donc la bonne solution est de payer avec  ,  et  , donc avec une pièce de 8 beavercoins, une pièce de 4 beavercoins et une pièce d'un beavercoin. La somme des pièces donne $8+4+1 = 13$. Ce n'est pas possible d'utiliser moins de pièces, car la pièce valant plus de 8 beavercoins vaut déjà 16 beavercoins et il n'y a pas de pièce valant les 5 beavercoins manquants. La plus petite pièce suivante est la pièce de 4 beavercoins qui fait qu'avec la pièce d'un beavercoin, on utilise ces trois pièces pour payer.
On peut aussi commencer par une autre combinaison pour trouver la bonne solution, par exemple avec deux pièces de 4 beavercoins, une pièce de 2 beavercoins et trois pièces d'un beavercoin. On peut ensuite remplacer deux pièces ayant la même valeur par une pièce valant le double jusqu'à arriver au bon résultat

C'est de l'informatique !

Les informaticiennes et informaticiens sont des experts pour représenter des informations par des suites de symboles. La représentation de chiffres en fait partie. Dans cet exercice, il s'agit de pouvoir payer une somme d'argent avec différentes combinaisons de pièces de monnaie. Cette combinaison n'est pas unique, plusieurs combinaisons de pièces avec différentes valeurs donnent la même somme d'argent. Il s'agit donc dans cet exercice aussi de trouver la combinaison avec le plus petit nombre de pièces de monnaie.
Les pièces utilisées dans cet exercice sont choisies pour que deux pièces de la même valeur aient ensemble la même valeur que la plus grande pièce suivante. Cela donne le *système binaire* avec les valeurs 1, 2, 4, 8, 16 et ainsi de suite. Dans le système binaire, n'importe quel nombre comme le 13 est représenté de manière unique : chaque valeur est utilisée ou non.
Le boulier fonctionne de la même manière. Un boulier est une machine à calculer qui a été utilisée pendant des centaines d'années et dont certaines versions sont encore utilisées aujourd'hui à l'âge des calculatrices dans certaines régions du monde.

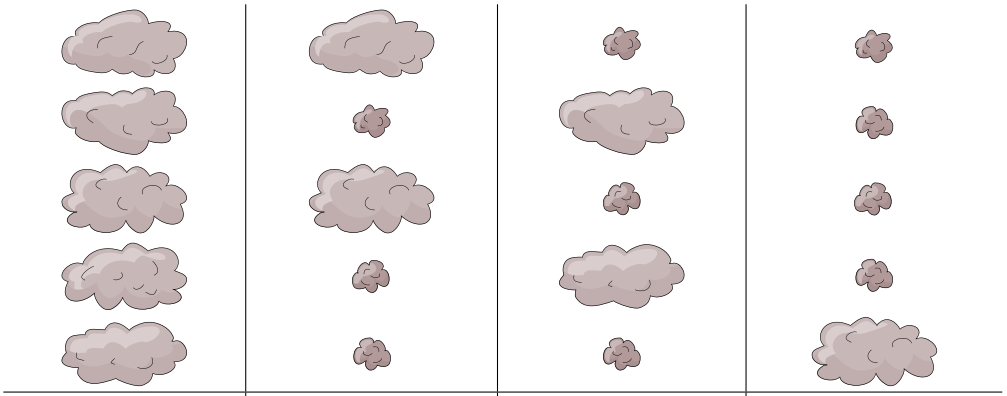
Mots clés et sites web

Système binaire, boulier

- `https://fr.wikipedia.org/wiki/Syst%C3%A8me_binaire`
- `https://fr.wikipedia.org/wiki/Boulier`

5. Signaux de fumée

Un castor est toujours en haut de la montagne et observe la météo. Il transmet les prévisions météo aux castors dans la vallée. Pour cela, il utilise des signaux de fumée qui sont composés de cinq nuages de fumée. Un nuage de fumée peut être soit petit, soit grand. Les castors se sont mis d'accord sur les signaux de fumée suivants :



Ce sera orageux.	Ce sera pluvieux.	Ce sera nuageux.	Ce sera ensoleillé.

Un jour où il y a beaucoup de vent, les castors dans la vallée n'arrivent pas bien à reconnaître les nuages de fumée. Il sont seulement sûrs que le deuxième et quatrième nuages sont grands, ils ont remplacé les autres par des points d'interrogation :

Qu'est-ce que cela aurait pu vouloir dire ?

A) « Ce sera orageux » ou « Ce sera pluvieux ».

B) « Ce sera pluvieux » ou « Ce sera nuageux ».

C) « Ce sera pluvieux » ou « Ce sera ensoleillé ».

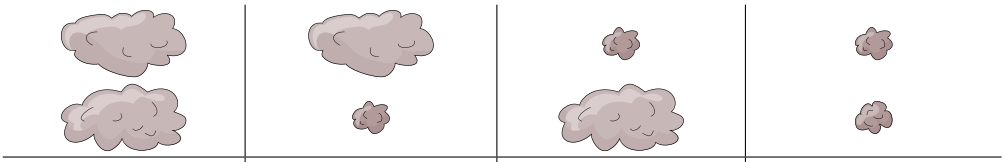
D) « Ce sera orageux » ou « Ce sera nuageux ».

Solution

Les castors dans la vallée ont reconnu de gros nuages de fumée en deuxième et quatrième places. Les signaux de fumée « Ce sera orageux » et « Ce sera nuageux » ont aussi de gros nuages de fumée à ces deux positions, donc à la deuxième et quatrième places. Les signaux « Ce sera pluvieux » et « Ce sera ensoleillé » ont de petits nuages de fumée à ces positions, ces signaux ne correspondent donc pas aux observations des castors dans la vallée.
La bonne réponse est donc D) « Ce sera orageux » ou « Ce sera nuageux ».

C'est de l'informatique !

Lorsque l'on doit transmettre un message, on aimerait que ce message arrive correctement à son destinataire. Les messages de cet exercice sont transmis à l'aide de petits et de grands nuages de fumée. Dans le cas général, on parle de *symboles*. C'est donc raisonnable de choisir une suite de symboles qui permette de comprendre le message même s'il est endommagé en cours de route. On peut faire cela en transmettant plus d'informations qu'il n'est strictement nécessaire. On appelle ces informations supplémentaires *redondantes*.
Lorsque l'on peut reconstruire un message avec au maximum n erreurs, on parle de code correcteur avec une capacité de correction n. La représentation de messages par des suites de symboles de manière à ce que l'on puisse les reconstruire même lorsque cette représentation a été endommagée lors de la transmission est une tâche typique pour les informaticiens. Ils nous permettent ainsi par exemple de lire de la musique à partir de CD ou des vidéos à partir de DVD même lorsque quelques erreurs ont eu lieu lors de la transmission.
Dans cet exercice, deux nuages de fumée auraient suffi pour transmettre les quatre messages différents :



Ce sera orageux.	Ce sera pluvieux.	Ce sera nuageux.	Ce sera ensoleillé.

Les castors utilisent cependant cinq nuages de fumée. Cela leur permet de comprendre le message même dans les cas où deux voire parfois trois des nuages de fumée sont « illisibles ». Les castors ont de plus choisi les messages de manière à ce qu'il y ait au moins trois positions différentes entre chaque paire de messages.

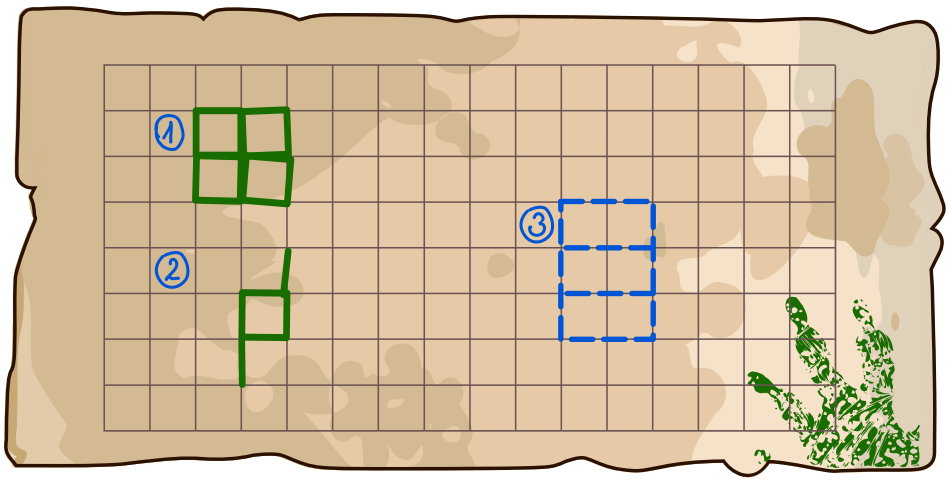
Mots clés et sites web

Code correcteur
— `https://fr.wikipedia.org/wiki/Code_correcteur`

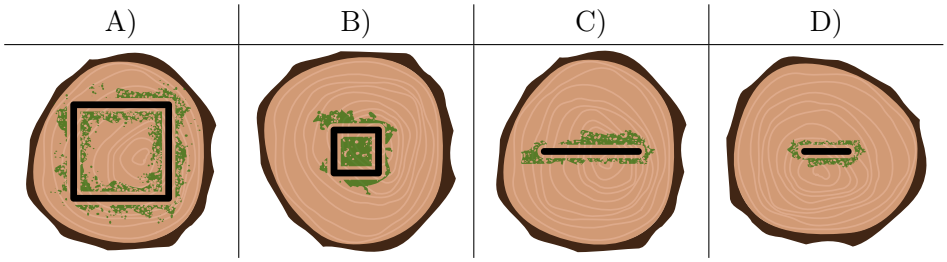
6. Tampon

Le castor Paul a les quatre tampons A, B, C et D comme montré plus bas. Paul a tamponné les deux motifs ① et ② avec ces tampons.

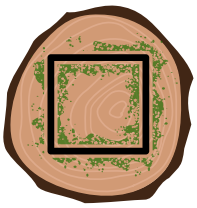
— Pour le motif ①, Paul a utilisé quatre fois le tampon B.
— Pour le motif ②, Paul a utilisé une fois le tampon B et deux fois le tampon D.

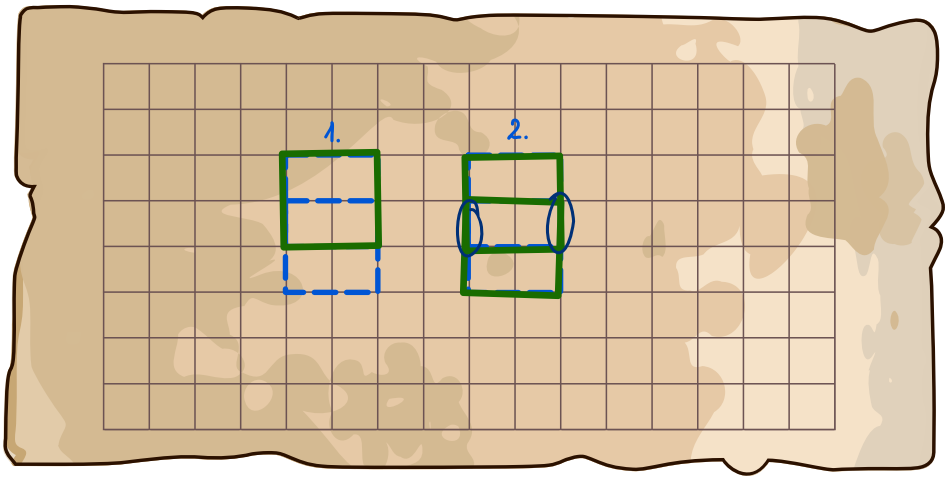


Paul aimerait à présent obtenir le motif ③. Sa sœur Marie affirme qu'elle ne doit tamponner que deux fois pour faire le motif.
Quel tampon Marie utiliserait-elle ?



Solution

La bonne réponse est A)  . Lorsque Marie tamponne un grand carré et décale le tampon d'une case vers le haut ou le bas, cela donne exactement le motif voulu. Deux parties de ligne se superposent, mais si elle tamponne proprement, cela ne se voit pas :



On n'arrive pas à dessiner le motif voulu en seulement deux tamponnages avec les autres tampons :

- C'est impossible de dessiner un rectangle large de deux cases sans ligne au milieu avec le tampon B.
- Elle pourrait dessiner le motif avec le tampon C, mais comme les lignes du motif sont en tout longues de quatorze cases et qu'elle ne peut tamponner que deux cases par utilisation du tampon, elle devrait tamponner au moins sept fois. Si l'on regarde exactement, on voit qu'elle devrait même tamponner huit fois car elle a besoin de tamponner deux fois pour dessiner chaque ligne verticale (avec superposition) en plus des quatre lignes horizontales.
- Elle pourrait dessiner le motif avec le tampon D, mais comme les lignes du motif sont en tout longues de quatorze cases et qu'elle ne peut tamponner qu'une case par utilisation du tampon, elle devrait tamponner au moins quatorze fois.

C'est de l'informatique !

Beaucoup de problèmes ont plusieurs solutions qui permettent d'atteindre le but. Certaines d'entre elles peuvent souvent être trouvées rapidement, comme par exemple les solutions avec les tampons C ou D. Mais toutes les solutions ne sont pas de « qualité » égale : les solutions sont par exemple clairement différentes au niveau du nombre de tamponnages nécessaires.
Une des tâche de l'informatique est de trouver la « meilleure » solution parmi toutes les solutions possibles d'un problème. C'est très important en pratique : si l'on peut compléter une tâche en une heure au lieu d'un jour, il reste de nombreuse heures pour s'occuper d'autres tâches.
Pour mesurer l'efficacité d'un processus, les informaticiens l'analysent et décrivent sa durée en fonction de la quantité ou de la taille des données à traiter. Par exemple, si l'on cherche une entrée dans une liste triée contenant 1 000 000 d'entrées, on peut soit regarder une entrée après l'autre et faire en moyenne 500 000 comparaisons, ou alors commencer au milieu et continuer à chercher dans la

moitié de la liste correspondant à l'entrée recherchée… Et on trouve l'entrée après au plus 20 comparaisons ! Si une comparaison prend trois secondes, la différence est entre 17 jours ininterrompus et une minute de recherche.

Mots clés et sites web

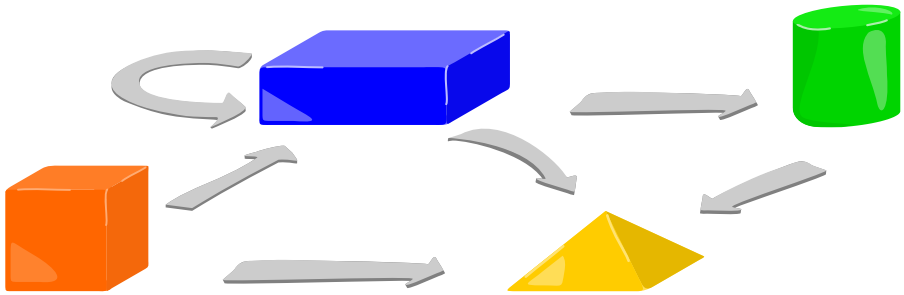
Efficacité, théorie de la complexité

- `https://fr.wikipedia.org/wiki/Analyse_de_la_complexit%C3%A9_des_algorithmes`
- `https://fr.wikipedia.org/wiki/Th%C3%A9orie_de_la_complexit%C3%A9_(informatique_th%C3%A9orique)`

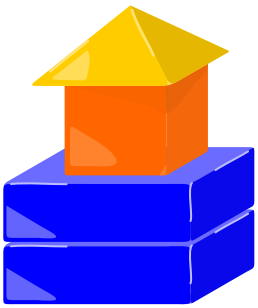
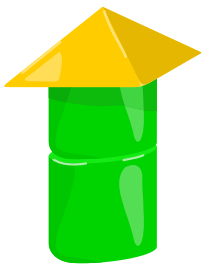
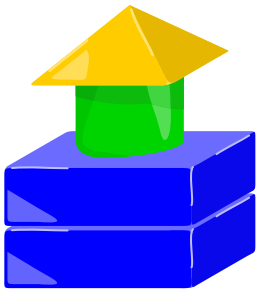
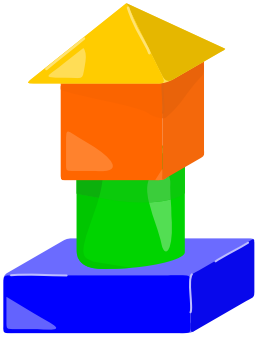
7. Quelle tour ?

La petite sœur de Léon a établi des règles selon lesquelles des plots peuvent être empilés. Elle les a représentées par des flèches sur un dessin. Il y a en plus les règles suivantes :

- Léon peut commencer avec n'importe quel plot.
- Léon doit toujours suivre les flèches. Lorsque plusieurs flèches partent d'un plot, Léon peut choisir laquelle il suit. Lorsqu'une flèche revient au même plot, il peut ajouter un autre plot de la même sorte à la tour.
- Léon doit s'arrêter lorsqu'aucune flèche ne part du plot ajouté à la tour.



Léon empile quatre tours différentes pour sa petite sœur.
Laquelle des quatre tours a-t-il construite en suivant les règles de sa petite sœur ?

A)	B)	C)	D)
			

Solution

La tour de la réponse A) commence correctement avec deux pavés droits bleus. Après le deuxième pavé droit bleu vient cependant un cube orange alors qu'il n'y a pas de flèche allant du pavé droit bleu au cube orange. La réponse A) est donc fausse.
La tour de la réponse B) commence correctement avec un cylindre vert. Après le cylindre vert vient cependant un autre cylindre vert alors qu'il n'y pas de flèche revenant vers le cylindre vert depuis le cylindre vert. La réponse B) est donc fausse.
La tour de la réponse C) commence correctement avec deux pavés droits bleus. Après le deuxième pavé droit bleu vient correctement un cylindre vert, et après le cylindre vert, aussi correctement, une pyramide jaune. Comme aucune flèche ne part de la pyramide jaune, c'est correct qu'aucun autre plot n'y soit empilé. La réponse C) est donc juste.
La tour de la réponse D) commence correctement avec un pavé droit bleu. Après le pavé droit bleu vient correctement un cylindre vert. Après le cylindre vert vient cependant un cube orange alors qu'il n'y a pas de flèche allant du cylindre vert au cube orange. La réponse D) est donc fausse.

C'est de l'informatique !

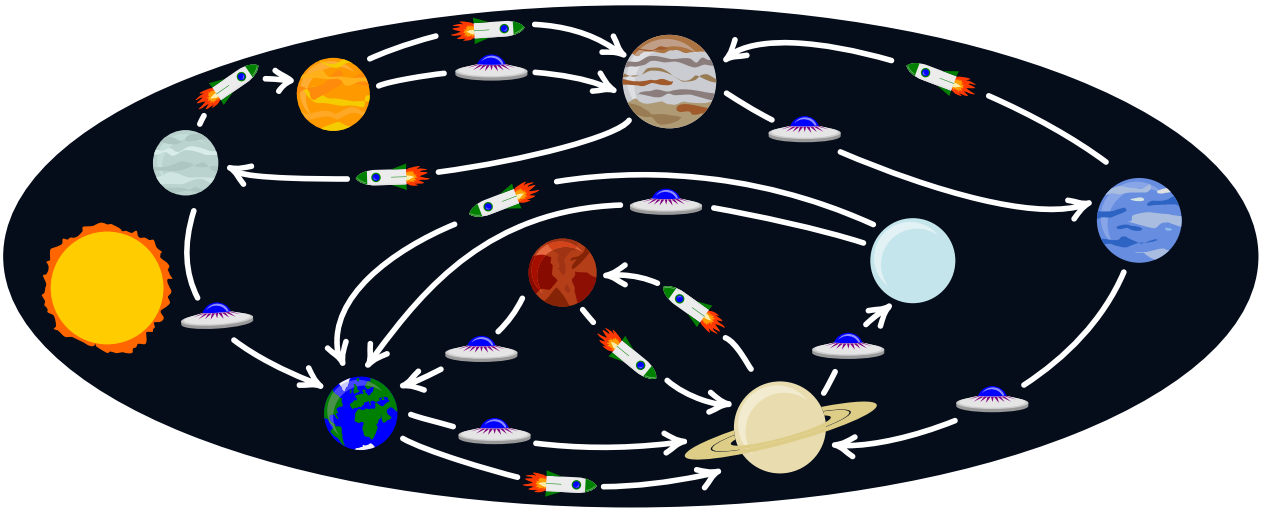
Les règles pour construire une tour sont basées sur le fait que le plot tout en haut de la tour détermine quels plots sont autorisés ensuite. Le plot le plus haut est donc l'*état actuel* de la tour. Les règles fixent vers quels états la *transition* suivante peut se faire. L'illustration avec les flèches est appelée un *diagramme états-transitions*. Comme tous les plots peuvent être utilisés tout en bas de la tour, ils sont tous des *états initiaux* possibles. La pyramide jaune est le seul plot étant un *état final* avec lequel la tour est terminée (si elle n'est pas tombée avant). La décision d'ajouter un plot de plus sur la tour est une *entrée* du constructeur.
Ces aspects de la construction d'une tour décrivent un *automate fini non déterministe*. Il est appelé non déterministe car il existe des états depuis lesquels plusieurs chemins peuvent être choisis : après un pavé droit bleu, il peut y avoir un autre pavé droit bleu, un cylindre vert ou une pyramide jaune. Il est appelé fini car il n'y a qu'un ensemble fini d'états : l'un des quatre plots peut être en haut de la tour. Théoriquement, il permet de construire une tour infiniment haute... Mais pour cela, on aurait besoin d'une part d'une infinité de pavés droits bleus, et d'autre part, les tours hautes ont tendance à tomber (souvent au grand plaisir du constructeur).
Le modèle de l'automate fini non déterministe est souvent utilisé en informatique. Il permet de décrire des choses complètement différentes les unes des autres : le comportement de modules logiciels ou de programmes entiers, de simples structures de langages, les interactions entre des composantes de hardware et beaucoup d'autre choses. On peut tester à l'aide de telles descriptions formelles si un logiciel se comporte comme souhaité... ou si une tour est construite de la bonne manière.

Mots clés et sites web

Automate fini non déterministe
- `https://fr.wikipedia.org/wiki/Automate_fini_non_d%C3%A9terministe`
- `https://www.swisseduc.ch/informatik/karatojava/kara/index.html`
- `https://educ.ethz.ch/unterrichtsmaterialien/informatik/kara.html`

8. Voyage dans l'espace

Des astronautes peuvent voler entre les planètes de notre système solaire en fusée ou en OVNI . La carte suivante représente les itinéraires de vols possibles :



Un astronaute qui veut voyager de Vénus à Saturne peut voler jusqu'à Jupiter en fusée ou en OVNI . Ensuite, il peut voler jusqu'à Neptune en OVNI et finalement jusqu'à son but, la planète Saturne , en OVNI . Lorsque l'astronaute vole d'abord avec une fusée et ensuite avec deux OVNI, il décrit son voyage ainsi :

En ce moment, l'astronaute Heidi est sur la planète Neptune et aimerait voyager jusqu'à la Terre . L'agence de voyage astronautique lui envoie quatre propositions.
Quelle proposition ne *ramène* pas *Heidi sur Terre* *?*

A)

B)

C)

D)

Solution

La réponse B) est la seule réponse ne permettant pas à Heidi de rentrer sur Terre. Si Heidi suit cette proposition, elle atterrit de nouveau sur Neptune à la fin du voyage. Elle commence en effet par voler en fusée jusqu'à Jupiter, puis à nouveau jusqu'à Neptune en OVNI, puis à nouveau jusqu'à Jupiter en fusée, et finalement à nouveau en OVNI jusqu'à Neptune.
Les trois autres propositions la ramènent toutes sur Terre. Les étapes sont :
Réponse A) : De Neptune en OVNI jusqu'à Saturne, en OVNI jusqu'à Uranus et en fusée jusqu'à la Terre.
Réponse C) : De Neptune en fusée jusqu'à Jupiter, en OVNI jusqu'à Neptune, en OVNI jusqu'à Saturne, en OVNI jusqu'à Uranus et en fusée jusqu'à la Terre.
Réponse D) : De Neptune en fusée jusqu'à Jupiter, en fusée jusqu'à Mercure et en OVNI jusqu'à la Terre.

C'est de l'informatique !

La carte des itinéraires possibles d'une planète à l'autre a une caractéristique particulière : il y a toujours exactement deux routes qui quittent chaque planète, l'une avec une fusée et l'autre avec un OVNI. De cette manière, on sait toujours sur quelle planète on atterrit si le moyen de transport utilisé est indiqué.
Une carte comme celle-ci décrit un *automate fini déterministe*. Un tel automate consiste en un ensemble d'*états* possibles (dans ce cas, ce sont les noms des planètes comme emplacement d'un astronaute), un ensemble de *transitions* entre les états (dans ce cas, il s'agit des flèches sur la carte permettant à un astronaute de changer d'emplacement) qui dépendent d'*entrées* définies (« fusée » ou « OVNI ») ainsi qu'un *état initial* (dans ce cas l'état « Neptune ») et un ensemble d'*états finaux* (dans ce cas seulement l'état « Terre »). On appelle aussi cette carte un *diagramme états-transitions*.
Les automates finis déterministes sont souvent utilisés parce qu'ils sont très faciles à programmer. Des exemples typiques sont les machines à café, les lave-vaisselle ou encore les automates à boissons. Ils sont également utilisés pour reconnaître des mots correctement (par exemple pour déterminer si un texte représente une adresse e-mail). On peut mettre les automates finis en lien avec un certain type de grammaires (appelées *grammaires régulières*) et un certain type de langages artificiels (appelés *langages réguliers*) et passer d'un « monde » à l'autre. C'est utile pour résoudre beaucoup de problèmes.
L'agence de voyage astronautique a par ailleurs une autre tâche : elle doit trouver un chemin possible allant d'un état à l'autre sur le diagramme états-transitions. Pour cela, c'est utile de voir le diagramme états-transitions comme un *graphe orienté* et d'y chercher un chemin allant d'un *nœud* nœud à l'autre en passant par les *arêtes* données. Il existe pour cela des algorithmes standards qui font que l'agence de voyage astronautique ne doit pas recommencer à chercher depuis le départ à chaque fois...

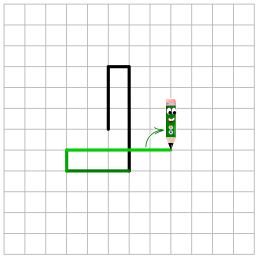
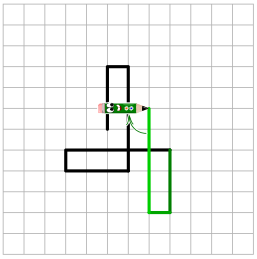
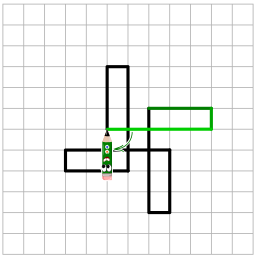
Mots clés et sites web

Automate fini déterministe, graphe

- `https://fr.wikipedia.org/wiki/Automate_fini_d%C3%A9terministe`
- `https://www.swisseduc.ch/informatik/karatojava/kara/index.html`
- `https://educ.ethz.ch/unterrichtsmaterialien/informatik/kara.html`
- `https://fr.wikipedia.org/wiki/Graphe_(math%C3%A9matiques_discr%C3%A8tes)`

9. Robot graphique

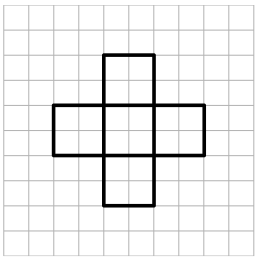
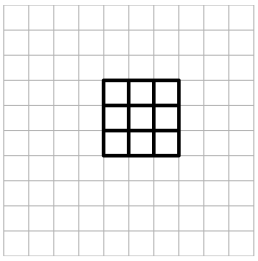
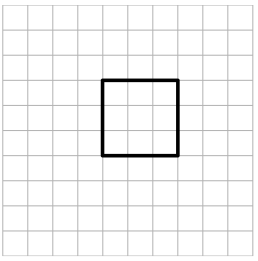
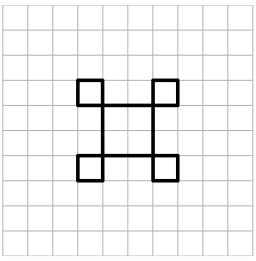
Un robot se déplace sur une grille en dessinant des lignes. Il peut être commandé à l'aide de trois nombres. Si on lui donne les chiffres 3↱1↱5↱, il dessine la figure suivante :

Première exécution :	Deuxième exécution :	Troisième exécution :	Quatrième exécution :
			

Pour cela, il répète quatre fois les étapes suivantes :

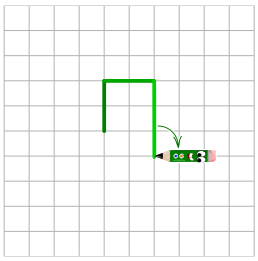
— Avance sur la grille du nombre de cases indiqué par le premier nombre.
— Fais un quart de tour vers la droite.
— Avance sur la grille du nombre de cases indiqué par le deuxième nombre.
— Fais un quart de tour vers la droite.
— Avance sur la grille du nombre de cases indiqué par le troisième nombre.
— Fais un quart de tour vers la droite.

On donne les nombres 2↱2↱3↱ au robot. À quoi les lignes dessinées ressemblent-elles ?

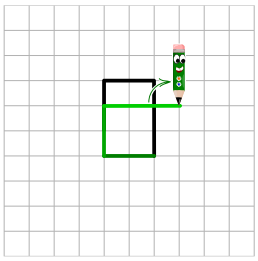
A)	B)	C)	D)
			

Solution

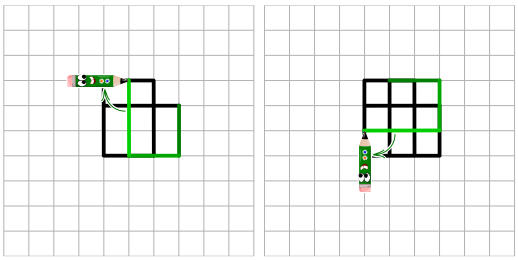
Les nombres 2↱2↱3↱ signifient que le robot commence par avancer de deux cases, fais un quart de tour vers la droite, avance à nouveau de deux cases, fais un quart de tour vers la droite, avance de trois cases et fais encore une fois un quart de tour vers la droite. Quand il a terminé, il a dessiné les lignes suivantes :



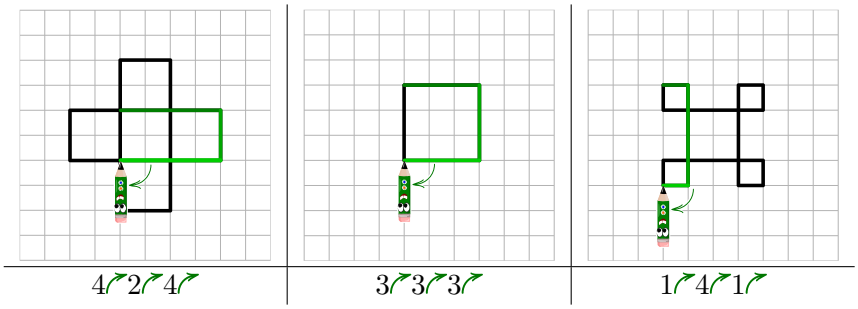
Après avoir répété ces étapes, il a dessiné en tout les lignes suivantes :



Après les deux répétitions suivantes, l'image ressemble à cela :



La bonne réponse est donc la réponse B).
Le robot peut par ailleurs aussi dessiner les trois autres figures, il faut simplement lui donner d'autres nombres :



C'est de l'informatique !

Le robot graphique de cet exercice ne peut exécuter que des programmes très simples. Le langage de programmation que le robot comprend n'a que trois nombres comme instructions. Chaque programme doit être composé d'exactement trois nombres suivis par le symbole de rotation ↱. C'est par ailleurs intégré de manière fixe dans le programme que le robot répète les instructions quatre fois, que cela soit désiré ou non.
La plupart des robots et ordinateurs comprennent des langages (de programmation) beaucoup plus complexes. La plupart de ces langages ont les même propriétés de base :

1. Les programmes peuvent être composés de n'importe quel nombre d'instructions qui sont exécutées les unes après les autres en tant que *séquence*.
2. Des instructions de répétition, appelées *boucles*, peuvent être utilisées, sans que cela soit obligatoire.
3. Il existe également des *instructions conditionnelles* qui permettent différentes exécutions du programme suivant son état.

Lorsqu'un langage de programmation contient des boucles et des instructions conditionnelles, on peut les utiliser pour calculer tout ce qui est calculable. En informatique, on dit de tels langages de programmation qu'ils sont universels — ou *Turing-complets*.
Le robot de cet exercice est un environnement classique avec lequel on peut apprendre à programmer. On s'imagine une tortue qui dessine des lignes à la place du robot. De telles *tortues graphiques* peuvent par exemple être réalisées avec XLogoOnline ou TigerJython.

Mots clés et sites web

Tortue graphique
- `https://fr.wikipedia.org/wiki/Programmation_structur%C3%A9e`
- `https://fr.wikipedia.org/wiki/Turing-complet`
- `https://en.wikipedia.org/wiki/Turtle_graphics`
- `https://fr.wikipedia.org/wiki/Logo_(langage)`
- `https://xlogo.inf.ethz.ch/`
- `http://www.tigerjython.ch/`

A. Auteurs des exercices

Haim Averbuch
Michelle Barnett
Michael Barot
Daniela Bezáková
Anton Chukhnov
Allira Crowe
Andrew Csizmadia
Christian Datzko
Susanne Datzko
Marissa Engels
Olivier Ens
Martin Guggisberg
Vernon Gutierrez
Juraj Hromkovič
Alisher Ikramov
Thomas Ioannou
Tiberiu Iorgulescu
Yong-ju Jeon
Felipe Jiménez
Anna Laura John
Mile Jovanov
Adem Khachnaoui
Injoo Kim
Vaidotas Kinčius
Jia-Ling Koh
Dennis Komm
Anja Koron
Bohdan Kudrenko
Regula Lacher
Karolína Mayerová
Anna Morpurgo
Tom Naughton
Pia Niemelä
Henry Ong
Wolfgang Pohl
Stavroula Prantsoudi
Nol Premasathian
J.P. Pretti
Taras Shpot
Jacqueline Staub
Bundit Thanasopon
Monika Tomcsányiová
Peter Tomcsányi
Nicole Trachsler
Troy Vasiga
Ela Veza
Florentina Voboril
Khairul A. Mohamad Zaki

B. Sponsoring : Concours 2019

http://www.haslerstiftung.ch/

http://www.roborobo.ch/

http://www.baerli-biber.ch/

http://www.verkehrshaus.ch/
Musée des transports, Lucerne

Standortförderung beim Amt für Wirtschaft und Arbeit Kanton Zürich

i-factory (Musée des transports, Lucerne)

http://www.ubs.com/

http://www.bbv.ch/

http://www.presentex.ch/

http://www.oxocard.ch/
OXOcard
OXON

http://www.diartis.ch/
Diartis AG

https://educatec.ch/
educaTEC

http://senarclens.com/
Senarclens Leu & Partner

http://www.abz.inf.ethz.ch/
Ausbildungs- und Beratungszentrum für Informatikunterricht der ETH Zürich.

http://www.hepl.ch/
Haute école pédagogique du canton de Vaud

http://www.phlu.ch/
Pädagogische Hochschule Luzern

https://www.fhnw.ch/de/die-fhnw/hochschulen/ph
Pädagogische Hochschule FHNW

http://www.supsi.ch/home/supsi.html
La Scuola universitaria professionale della Svizzera italiana (SUPSI)

https://www.zhdk.ch/
Zürcher Hochschule der Künste

C. Offres ultérieures

Devenez vous aussi membre de la SSIE
`http://svia-ssie-ssii.ch/la-societe/devenir-membre/`
et soutenez le Castor Informatique par votre adhésion
Peuvent devenir membre ordinaire de la SSIE toutes les personnes qui enseignent dans une école primaire, secondaire, professionnelle, un lycée, une haute école ou donnent des cours de formation ou de formation continue.
Les écoles, les associations et autres organisations peuvent être admises en tant que membre collectif.