



**INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA**

Exercices et solutions 2019 Tous les âges

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

Éditeurs :

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

010100110101011001001001
010000010010110101010011
010100110100100101000101
001011010101001101010011
010010010100100100100001

SS!E

www.svia-ssie-ssii.ch
schweizerischerverein für informatik in d
erausbildung // société suisse pour l'infor
matique dans l'enseignement // società sviz
zera per l'informatica nell'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 1^{er} novembre 2019. Ce cahier d'exercice a été produit le 2 janvier 2020 avec le logiciel de mise en page L^AT_EX.



Les exercices sont protégés par une licence Creative Commons Paternité – Pas d'Utilisation Commerciale – Partage dans les Mêmes Conditions 4.0 International. Les auteurs sont cités en p. 104.



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.bebas.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 fautive 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 fautive	-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

Ont collaboré au Castor Informatique 2019	i
Préambule	ii
Table des matières	iv
1. Piscine !	1
2. Papier à gratter	5
3. Kiosque	7
4. Beavercoins	9
5. Signaux de fumée	11
6. Tampon	13
7. Quelle tour ?	17
8. Voyage dans l'espace	19
9. Robot graphique	21
10. Rangoli	25
11. Chapeaux et bonshommes de neige	27
12. Superstar	31
13. Drapeaux colorés	35
14. Rangement du lave-vaisselle	37
15. Message des anciens castors	39
16. Caractères chinois colorés	41
17. Garniture de hamburger	45
18. Signaux de fumée	49
19. Tours particulières	51
20. Boules instables	53
21. Caméra de surveillance	57
22. Un sac de bonbons	61



23. Réseau de castors	65
24. Signaux lumineux	69
25. Quipu	73
26. Tempête de neige	75
27. Quel bonheur que les arbres !	77
28. Compression vidéo	81
29. Scierie	85
30. Gare de triage	87
31. Jeu de billes	91
32. Quatre poissons	93
33. Job de vacances	97
34. Carte au trésor	101
A. Auteurs des exercices	104
B. Sponsoring : Concours 2019	105
C. Offres ultérieures	107



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 :

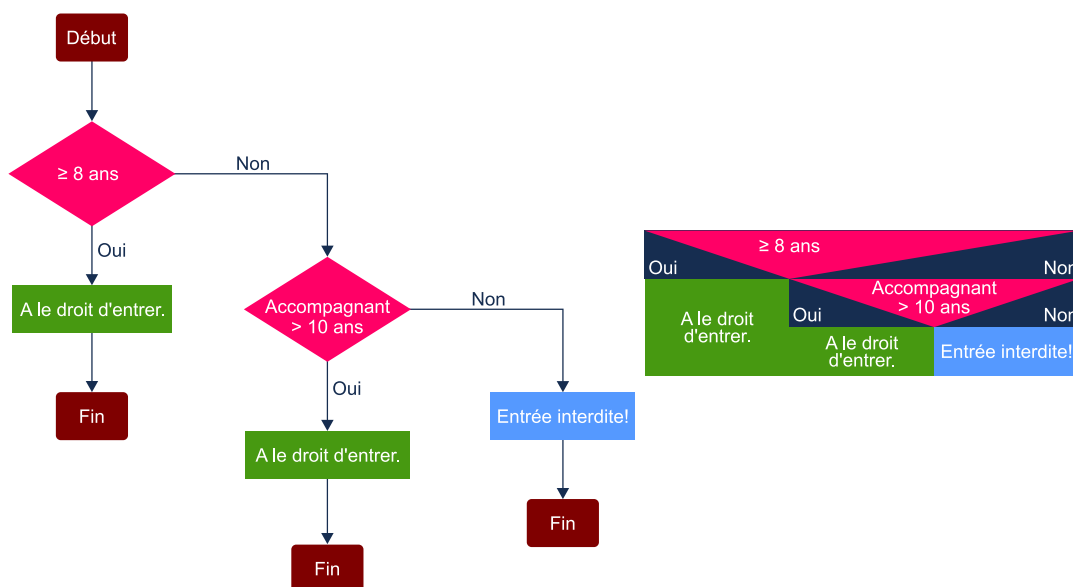
Alors elle a le droit d'entrer dans la piscine

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/Instruction_conditionnelle_(programmation))



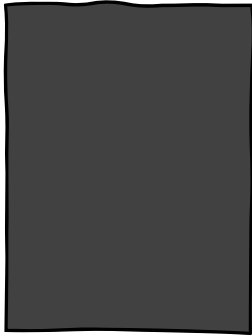
- https://fr.wikipedia.org/wiki/Organigramme_de_programmation
- <https://fr.wikipedia.org/wiki/Structogramme>



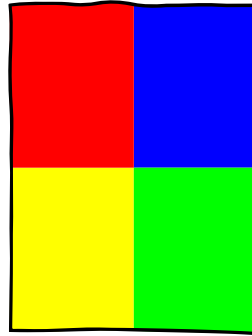


2. Papier à gratter

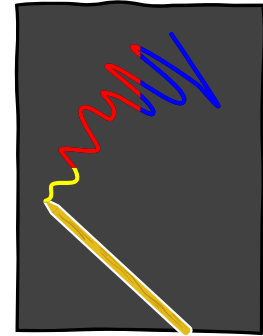
Tu peux faire des dessins colorés facilement avec du papier à gratter. Tu enlèves la couche du dessus avec un stylet en bois et la couche colorée du dessous apparaît.



Au départ, le papier à gratter est recouvert d'une couche noire.



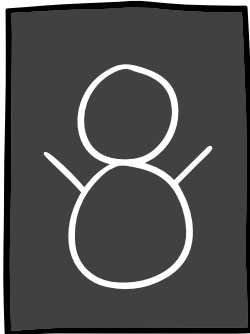
Derrière la couche noire se cachent ces quatre couleurs.



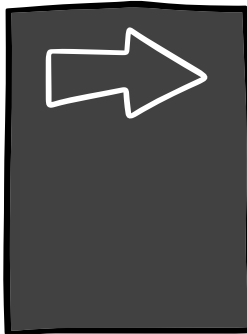
Une partie de la couche noire a été grattée avec le stylet en bois. Tu peux y voir les couleurs cachées en dessous.

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

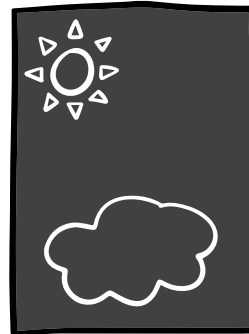
A)



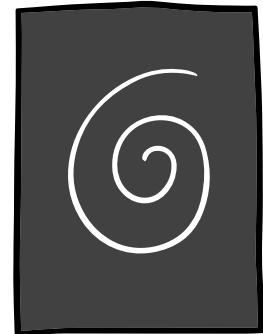
B)



C)

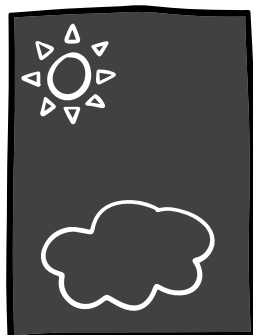


D)



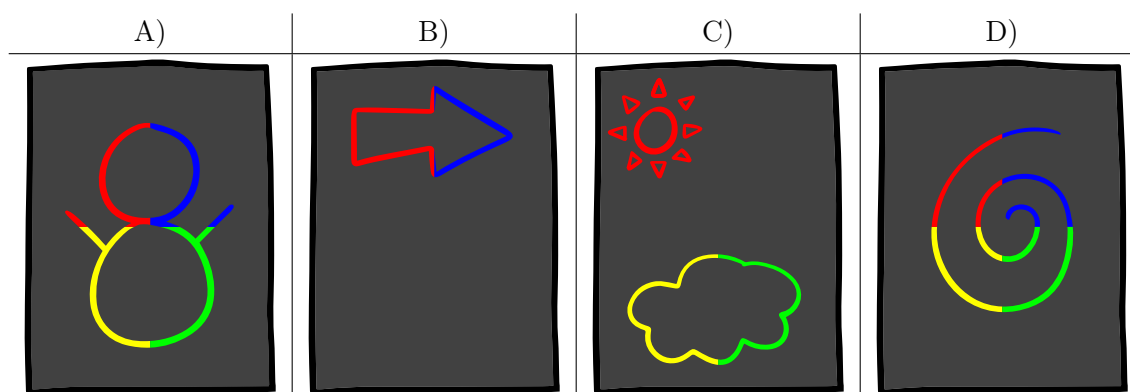


Solution



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

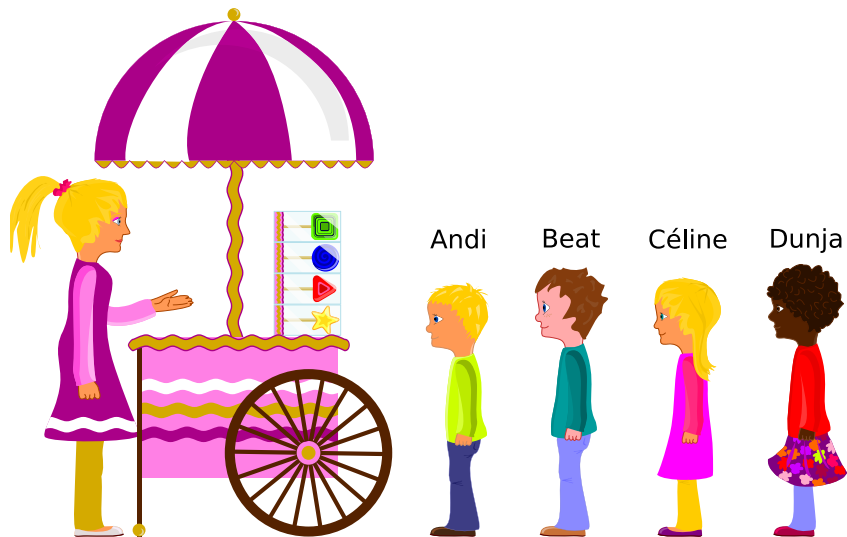
Retouche d'image, calque, décomposition

- [https://fr.wikipedia.org/wiki/Calque_\(infographie\)](https://fr.wikipedia.org/wiki/Calque_(infographie))
- [https://en.wikipedia.org/wiki/Decomposition_\(computer_science\)](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

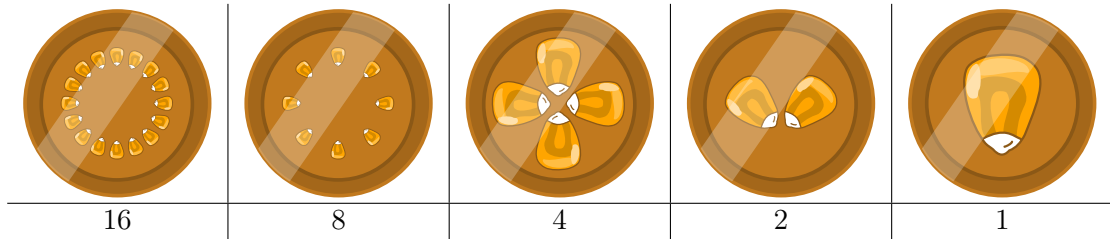
File, pile

- [https://fr.wikipedia.org/wiki/File_\(structure_de_donn%C3%A9es\)](https://fr.wikipedia.org/wiki/File_(structure_de_donn%C3%A9es))
- [https://fr.wikipedia.org/wiki/Pile_\(informatique\)](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. Ils 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

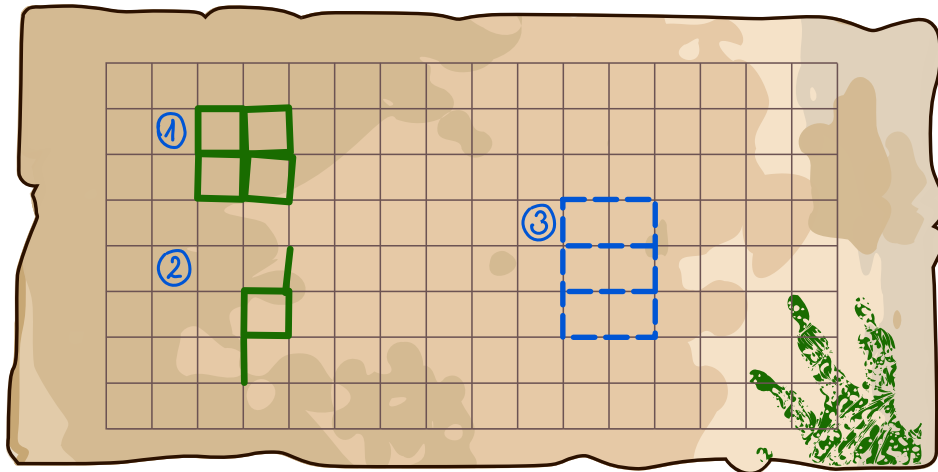
— https://fr.wikipedia.org/wiki/Code_correcteur



6. Tampon

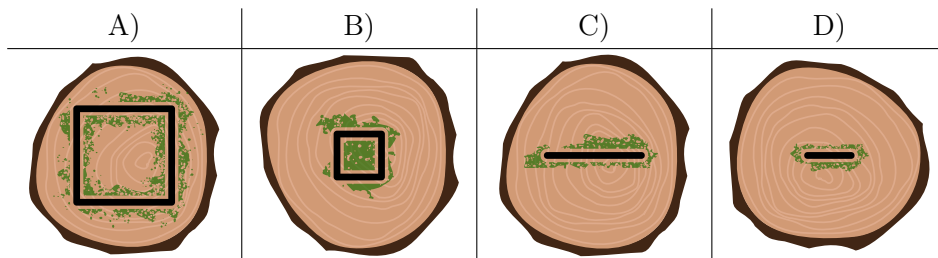
Le castor Paul a les quatre tampons A, B, C et D comme montré plus bas. Paul a tamponné les deux motifs ① et ② avec ces tampons.

- 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.



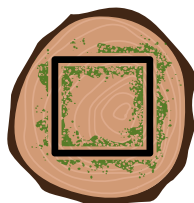
Paul aimerait à présent obtenir le motif ③. Sa sœur Marie affirme qu'elle ne doit tamponner que deux fois pour faire le motif.

Quel tampon Marie utiliserait-elle ?

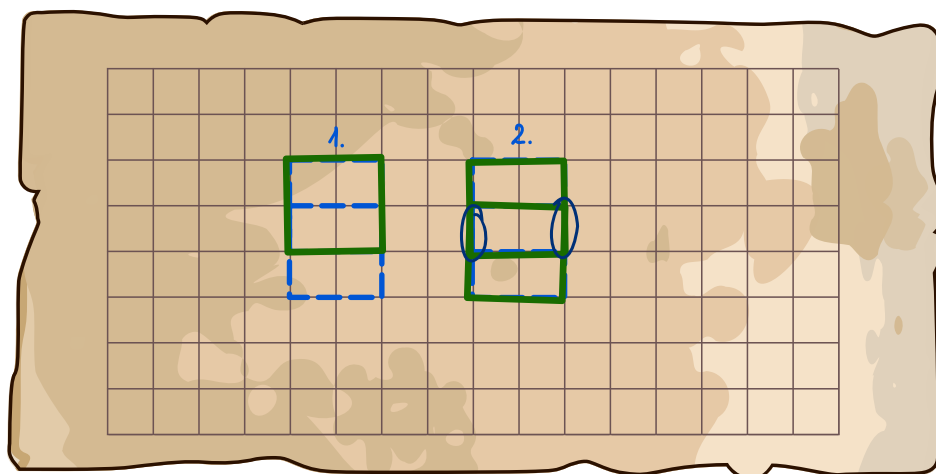




Solution



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 tâches 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 nombreuses 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_\(informatique_th%C3%A9orique\)](https://fr.wikipedia.org/wiki/Th%C3%A9orie_de_la_complexit%C3%A9_(informatique_th%C3%A9orique))

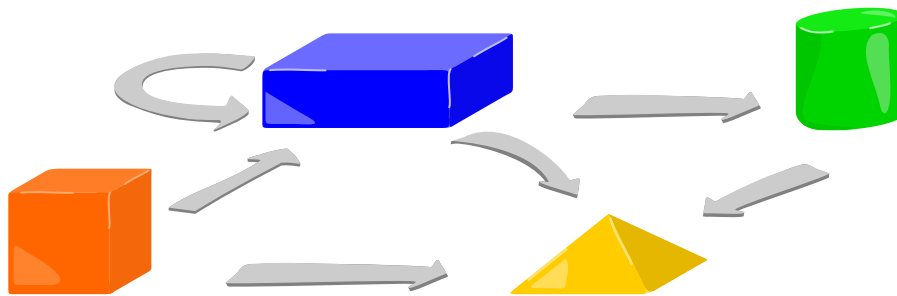




7. Quelle tour ?

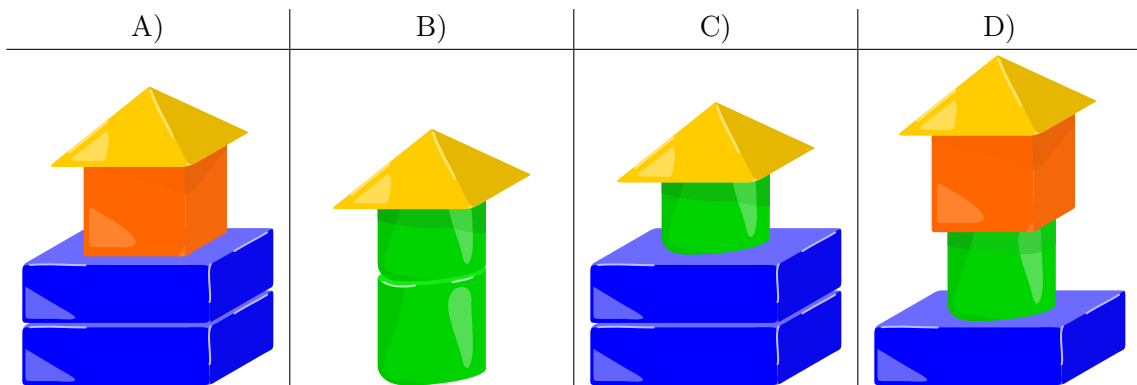
La petite sœur de Léon a établi des règles selon lesquelles des plots peuvent être empilés. Elle les a représentées par des flèches sur un dessin. Il y a en plus les règles suivantes :

- 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 ?





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 a 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'autres 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

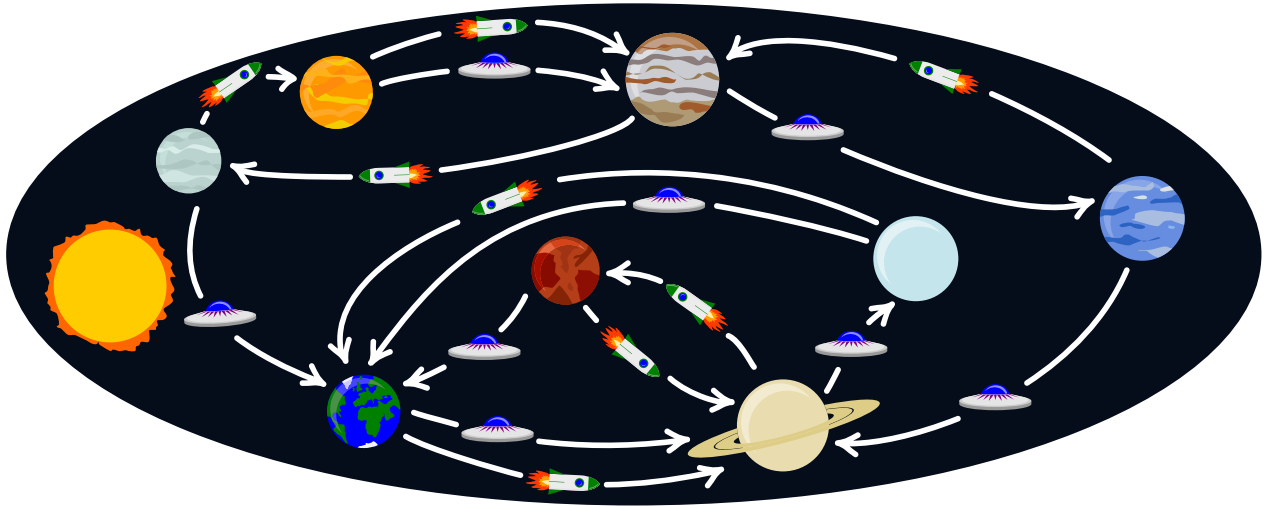
Automate fini non déterministe

- 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_\(math%C3%A9matiques_discr%C3%A8tes\)](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.

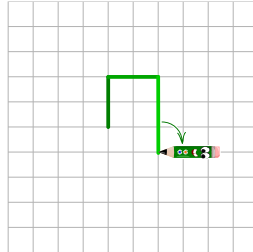
On donne les nombres 2↗2↗3↗ au robot. À quoi les lignes dessinées ressemblent-elles ?

A)	B)	C)	D)

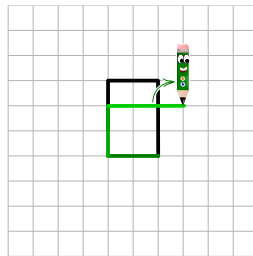


Solution

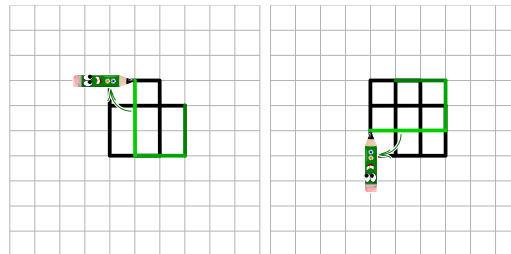
Les nombres 2↗2↗3↗ signifient que le robot commence par avancer de deux cases, fais un quart de tour vers la droite, avance à nouveau de deux cases, fais un quart de tour vers la droite, avance de trois cases et fais encore une fois un quart de tour vers la droite. Quand il a terminé, il a dessiné les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



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

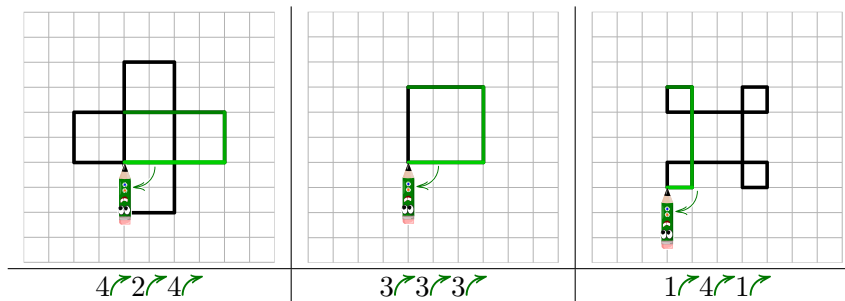


Après les deux répétitions suivantes, l'image ressemble à cela :



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'exactly 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 mêmes 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://fr.wikipedia.org/wiki/Logo_(langage))
- <https://xlogo.inf.ethz.ch/>
- <http://www.tigerjython.ch/>

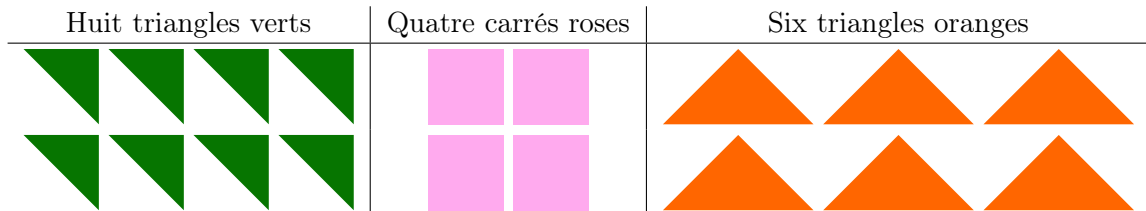




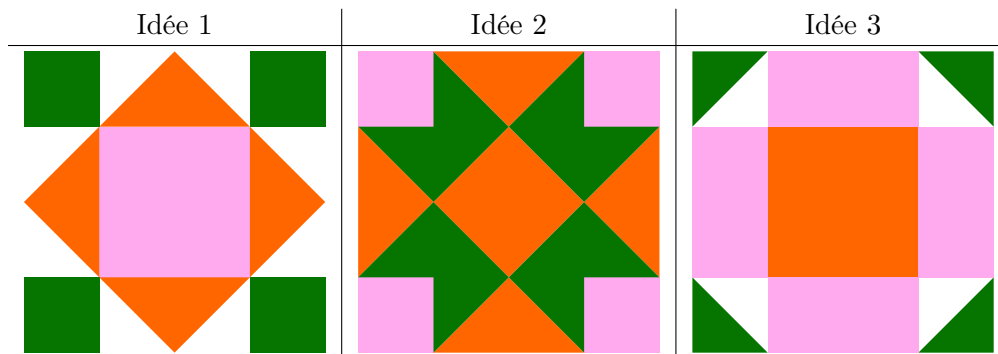
10. Rangoli

Le rangoli est une forme d'art venant d'Inde. Les rangolis sont des motifs faits sur le sol. Ces motifs sont le plus souvent symétriques.

Priya a des pierres de trois formes différentes pour son rangoli : huit triangles verts, quatre carrés roses et six triangles oranges. Les pierres de la même couleur ont la même taille :



Elle trouve les idées de rangoli suivantes sur un site Internet (les surfaces blanches restent vides) :



Laquelle des trois idées de rangoli Priya peut-elle faire avec ses pierres ?

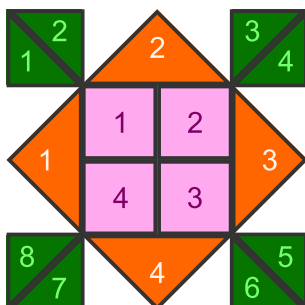
- A) Seulement l'idée 1.
- B) Seulement l'idée 2.
- C) Seulement l'idée 3.
- D) Les trois idées.



Solution

Priya ne peut faire que A) seulement l'idée 1 avec ses pierres.

L'illustration suivante compte les différentes pierres utilisées pour l'idée 1. Comme elle n'a besoin au maximum que du nombre de pierres qu'elle a à disposition, elle peut faire le motif de l'idée 1 :



Pour le motif de l'idée 2, elle aurait besoin de douze triangles verts, car chacune des quatre figures vertes de l'idée 2 nécessite trois triangles. Priya n'a que huit triangles verts et ne peut donc pas faire le motif de l'idée 2.

Pour le motif de l'idée 3, elle aurait besoin de huit triangles roses, car chacune des quatre figures roses de l'idée 3 nécessite deux carrés roses. Priya n'a que quatre carrés roses et ne peut donc pas faire le motif de l'idée 3.

Comme elle ne peut faire ni le motif de l'idée 2, ni celui de l'idée 3, la réponse D) ne peut pas être correcte.

C'est de l'informatique !

Le *rangoli* est une forme d'art venant d'Inde pour laquelle on utilise traditionnellement du riz et de la farine colorée, mais aussi du sable de couleur et des fleurs. Les rangolis ont surtout un but décoratif, mais sont aussi associés à des traditions locales et familiales ainsi qu'à des vœux de protection. Certaines traditions religieuses sont aussi associées aux rangolis.

Dans cet exercice, il fallait décomposer une forme complexe en formes plus petites que l'on pouvait comparer aux formes de base. On savait alors combien des formes de base étaient nécessaires. On appelle ce processus *décomposition*, il est souvent utilisé en informatique.

On appelle la comparaison des formes décomposées avec des formes de base *filtrage par motif*. Le filtrage par motif a une grande importance en informatique, il n'est pas utilisé que pour la recherche de motifs graphiques, mais aussi par exemple pour la recherche de mots dans un texte, de noms de fichiers dans un système de fichiers ou pour la comparaison de séquences génétiques lors de la recherche de criminels.

Mots clés et sites web

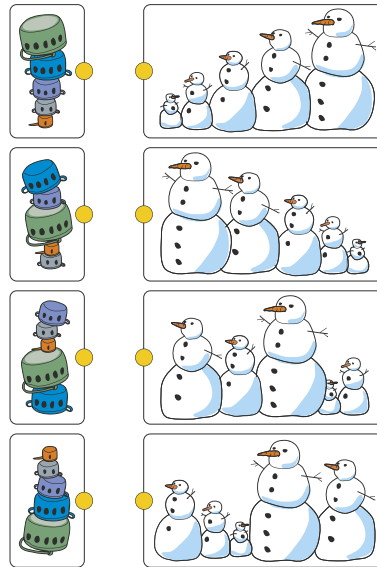
Décomposition, filtrage par motif

- <https://fr.wikipedia.org/wiki/Rangoli>
- https://fr.wikipedia.org/wiki/Filtrage_par_motif
- [https://en.wikipedia.org/wiki/Decomposition_\(computer_science\)](https://en.wikipedia.org/wiki/Decomposition_(computer_science))

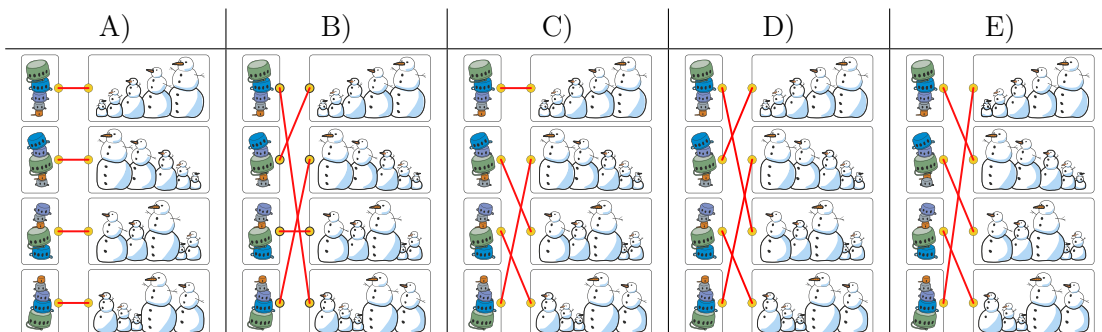


11. Chapeaux et bonshommes de neige

Cinq chapeaux empilés sont distribués, en commençant en haut et allant vers le bas, à cinq bonshommes de neige en commençant à gauche et finissant à droite. À la fin, chaque bonhomme de neige devrait recevoir un chapeau à sa taille.

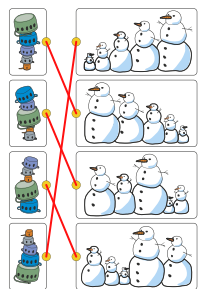


Quelle pile de chapeaux correspond à quelle rangée de bonshommes de neige ?





Solution



La bonne réponse est E)

La première pile de chapeaux correspond à la deuxième rangée de bonshommes de neige. Le premier bonhomme de neige est le plus grand bonhomme de neige et le premier chapeau est le plus grand chapeau. Le deuxième bonhomme de neige est le deuxième plus grand bonhomme de neige et reçoit le deuxième plus grand chapeau, et ainsi de suite.

La deuxième pile de chapeaux correspond à la troisième rangée de bonshommes de neige. Le premier bonhomme de neige est le deuxième plus grand bonhomme de neige et le premier chapeau est le deuxième plus grand chapeau. Les bonshommes de neige restants reçoivent également des chapeaux à leur taille.

La troisième pile de chapeaux correspond à la quatrième rangée de bonshommes de neige. Le premier bonhomme de neige reçoit le troisième chapeau par ordre de grandeur qui est situé tout en haut de la pile. Ici aussi, les bonshommes de neige restants reçoivent des chapeaux à leur taille.

La quatrième pile de chapeaux correspond à la première rangée de bonshommes de neige. Ici, les bonshommes de neige sont classés du plus petit au plus grand de gauche à droite. De même, les chapeaux sont classés du plus petit au plus grand de haut en bas. Ainsi, chaque bonhomme de neige reçoit le chapeau à sa taille.

La réponse A) n'est pas juste, car dans la première rangée, le plus petit bonhomme de neige recevrait le plus grand chapeau.

La réponse B) n'est pas juste, car dans la première rangée, le plus petit bonhomme de neige recevrait le deuxième plus grand chapeau.

La réponse C) n'est pas juste, car comme pour la réponse A), dans la première rangée, le plus petit bonhomme de neige recevrait le plus grand chapeau.

La réponse D) n'est pas juste, car comme pour la réponse B), dans la première rangée, le plus petit bonhomme de neige recevrait le deuxième plus grand chapeau.

C'est de l'informatique !

Lorsque l'on distribue comme dans cet exercice «son» chapeau à chaque bonhomme de neige sans changer l'ordre des bonshommes de neige ou des chapeaux, on appelle cela un *homomorphisme*. Dans chaque rangée de bonshommes de neige et chaque pile de chapeaux, un premier, deuxième, troisième, quatrième et cinquième éléments qui doivent se correspondre lorsqu'on les met en relation sont définis.

On peut considérer les suites des tailles des bonshommes de neige et des chapeaux comme des *tuples* : des collections ordonnées d'éléments. On ne considère alors que la propriété intéressante, c'est-à-dire la taille des bonshommes de neige et des chapeaux. Par exemple, les tailles des chapeaux de la première pile sont (5, 4, 3, 2, 1) et les tailles des bonshommes de neige de la première rangée sont (1, 2, 3, 4, 5). Pour mettre la pile en relation avec la première rangée, les quintuplets devraient être égaux.

Les chapeaux sont représentés comme une *pile* (*stack* en anglais). Une pile est une structure de données avec laquelle on ne peut faire que peu de choses (appelées *opérations*) : *dépiler* (enlever



l'élément du dessus, *pop* en anglais) et *empiler* (ajouter un élément sur la pile, *push* en anglais). Parfois, on peut aussi *voir l'élément de tête* (*top* en anglais) et *vérifier si la pile est vide* (*empty* en anglais).

Les bonshommes de neige sont représentés comme une *file* (*queue* en anglais). Une file est une structure de données avec laquelle on ne peut pas non plus faire beaucoup d'opérations: *défiler* (enlever l'élément du devant, *dequeue* en anglais) et *enfiler* (ajouter un élément à la fin de la file, *enqueue* en anglais). Parfois, on peut aussi *voir l'élément du devant* (*first* en anglais) et *vérifier si la file est vide* (*empty* en anglais).

On peut de cette manière enlever et comparer un élément à chaque fois: dès que le chapeau et le bonhomme de neige ne correspondent plus, la solution ne peut plus être correcte. Cet argument a été utilisé dans l'explication de la solution: pour les réponses fausses, on a expliqué pourquoi le premier chapeau de la pile correspondant à la première rangée de bonshommes de neige n'allait pas au premier bonhomme de neige de la rangée.

On ne sait du reste pas exactement si la pile de chapeaux est vraiment une pile: Les chapeaux pourraient avoir été ajoutés chaque fois en dessous de la pile (ce qui en ferait en réalité une file). De même, les bonshommes de neige pourraient avoir dépassé les autres et s'être mis à l'avant (et la rangée de bonshommes de neige fonctionnerait comme une pile).

Mots clés et sites web

Homomorphisme, pile, file

- <https://en.wikipedia.org/wiki/Homomorphism>
- [https://fr.wikipedia.org/wiki/File_\(structure_de_donn%C3%A9es\)](https://fr.wikipedia.org/wiki/File_(structure_de_donn%C3%A9es))
- [https://fr.wikipedia.org/wiki/Pile_\(informatique\)](https://fr.wikipedia.org/wiki/Pile_(informatique))



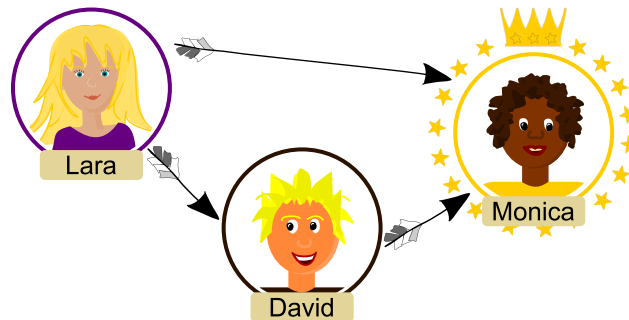


12. Superstar

Le réseau social AdoGram permet à ses membres de suivre d'autres membres. Il y a aussi des groupes de membres sur AdoGram. Dans un groupe, un membre est une superstar si...

- ... la superstar est suivie par tous les membres du groupe, et...
- ... elle-même ne suit aucun membre du groupe.

Dans le groupe suivant, Lara suit Monica et David, David suit Monica, mais Monica ne suit personne. Monica est donc une superstar :



Un autre groupe est composé de six membres : Andrea, Dican, Françoise, Gianni, Robin et Stefan. Il se suivent ainsi :

- Andrea suit Dican, Françoise et Gianni.
- Dican suit Françoise, Gianni et Robin.
- Françoise suit Gianni.
- Robin suit Dican, Françoise et Gianni.
- Stefan suit Andrea, Dican, Françoise, Gianni et Robin.

Est-ce qu'il y a une superstar dans ce groupe ?

- A) Oui, Françoise est une superstar dans ce groupe.
- B) Oui, Gianni est une superstar dans ce groupe.
- C) Oui, Stefan est une superstar dans ce groupe.
- D) Oui, Françoise et Gianni sont les deux des superstars dans ce groupe.
- E) Non, ce groupe n'a pas de superstar.



Solution

La bonne réponse est B) Oui, Gianni est une superstar dans ce groupe.

Les deux conditions sont remplies :

- Tous les autres membres du groupe suivent Gianni.
- Gianni lui-même ne suit personne dans le groupe.

Les autres réponses sont fausses.

La réponse A) ne peut pas être juste, car Françoise suit Gianni. En plus, Gianni ne suit pas Françoise.

La réponse C) ne peut pas être juste. Stefan est même une sorte d'anti-superstar dans le groupe : il suit tous les autres membres, mais aucun autre membre du groupe ne le suit.

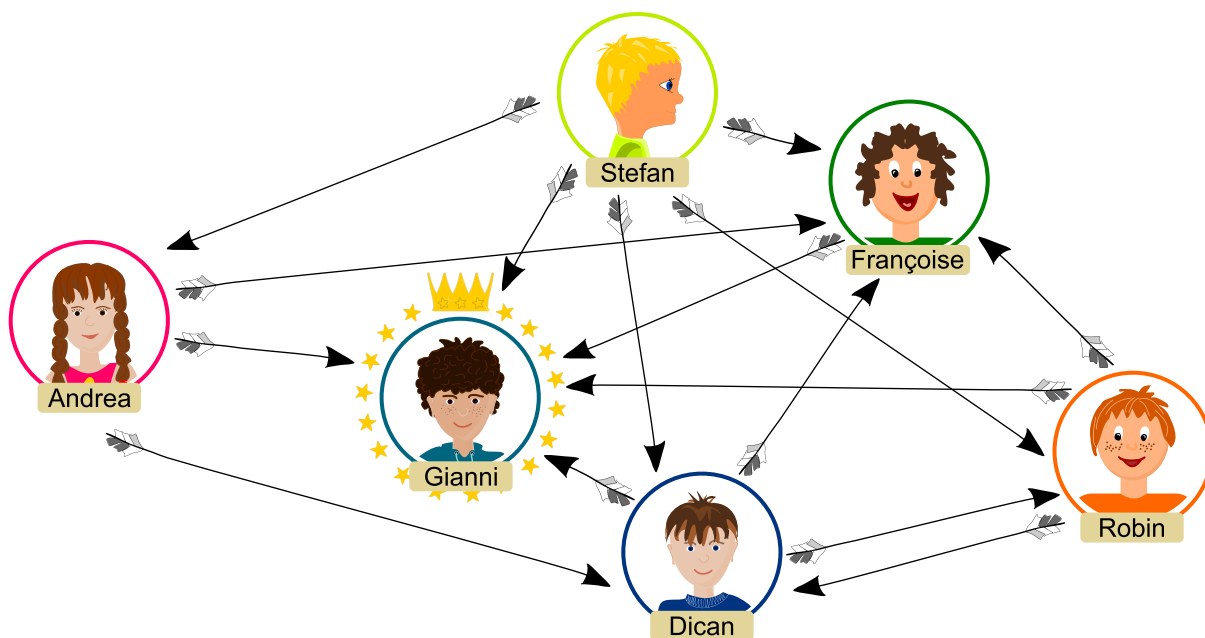
La réponse D) ne peut pas être juste. Non seulement Françoise n'est pas une superstar du groupe, comme décrit ci-dessus, mais un groupe ne peut avoir qu'une superstar : une superstar ne suit personne dans le groupe, mais tous les autres membres du groupe la suivent ; s'il y avait deux superstars, elles devraient se suivre l'une l'autre et ne seraient donc plus des superstars.

La réponse E) est également fautive : comme décrit plus haut, Gianni est une superstar dans le groupe.

C'est de l'informatique !

Les réseaux sociaux comme le fictif AdoGram fonctionnent car leurs membres ont des *relations* entre eux. Souvent, dans les réseaux sociaux, ces relations vont dans une direction (Andrea *suit* Dican). Bien sûr, cela arrive aussi que deux membres se suivent l'un l'autre (Dican suit Robin et Robin suit Dican).

On peut représenter ces relations par un *graphe*, comme dans l'exemple de cet exercice. On utilise des flèches pour montrer qui suit qui. Dans un graphe, on appelle les membres des *nœuds* et les flèches des *arêtes*. Comme les arêtes ont un sens, il s'agit d'un *graphe orienté*. Le graphe pour cet exercice serait donc comme cela :



Les réseaux sociaux ayant beaucoup de membres correspondent souvent à de très grands graphes. Les entreprises qui gèrent ces réseaux sociaux s'intéressent aux particularités que l'on peut trouver dans ces graphes. Une superstar n'est alors peut-être plus quelqu'un que tout le monde suit, mais quelqu'un que beaucoup de personnes suivent. Si une superstar fait par exemple de la publicité pour un certain produit, cette publicité est vue par beaucoup plus de membres que si un membre



quelconque la faisait. C'est pour cela que les superstars se vantent d'avoir beaucoup de *followers* et ont parfois recours à des méthodes douteuses pour augmenter leur nombre de followers : plus ils ont de followers, plus ils peuvent faire de profit en faisant de la publicité ou du placement de produit. Ils deviennent des *influenceurs*, des personnes qui en influencent d'autres.

Pour exploiter ce marché, les grands réseaux sociaux emploient déjà beaucoup de moyens afin d'augmenter la qualité de la relation de follower. Il suffit parfois d'avoir cherché un certain nom sur un navigateur ou d'avoir été localisé près d'un certain endroit par smartphone pour que les réseaux sociaux nous proposent de «suivre» une personne ou son magasin.

Mots clés et sites web

Réseaux sociaux, graphes

- https://fr.wikipedia.org/wiki/Analyse_des_r%C3%A9seaux_sociaux
- [https://fr.wikipedia.org/wiki/Graphe_\(math%C3%A9matiques_discr%C3%A8tes\)](https://fr.wikipedia.org/wiki/Graphe_(math%C3%A9matiques_discr%C3%A8tes))





13. Drapeaux colorés

Le constructeur de bateaux des castors fait d'excellents bateaux. Chaque castor veut en avoir un. Mais comment peuvent-ils faire la différence entre leurs bateaux s'ils sont tous pareils ?

Les castors décident de personnaliser chaque bateau avec un drapeau. Un drapeau des castors ressemble à cela :

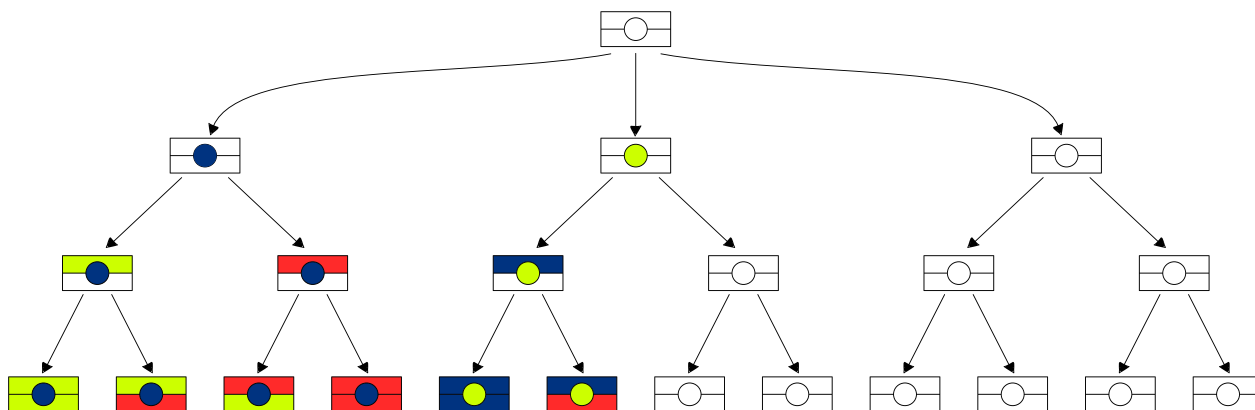


Ils se mettent d'accord sur trois couleurs pour les différentes surfaces du drapeau : rouge, vert clair et bleu foncé. Les deux bandes peuvent avoir la même couleur, mais le cercle au centre doit être d'une autre couleur que les deux bandes :



Pour ne pas perdre la vue d'ensemble, les castors dessinent un diagramme qui montre toutes les combinaisons de couleurs possibles pour les drapeaux, mais ils ne l'ont pas terminé.

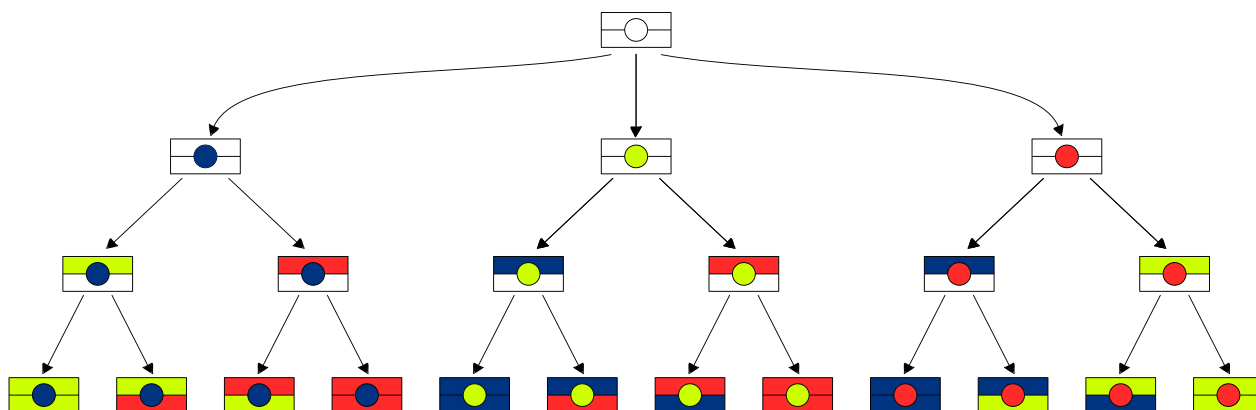
Complète le diagramme pour les castors. Il y a plusieurs bonnes solutions, cela suffit si tu en indiques une. Colorie les surfaces sur le diagramme.





Solution

Une bonne solution possible est :



De manière générale, les combinaisons du diagramme sont justes si...

- ... dans la deuxième rangée, le cercle à droite est rouge, ...
- ... dans la troisième rangée, en fonction de la couleur du cercle, la bande du haut a une couleur différente pour chaque couleur du cercle (l'ordre est égal), ...
- ... dans la quatrième rangée, en fonction de la couleur du cercle, la bande du bas a une couleur différente pour chaque couleur du cercle (l'ordre est égal).

C'est de l'informatique !

On doit parfois résoudre des problèmes compliqués. Pour cela, c'est utile de faire une liste de toutes les solutions possibles. En informatique, c'est spécialement important de pouvoir faire une liste de toutes les solutions possibles de manière efficace.

C'est souvent utile d'avoir une méthode systématique pour faire une liste afin de n'oublier aucune solution possible et de ne pas avoir de solution à double. Les structures de données comme l'*arbre* utilisé par les castors aident à trouver systématiquement toutes les solutions. Dans chaque rangée, toutes les valeurs possibles (ici les couleurs) d'une partie de l'objet (ici une surface du drapeau) sont dessinées côte à côte. On appelle le drapeau de la première rangée (pas coloré) la *racine*, et les drapeaux complètement colorés de la rangée du bas des *feuilles*. Un branchement est appelé *branche*. Comme chaque branches correspond à une valeur possible pour remplir la surface à colorier, on peut être sûr que toutes les solutions possibles sont sur les feuilles.

Mots clés et sites web

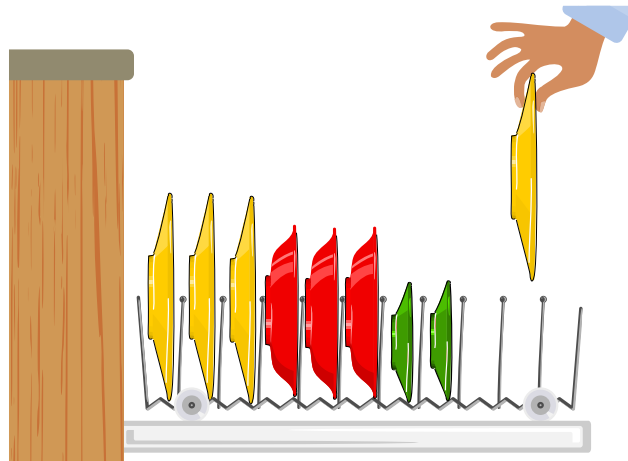
Arbre

- https://fr.wikipedia.org/wiki/Arbre_enracin%C3%A9
- <https://en.wikipedia.org/wiki/Enumeration>



14. Rangement du lave-vaisselle

Tobias range ses assiettes dans le lave-vaisselle de manière à ce que les grandes assiettes soient tout à gauche, les assiettes à soupe au milieu et les petites assiettes à droite. Il n'y a pas de trou entre les assiettes. Après le souper, il doit ajouter une grande assiette dans le lave-vaisselle. Il veut toucher le moins d'assiettes possible dans le lave-vaisselle en les déplaçant mais veut conserver le même classement.



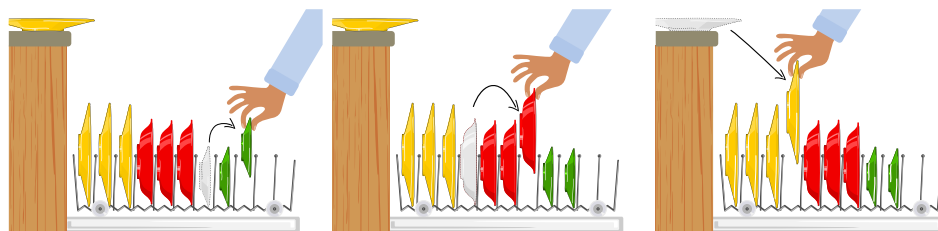
Combien d'assiettes du lave-vaisselle doit-il toucher pour pouvoir ensuite ranger la grande assiette à la bonne place ?

- A) 0
- B) 1
- C) 2
- D) 3
- E) 5
- F) 8



Solution

La méthode la plus rapide pour Tobias est de déplacer la petite assiette de gauche vers la droite, de mettre l'assiette à soupe de gauche dans la place ainsi libérée et de mettre la grande assiette supplémentaire dans la place libérée par l'assiette à soupe de manière à ce que la nouvelle assiette soit tout à droite des grandes assiettes. Il a ainsi touché deux assiettes du lave-vaisselle, la réponse C) est donc juste.



Cela ne peut pas aller plus vite, car la grande assiette doit être mise à la place d'une grande assiette ou de l'assiette à soupe de gauche (il faut donc toucher au moins une assiette du lave-vaisselle). De plus, il faut remettre l'assiette touchée à une autre place : si c'est une grande assiette, le problème est à nouveau le même, et si c'est une assiette à soupe, celle-ci doit être mise à la place d'une autre assiette à soupe ou de la petite assiette de gauche (il faut donc toucher au moins une deuxième assiette du lave-vaisselle).

C'est de l'informatique !

Dans cet exercice, il s'agit en fin de compte d'insérer un nouvel *Element* dans une *liste déjà triée*. De tels procédés ont souvent lieu dans les ordinateurs, cela vaut donc la peine de réfléchir à comment le faire efficacement.

Dans cet exercice, le « coût » pour le déplacement d'une seule assiette dans le lave-vaisselle est relativement haut. Par contre, le type de chaque assiette est déterminé très rapidement. Beaucoup d'assiettes du même type sont présentes simultanément dans le lave-vaisselle. Cela vaut donc la peine de trouver une solution particulière pour ce problème spécial en ne déplaçant que deux éléments. Pour un ordinateur, c'est en règle générale plus facile de trouver une place correcte pour l'élément à ajouter dans une liste triée et de décaler ensuite d'une place tous les éléments suivants.

Cette sorte de tri basé sur la comparaison s'appelle dans ce cas *tri par insertion* (*insertion sort* en anglais). Elle fait partie des méthodes de tri simples mais pas particulièrement efficaces. D'autres méthodes de tri semblables sont le *tri à bulles* (*bubble sort* en anglais) ou le *tri par sélection* (*selection sort* en anglais). Le *tri rapide* (*quicksort* en anglais), également courant, se base sur le principe *diviser pour régner* et est en revanche nettement plus rapide, surtout pour les longues listes.

Mots clés et sites web

Tri

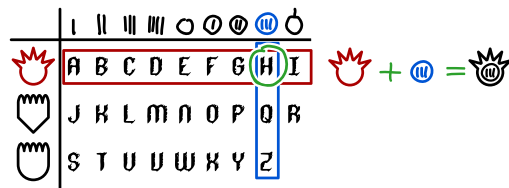
- https://fr.wikipedia.org/wiki/Algorithme_de_tri
- https://fr.wikipedia.org/wiki/Tri_par_insertion
- <https://www.youtube.com/watch?v=R0alU37913U>



15. Message des anciens castors

La castor Dara a trouvé un très ancien morceau de bois tout en bas du barrage des castors. Des symboles inconnus sont gravés dans le bois. Dara suppose qu'il s'agit d'une table de chiffrement datant du temps où les anciens castor habitaient le barrage.

Dara observe la table longtemps et pense savoir comment elle fonctionne: les symboles inconnus sont une combinaison des symboles indiqués dans les lignes et colonnes. La lettre «H» serait donc chiffrée de la manière suivante:



Dara se rappelle qu'elle a déjà vu de tels symboles à un autre endroit du barrage. En effet, il y est écrit:






Que signifie le message des anciens castors ?



- A) SAVEWATER
- B) CLEARDAYS
- C) SAVEMYDAM
- D) CAREFORME



Solution

Le premier symbole est constitué des formes  et . On le trouve donc dans la troisième ligne et première colonne : c'est la lettre S. Seules les réponses A) ou C) peuvent donc être justes.

Les deuxièmes, troisièmes et quatrièmes symboles sont les mêmes dans les réponses A) et C). Les symboles correspondent aux lettres A, V et E. La cinquième lettre est différente entre les deux réponses. Le symbole est constitué des formes  et . Cela correspond à la lettre W. La réponse A) est donc correcte. Les quatre derniers symboles correspondent en effet aux lettres A, T, E et R.

Il existe un raccourci pour résoudre l'exercice. Si l'on commence par le dernier symbole à la place de commencer par le premier, on voit qu'il est différent pour chaque réponse. Les formes  et  du dernier symbole correspondent à la lettre R et seule la réponse A) se termine par cette lettre.

C'est de l'informatique !

La sécurité des données est aujourd'hui une tâche sociale importante. Une des méthodes pour empêcher la lecture non-autorisée de données est de les chiffrer. La science étudiant le chiffrement de données (la *cryptographie*) a déjà au moins 3500 ans. Une des plus vieilles méthodes de *chiffrement* (parfois aussi appelé *cryptage*) est basée sur le remplacement de lettres par d'autres lettres ou symboles. Lors du chiffrement, un *texte clair* est transformé en un *texte chiffré* à l'aide d'une *clé*. On appelle *déchiffrement* la reconstruction du texte clair à partir du texte chiffré à l'aide de la clé. Lorsque l'on retrouve le texte clair à partir du texte chiffré sans avoir connaissance de la clé, on appelle cela *décryptage*.

La méthode de chiffrement utilisée dans cet exercice est appelée *chiffrement par substitution mono-alphabétique*. Cette méthode remplace chaque lettre par un seul nouveau symbole. On utilise pour ça souvent des systèmes dont on se rappelle facilement. Le système utilisé dans cet exercice ressemble à l'alphabet franc-maçon. Les cryptanalystes, qui décryptent de tels textes, utiliseraient des techniques spéciales comme l'analyse fréquentielle ou les *n*-grammes avec le texte chiffré pour associer la bonne lettre à chaque symbole. Edgar Allan Poe a montré dans sa nouvelle de 1843 *Le Scarabée d'or* (*The Gold-Bug* en version originale) que cela était généralement possible avec les chiffrements par substitution monoalphabétique

Qu'est-ce que Dara aurait pu faire si elle n'avait pas eu la table à disposition, mais connaissait les quatre significations possibles ? Elle aurait pu déterminer que les deuxième et sixième ainsi que les quatrième et huitième symboles sont les mêmes. En partant du principe que le chiffrement est une substitution monoalphabétique, elle n'aurait plus eu qu'à trouver le texte ayant les mêmes deuxième et sixième ainsi que quatrième et huitième lettres, ce qui n'est le cas que pour la réponse A).

Mots clés et sites web

Cryptographie, chiffrement par substitution monoalphabétique

- <https://fr.wikipedia.org/wiki/Cryptographie>
- https://fr.wikipedia.org/wiki/Chiffrement_par_substitution
- https://fr.wikipedia.org/wiki/Chiffre_des_francs-ma%C3%A7ons
- https://fr.wikipedia.org/wiki/Analyse_fr%C3%A9quentielle
- <https://fr.wikipedia.org/wiki/N-gramme>
- https://fr.wikipedia.org/wiki/Le_Scarab%C3%A9e_d'or
- <http://users.telenet.be/d.rijmenants/en/goldbug.htm>



16. Caractères chinois colorés

La structure des caractères chinois nous paraît étrange. Pour mieux comprendre comment certains caractères chinois sont assemblés, on peut s’imaginer le schéma suivant qui les sépare en cinq parties, haut ▲, bas ▣, gauche ▢, droite ▣ et centre ★ :



Ces parties peuvent être assemblées en quatre structures :

Structure	Structure gauche-centre-droite	Structure gauche-droite	Structure haut-centre-bas	Structure haut-bas
Exemple de caractère	川	儿	三	吕
Exemple d’analyse				

Quelle analyse montre la bonne structure pour les trois caractères chinois 劳, 二 et 八 d’après le schéma ?

- A)
- B)
- C)
- D)

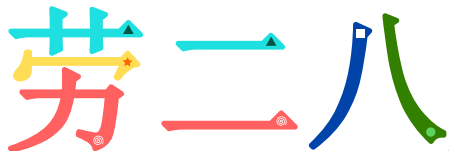


Solution

Le premier caractère, 劳, a une structure haut-centre-bas, le trait du haut est donc bleu clair , celui du centre jaune et celui du bas rose .

Le deuxième caractère, 二, a une structure haut-bas, le trait du haut est donc bleu clair et celui du bas rose .

Le troisième caractère, 八, a une structure gauche-droite, le trait de gauche est donc bleu foncé et celui de droite vert .



La bonne réponse est donc B)

Dans la réponse A), le deuxième caractère, 二, est bien analysé, mais les mauvaises couleurs sont associées aux caractères 劳 et 八: la couleur du haut est fausse pour 劳 et les deux couleurs sont inversées pour 八.

Dans la réponse C), tous les caractères sont mal analysés. Les couleurs du centre et du bas du premier caractère ne sont pas justes, celle du haut du deuxième caractère est fausse ainsi que les deux couleurs du troisième caractère.

Dans la réponse D), le caractère 八 est bien analysé, mais les couleurs du haut et du bas de 劳 sont fausses ainsi que les deux couleurs de 二.

C'est de l'informatique !

Les caractères chinois (*sinogrammes*) sont composés de caractères assemblés de manière complexe. Même les versions simplifiées ont plus de 200 éléments de base (*radicaux* ou *clés des sinogrammes*) différents, à partir desquels les caractères sont assemblés. Ceux-ci sont écrits l'un à côté de l'autre ou l'un en dessous de l'autre, formant des structures comme expliqué dans cet exercice. Ainsi, des milliers de caractères différents peuvent être assemblés. Si l'on doit apprendre à écrire avec ces caractères, il faut comprendre comment fonctionne l'assemblage. Pour cela, des couleurs sont souvent utilisées, comme dans cet exercice. L'alphabet latin utilisé chez nous fonctionne différemment: une lettre correspond à un son (avec des exceptions comme «eau», qui est prononcé [o] et non pas [eay]).

Qu'est-ce que cela a à voir avec l'informatique? Premièrement, de tels caractères doivent pouvoir être représentés à l'ordinateur. Il existe plusieurs approches pour cela, l'une d'entre elles utilise les radicaux décrits dans cet exercice. Deuxièmement, on doit avoir la possibilité de chercher des mots, par exemple dans un dictionnaire ou un lexique. Les radicaux les plus utilisés aujourd'hui viennent d'un dictionnaire qui a été élaboré entre 1710 et 1716 sous l'empereur Kangxi. Ce dictionnaire est ordonné d'après le nombre de traits de chaque radical.

Mots clés et sites web

Sinogramme

- https://fr.wikipedia.org/wiki/C1%C3%A9_d%27un_sinogramme
- https://fr.wikipedia.org/wiki/Sinogramme_simplifi%C3%A9
- https://fr.wikipedia.org/wiki/Codage_des_caract%C3%A8res_chinois
- https://fr.wikipedia.org/wiki/M%C3%A9thodes_de_saisie_et_d%27encodage_du_chinois
- https://fr.wikipedia.org/wiki/Dictionnaire_de_caract%C3%A8res_de_Kangxi
- https://fr.wikipedia.org/wiki/Alphabet_latin
- [https://fr.wikipedia.org/wiki/Eau_\(trigramme\)](https://fr.wikipedia.org/wiki/Eau_(trigramme))



Les caractères chinois sont :





- 川 : https://fr.wikipedia.org/wiki/Radical_47
- 儿 : https://fr.wikipedia.org/wiki/Radical_10
- 吕 : [https://en.wikipedia.org/wiki/L%C3%BC_\(surname\)](https://en.wikipedia.org/wiki/L%C3%BC_(surname))
- 二 : https://fr.wikipedia.org/wiki/Radical_7
- 三 : [https://fr.wikipedia.org/wiki/3_\(nombre\)](https://fr.wikipedia.org/wiki/3_(nombre))
- 八 : https://fr.wikipedia.org/wiki/Radical_12
- 劳 : <https://fr.wiktionary.org/wiki/%E5%8A%B3>



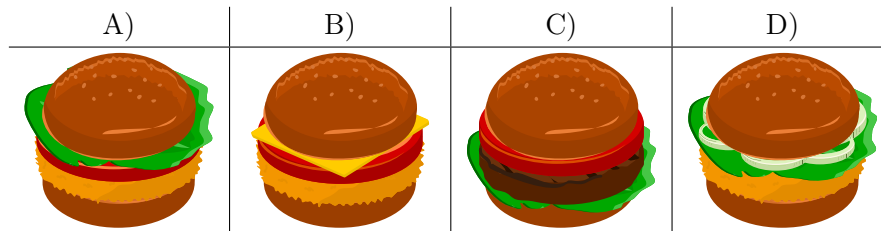


17. Garniture de hamburger

CastorBurger propose six ingrédients (A, B, C, D, E et F) pour ses hamburgers faits maison. Le tableau suivant liste les ingrédients de quatre exemples de hamburgers, pas forcément dans le même ordre que dans l'exemple de hamburger :

Hamburger				
Ingrédients	C, F	A, B, E	B, E, F	B, C, D

Quel hamburger a les ingrédients A, E et F ?





Solution

Pour déterminer à quel ingrédient correspond chaque lettre, il faut toujours comparer deux hamburgers :

Hamburgers comparés		Lettre en commun	Ingrédient en commun
		F	
		C	
		B	
		B (déjà identifié)	
		E	

Deux des ingrédients ne sont présents que dans un seul hamburger. Comme l'on a déjà identifié toutes les autres lettres, on peut identifier les ingrédients correspondants comme cela :

Hamburger particulier	Lettre particulière	Ingrédient particulier
	A	
	D	

Le hamburger recherché avec les ingrédients A, E et F doit donc contenir les ingrédients ,

et , et ce n'est le cas que du hamburger de la réponse A)





C'est de l'informatique !

L'*inférence* est la base de beaucoup de raisonnements, y compris en informatique. Pour résoudre cet exercice, on doit l'appliquer intensément : en comparant des hamburgers ayant des ingrédients en commun, on peut déduire des informations inconnues jusque-là (quel ingrédient correspond à quelle lettre).

Dans le cas de cet exercice, les ingrédients communs entre deux hamburgers correspondent à l'*intersection* des deux *ensembles* que sont les hamburgers. Elle ne contient que les ingrédients présents dans les deux ensembles. La première comparaison s'écrirait $\{C, F\} \cap \{B, E, F\} = \{F\}$. La contrepartie de l'intersection serait l'*union* $\{C, F\} \cup \{B, E, F\} = \{B, C, E, F\}$, elle contient tous les éléments qui sont présents dans au moins l'un des ensembles.

Pour déterminer quels ingrédients ne sont présents que dans un seul hamburger, on peut utiliser la *différence ensembliste*. Elle ne contient que les ingrédients du premier ensemble qui ne sont pas présents dans le deuxième ensemble. Par exemple, pour le premier hamburger particulier, on pourrait écrire : $\{A, B, E\} \setminus (\{C, F\} \cup \{B, E, F\} \cup \{B, C, D\}) = \{A, B, E\} \setminus \{B, C, D, E, F\} = \{A\}$.

La théorie des ensembles est peut-être connue du cours de mathématique. En informatique, elle est par exemple utilisée avec les bases de données. On peut aussi appliquer la théorie des ensembles telle quelle dans le domaine de la logique, appelé aussi algèbre de Boole, qui est utilisé souvent en informatique.

Mots clés et sites web

Inférence, théorie des ensembles, logique





- [https://fr.wikipedia.org/wiki/Inf%C3%A9rence_\(logique\)](https://fr.wikipedia.org/wiki/Inf%C3%A9rence_(logique))
- <https://fr.wikipedia.org/wiki/Ensemble>
- [https://fr.wikipedia.org/wiki/Alg%C3%A8bre_de_Boole_\(logique\)](https://fr.wikipedia.org/wiki/Alg%C3%A8bre_de_Boole_(logique))





18. 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. Ils interprètent le signal de fumée comme cela :



Comme ce n'est aucun des signaux de fumée convenus, ils supposent qu'ils ont mal interprété l'un des nuages de fumée : l'un des petits nuages de fumée devrait en fait être grand ou l'un des grands nuages de fumée devrait en fait être petit.

Que voudrait dire le signal de fumée si exactement un nuage de fumée avait été mal interprété ?

- A) Ce sera orageux.
- B) Ce sera pluvieux.
- C) Ce sera nuageux.
- D) Ce sera ensoleillé.



Solution

Cinq signaux de fumée différents sont possibles si exactement un nuage de fumée a été mal interprété. Si l'on interprète le premier, deuxième, quatrième ou cinquième nuage de fumée différemment, on n'obtient aucun des quatre signaux de fumée convenus. Par contre, si l'on interprète le troisième nuage de fumée comme un petit nuage, cela donne la bonne réponse C) «Ce sera nuageux».

On peut aussi comparer le signal de fumée interprété aux quatre signaux de fumée convenus et regarder combien de nuages de fumée sont différents. Pour le signal «Ce sera orageux», deux nuages de fumée sont différents (le plus haut et le plus bas), pour le signal «Ce sera pluvieux», trois nuages de fumée sont différents (les deux plus hauts et le plus bas), pour le signal «Ce sera nuageux», un nuage de fumée est différent (celui du milieu, ce qui en fait la bonne réponse comme décrit ci-dessus) et pour le signal «Ce sera ensoleillé», quatre nuages de fumée sont différents (tous sauf le plus haut).

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

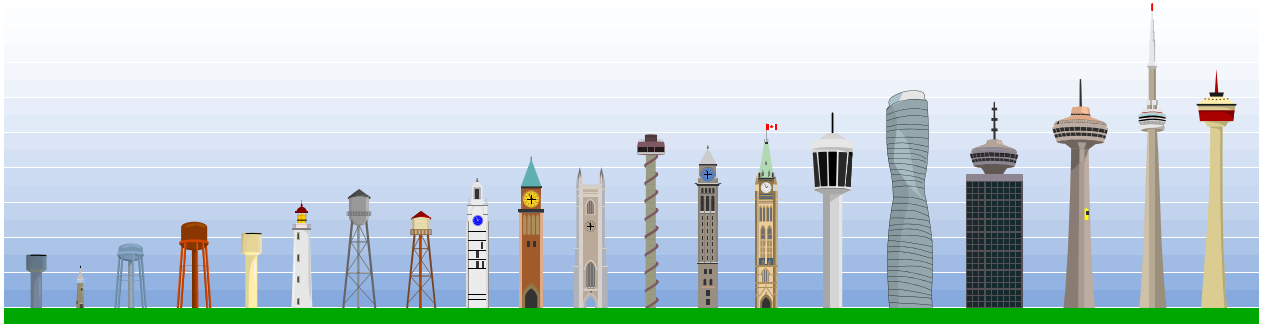
Code correcteur

— https://fr.wikipedia.org/wiki/Code_correcteur



19. Tours particulières

Une tour est considérée comme particulière lorsque toutes les tours à sa gauche sont plus petites qu'elle et toutes les tours à sa droite sont plus grandes qu'elle.



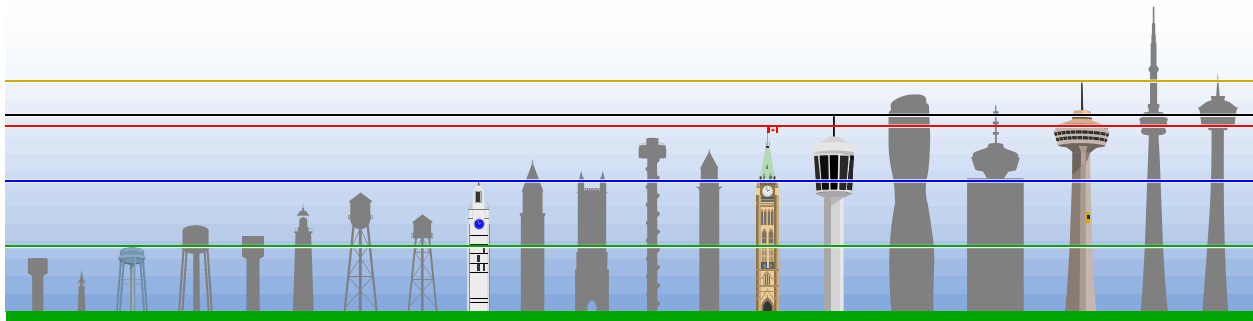
Combien de tours sont particulières sur le dessin ?

- A) 4
- B) 5
- C) 6
- D) 7



Solution

Comme on peut le voir aux lignes colorées, les cinq tours suivantes sont particulières, la réponse B) est donc correcte :



C'est de l'informatique !

Dans cet exercice, des tours sont comparées sur la base de leur hauteur. De telles comparaisons ont lieu lors de processus de *tri et recherche*, un domaine de l'informatique qui a été étudié en détail.

Il existe beaucoup d'algorithmes de tri différents qui sont adaptés à différentes applications. L'algorithme de *tri rapide* en est un exemple connu. L'identification de valeurs n'ayant que des valeurs plus petites à leur gauche et plus grandes à leur droite est un élément crucial de l'algorithme de tri rapide. Un tel élément partitionne le tableau à trier en deux sous-tableaux et divise ainsi le problème de tri de départ en deux problèmes plus petits. Cet élément entre les deux sous-tableaux est appelé *pivot*. Contrairement à cet exercice, dans l'algorithme de tri, les éléments à gauche et à droite du pivot ne sont pas plus petits et plus grand que le pivot dès le départ, mais doivent le devenir en procédant à des permutations d'éléments. Ce processus est répété pour chaque sous-tableau jusqu'à ce que chaque sous-tableau ne contienne plus qu'un seul élément... qui est déjà trié. Ce processus *récuratif* consistant à diviser un problème en problèmes plus petits avant de les résoudre est appelé *diviser pour régner*. C'est une manière très répandue de résoudre des problèmes difficiles.

L'algorithme de tri rapide est plus rapide que beaucoup d'autres algorithmes de tri, d'où son nom. Cela vient du fait que, dans le cas habituel, le choix du pivot permet de diviser la taille du tableau par deux. Un tableau de 1000 éléments nécessite donc environ 10 niveaux de division (mathématiquement, ce sont $\log_2(1000)$ niveaux de division). Étant donné que chaque élément doit être comparé au pivot, cela nécessite en tout 10 000 comparaisons. L'ordre de grandeur du nombre de comparaisons nécessaires en utilisant d'autres algorithmes répandus est plutôt de 1 000 000 comparaisons !

Mots clés et sites web

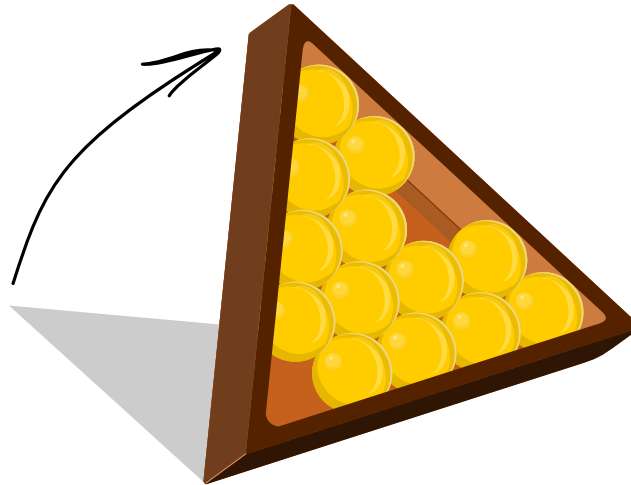
Tri rapide, pivot, diviser pour régner (Divide & Conquer)

- https://fr.wikipedia.org/wiki/Tri_rapide
- [https://fr.wikipedia.org/wiki/Diviser_pour_r%C3%A9gner_\(informatique\)](https://fr.wikipedia.org/wiki/Diviser_pour_r%C3%A9gner_(informatique))
- <https://www.youtube.com/watch?v=ywWBy6J5gz8>



20. Boules instables

Une boîte triangulaire peut contenir quinze boules de la même taille. Deux boules sont retirées de la boîte comme dans le dessin ci-dessous. La boîte est ensuite inclinée sur le côté.

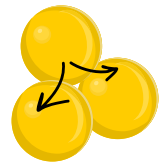


Lorsque l'on incline la boîte, certaines boules peuvent devenir « instables ». Un boule est instable lorsque...

- ... la boule à gauche ou à droite en dessous d'elle a été retirée, ...
- ... ou la boule à gauche ou à droite en dessous d'elle est instable.

Les boules de la rangée du bas sont stables.

Combien des treize boules sont instables ?

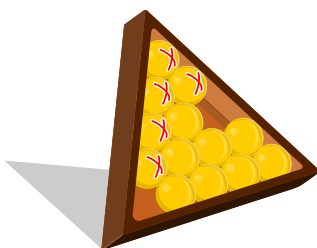


- | | | |
|-----------------|-------------|----------------------|
| A) Aucune boule | F) 5 boules | K) 10 boules |
| B) 1 boule | G) 6 boules | L) 11 boules |
| C) 2 boules | H) 7 boules | M) 12 boules |
| D) 3 boules | I) 8 boules | N) Toutes les boules |
| E) 4 boules | J) 9 boules | |



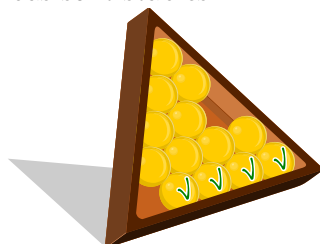
Solution

Cinq boules sont instables. Elles sont marquées d'une croix dans le dessin suivant :

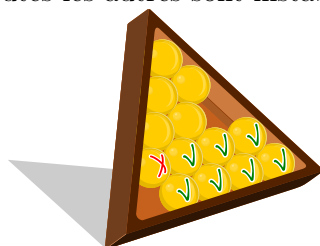


Le plus simple est d'y réfléchir en allant du bas vers le haut :

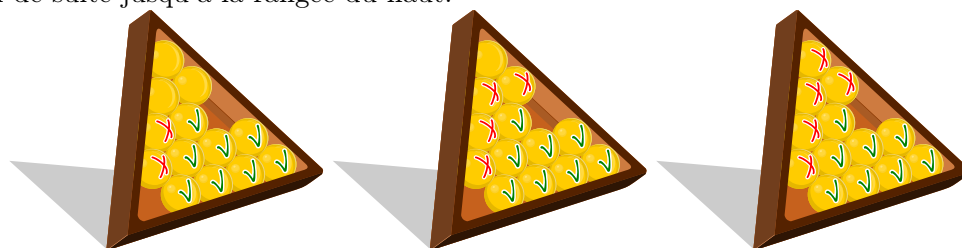
- Toutes les boules de la rangée du bas sont stables.



- Toutes les boules de la deuxième rangée depuis le bas qui ont deux boules stables en dessous d'elles sont également stables. Toutes les autres sont instables.



- Et ainsi de suite jusqu'à la rangée du haut.



C'est de l'informatique !

Il existe deux conditions pour qu'une boule soit classifiée comme instable. La première condition peut être testée directement. Afin de pouvoir tester la deuxième condition, on doit d'abord savoir si l'une des boules de la rangée en dessous est instable. C'est facile pour la rangée du bas, car comme il n'y a aucune boule en dessous, toutes les boules y sont stables. Comme expliqué dans la solution, on peut ensuite tester la rangée en dessus et déterminer quelles boules sont instables. De cette manière, on peut tester systématiquement chaque rangée du bas vers le haut et déterminer si chaque boule est instable.

Le principe d'après lequel une condition dépend du résultat d'une condition similaire s'appelle *récur-sivité*. Les conditions récursives sont construites de manière à ce que le résultat soit évident (*cas de base*, dans notre cas, toutes les boules de la dernière rangée sont stables) soit dépendant du résultat d'autres conditions récursives (*cas général*, dans notre cas, il s'agit de toutes les boules qui ne se



trouvent pas dans la dernière rangée et pour lesquelles il faut donc d'abord tester les boules du dessous).

Le principe de la récursivité est souvent utilisé en informatique. Il permet de résoudre de manière simple et élégante beaucoup de problèmes complexes. C'est également possible de transformer une solution récursive en une solution pas à pas (*itérative*). Un exemple classique pour lequel il existe une solution récursive simple est le jeu des tours de Hanoï.

Mots clés et sites web

Récursivité, tours de Hanoï

— <https://fr.wikipedia.org/wiki/R%C3%A9cursivit%C3%A9>

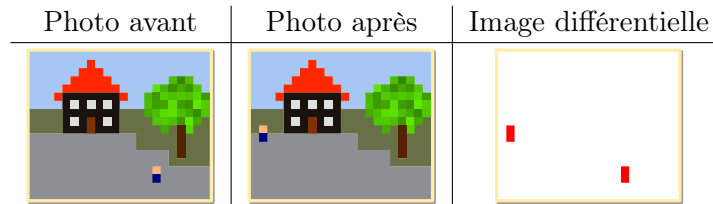
— https://fr.wikipedia.org/wiki/Tours_de_Hano%C3%AF





21. Caméra de surveillance

Sur la place de la gare, une caméra de surveillance fait des photos à intervalles réguliers. Ces photos sont analysées par un ordinateur qui génère une image appelée *image différentielle*. Dans une telle image, les points de l'image qui sont différents entre deux photos consécutives sont mis en évidence. Une personne traverse les deux photos suivantes. C'est mis en évidence sur l'image différentielle à droite :



Cinq événements ont lieu entre la photo suivante et les cinq images différentielles.



Dans quel ordre les événements ont-ils lieu ?

- A) Deux personnes se rencontrent.
La porte de la maison s'ouvre.
Deux personnes partent vers la droite bras dessus, bras dessous.
Le vent se lève.
La porte de la maison se ferme.
- B) La porte de la maison se ferme.
Deux personnes se rencontrent.
Deux personnes partent vers la droite bras dessus, bras dessous.
La porte de la maison s'ouvre.
Le vent se lève.
- C) La porte de la maison s'ouvre.
Deux personnes partent vers la droite bras dessus, bras dessous.
Deux personnes se rencontrent.
Le vent se lève.
La porte de la maison se ferme.
- D) Le vent se lève.
La porte de la maison s'ouvre.
Deux personnes se rencontrent.
Deux personnes partent vers la droite bras dessus, bras dessous.
La porte de la maison se ferme.



Solution

La bonne réponse est B) «La porte de la maison se ferme. Deux personnes se rencontrent. Deux personnes partent vers la droite bras dessus, bras dessous. La porte de la maison s'ouvre. Le vent se lève.»

La première image différentielle montre que le secteur de la photo autour de la porte de la maison a changé. Cela peut être parce que la porte s'est ouverte ou fermée. Théoriquement, on peut ici encore s'imaginer que deux personnes se sont rencontrées dans le secteur de la porte ou partent vers la droite bras dessus, bras dessous.

La deuxième image différentielle montre deux changements à gauche près de l'arbre. Cela ne peut être que parce deux personnes se sont rencontrées, mais ne partent pas encore vers la droite bras dessus, bras dessous. Rien n'a changé dans le secteur de la porte. La porte doit donc soit avoir été ouverte ou fermée plus tôt (et être restée ouverte ou fermée depuis), ou alors, si deux personnes s'y sont rencontrées, elle ne doivent pas avoir bougé depuis.

La troisième image différentielle montre trois changements dans le secteur gauche sur le devant et en bas près de l'arbre. Il y a à nouveau un changement à l'endroit exact où les deux personnes se sont rencontrées plus tôt : soit elles ont les deux bougé, soit elles sont parties. Il y a un large bloc à droite qui montre certainement deux personnes partant vers la droite bras dessus, bras dessous.

La quatrième image montre à nouveau un changement dans le secteur de la porte de la maison et aussi là où les deux personnes sont parties vers la droite bras dessus, bras dessous. Cela signifie que la porte de la maison s'est ouverte ou fermée. Ce n'est pas possible qu'il s'agisse des deux personnes se rencontrant ou partant vers la droite bras dessus, bras dessous, car ces événements ont déjà eu lieu. Là où les deux personnes étaient présentes plus tôt, le changement montre qu'elles ont soit bougé, soit qu'elles sont parties.

La cinquième image montre des changements dans l'arbre qui ont l'air aléatoires. Cela n'a pu être causé que par le vent.

Les photos auraient donc pu ressembler à celles-ci :



C'est de l'informatique !

Aujourd'hui, beaucoup de lieux publics et privés sont surveillés à l'aide de caméras. Comme cela coûterait beaucoup trop cher de faire observer chaque caméra par quelqu'un tout le temps, elle sont analysées directement par des ordinateurs qui détectent les changements dans l'image et informent le propriétaire ou le service de sécurité en cas de besoin. Naturellement, ces ordinateurs sont meilleurs que ne le décrit cet exercice : en règle générale, les petits changements (comme lorsqu'un oiseau vole très loin à travers l'image) ou les changements longs et réguliers (comme lorsque la nuit tombe) sont ignorés et l'information n'est donnée qu'en cas de changements plus importants ou plus rapides. Si l'image actuelle et l'image différentielle sont ensuite également envoyées, une personne peut rapidement décider s'il faut réagir.

La surveillance de lieux publics et privés est controversée. D'un côté, beaucoup d'endroits peuvent être surveillés comme décrit plus haut et une intervention rapide est possible. Si les photos sont aussi enregistrées, on peut ensuite éventuellement avoir des preuves pour confondre les criminels. D'un autre côté, de telles caméras de surveillance peuvent être utilisées abusivement. En 2013 déjà, des poubelles qui n'avaient pas seulement des écrans publicitaires, mais qui enregistraient aussi les profils de déplacement des passants grâce au module WIFI de leurs smartphones, ont été installées dans



des lieux publics de Londres... Sans que les passants ne le sachent ou ne l'autorisent. Les caméras de surveillance peuvent maintenant identifier ces passants à l'aide de la reconnaissance faciale, ou du moins identifier certains groupes de clients potentiels et leur montrer des publicités ciblées sur ces poubelles. Ce qui est raisonnable pour la surveillance de sécurité de bâtiments peut être acceptable dans un supermarché, mais doit au moins être discuté dans la société quand il s'agit de l'espace public pour trouver une voie entre les besoins liés à la sécurité, les intérêts commerciaux et le droit à la sphère privée dans l'espace public.

Mots clés et sites web

Traitement d'images, surveillance, sphère privée

- https://fr.wikipedia.org/wiki/Traitement_d%27images
- https://en.wikipedia.org/wiki/Image_differencing
- <https://gizmodo.com/londons-shutting-down-those-creepy-phone-tracking-sm-1107706580>

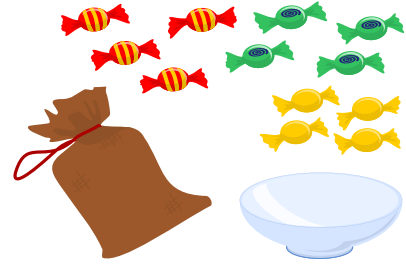




22. Un sac de bonbons

Petra a quatre bonbons rouges, quatre bonbons verts et quatre bonbons jaunes dans un sac opaque. Elle a aussi une coupe vide. Petra et Moritz jouent à un jeu. Pendant trois tours, Moritz peut tirer un bonbon du sac. Les règles suivantes valent pour chaque bonbon :

- Tant que le bonbon tiré est vert, il le met dans la coupe et peut tirer un autre bonbon pendant le même tour.
- Si le bonbon tiré est rouge, Moritz le met dans la coupe et termine le tour.
- Si le bonbon tiré est jaune, Moritz le mange directement sans le mettre dans la coupe et termine le tour.



Combien de bonbons au maximum Moritz peut-il avoir mis dans la coupe à la fin du jeu ?

- | | | |
|------|------|-------|
| A) 0 | F) 5 | K) 10 |
| B) 1 | G) 6 | L) 11 |
| C) 2 | H) 7 | M) 12 |
| D) 3 | I) 8 | |
| E) 4 | J) 9 | |



Solution

La bonne réponse est H) 7.

Dans le meilleur des cas, les quatre bonbons verts sont tirés. Cela signifie d'une part que les quatre bonbons verts sont dans la coupe et d'autre part que Moritz a pu tirer quatre fois un bonbon de plus au cours des trois tours, donc sept bonbons en tout.

Pour les trois bonbons restants, Moritz tire dans le meilleur des cas un bonbon rouge à chaque fois, qui sont ensuite aussi dans la coupe. Cela fait en tout quatre bonbons verts et trois bonbons rouges, donc sept bonbons dans la coupe en tout.

Il ne peut pas y avoir plus de sept bonbons dans la coupe. À chaque tirage, un bonbon au maximum peut être mis dans la coupe, et comme il n'y a que quatre bonbons avec lesquels on peut tirer un bonbon supplémentaire, cela fait au maximum sept bonbons.

L'ordre dans lequel les bonbons sont tirés dans le meilleur des cas est relativement égal tant que le dernier bonbon tiré est un bonbon rouge, car on peut dans ce cas toujours en tirer un de plus grâce aux bonbons verts.

C'est de l'informatique !

Deux des trois règles de l'exercice sont formulées comme des *instructions conditionnelles* : si une certaine condition est remplie, alors une certaine action est exécutée. De telles instructions conditionnelles sont très souvent utilisées en informatique. Souvent, les mots clés de langue anglaise *if* («si» en anglais) et *then* («alors» en anglais) sont utilisés. L'une des règles est formulée de manière à ce que quelque chose soit répété tant qu'une condition est remplie. On appelle cela une *boucle* pour laquelle on utilise souvent le mot de langue anglaise *while* («tant que» en anglais). De tels boucles peuvent aussi être formulées comme des *boucles itératives* qui déterminent un nombre de répétitions fixe.

On pourrait donc formuler le jeu de Petra de la manière suivante :

fixe Tours égal à 3

*tant qu'*il reste un tour :

diminue Tours de 1

tire un bonbon

tant que le bonbon est vert, alors mets-le dans la coupe et tire un bonbon

si le bonbon est rouge, alors mets-le dans la coupe

si le bonbon est jaune, alors mange-le

Pour résoudre l'exercice, on doit *analyser* le programme. Dans un cas simple comme celui de ce programme, on pourrait simplement essayer tous les ordres possibles de bonbons. Cela pourrait même être exécuté par un ordinateur de manière automatisée. L'explication de la solution, par contre, se base sur la compréhension des relations qui permet de prouver qu'une certaine solution est vraie sans exécuter le programme. De telles analyses ne peuvent pas toujours être réalisées par un ordinateur, comme l'a montré la *théorie de la calculabilité*. Donald Knuth, un des grands informaticiens du XX^e siècle, l'a exprimé ainsi : «Faites attention aux erreurs dans ce code ; je n'ai fait que démontrer qu'il était correct, je ne l'ai pas essayé.»

Mots clés et sites web

Instruction conditionnelle, boucle, théorie de la calculabilité

- [https://fr.wikipedia.org/wiki/Instruction_conditionnelle_\(programmation\)](https://fr.wikipedia.org/wiki/Instruction_conditionnelle_(programmation))
- https://fr.wikipedia.org/wiki/Structure_de_contr%C3%B4le#Boucles



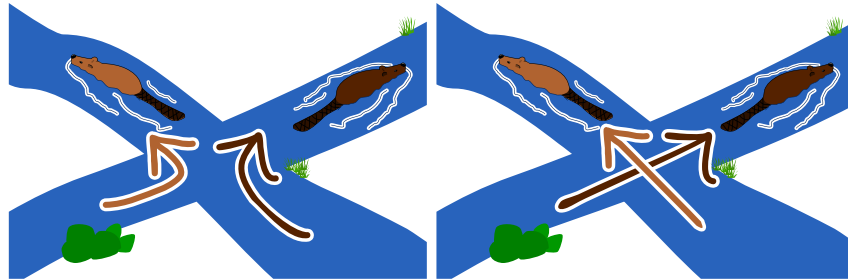
- <https://fr.wikipedia.org/wiki/Calculabilit%C3%A9>
- https://en.wikiquote.org/wiki/Donald_Knuth





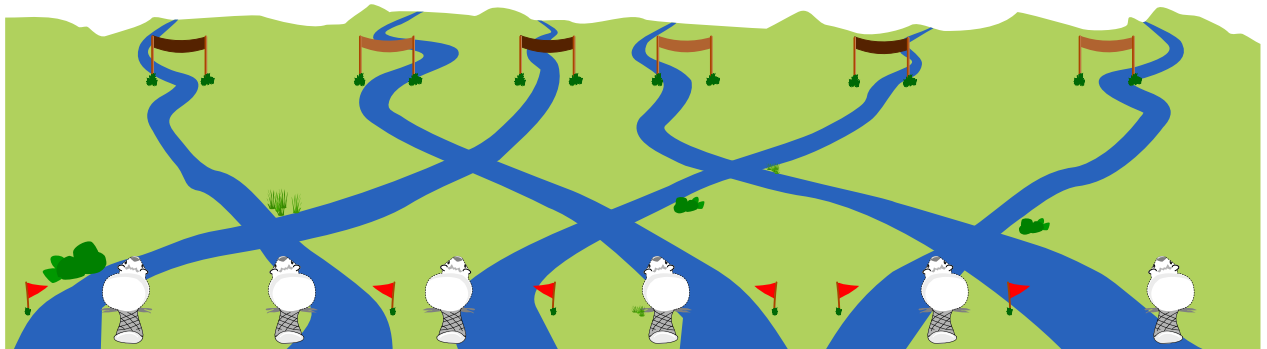
23. Réseau de castors

Trois castors brun clair et trois castors brun foncé nagent dans un système de canaux du bas vers le haut. Deux castors se rencontrent à chaque croisement de deux canaux. Si les deux castors qui se rencontrent sont de couleurs différentes, le castor brun clair continue vers la gauche et le castor brun foncé vers la droite. Sinon, ils continuent simplement chacun dans la même direction.



À la fin, les castors doivent être ordonnés de gauche à droite de la manière suivante : brun foncé, brun clair, brun foncé, brun clair.

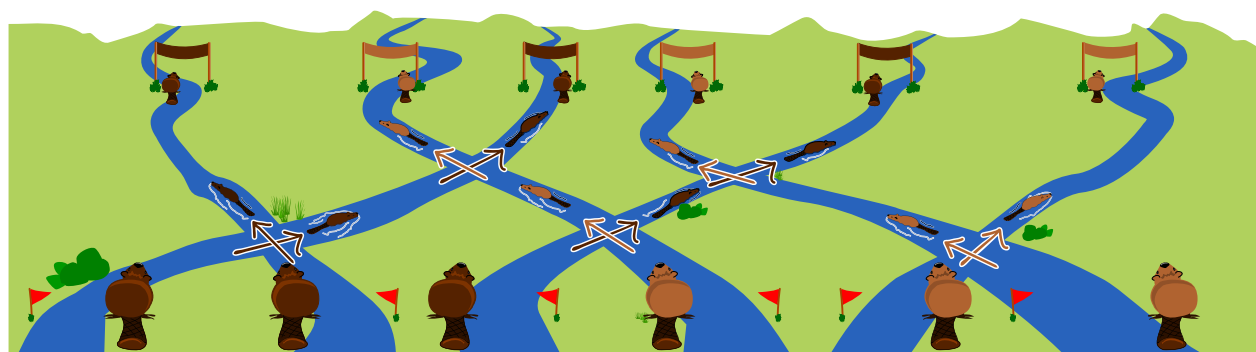
À quelles positions les castors brun clair et brun foncé doivent-ils commencer afin d'arriver dans le bon ordre ?





Solution

Il y a deux bonnes réponses :



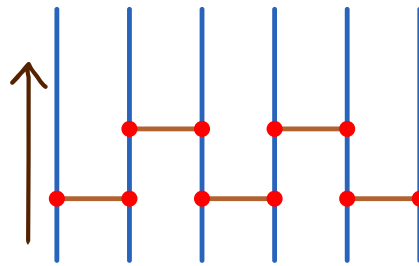
Ce sont également les deux seules réponses justes. En effet, pour que le castor arrivant tout à gauche soit brun foncé, il ne doit pas y avoir de castor brun clair qui passe par le premier croisement à gauche, car celui-ci devrait prendre le canal de gauche. Les deux positions de départ de gauche doivent donc être occupées par des castors brun foncé.

Le même raisonnement vaut pour la position d'arrivée du castor brun clair tout à droite : pour que le castor arrivant tout à droite soit brun clair, deux castors brun clair doivent se rencontrer au premier croisement depuis la droite. Les deux positions de départ de droite doivent donc être occupées par des castors brun clair.

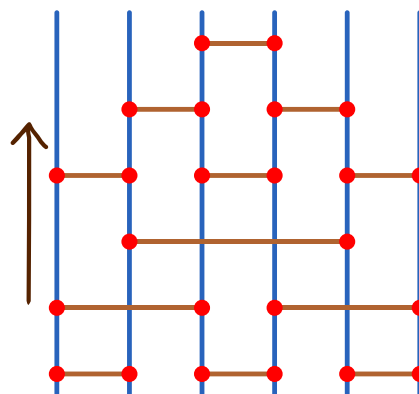
Pour les castors du milieu, cela ne fait pas de différence si le troisième castor brun clair part à gauche et le troisième castor brun foncé part à droite ou l'inverse, car après le premier croisement, le castor brun clair continue de toute façon à gauche et le castor brun foncé à droite.

C'est de l'informatique !

Le système de canaux des castors et la règle déterminant qui nage à gauche et qui nage à droite forment une partie d'un *réseau de tri*. Dans un réseau de tri, des données se déplacent le long d'un fil (les canaux dans cet exercice), et à chaque comparateur (les croisements dans cet exercice), on teste pour décider si les données doivent être échangées ou non. On peut par exemple représenter le castor brun foncé par le chiffre 0 et le castor brun clair par le chiffre 1. Le réseau de tri ressemble alors à cela :



Dans ce diagramme, les lignes bleues représentent les canaux des castors et les points orange reliés par un trait représentent les croisements où les castors changent potentiellement de direction. Ce réseau de tri n'est pas *complet* : il ne va pas forcément disposer tous les castors brun clair à gauche et tous les castors brun foncé à droite. En ce sens, il n'effectue qu'un tri *partiel* des castors. Le réseau de tri suivant est un réseau de tri optimal (complet et minimal) pour cet exercice : en suivant les mêmes règles, ce réseau serait toujours capable de trier tous les castors, quel que soit leur disposition initiale. On peut voir que le réseau de tri ci-dessus est intégré dans le réseau optimal (aux deuxième et troisième lignes depuis le haut) :



Les réseaux de tri sont particulièrement efficaces lorsque les comparaisons peuvent être effectuées parallèlement l'une à l'autre. De tels réseaux de tri optimaux sont difficiles à trouver pour de grands ensembles de données.

Si l'on généralise, on peut aussi se représenter le système de canaux des castors comme le système de câbles d'un réseau informatique comme Internet. Les canaux représentent alors les connexions par câbles directes entre deux routeurs, les intersections. Généralement, des tables de routage fixes permettant de diriger les données vers leur but sont programmées dans de tels routeurs.

Mots clés et sites web

Réseau de tri, réseau informatique, routeur, table de routage

- https://fr.wikipedia.org/wiki/R%C3%A9seau_de_tri
- <http://www.inf.fh-flensburg.de/lang/algorithmen/sortieren/networks/optimal/optimal-sorting-networks.htm>
- <https://www.computernetworkingnotes.com/ccna-study-guide/basic-routing-concepts-and-protocols-explained.html>
- <https://fr.wikipedia.org/wiki/Routage>
- https://fr.wikipedia.org/wiki/Table_de_routage



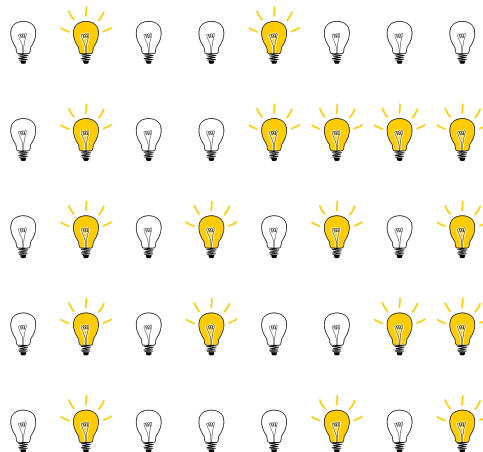


24. Signaux lumineux

Sophie a huit lampes reliées par des interrupteurs et des câbles. Elle peut les utiliser pour envoyer des messages. Pour cela, elle utilise la table de codage suivante, dans laquelle 0 signifie que la lampe correspondante est éteinte (💡) et 1 que la lampe correspondante est allumée (💡):

A : 01000001	J : 01001010	S : 01010011
B : 01000010	K : 01001011	T : 01010100
C : 01000011	L : 01001100	U : 01010101
D : 01000100	M : 01001101	V : 01010110
E : 01000101	N : 01001110	W : 01010111
F : 01000110	O : 01001111	X : 01011000
G : 01000111	P : 01010000	Y : 01011001
H : 01001000	Q : 01010001	Z : 01011010
I : 01001001	R : 01010010	

Sophie envoie à présent les signaux lumineux suivants :



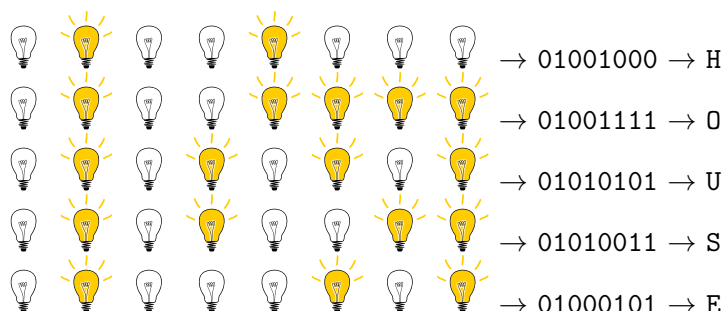
Que signifient les signaux lumineux de Sophie ?

- A) HOUSE
- B) HAPPY
- C) HORSE
- D) HONEY



Solution

Les signaux lumineux signifient :



La bonne réponse est donc le mot A) HOUSE.

On peut par ailleurs trouver cette réponse très facilement : la lettre du milieu est différente dans chaque mot : A) U, B) P, C) R et D) N. Comme le troisième signal lumineux signifie U, seule la réponse A) peut être juste.

C'est de l'informatique !

Le codage de Sophie n'a pas été choisi au hasard. Elle utilise une partie du code appelé ASCII qui a été développé pour l'échange de messages il y a plus de 50 ans. Il est basé sur le principe du code binaire que Gottfried Leibnitz (1646-1716) a déjà décrit en 1679 et 1703 sur la base de systèmes précurseurs indiens et chinois pour la représentation de nombres et le calcul avec ces nombres. Claude Shannon (1916-2001) l'appliqua ensuite lors du développement de l'ordinateur.

Aujourd'hui, les ordinateurs utilisent une version mise à jour de l'ASCII. Comme l'ASCII ne contenait que 95 caractères imprimables (les lettres latines majuscules et minuscules, les chiffres de 0 à 9 ainsi que quelques signes de ponctuation) et que les 33 autres étaient des caractères de contrôle (pour les imprimantes par exemple), on a bientôt eu besoin d'additions pour les accents et les autres alphabets. Cela eut lieu d'abord sous la forme d'un premier code utilisant 8 bits par caractère (ANSI) et ensuite sous la forme du standard Unicode utilisé presque universellement. Les lettres de Sophie sont encore codées exactement de la même manière en UTF-8, la version de l'Unicode la plus répandue.

Le premier groupe de lettres (identique en ASCII, 8 bits et Unicode) est par ailleurs (les caractères de contrôle sont laissés vides, `␣` représente l'espace) :



	000 ...	001 ...	010 ...	011 ...	100 ...	101 ...	110 ...	111 ...
...0000			□	0	@	P	'	p
...0001			!	1	A	Q	a	q
...0010			"	2	B	R	b	r
...0011			#	3	C	S	c	s
...0100			\$	4	D	T	d	t
...0101			%	5	E	U	e	u
...0110			&	6	F	V	f	v
...0111			'	7	G	W	g	w
...1000			(8	H	X	h	x
...1001)	9	I	Y	i	y
...1010			*	:	J	Z	j	z
...1011			+	;	K	[k	{
...1100			,	<	L	\	l	
...1101			-	=	M]	m	}
...1110			.	>	N	^	n	~
...1111			/	?	O	_	o	

Mots clés et sites web

ASCII, Unicode, Codage

- https://fr.wikipedia.org/wiki/American_Standard_Code_for_Information_Interchange
- https://fr.wikipedia.org/wiki/Code_binaire
- https://fr.wikipedia.org/wiki/Gottfried_Wilhelm_Leibniz
- https://fr.wikipedia.org/wiki/Claude_Shannon
- <https://fr.wikipedia.org/wiki/Unicode>
- <https://fr.wikipedia.org/wiki/UTF-8>
- <https://www.unicode.org/charts/PDF/U0000.pdf>



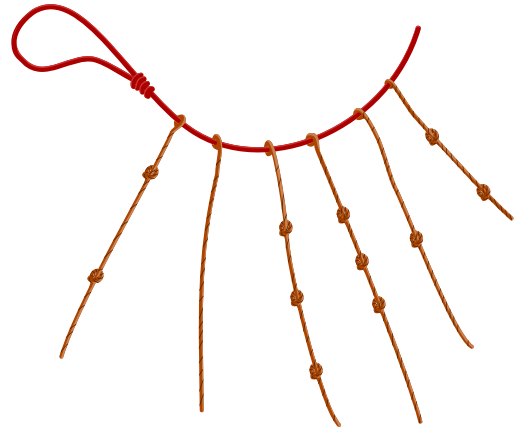


25. Quipu

Les Incas utilisaient à l'époque des nœuds pour la transmission de messages. Plusieurs cordelettes sur lesquelles des nœuds étaient noués étaient attachées à une corde. Ces assemblages de cordelettes appelés « quipus » étaient grands et difficiles à fabriquer.

Imagine qu'il faut développer une version simplifiée des quipus. Les conditions sont :

- Il y a toujours le même nombre de cordelettes attachées à la corde.
- Les cordelettes ne diffèrent que par le nombre de nœuds.
- Une cordelette a 0, 1, 2 ou 3 nœuds.
- L'ordre des cordelettes est déterminé à l'aide d'un nœud sur la corde.
- Il doit pouvoir y avoir 30 quipus discernables pour différents messages.



Quel est le nombre minimum de cordelettes de la version simplifiée des quipus dans ces conditions ?

- A) 2
- B) 3
- C) 4
- D) 5
- E) 8
- F) 10



Solution

La réponse B) 3 est correcte.

Chaque cordelette peut enregistrer une des 4 valeurs différentes (0, 1, 2 ou 3). Avec deux cordelettes, on aurait $4 \cdot 4 = 16$ combinaisons possibles, avec trois cordelettes $4 \cdot 4 \cdot 4 = 64$ combinaisons possibles, et ainsi de suite. Trois cordelettes sont donc suffisantes, plus de cordelettes seraient en contradiction avec la condition stipulant qu'il faut attacher le moins de cordelettes possible à la corde. Comme l'ordre des valeurs est déterminé par le nœud sur la corde, on ne doit pas s'inquiéter du fait que l'on pourrait lire la corde dans un sens ou dans l'autre.

C'est de l'informatique !

Les *quipus* étaient en effet utilisés par les Incas en Amérique du Sud. Des quipus gris étaient utilisés pour la comptabilité et la perception d'impôts. On suppose que jusqu'à 95 syllabes différentes pouvaient être codées à l'aide de cordelettes colorées, permettant une correspondance écrite. Au contraire de la version simplifiée de cet exercice, différentes sortes de nœuds et dans certains cas des sous-cordelettes attachées aux cordelettes étaient utilisées

L'exemple de cet exercice est une version simplifiée. Comme l'ordre est fixé par le nœud sur la corde, les valeurs individuelles (0, 1, 2 ou 3) créent une *notation positionnelle*, dans ce cas un système en base 4. Les notations positionnelles sont très répandues : en règle générale, le système décimal (en base 10) est utilisé, les ordinateurs utilisent le *système binaire* (en base 2). Au temps des premiers ordinateurs, il y a eu des essais de construction d'ordinateurs basés sur le *système ternaire* (en base 3, représentés dans ce cas par -1 , 0 et $+1$). En utilisant un système en base b avec n positions, on peut représenter exactement b^n valeurs différentes. Un octet (8 bits qui peuvent valoir chacun 0 ou 1) peut de cette manière enregistrer $2^8 = 256$ valeurs différentes (de 0 à 255), le quipu de cet exercice $4^3 = 64$ valeurs différentes

Les Incas n'auraient eu besoin que d'une seule cordelette pour enregistrer les valeurs de 1 à 30. Ils utilisaient également un système décimal comme nous pour l'écriture des nombres, simplement en utilisant différents nœuds sur une cordelette. La position des unités aurait été codée entre autres avec des demi-nœuds et la position des dizaines avec un demi-nœud auquel un nombre de tours correspondants était ajouté. Cependant, ils auraient eu besoin d'au moins 4 nœuds et de plusieurs sortes de nœuds.

Mots clés et sites web

Quipu, notation positionnelle

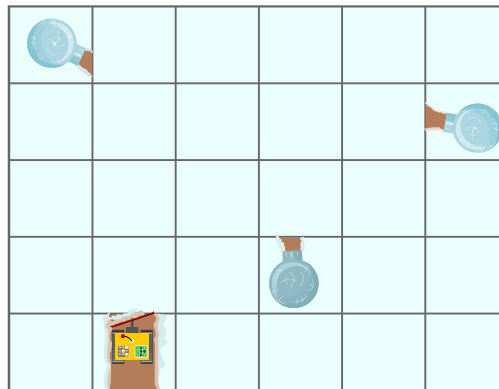
- <https://fr.wikipedia.org/wiki/Quipu>
- https://fr.wikipedia.org/wiki/N%C5%93ud_double
- https://en.wikipedia.org/wiki/Stopper_knot
- https://fr.wikipedia.org/wiki/Notation_positionnelle
- https://fr.wikipedia.org/wiki/Ordinateur_ternaire



26. Tempête de neige

Il y a des congères partout après une forte tempête de neige et les habitants des trois iglous sont isolés. Les habitants peuvent dégager des chemins à l'aide de leur chasse-neige télécommandé. Cela fonctionne ainsi :

- Le chasse-neige a besoin de 4 minutes pour passer d'une case à une case voisine en la dégageant.
- Le chasse-neige a besoin d'une minute pour passer d'une case déjà déneigée à une case voisine.
- Les case voisines ne sont que les cases qui sont situées directement en dessus, en dessous, à gauche ou à droite d'une case sur la carte. La chasse-neige ne peut donc pas rouler en diagonale.
- Dès que la case devant l'entrée d'un iglou est dégagée, les habitants de l'iglou peuvent en dégager l'entrée avec une pelle et ne sont plus isolés.

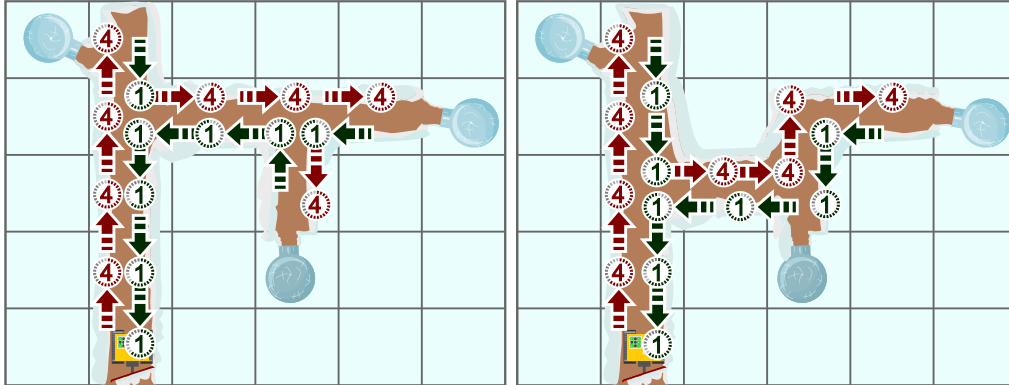


Dans le cas idéal, de combien de minutes le chasse-neige a-t-il besoin pour libérer tous les iglous de leur isolement et retourner à sa case de départ ?



Solution

La bonne réponse est 40. Les illustrations suivantes montrent les deux chemins optimaux pour le chasse-neige :



Pourquoi est-ce que ça ne va pas plus rapidement ? Pour atteindre l'iglou en haut à gauche, il faut dégager quatre cases. Cela prend 16 minutes. Pour atteindre l'iglou à droite, trois cases de plus doivent être dégagées. Cela prend 12 minutes de plus. Pour atteindre l'iglou du bas, une case doit encore être dégagée, car il faut soit dégager un petit chemin jusqu'au chemin transversal, soit ajouter un virage au chemin transversal. Cela prend encore 4 minutes. Pour que le chasse-neige retourne à la case départ, il doit encore parcourir quatre cases vers le bas et trois cases vers la gauche. Cela dure encore 7 minutes. Le détour par le petit chemin ou le virage dans le chemin transversal dure encore une minute supplémentaire. Il a donc besoin de 40 minutes en tout.

Si le chasse-neige dégageait le chemin plus vite, ce serait éventuellement plus efficace s'il rentrait de l'iglou du bas en passant par la case enneigée à gauche en la dégageant. Mais il a besoin de 4 minutes pour la dégager et d'une minute pour passer sur la case déjà déneigée, donc de 5 minutes, alors que le détour par les cases déjà déneigées ne lui prend que 4 minutes.

C'est de l'informatique !

Il s'agit dans cet exercice de trouver un réseau de chemins qui relie tous les endroits (les iglous et la case de départ) à un coût minimal (le temps dont le chasse-neige a besoin). De tels réseaux ne sont pas forcément composés des chemins les plus courts entre tous les endroits, mais les coûts nécessaires à la réalisation du réseau doivent être aussi bas que possible. On appelle de tels réseaux des *arbres de Steiner*. Ils sont utilisés, par exemple, pour la fabrication des cartes mères des ordinateurs ou la construction de réseaux ferroviaires peu exploités pour le transport de marchandises. La recherche d'arbres de Steiner est un problème d'optimisation difficile et qui prend beaucoup de temps en informatique, ce qui fait que l'on utilise souvent des algorithmes qui trouvent des solutions suffisamment bonnes, mais pas forcément la meilleure solution.

Dans cet exercice, les coûts sont calculés de manière spéciale, car il n'y a pas seulement un coût fixe pour l'élaboration d'un chemin (les 4 minutes pour dégager une case), mais également un coût pour le déplacement de la machine sur les cases déneigées. Cet exercice est donc une généralisation du problème de l'arbre de Steiner.

Mots clés et sites web

Problème de l'arbre de Steiner

— https://fr.wikipedia.org/wiki/Probl%C3%A8me_de_l%27arbre_de_Steiner



27. Quel bonheur que les arbres !

Sergio a écrit une chanson qui décrit comment plusieurs choses différentes peuvent se former sur un arbre. Voici un couplet :

Quel bonheur que les arbres !
Sur un arbre poussent des feuilles,
Sur un arbre poussent des fleurs,
Les fleurs donnent des fruits,
Avec des feuilles et des fleurs, je peux tresser des couronnes.

C'était important pour Sergio de n'utiliser après le premier vers que des objets déjà mentionnés auparavant.

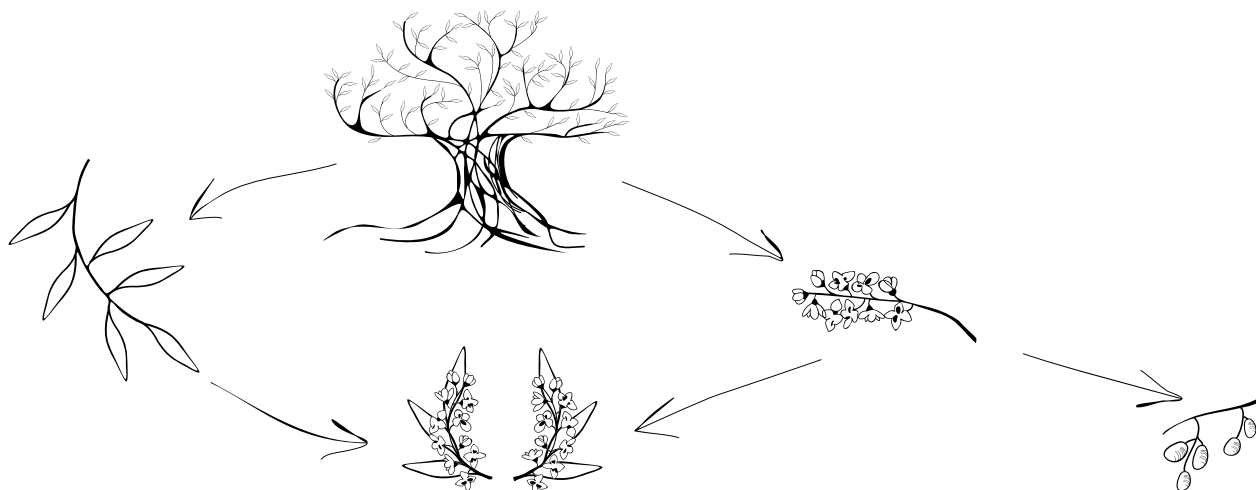
Lequel des couplets suivants est faux d'après Sergio ?

- A) Quel bonheur que les arbres !
Sur un arbre poussent des fleurs,
Sur un arbre poussent des feuilles,
Avec des feuilles et des fleurs, je peux tresser des couronnes,
Les fleurs donnent des fruits.
- B) Quel bonheur que les arbres !
Sur un arbre poussent des fleurs,
Sur un arbre poussent des feuilles,
Les fleurs donnent des fruits,
Avec des feuilles et des fleurs, je peux tresser des couronnes.
- C) Quel bonheur que les arbres !
Sur un arbre poussent des feuilles,
Les fleurs donnent des fruits,
Sur un arbre poussent des fleurs,
Avec des feuilles et des fleurs, je peux tresser des couronnes.
- D) Quel bonheur que les arbres !
Sur un arbre poussent des fleurs,
Les fleurs donnent des fruits,
Sur un arbre poussent des feuilles,
Avec des feuilles et des fleurs, je peux tresser des couronnes.
- E) Quel bonheur que les arbres !
Sur un arbre poussent des feuilles,
Sur un arbre poussent des fleurs,
Avec des feuilles et des fleurs, je peux tresser des couronnes,
Les fleurs donnent des fruits.



Solution

On peut représenter les dépendances des objets «arbre», «feuilles», «fleurs», «couronnes» et «fruits» à l'aide d'un graphe sur lequel une flèche signifie qu'un objet est nécessaire à la formation de l'autre :



D'après ce graphe, les fleurs doivent être mentionnées avant les fruits, et les feuilles et les fleurs avant les couronnes.

La séquence des objets dans les différentes réponses est la suivante :

- A) Arbre, fleurs, feuilles, couronnes, fruits
- B) Arbre, fleurs, feuilles, fruits, couronnes
- C) Arbre, feuilles, *fruits*, *fleurs*, couronnes
- D) Arbre, fleurs, fruits, feuilles, couronnes
- E) Arbre, feuilles, fleurs, couronnes, fruits

Dans la réponse C), les fruits sont mentionnés avant les fleurs (en italique plus haut), ce qui est une contradiction car il faut des fleurs pour faire des fruits. Tous les autres vers respectent les conditions.

C'est de l'informatique !

En 1974, le musicien italien Sergio Endrigo (1933–2005) a écrit la chanson pour enfants « Ci vuole un fiore » d'après un texte de Gianni Rodari (1920–1980). Dans cette chanson, il chante que l'on a d'abord besoin de bois pour avoir une table, d'un arbre pour avoir du bois, d'une graine pour avoir un arbre, d'un fruit pour avoir une graine et d'une fleur pour avoir un fruit. Il décrit aussi que pour avoir une fleur, on a également besoin d'une fleur en passant par une branche, un arbre, une forêt, une montagne et la terre. Il finit par dire qu'on a en fin de compte besoin d'une fleur pour tout.

La précedence d'un objet sur un autre peut être décrite à l'aide d'un *graphe orienté*. Un tel graphe est dessiné dans l'explication de la réponse. C'est un *graphe orienté acyclique* qui décrit un *ensemble de séquences admissibles*. Si l'on veut un nœud (l'un des objets), on doit déjà avoir tous les objets pointant vers lui. Le même chose est vraie pour ces objets-là, ce qui fait que l'on doit remonter de manière *réursive* jusqu'à ce que l'on arrive à un objet sur lequel rien ne pointe. On peut utiliser cet objet comme objet de départ

On ne peut d'ailleurs pas décrire la chanson de Sergio Endrigo à l'aide d'un graphe orienté acyclique. Dans la deuxième partie de la chanson décrite plus haut, il chante que l'on a en fin de compte besoin d'une fleur pour avoir une fleur. C'est en contradiction avec le fait que le graphe doit être acyclique



et ne peut donc pas contenir de circuit. Avec cette rupture de logique, il rend sa déclaration encore plus claire: «Ci vuole un fiore» — «Nous avons besoin d'une fleur»!

Mots clés et sites web

Graphe orienté acyclique, tri topologique

- <https://www.filastrocche.it/contenuti/ci-vuole-un-fiore/>
- <https://www.youtube.com/watch?v=9ht4tIot8XY>
- https://en.wikipedia.org/wiki/Precedence_graph
- https://fr.wikipedia.org/wiki/Tri_topologique

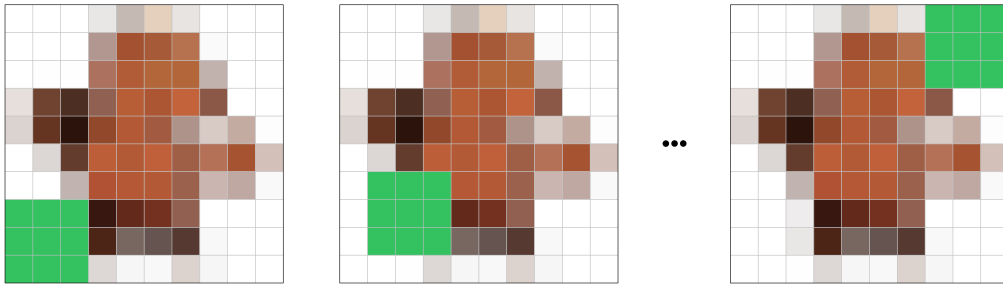




28. Compression vidéo

Les vidéos occupent beaucoup d'espace de stockage. Pourtant, deux images fixes consécutives se ressemblent souvent beaucoup.

La vidéo suivante a une taille de 10×10 points. Le carré vert dans le coin en bas à gauche fait 3×3 points. Il se déplace d'image fixe à image fixe d'un point vers la droite et d'un point vers le haut à chaque image jusqu'à ce qu'il arrive dans le coin en haut à droite.



Pour économiser de l'espace de stockage, à partir de la deuxième image, seuls les points qui ont changé sont enregistrés.

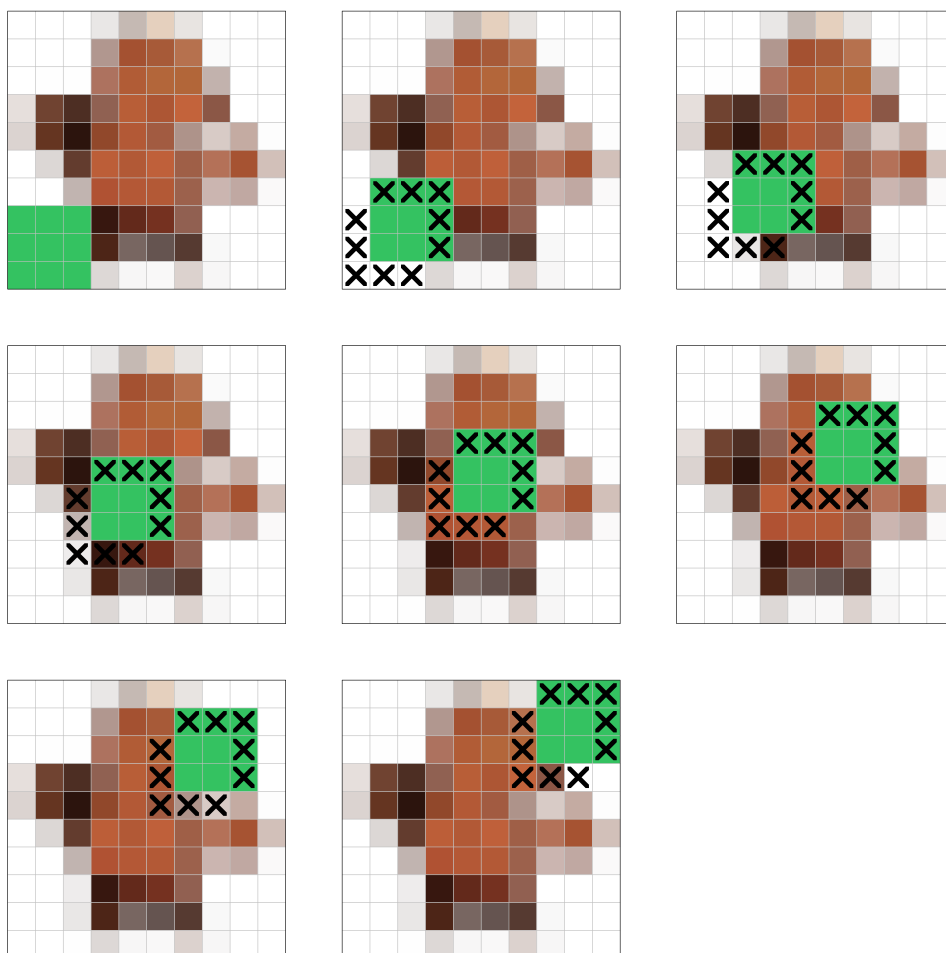
Combien de points doivent être enregistrés pour toute la vidéo ?

- | | | |
|--------|--------|---------|
| A) 100 | D) 170 | G) 800 |
| B) 135 | E) 180 | H) 1000 |
| C) 140 | F) 700 | |



Solution

Les images fixes individuelles ressemblent à cela lorsque l'on marque les points ayant changé :



On remarque tout d'abord que la première image contient $10 \cdot 10 = 100$ points. Pour chacune des images suivantes, seuls les points qui ont changé doivent être enregistrés. Il s'agit des cinq points en bas à gauche du carré qui sont remplacés par les points de l'arrière-plan et des cinq points en haut à droite sur le carré qui représentent la nouvelle position du carré. 10 points changent donc par image fixe. Le carré a besoin de 7 images fixes supplémentaires pour se déplacer d'en bas à gauche vers en haut à droite, il faut donc ajouter $10 \cdot 7 = 70$ points pour les points qui changent aux 100 points de la première image. La bonne réponse est donc D) 170 points.

C'est de l'informatique !

Comme décrit dans l'exercice, la compression vidéo joue un rôle important de nos jours. La méthode décrite ici n'est qu'une des approches permettant de compresser des vidéos. Une autre approche consiste à enlever certaines informations qu'une personne ne perçoit habituellement pas. Le format d'image JPEG utilise de telles relations. Sur des images compressées spécialement fortement, on peut le reconnaître à la formation de blocs, car la différence entre les couleurs du bloc a été interprétée à tort comme imperceptible. Une autre possibilité est de réduire la dimension de l'espace de couleur. C'est sur ces idées qu'est basé le standard MPEG. Comme dans cet exercice, il fait la différence entre différents types d'images fixes. Un type d'images fixes (appelées images «intra») représente une image fixe complète (similairement à notre première image fixe). Un autre type d'images fixes



est basé sur la différence avec l'image fixe précédente (images «P», comme nos autres images fixes) ou même sur la différence avec l'image fixe suivante en plus (images «D», ne sont pas utilisées dans cet exercice). Pour minimiser l'utilisation de mémoire tampon et pour pouvoir «retrouver le fil» en cas d'erreurs de transmission, des images intra sont insérées à intervalles réguliers. Sur des vidéos fortement compressées, on peut reconnaître les images P et D lorsque qu'un arrière-plan sombre «saute» tout à coup alors que la scène n'a changé que lentement pendant un certain temps.

Le besoin en espace de stockage n'est pas vraiment aussi petit que suggéré dans l'exercice : en plus des valeurs des couleurs, la position du pixel qui a changé doit aussi être enregistrée. Cela multiplie peut-être par deux l'espace de stockage nécessaire à un pixel changé. Même ainsi, 240 unités de stockage comparées à 800 unités de stockage représentent une économie de place impressionnante, d'autant plus que la méthode décrite dans l'exercice est, contrairement au MPEG, une méthode sans perte !

Mots clés et sites web

Compression vidéo

- https://fr.wikipedia.org/wiki/Compression_vid%C3%A9o
- <https://fr.wikipedia.org/wiki/JPEG>
- <https://fr.wikipedia.org/wiki/MPEG-1>

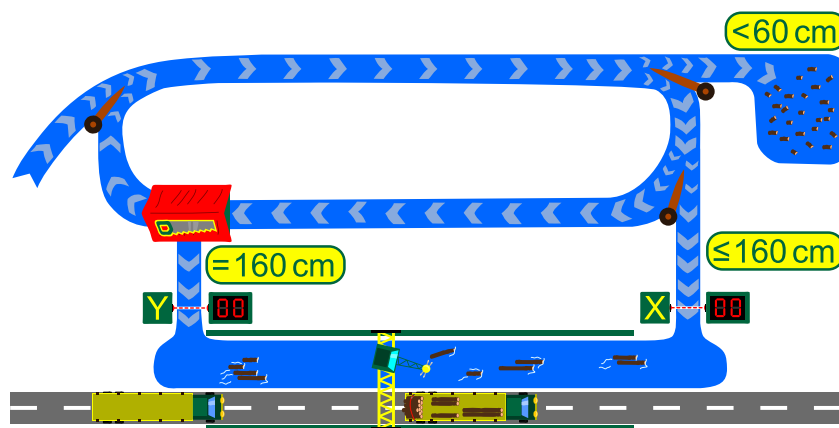




29. Scierie

Dans une scierie, des troncs d'arbres sont raccourcis pour avoir une longueur entre 60 cm et 160 cm avant d'être chargés sur des camions. À l'intérieur de la scierie, le transport des troncs d'arbres se fait par des canaux. Il y existe les stations de travail suivantes :

- En haut à gauche, les troncs d'arbres sont livrés.
- En haut à droite, les troncs plus courts que 60 cm sont mis à part (<60 cm).
- Au milieu à droite, tous les troncs de 160 cm ou moins sont chargés sur des camions (≤160 cm). Ceux-ci sont comptés à hauteur du détecteur X.
- Au milieu à gauche, un tronçon de 160 cm est coupé à l'extrémité de chaque tronc. Ces troncs sont chargés sur des camions (=160 cm) et comptés à hauteur du détecteur Y. Le reste du tronc est remis en circulation.



Trois troncs d'arbres de 60 cm, 140 cm et 360 cm sont livrés et traités dans la scierie. Combien de troncs d'arbres sont comptés à hauteur du détecteur X et combien à hauteur du détecteur Y ?

- A) Détecteur X : aucun tronc, détecteur Y : 4 troncs
- B) Détecteur X : 1 tronc, détecteur Y : 3 troncs
- C) Détecteur X : 2 troncs, détecteur Y : 2 troncs
- D) Détecteur X : 3 troncs, détecteur Y : 1 tronc



Solution

Le tronc de 60 cm n'est pas mis à part en haut à droite, car il n'est pas plus court que 60 cm. Il est par contre chargé sur un camion au milieu à droite, car il mesure 160 cm ou moins. Un tronc a donc déjà été compté à hauteur du détecteur.

Le tronc de 140 cm n'est pas mis à part en haut à droite non plus, car il n'est pas plus court que 60 cm. Par contre, comme il mesure 160 cm ou moins, il est chargé sur un camion au milieu à droite. Un deuxième tronc a ainsi été compté à hauteur du détecteur X.

Le tronc de 360 cm n'est pas mis à part en haut à droite non plus, car il n'est pas plus court que 60 cm. Par contre, vu qu'il fait plus de 160 cm, il est dirigé vers la scie lorsqu'il passe au milieu à droite. La scie en coupe un tronçon de 160 cm qui est chargé sur un camion. Ainsi, un tronc a été compté à hauteur du détecteur Y. Le reste du tronc mesurant 200 cm est remis en circulation. Le tronc mesurant à présent 200 cm n'est pas mis à part en haut à droite non plus, car il n'est pas plus court que 60 cm. Vu qu'il fait plus de 160 cm, il est également dirigé vers la scie lorsqu'il passe au milieu à droite. La scie en coupe un deuxième tronçon de 160 cm qui est chargé sur un camion. Ainsi, un deuxième tronc est compté à hauteur du détecteur Y. Le reste du tronc mesurant 40 cm est remis en circulation. Le tronc mesurant à présent 40 cm est mis à part en haut à droite.

La réponse correcte est donc C) Détecteur X : 2 troncs, détecteur Y : 2 troncs.

C'est de l'informatique !

La seule caractéristique importante des troncs en circulation est leur longueur. On peut donc considérer la scierie comme un programme dans lequel des nombres entiers sont entrés et qui fait certaines mesures. Dans ce cas, on peut voir la scierie comme un *programme réactif* : pendant que le nombre est traité par le programme, les mesures changent avec le temps. La programmation réactive joue un rôle majeur dans les tableurs. Les valeurs calculées à l'aide de formules réagissent aux changements de valeurs dans d'autres cases.

Concrètement, plusieurs opérations réactives sont utilisées dans cet exercice : en haut à gauche, deux trains de données convergent (*merge* en anglais), en haut à droite, les données sont filtrées (*filter* en anglais), en bas à droite aussi, et en bas à gauche les données sont transformées (*transform* en anglais). Les deux détecteurs font des mesures (*scan* en anglais).

L'analyse de processus dynamiques comme dans cet exercice est centrale à l'informatique. Bien avant que le terme *pensée computationnelle* (*computational thinking* en anglais) n'apparaisse et ne soit rendu mondialement connu par Jeanette Wing en 2006, des termes tels que *procedural thinking* ou *algorithmic thinking* (*pensée procédurale* ou *pensée algorithmique*) étaient utilisés comme paradigmes propres à l'informatique.

Mots clés et sites web

Reaktive Programmierung

- https://fr.wikipedia.org/wiki/Programmation_r%C3%A9active
- https://fr.wikipedia.org/wiki/Pens%C3%A9e_computationnelle
- <https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf>

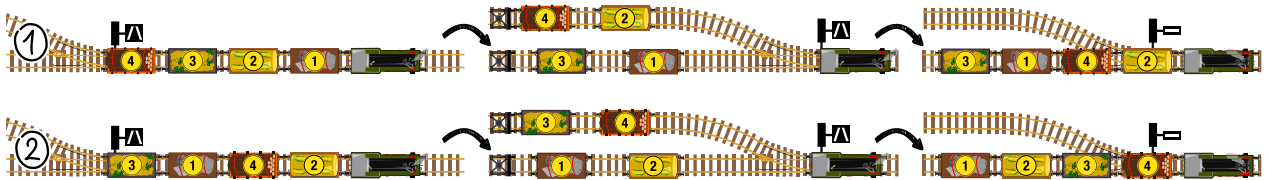


30. Gare de triage

Un train de marchandises doit amener des wagons de marchandises individuels sur des voies de raccordement le long de la ligne principale. Pour économiser du temps et éviter de manœuvrer sur la ligne principale, les wagons de marchandises doivent être classés d'après leur numéro de façon à ce que le wagon portant le numéro 1 soit tout à gauche.

Dans la gare de triage, il y a une butte sur laquelle les wagons de marchandises sont refoulés vers le bas de gauche à droite. Sur la butte, on décide pour chaque wagon sur laquelle des deux voies de garage il est orienté. Ensuite, la locomotive tracte les wagons hors des voies de garage : d'abord tous ceux se trouvant sur une voie, ensuite tous ceux se trouvant sur l'autre voie de garage. On considère ces actions comme une étape de tri.

Par exemple, lorsque quatre wagons de marchandises doivent être triés, deux étapes de tri suffisent (étape ① et étape ②) :



Ce n'est pas possible de trier les wagons en une seule étape de tri.

Si les wagons de marchandises sont dans l'ordre 2 – 8 – 3 – 1 – 5 – 7 – 6 – 4, de combien d'étapes de tri a-t-on besoin au minimum pour trier le train de marchandises ?



- A) 3
- B) 4
- C) 5
- D) 6
- E) 7
- F) 8



Solution

La bonne réponse est que l'on a besoin de A) 3 étapes de tri.

Bien sûr que l'on peut utiliser plusieurs méthodes pour trier le train de marchandises, mais l'une des meilleures méthodes est de commencer par refouler les wagons de marchandises 1, 3, 5 et 7 sur la voie du haut et les wagons 2, 4, 6, et 8 sur la voie du bas, puis de tracter d'abord les wagons de la voie du bas puis ceux de la voie du haut :



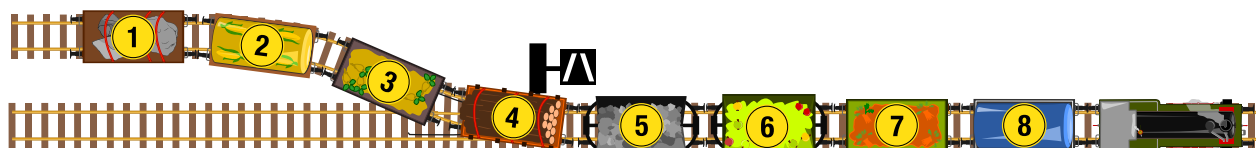
Ainsi, pour chaque paire (1 et 2, 3 et 4, 5 et 6, 7 et 8) de wagons de marchandises, le wagon portant le plus petit numéro se trouve toujours à gauche de celui portant le plus grand numéro.

Ensuite, c'est raisonnable de refouler les wagons de marchandises 1, 2, 5 et 6 sur la voie du haut et les wagons 3, 4, 7 et 8 sur la voie du bas, puis de tracter d'abord les wagons de la voie du bas puis ceux de celle du haut :



On n'a ainsi pas changé l'ordre des paires obtenues auparavant, car chaque paire a été refoulée sur la même voie. On a en plus mis les wagons 1 à 4 et 5 à 8 dans le bon ordre relatif les uns aux autres, mais les deux groupes ne sont pas encore mélangés.

Finalement, il ne faut plus que refouler les wagons de marchandises 1 à 4 sur la voie du haut et les wagons 5 à 8 sur la voie du bas, puis de tracter d'abord les wagons de marchandise de la voie du bas, puis de la voie du haut :



L'ordre dans chaque groupe n'a pas été modifié, car tous les wagons de marchandises du groupe 1 à 4 ont été refoulés sur une voie et tous les wagons du groupe 5 à 8 sur l'autre voie. Maintenant, les deux groupes sont composés de wagons de marchandises ordonnés et tous les wagons de l'un des deux groupes ont des numéros plus petits que tous les wagons de l'autre groupe.

On ne peut pas trier les huit wagons plus rapidement. Une preuve complète de cela serait trop longue pour cet exercice, mais l'idée de base est la suivante: lors d'une étape de tri, on peut changer l'ordre d'un sous-ensemble relativement aux autres sous-ensembles, mais pas l'ordre dans les sous-ensembles mêmes. On ne peut donc trier que deux wagons de marchandises dans un ordre défavorable lors de la première étape de tri. Chaque étape de tri suivante double le nombre de wagons ordonnés défavorablement que l'on peut trier. Les huit wagons de marchandises de cet exercice sont choisis de manière à être ordonnés défavorablement, donc deux étapes de tri ne suffisent pas.

C'est de l'informatique !

Les cheminots du monde entier doivent résoudre de tels problèmes quotidiennement, car le tri de wagons de marchandises est une tâche qui demande beaucoup de temps et de travail: chaque fois,



les wagons doivent être connectés et déconnectés, ce qui est encore un travail manuel. Cela prend du temps et bloque la ligne principale, surtout lorsque plusieurs wagons de marchandises doivent être sécurisés et déconnectés. C'est pour cela que des cheminots ont très tôt développé de grandes gares de triage avec beaucoup de voies de garage. En Suisse, il y a des gares de triage à MuttENZ près de Bâle, à Buchs (Saint-Gall), entre Spreitenbach et Dietikon près de Zurich, à Denges près de Lausanne et à Chiasso. Dans cet exercice, la gare de triage n'a que deux voies de garage, un défi pour les grands trains de marchandises mais une situation typique pour les lignes ferroviaires secondaires, en particulier pour les lignes de chemin de fer à voie étroite qui n'ont pas de connexion directe aux grandes entreprises ferroviaires.

L'informatique peut être très utile pour trier efficacement les trains de marchandise. Dans ce cas, le principe consistant à résoudre encore et encore le même problème simplifie fortement l'exercice : une méthode connue en informatique sous le nom de «diviser pour régner» (*divide & conquer* en anglais). Dans ce cas, on trie d'abord les paires de wagons de marchandises, puis des groupes de quatre wagons, puis le groupe de huit wagons.

Les voies de garage pour les wagons de marchandises fonctionnent comme le type de données abstrait *pile*, qui est beaucoup utilisé en informatique. Les seules opérations autorisées sont : *dépiler* (enlever l'élément du dessus, *pop* en anglais) et *empiler* (ajouter un élément sur la pile, *push* en anglais). Parfois, on peut aussi *voir l'élément de tête* (*top* en anglais) et *vérifier si la pile est vide* (*empty* en anglais).

Mots clés et sites web

Diviser pour régner (Divide & Conquer), pile

- https://fr.wikipedia.org/wiki/Gare_de_triage
- [https://fr.wikipedia.org/wiki/Diviser_pour_r%C3%A9gner_\(informatique\)](https://fr.wikipedia.org/wiki/Diviser_pour_r%C3%A9gner_(informatique))
- [https://fr.wikipedia.org/wiki/Pile_\(informatique\)](https://fr.wikipedia.org/wiki/Pile_(informatique))





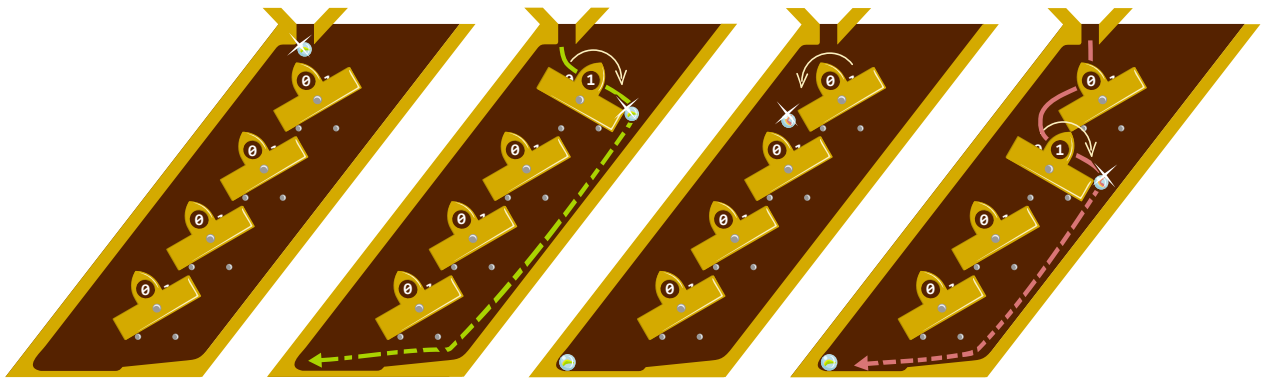
31. Jeu de billes

Un jeu de billes contient quatre balançoires qui peuvent s'incliner de deux manières :

- Si la balançoire est penchée vers la gauche, son inclinaison est 0.
- Si la balançoire est penchée vers la droite, son inclinaison est 1.

Lorsqu'une bille atterrit sur une balançoire, celle-ci change son inclinaison et la bille roule vers le bas.

Lorsque deux billes sont mises en jeu, les balançoires se penchent de façon à ce que l'inclinaison de la balançoire la plus haute soit 1 après la première bille, et qu'après la deuxième bille, l'inclinaison de la balançoire du haut soit à nouveau 0 et que celle de la deuxième balançoire soit 1 :



À la fin, les inclinaisons des balançoires (d'en bas à gauche vers en haut à droite) sont 0, 0, 1 et 0. Toutes les balançoires sont remises à l'inclinaison 0. Quelle sera l'inclinaison finale des quatre balançoires si dix billes sont mises en jeu ?

- | | | |
|-----------------|-----------------|-----------------|
| A) 0, 0, 0 et 0 | G) 1, 0, 1 et 0 | M) 1, 1, 0 et 1 |
| B) 1, 0, 0 et 0 | H) 1, 0, 0 et 1 | N) 1, 0, 1 et 1 |
| C) 0, 1, 0 et 0 | I) 0, 1, 1 et 0 | O) 0, 1, 1 et 1 |
| D) 0, 0, 1 et 0 | J) 0, 1, 0 et 1 | P) 1, 1, 1 et 1 |
| E) 0, 0, 0 et 1 | K) 0, 0, 1 et 1 | |
| F) 1, 1, 0 et 0 | L) 1, 1, 1 et 0 | |



Solution

La balançoire la plus haute change d'inclinaison à chaque bille. Elle se retrouve donc en position 0 quand, comme dans ce cas, un nombre pair de billes (10) sont mises en jeu. Le dernier chiffre est donc 0.

La deuxième balançoire depuis le haut ne change d'inclinaison que si la balançoire la plus haute se trouve en position 1 au départ de la bille. Ce n'est le cas qu'une fois sur deux. Elle change donc cinq fois d'inclinaison et termine avec l'inclinaison 1. L'avant-dernier chiffre est donc 1.

La troisième balançoire depuis le haut ne change d'inclinaison que si la deuxième balançoire a l'inclinaison 1 au départ de la bille et qu'une bille y atterrit, c'est-à-dire si la balançoire la plus haute a également l'inclinaison 1 au départ. La troisième balançoire ne change donc d'inclinaison qu'une fois sur quatre. C'est le cas lors du passage de la quatrième et huitième billes, donc deux fois en tout. Elle termine donc avec l'inclinaison 0 et le deuxième chiffre est 0.

La balançoire du bas ne change d'inclinaison que si les trois balançoires précédentes ont l'inclinaison 1 au départ de la bille et qu'une bille peut donc y atterrir. Ce n'est le cas que lors du passage de la huitième bille. La dernière balançoire ne change donc qu'une fois d'inclinaison et le premier chiffre est 1.

L'inclinaison des balançoires est donc 1, 0, 1 et 0.

C'est de l'informatique !

Les balançoires du jeu de billes représentent un interrupteur électronique capable d'alterner entre deux états. De tels interrupteurs sont des éléments de bases d'appareils électroniques, celui de l'exercice est une variante de *bascule*.

Les balançoires fonctionnent ensemble comme un *compteur* (binaire). Les bascules y sont reliées de manière à former des *demi-additionneurs*. Un tel demi-additionneur prend comme valeur d'entrée l'état enregistré de la bascule ainsi qu'une impulsion. Sa sortie est un nouvel état à enregistrer ainsi qu'un reste.

État enregistré	Impulsion	État à enregistrer	Reste
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Le circuit doit assurer que le changement des états enregistrés se passe dans le bon ordre et que les valeurs des états ne changent pas de manière automatique après un changement.

Les états des balançoires représentent donc finalement un nombre binaire qui augmente de 1 au passage de chaque bille. Dix billes donnent donc, si on inclut l'état de départ, les 11 premiers nombres binaires : 0000 → 0001 → 0010 → 0011 → 0100 → 0101 → 0110 → 0111 → 1000 → 1001 → 1010.

Mots clés et sites web

Système binaire, compteur, demi-additionneur, bascule

- <https://fr.wikipedia.org/wiki/Compteur>
- <https://www.youtube.com/watch?v=zELAfmp3fXY>
- [https://fr.wikipedia.org/wiki/Basculer_\(circuit_logique\)](https://fr.wikipedia.org/wiki/Basculer_(circuit_logique))
- <https://fr.wikipedia.org/wiki/Additionneur#Demi-additionneur>
- https://fr.wikipedia.org/wiki/Syst%C3%A8me_binaire



32. Quatre poissons

En informatique, le mode de fonctionnement d'opérateurs tels que + ou * dépend en partie de quels types de données sont impliqués. La table suivante montre différentes combinaisons typiques utilisées dans des expressions :

Général	Exemple
Nombre + Nombre → Nombre (additionner)	2+3 → 5
Nombre + Texte → Erreur	2+"3" → erreur
Texte + Nombre → Erreur	"2"+3 → erreur
Texte + Texte → Texte (mettre bout à bout)	"2"+"3" → "23"
Nombre * Nombre → Nombre (multiplier)	2*3 → 6
Nombre * Texte → Text (mettre le texte nombre fois bout à bout)	2*"3" → "33"
Texte * Nombre → Text (mettre le texte nombre fois bout à bout)	"2"*3 → "222"
Texte * Texte → Erreur	"2"*"3" → erreur

Lorsque le résultat est «erreur», cela signifie qu'aucun mode de fonctionnement n'est défini pour cette combinaison. S'il y a une erreur dans une expression, le résultat est également «erreur». Les opérateurs sont combinés suivant l'ordre habituel des opérations, les multiplications et divisions avant les additions et soustractions. L'opération * est donc effectuée avant l'opération +. Cet ordre peut être modifié à l'aide de parenthèses. Les opérations entre parenthèses sont effectuées de l'intérieur vers l'extérieur.

Quelle expression, parmi les expressions suivantes, génère cette ligne de texte ?

"...>(((°>.....>(((°>.....>(((°>.....>(((°>...."

- A) (3*" "+">" +3*" ("+"°>" +3*" .") *2"*2
- B) (3*" "+">" +3*" ("+"°>") *2*2+3*" ."
- C) (3*" "+">" +3*" ("+"°>" +3*" .") *2*2
- D) (3*" "+">" +3*" ("+"°>" +3*" .") *2*2



Solution

La bonne réponse est D) $(3*"."+"><"+3*"("+"°>"+3*"."))*2*2$.

Les cinq termes dans la parenthèse génèrent chacun le texte suivant :

- $3*"." \rightarrow "..."$
- $"><" \rightarrow "><"$
- $3*"(" \rightarrow "(((("$
- $"°>" \rightarrow "°>"$
- $3*"." \rightarrow "..."$

Mis bout à bout, cela donne $"...><(((°>..."$.

Les deux multiplications suivantes $*2*2$ causent chacune la mise bout à bout du texte, ce qui donne d'abord $"...><(((°>.....><(((°>..."$ et finalement $"...><(((°>.....><(((°>.....><(((°>.....><(((°>..."$.

La réponse A) génère une erreur, car l'expression entre parenthèses est un texte ($"...><(((°>..."$) qui doit être multiplié par un autre texte ($"2"$).

La réponse B) génère le texte $"...><(((°>...><(((°>...><(((°>...><(((°>..."$. C'est très proche du résultat souhaité, mais le nombre de points entre les poissons ne correspond pas au résultat souhaité.

La réponse C) génère une erreur, car l'expression $"3*"("$ a déjà « erreur » comme résultat.

C'est de l'informatique !

On parle de *surcharge* lorsque plusieurs modes de fonctionnement sont assignés à un opérateur (ou à un sous-programme) en fonction des opérandes (ou des paramètres). Cela arrive le plus souvent dans le cas d'opérateurs qui sont également utilisés fréquemment en dehors du langage de programmation. La forme de surcharge des opérateurs + et * décrite plus haut est présente dans beaucoup de langages de programmation. Aussi pratique la surcharge d'opérateur soit-elle, elle comporte le risque de rendre le code du programme moins lisible.

Le court programme suivant génère la même chaîne de caractères que celle de l'exercice (les poissons), mais le type de chaque variable et le mode de fonctionnement de chaque opérateur en fonction du type de variable est difficile à déterminer et vérifier :

```

a ← 3
b ← "."
c ← "><"
d ← "("
e ← "°>"
f ← 2
Sortie (a*b+c+a*d+e+a*b)*f*f

```

La surcharge de termes avec différentes significations n'est d'ailleurs pas un phénomène propre à l'informatique. Dans le contexte de la langue, cela s'appelle *polysémie*. Le mot « figure », par exemple, a un sens différent si l'on parle de géométrie, de ballet, de littérature ou d'un visage.

Le poisson généré ici est un exemple classique d'art ASCII. C'est une variante du « hareng rouge », qui est parfois envoyé aux trolls sur Internet.

Mots clés et sites web

Surcharger, art ASCII

- https://fr.wikipedia.org/wiki/D%C3%A9finition_d%27op%C3%A9rateur



- <https://fr.wikipedia.org/wiki/Polys%C3%A9mie>
- https://fr.wikipedia.org/wiki/Art_ASCII

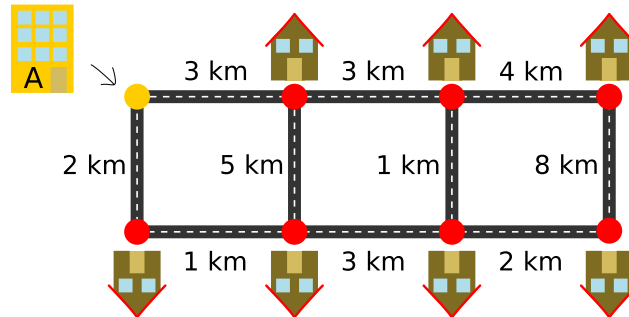




33. Job de vacances

Ton job de vacances consiste à livrer des paquets à vélo. Tu commences à l'endroit A et livres un paquet à chacun des sept autres endroits. Ta tournée se termine au dernier endroit et ton employeur vient te chercher ainsi que ton vélo.

Pour rester en forme, tu aimerais faire le plus long trajet possible avec les paquets. La longueur de chaque chemin est inscrite sur la carte ci-dessous. Ton employeur te laisse décider quels chemins prendre, mais tu ne veux pas passer deux fois au même endroit.



Quelle est la longueur du plus long chemin qui ne passe pas deux fois au même endroit ?

- A) 22 km
- B) 23 km
- C) 24 km
- D) 25 km
- E) 26 km



Solution

Il n'y a en tout que quatre chemins possibles qui ne passent pas deux fois au même endroit. Leurs longueurs sont :

Chemin	Longueur
	$2 \text{ km} + 1 \text{ km} + 3 \text{ km} + 2 \text{ km} + 8 \text{ km} + 4 \text{ km} + 3 \text{ km} = 23 \text{ km}$
	$2 \text{ km} + 1 \text{ km} + 5 \text{ km} + 3 \text{ km} + 4 \text{ km} + 8 \text{ km} + 2 \text{ km} = 25 \text{ km}$
	$2 \text{ km} + 1 \text{ km} + 5 \text{ km} + 3 \text{ km} + 1 \text{ km} + 2 \text{ km} + 8 \text{ km} = 22 \text{ km}$
	$3 \text{ km} + 3 \text{ km} + 4 \text{ km} + 8 \text{ km} + 2 \text{ km} + 3 \text{ km} + 1 \text{ km} = 24 \text{ km}$

Il ne peut pas y avoir d'autre chemin : dans tous les autres cas, il faut passer deux fois par l'un des endroits afin d'arriver à un autre.

C'est donc clair que D) 25 km est la bonne réponse.

C'est de l'informatique !

Dans cet exercice, il s'agit de trouver un chemin passant exactement une fois par chaque endroit sur une carte. On appelle un tel chemin un *chemin hamiltonien*. Comme on peut interpréter une carte comme un *graphe*, les endroits comme des *nœuds* et les chemins comme des *arêtes*, la recherche d'un chemin hamiltonien relève de la théorie des graphes. La recherche d'un chemin hamiltonien dans un graphe quelconque, ou le fait de déterminer si un tel chemin existe, est un problème NP-complet qui ne peut donc pas être résolu de manière efficace par ordinateur.

Dans ce cas, on ne recherche pas n'importe quel chemin hamiltonien, mais celui qui a comme somme des arêtes la valeur la plus haute parmi tous les chemins hamiltoniens possibles. Cela rend le problème encore plus difficile à résoudre.

Les approches trouvant une solution acceptable mais suboptimale dans d'autres cas ne peuvent pas résoudre ce problème. Une approche classique serait un *algorithme glouton*, qui sélectionne d'abord les valeurs les plus élevées, avec lequel on commencerait donc par le chemin de 3 km. Cependant, cette approche ne trouverait que le chemin de 24 km. Et si l'on continue de manière si « gloutonne »



en prenant ensuite le chemin de 5 km, on élimine complètement la possibilité de ne pas devoir passer deux fois au même endroit.

Mots clés et sites web

Chemin hamiltonien, NP-complet

- https://fr.wikipedia.org/wiki/Graphe_hamiltonien
- https://fr.wikipedia.org/wiki/Probl%C3%A8me_NP-complet
- https://fr.wikipedia.org/wiki/Algorithme_glouton





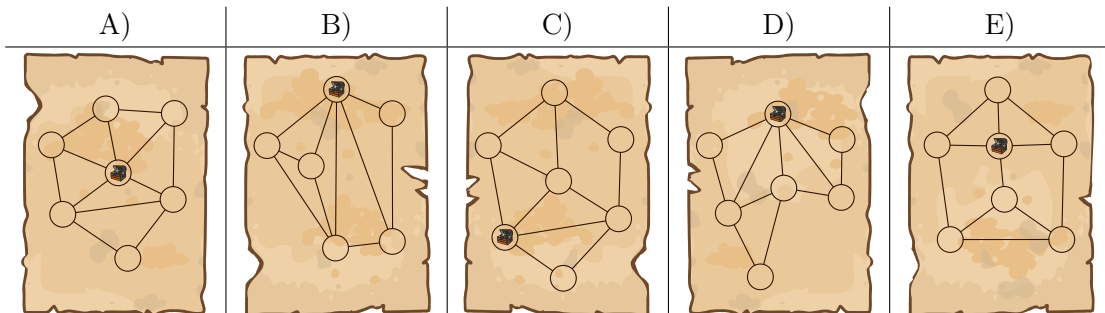
34. Carte au trésor

Le roi des castors règne sur sept provinces dont les frontières sont représentées sur la carte ci-dessous. Il a caché son trésor dans l'une des provinces :



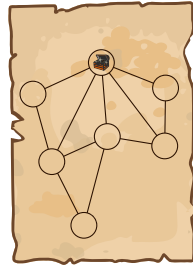
Le roi a fait dessiner une carte au trésor sur laquelle les provinces sont représentées par des cercles. Il a mis la province où se trouve le trésor en évidence. Deux cercles sont reliés si les provinces correspondantes ont une frontière en commun. Pour empêcher des voleurs de trouver le trésor, le roi a fait dessiner quatre fausses cartes au trésor supplémentaires.

Quelle carte au trésor est la bonne ?



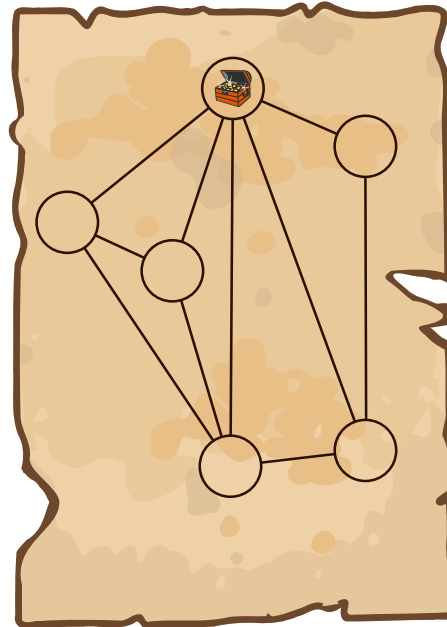
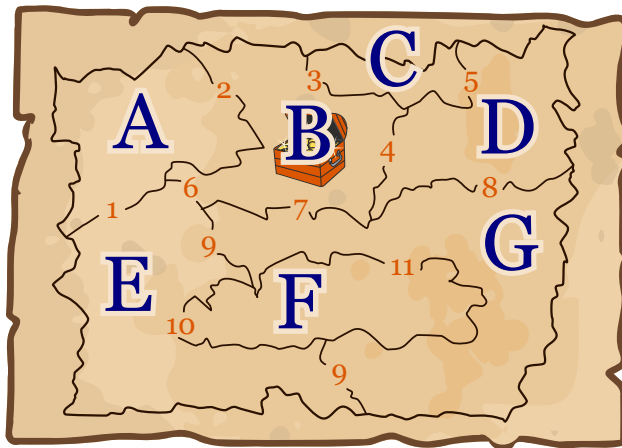


Solution



La bonne réponse est D)

Les provinces sont annotées des lettres A, B, C, D, E, F et G dans la représentation suivante. Les frontières entre les provinces sont annotées des nombres de 1 à 11. Comme il y a deux frontières communes entre les provinces E et G mais que la condition ne demande qu'une seule frontière commune, les deux frontières sont annotées du chiffre 9. On est rapidement convaincu que la carte D) est la bonne carte à l'aide de cette représentation.



La réponse A) est fausse : les trois provinces A, C et F n'ont de frontières communes qu'avec deux autres provinces. Il devrait donc y avoir trois cercles desquels deux lignes partent sur la carte. Il n'y a pourtant qu'un seul cercle dont deux lignes partent.

La réponse B) ne peut pas être juste, car elle ne contient que six cercles alors qu'il y a sept provinces.

La réponse C) ne peut pas être juste non plus : la province où est le trésor est voisine des cinq provinces A, C, D, E et G, il devrait donc y avoir cinq lignes partant du cercle mis en évidence. Il n'y en a pourtant que quatre.

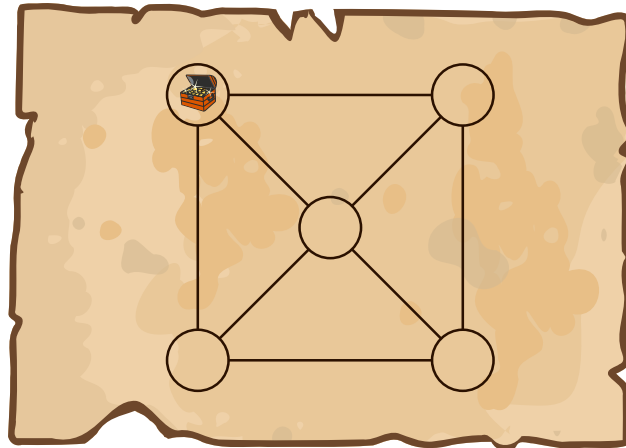
La réponse E) ressemble le plus à la carte des provinces. Les frontières n'y sont pourtant pas représentées correctement, il manque une frontière à la province B où est le trésor, par exemple.

C'est de l'informatique !

Dans cet exercice, il s'agit de représenter une carte à l'aide d'un *graphe*. Un tel graphe est une *abstraction* de la carte (comme la carte qui est elle-même déjà une abstraction de la réalité). Comme dans toutes les abstractions, les informations sans importance sont perdues. Dans cet exercice, il s'agit des positions géographiques des provinces les unes par rapport aux autres : même si les provinces



B et D se trouvent à la même hauteur sur la carte, elles se trouvent à des hauteurs différentes sur le graphe. Par contre, les informations essentielles sont conservées dans l'abstraction, dans le cas de cet exercice, il s'agit de savoir quelles provinces sont voisines. Cependant, même une abstraction correcte comporte le risque de perdre une information apparemment sans importance : le graphe suivant d'un autre état est symétrique et ne permet donc pas de déterminer où le trésor est caché exactement :



Un graphe est composé de *nœuds* (les cercles dans cet exercice) et d'*arêtes* (les lignes dans cet exercice). Dans cet exercice, les arêtes du graphe ne peuvent que relier deux nœuds. On ne peut pas non plus relier deux nœuds avec plusieurs arêtes, ce qui aurait été envisageable entre les provinces E et G.

Les graphes sont souvent utilisés en informatique pour abstraire des informations. Dans de nombreux cas, cette étape d'abstraction est déjà presque la solution du problème : comme les graphes sont beaucoup étudiés et très bien définis en informatique, il suffit souvent de ramener un problème à un problème de graphe typique et d'y appliquer la solution que l'on connaît déjà pour ce problème.

Mots clés et sites web

Graphe, théorie des graphes

- [https://fr.wikipedia.org/wiki/Graphe_\(math%C3%A9matiques_discr%C3%A8tes\)](https://fr.wikipedia.org/wiki/Graphe_(math%C3%A9matiques_discr%C3%A8tes))
- [https://fr.wikipedia.org/wiki/Abstraction_\(philosophie\)](https://fr.wikipedia.org/wiki/Abstraction_(philosophie))



A. Auteurs des exercices

Tony René Andersen	Takeharu Ishizuka	Henry Ong
Haim Averbuch	M. Faiz Ahmad Ismail	Margot Phillipps
Michelle Barnett	Yong-ju Jeon	Zsuzsa Pluhár
Michael Barot	Felipe Jiménez	Wolfgang Pohl
Wilfried Baumann	Anna Laura John	Sergei Pozdniakov
Jan Berki	Mile Jovanov	Stavroula Prantsoudi
Linda Bergsveinsdóttir	Ungyeol Jung	Nol Premasathian
Daniela Bezáková	Ilya Kaysin	J.P. Pretti
Laura Braun	Adem Khachnaoui	Milan Rajković
Špela Cerar	Injoo Kim	Chris Roffey
Mony Chanroath	Jihye Kim	Andrea Schrijvers
Marios Choudary	Vaidotas Kinčius	Eljakim Schrijvers
Anton Chukhnov	Mária Kiss	Humberto Sermeno
Sébastien Combéfis	Jia-Ling Koh	Vipul Shah
Kris Coolsaet	Sophie Koh	Daigo Shirai
Allira Crowe	Dennis Komm	Taras Shpot
Andrew Csizmadia	Anja Koron	Jacqueline Staub
Valentina Dagienė	Bohdan Kudrenko	Nikolaos Stratis
Christian Datzko	Regula Lacher	Gabrielė Stupurienė
Maria Suyana Datzko	Anh Vinh Lê	Maciej M. Sysło
Sarah Estrella Datzko	Greg Lee	Bundit Thanasopon
Susanne Datzko	Inggriani Liem	Monika Tomcsányiová
Guillaume de Moffarts	Judith Lin	Peter Tomcsányi
Lanping Deng	Lynn Liu	Nicole Trachsler
Marissa Engels	Violetta Lonati	Jiří Vaníček
Olivier Ens	Vũ Văn Luân	Troy Vasiga
Gerald Futschek	Karolína Mayerová	Ela Veza
Sonali Gogate	Mattia Monga	Márton Visnovitz
Arnheiður Guðmundsdóttir	Samart Moodleah	Florentina Voboril
Martin Guggisberg	Anna Morpurgo	Michael Weigend
Vernon Gutierrez	Madhavan Mukund	Jing-Jing Yang
Juraj Hromkovič	Tom Naughton	Xing Yang
Alisher Ikramov	Pia Niemelä	Khairul A. Mohamad Zaki
Thomas Ioannou	Tomohiro Nishida	
Tiberiu Iorgulescu	Assylkan Omashev	



B. Sponsoring : Concours 2019


HASLERSTIFTUNG <http://www.haslerstiftung.ch/>

ROBOROBO <http://www.roborobo.ch/>


**bischof
berger** <http://www.baerli-biber.ch/>



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


**Kanton Zürich
Volkswirtschaftsdirektion
Amt für Wirtschaft und Arbeit** Standortförderung beim Amt für Wirtschaft und Arbeit Kanton Zürich


i-factory (Musée des transports, Lucerne)


UBS <http://www.ubs.com/>


bbv <http://www.bbv.ch/>
Software Services


PRESENTEX <http://www.presentex.ch/>
Das Geschenk - die gute Werbung


OXOCARD <http://www.oxocard.ch/>
OXOcard
OXON


DIARTIS <http://www.diartis.ch/>
Diartis AG



<https://educatec.ch/>
educatec



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



AUSBILDUNGS- UND BERATUNGSZENTRUM
FÜR INFORMATIKUNTERRICHT

<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



Fachhochschule
Nordwestschweiz

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

Scuola universitaria professionale
della Svizzera italiana



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



Zürcher Hochschule der Künste
Game Design

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



C. Offres ultérieures

010100110101011001001001
010000010010110101010011
010100110100100101000101
001011010101001101010011
010010010100100100100001

SS!E

www.svia-ssie-ssii.ch
schweizerischerverein für informatik und
erausbildung // société suisse pour l'infor-
matique dans l'enseignement // società sviz-
zera per l'informatica nell'insegnamento

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.