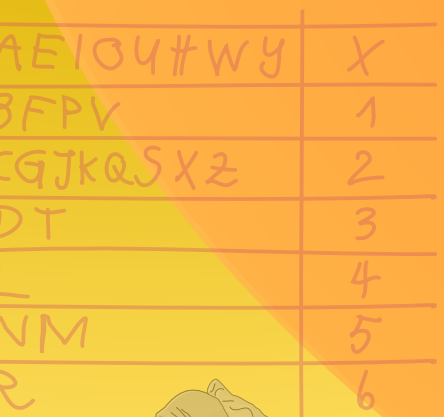
INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA

Exercices et solutions 2018
Années HarmoS 9/10

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

Éditeurs :
Gabriel Parriaux, Jean-Philippe Pellet, Elsa Pellet, Julien Ragot, Christian Datzko, Susanne Datzko, Hanspeter Erni



SSIE
www.svia-ssie-ssii.ch
schweizerischervereinfürinformatikind
erausbildung//sociétésuissepourl'infor
matiquedansl'enseignement//societàsviz
zeraperl'informaticanell'insegnamento

Ont collaboré au Castor Informatique 2018

Andrea Adamoli, Christian Datzko, Susanne Datzko, Olivier Ens, Hanspeter Erni, Martin Guggisberg, Carla Monaco, Gabriel Parriaux, Elsa Pellet, Jean-Philippe Pellet, Julien Ragot, Beat Trachsler.

Nous adressons nos remerciements à :
Juraj Hromkovič, Urs Hauser, Regula Lacher, Jacqueline Staub : ETHZ
Andrea Maria Schmid, Doris Reck : PH Luzern
Gabriel Thullen : Collège des Colombières
Valentina Dagienė : Bebras.org
Hans-Werner Hein, Ulrich Kiesmüller, Wolfgang Pohl, Kirsten Schlüter, Michael Weigend : Bundesweite Informatikwettbewerbe (BWINF), Allemagne
Chris Roffey : University of Oxford, Royaume-Uni
Anna Morpurgo, Violetta Lonati, Mattia Monga : ALaDDIn, Università degli Studi di Milano, Italie
Gerald Futschek, Wilfried Baumann : Oesterreichische Computer Gesellschaft, Austria
Zsuzsa Pluhár : ELTE Informatikai Kar, Hongrie
Eljakim Schrijvers, Daphne Blokhuis, Arne Heijenga, Dave Oostendorp, Andrea Schrijvers : Eljakim Information Technology bv, Pays-Bas
Roman Hartmann : hartmannGestaltung (Flyer Castor Informatique Suisse)
Christoph Frei : Chragokyberneticks (Logo Castor Informatique Suisse)
Andrea Adamoli (page web)
Andrea Leu, Maggie Winter, Brigitte Maurer : Senarclens Leu + Partner

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

INFORMATIK-BIBER SCHWEIZ
CASTOR INFORMATIQUE SUISSE
CASTORO INFORMATICO SVIZZERA

Le Castor Informatique 2018 a été réalisé par la Société Suisse de l'Informatique dans l'Enseignement SSIE. Le Castor Informatique est un projet de la SSIE, aimablement soutenu par la Fondation Hasler.

HASLERSTIFTUNG

Tous les liens ont été vérifiés le 1er novembre 2018. Ce cahier d'exercice a été produit le 9 octobre 2019 avec avec le logiciel de mise en page LaTeX.

Préambule

Très bien établi dans différents pays européens depuis plusieurs années, le concours « Castor Informatique » a pour but d'éveiller l'intérêt des enfants et des jeunes pour l'informatique. En Suisse, le concours est organisé en allemand, en français et en italien par la SSIE, la Société Suisse pour l'Informatique dans l'Enseignement, et soutenu par la Fondation Hasler dans le cadre du programme d'encouragement « FIT in IT ».
Le Castor Informatique est le partenaire suisse du concours « Bebras International Contest on Informatics and Computer Fluency » (`https://www.bebras.org/`), initié en Lituanie.
Le concours a été organisé pour la première fois en Suisse en 2010. Le Petit Castor (années HarmoS 5 et 6) a été organisé pour la première fois en 2012.
Le Castor Informatique vise à motiver les élèves à apprendre l'informatique. Il souhaite lever les réticences et susciter l'intérêt quant à l'enseignement de l'informatique à l'école. Le concours ne suppose aucun prérequis quant à l'utilisation des ordinateurs, sauf de savoir naviguer sur Internet, car le concours s'effectue en ligne. Pour répondre, il faut structurer sa pensée, faire preuve de logique mais aussi de fantaisie. Les exercices sont expressément conçus pour développer un intérêt durable pour l'informatique, au-delà de la durée du concours.
Le concours Castor Informatique 2018 a été fait pour cinq tranches d'âge, basées sur les années scolaires :

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

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

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

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

Pour de plus amples informations :

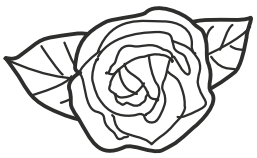
SVIA-SSIE-SSII Société Suisse de l'Informatique dans l'Enseignement
Castor Informatique

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

Table des matières

1. Les fleurs de Clara

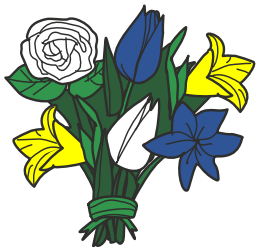
Clara va chez le fleuriste, car elle aime les bouquets de fleurs colorés. Elle y trouve les sortes de fleurs suivantes :

glaïeuls	lys	tulipes	roses
			

Chaque sorte de fleur est disponible en trois couleurs : blanc, bleu et jaune.
Clara aimerait un bouquet de six fleurs qui remplit les conditions suivantes :

1. Il doit y avoir deux fleurs de chaque couleur (blanc, bleu, jaune),
2. Les fleurs de la même sorte ne doivent jamais être de la même couleur,
3. Il ne doit pas y avoir plus de deux fleurs de la même sorte.

Quel est le bouquet qui remplit les trois conditions ?

A)	B)	C)	D)
			

Solution

La bonne réponse est D). Le bouquet A) contient trois fleurs blanches (contrevient à la règle 1), le bouquet B) trois roses (contrevient à la règle 3), et le bouquet C) deux glaïeuls jaunes (contrevient à la règle 2).

C'est de l'informatique !

Les problèmes informatiques courants sont décrits par un ensemble de restrictions, la tâche étant de trouver une solution qui respecte toutes ou le plus possible de ces restrictions.
On peut considérer des tâches plus complexes lors desquelles des opérateurs logiques comme le connecteur ET (A ET B signifie que les deux conditions A et B doivent être remplies, comme les trois règles dans notre exercice) ou le connecteur OU (A OU B signifie que seulement une des deux conditions doit être remplie).

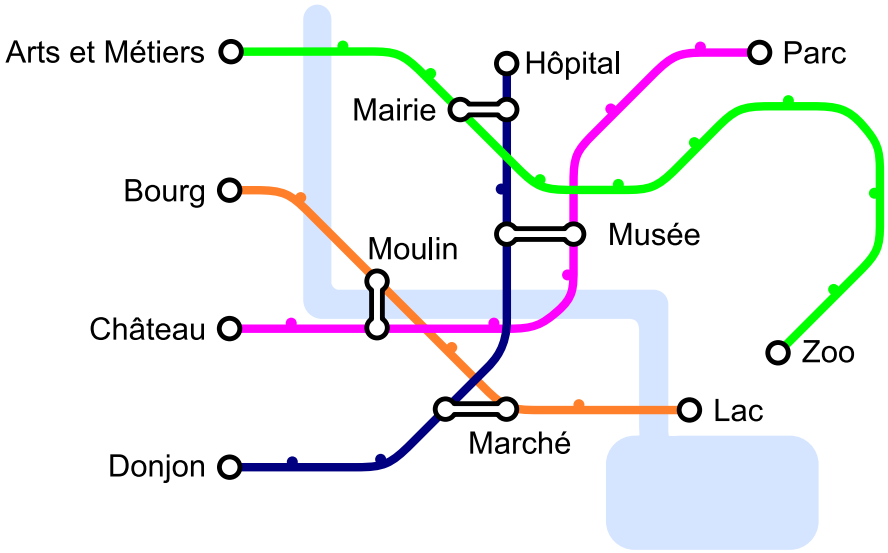
Mots clés et sites web

Conditions, opérateurs logiques

- `https://bookofbadarguments.com/`
- `https://fr.wikipedia.org/wiki/Connecteur_logique`
- `https://www.iep.utm.edu/prop-log/`

2. Réseau de lignes

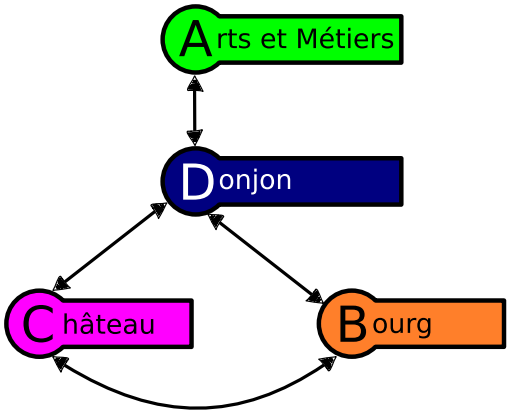
Dans la ville des castors, il y a quatre lignes avec pour point de départ quatre stations différentes : les stations « Arts et Métiers », « Bourg », « Château » et « Donjon ». Chaque ligne comprend au moins une station de transit qui permet de changer de ligne : la station « Musée », la station « Marché », la station « Moulin » et la station « Mairie ».



Aujourd'hui, Jean se rend au zoo. Il sait qu'il ne va changer de ligne qu'une seule fois. De quelle station de départ Jean est-il parti ?

Solution

La réponse correcte est « Donjon ». Quand on rebrousse chemin depuis la station « Zoo », on voit qu'il n'y a qu'une seule station de transit sur la ligne verte : « Mairie ». On en déduit que Jean est venu par la ligne bleue et que sa station de départ est « Donjon », étant donné qu'il n'a changé qu'une fois.
En informatique, il est possible de représenter le réseau de lignes à l'aide d'un graphe. Dans celui-ci, on représente les lignes par des nœuds et on les relie lorsqu'une station de transit permet de passer directement d'une ligne à l'autre. Par exemple, la ligne verte de « Arts et Métiers » à « Zoo », représentée par le A vert, est connectée à la ligne bleue de « Donjon » à « Hôpital », représentée par le D bleu, via la station de transit « Mairie » et donc le A est connecté au D dans ce schéma.



De cette ligne…	…on atteint ces lignes en un seul changement		
Arts et Métiers ↔ Zoo	Donjon ↔ Hôpital		
Bourg ↔ Lac	Château ↔ Place du parc	Donjon ↔ Hôpital	
Château ↔ Place du parc	Bourg ↔ Lac	Donjon ↔ Hôpital	
Donjon ↔ Hôpital	Bourg ↔ Lac	Château ↔ Place du parc	Arts et Métiers ↔ Zoo

Si on veut prendre la ligne « Arts et Métiers » ↔ « Zoo » pour arriver au zoo en ne changeant qu'une seule fois, on ne peut le faire qu'à partir de la station de départ « Donjon ». Ceci se lit dans le graphe en constatant que D est connecté uniquement à A, ou dans le tableau en voyant que A) n'apparaît qu'en regard de D), à la dernière ligne.

C'est de l'informatique !

Si tout cela te semble familier, c'est parce que beaucoup de réseaux de lignes de bus, de tramways ou de métros ressemblent à ce diagramme schématique qui représente les lignes et les stations de ces transports publics. Il s'agit là d'une véritable invention : en 1931, Henry Beck a élaboré un diagramme schématique pour le système de métro de Londres.
En informatique, un tel modèle abstrait est appelé un graphe. Un graphe se constitue par des nœuds (les stations) et des arêtes (le trajet entre deux stations). Dans notre tâche, il faut distinguer les nœuds qui présentent une ou deux arêtes (les stations de départ de fin ainsi que les stations intermédiaires) de ceux qui présentent un plus grand nombre d'arêtes (les stations de transit).

Dans la vie quotidienne, les graphes ont de nombreuses applications : les algorithmes élaborés d'un graphe peuvent par exemple résoudre des problèmes dans le domaine des réseaux sociaux, des guides routiers ou encore en matière de recherches sur les suggestions d'achats sur Internet. C'est la raison pour laquelle la maîtrise des graphes est une des compétences informatiques essentielles.

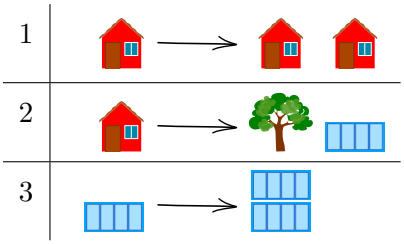
Mots clés et sites web

réseau de lignes, graphe

- `https://fr.wikipedia.org/wiki/Plan_du_métro_de_Londres`
- `https://fr.wikipedia.org/wiki/Graphe_(mathématiques_discrètes)`

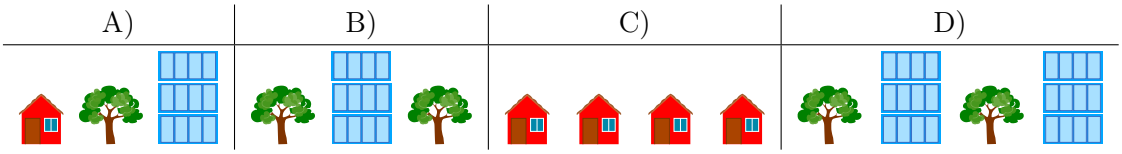
3. Planète Z

Les habitants de la planète Z construisent toutes leurs villes de la même manière. Ils commencent chaque ville avec une maison, puis remplacent les bâtiments construits les uns après les autres en suivant les règles suivantes :



Par exemple, en appliquant d'abord la règle 1, puis la règle 2, puis deux fois la règle 3, on obtient la ville à droite de l'image ci-dessous :

L'ordre dans lequel sont arrangés les bâtiments et les arbres ne peut pas être modifié.
Laquelle des villes suivantes ne *peut-elle pas se trouver sur la planète Z ?*



Solution

La bonne réponse est B). Les arbres ne sont plantés dans la ville qu'en suivant la règle 2, et la règle 2 spécifie qu'il doit y avoir un immeuble à droite de chaque arbre. Dans la ville B), il n'y a pas d'immeuble à droite du deuxième arbre. Comme il n'existe pas de règle permettant d'enlever un immeuble, la ville B) ne peut pas être sur la planète Z.
La ville A) peut être construite en appliquant la suite de règles 1, 2, 3, et 3.
La ville C) peut être construite en appliquant trois fois la règle 1.
La ville D) peut être construite en appliquant d'abord la règle 1, puis la règle deux sur chacune des deux maisons. Finalement, on applique deux fois la règle 3 sur chaque immeuble.

C'est de l'informatique !

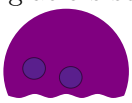
Les règles de cet exercices sont appelées règles de réécriture : un symbole ou un objet est remplacé par une série d'autres symboles ou objets. Si chaque règle ne remplace qu'un seul objet à la fois, le set de règles est appelé algébrique (ou non contextuel). Un symbole ou un objet est remplacé sans que le contexte (c'est-à-dire ce qui se trouve à sa gauche et à sa droite) ne joue de rôle.
En informatique, les règles de réécriture sont par exemple utilisées pour définir la syntaxe d'un langage de programmation. Les symboles ou objets sont des termes clés et les règles décrivent de quelle manière on peut les assembler en un programme (syntaxiquement) correct. Dans cet exercice, les symboles ou objets sont les maisons, les arbres et les immeubles. Les symboles ou objets et les règles de réécriture forment ensemble la grammaire décrivant un langage.
Lorsqu'un ordinateur traduit (compile) un programme en langage machine ou l'exécute directement (via un interpréteur pour les programmes écrits dans un langage de script), il commence par vérifier si le texte du programme suit bien les règles du langage de programmation. Il essaie donc, à l'aide d'un arbre syntaxique, de reconstruire les règles de réécriture qui transforment le symbole de départ (une maison dans cet exercice) en un texte de programme (les quatre réponses possibles dans cet exercice). Dans notre cas, c'est facilement réalisable, car les *mots* (des suites de symboles ou objets, dans cet exercice les villes) deviennent toujours plus grands et ne peuvent pas rétrécir.

Mots clés et sites web

Règle de réécriture, grammaire, langage algébrique

- `https://en.wikipedia.org/wiki/Production_(computer_science)`
- `https://fr.wikipedia.org/wiki/Langage_algébrique`
- `https://fr.wikipedia.org/wiki/Arbre_syntaxique`
- `https://fr.wikipedia.org/wiki/Grammaire_non_contextuelle`
- `https://fr.wikipedia.org/wiki/Règles_de_réécriture`

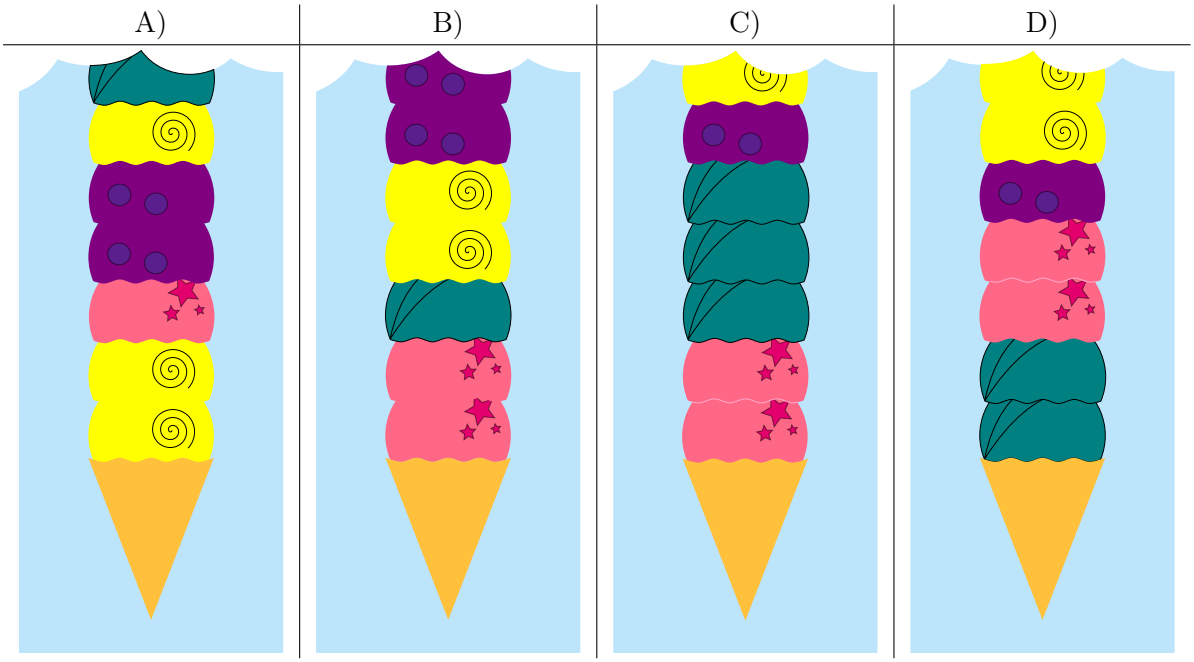
4. Glacier

Il y a deux glaciers sur la place du village. Ils offrent les quatre mêmes sortes de glace :  ,  ,  et  .

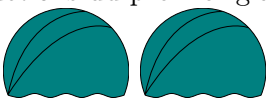
Le premier glacier suit les instructions suivantes pour préparer un cornet de glace :

1. Prends un cornet vide.
2. Choisis une sorte de glace au hasard et mets-en deux boules dans le cornet.
3. Ajoute une boule d'une des trois autres sortes de glace.
4. Si le nombre de boules souhaité est atteint, arrête. Sinon, recommence à l'étape 2.

Le deuxième glacier ne suit aucune règle.
L'image suivante représente les premières boules de quelques cornets. Lequel provient à coup sûr du deuxième glacier ?



Solution

La bonne réponse est D). C'est le seul cornet qui ne peut pas avoir été préparé en suivant les instructions du premier glacier. Il commence en accord avec les règles avec deux boules de la même sorte  suivies d'une boule d'une autre sorte  , mais ensuite viennent deux boules de deux sortes différentes  , alors que les instructions spécifient que deux boules de la même sorte devraient être ajoutées.
Les réponses A), B) et C) ne sont pas correctes. Tous ces cornets pourraient avoir été préparés d'après les instructions du premier glacier.

C'est de l'informatique !

Une série d'instructions permet de créer des motifs dans des cornets de glace, des textes ou des images. Les informaticiennes et informaticiens développent des programmes informatiques permettant de reconnaître des motifs et des variations dans ces motifs. Parfois, des motifs sont générés par la répétition d'instructions. Par exemple, ce simple motif  est généré par la répétition de  suivi de  . De tels motifs sont faciles à reconnaître. Le problème est plus compliqué dans cet exercice, car les instructions du premier glacier contiennent aussi des décisions laissées au hasard.
Il n'est de manière générale pas possible d'être sûr qu'une séquence a été générée par hasard ou en suivant une série d'instructions. Dans cet exercice, nous avons pu déterminer que l'un des cornets ne correspondait pas aux instructions et devait donc venir du deuxième glacier. On ne peut par contre jamais être sûr qu'une glace vient du premier glacier et pas du deuxième, car la composition correspondant aux instructions pourrait avoir été générée par hasard.

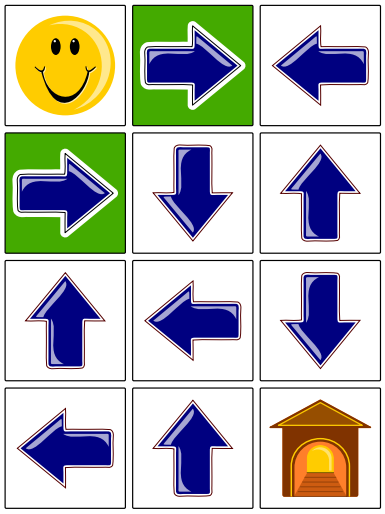
Mots clés et sites web

Reconnaissance de motifs

— `https://fr.wikipedia.org/wiki/Reconnaissance_de_formes`

5. Labyrinthe fléché

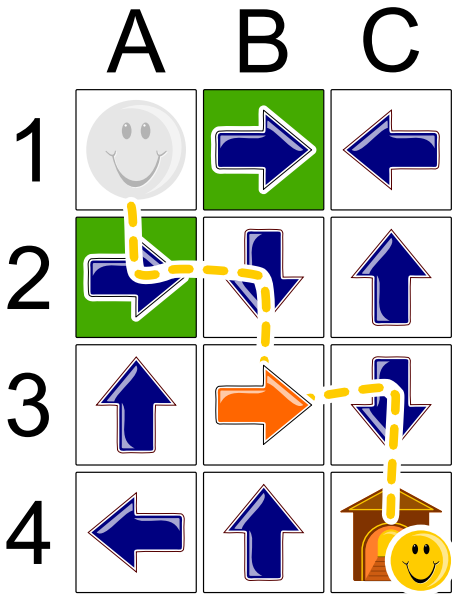
Le smiley aimerait rentrer à la maison, mais pour y arriver, il doit d'abord traverser un labyrinthe fléché. Il peut utiliser l'une des deux entrées (cases vertes). Lorsqu'il se trouve sur une case contenant une flèche, il doit quitter cette case dans le sens de la flèche. La position actuelle des flèches ne lui permet en aucun cas de rentrer à la maison.



Quelle flèche doit changer de direction pour permettre au smiley de rentrer à la maison ?

Solution

La flèche dont la direction doit changer est colorée en rouge. Le chemin du smiley jusqu'à la maison est indiqué par une ligne : A1 – A2 – B2 – B3 – C3 – C4.



Il est même possible de prouver que c'est la seule solution possible. Si l'on commence par la case cible C4 et se déplace à l'envers, on voit qu'il y a deux possibilités d'arriver à C4 : depuis C3 ou depuis B4. La flèche B4 ne pointe pas dans la bonne direction et doit donc être changée. Seulement, comme aucune des flèches voisines de B4 ne pointe en direction de B4, il faudrait changer la direction d'une deuxième flèche, ce qui n'est pas permis.
On ne peut donc arriver à C4 qu'en passant par la case C3. La flèche C3 pointe déjà en direction de la maison et ne doit pas être modifiée, mais les cases voisines de C3 ne contiennent pas de flèche pointant vers C3. Il faut donc changer la direction de la flèche B3 ou C2, les deux voisines de C3. Comme il n'y a pas de flèche pointant vers C2, C3 ne peut pas être atteinte en passant par C2 et en ne modifiant qu'une seule flèche. Il faut donc passer par la case B3. On peut atteindre B3 depuis l'une des deux entrées sans modifier la direction de flèches supplémentaires (A1 – A2 – B2). C'est donc la seule solution possible.

C'est de l'informatique !

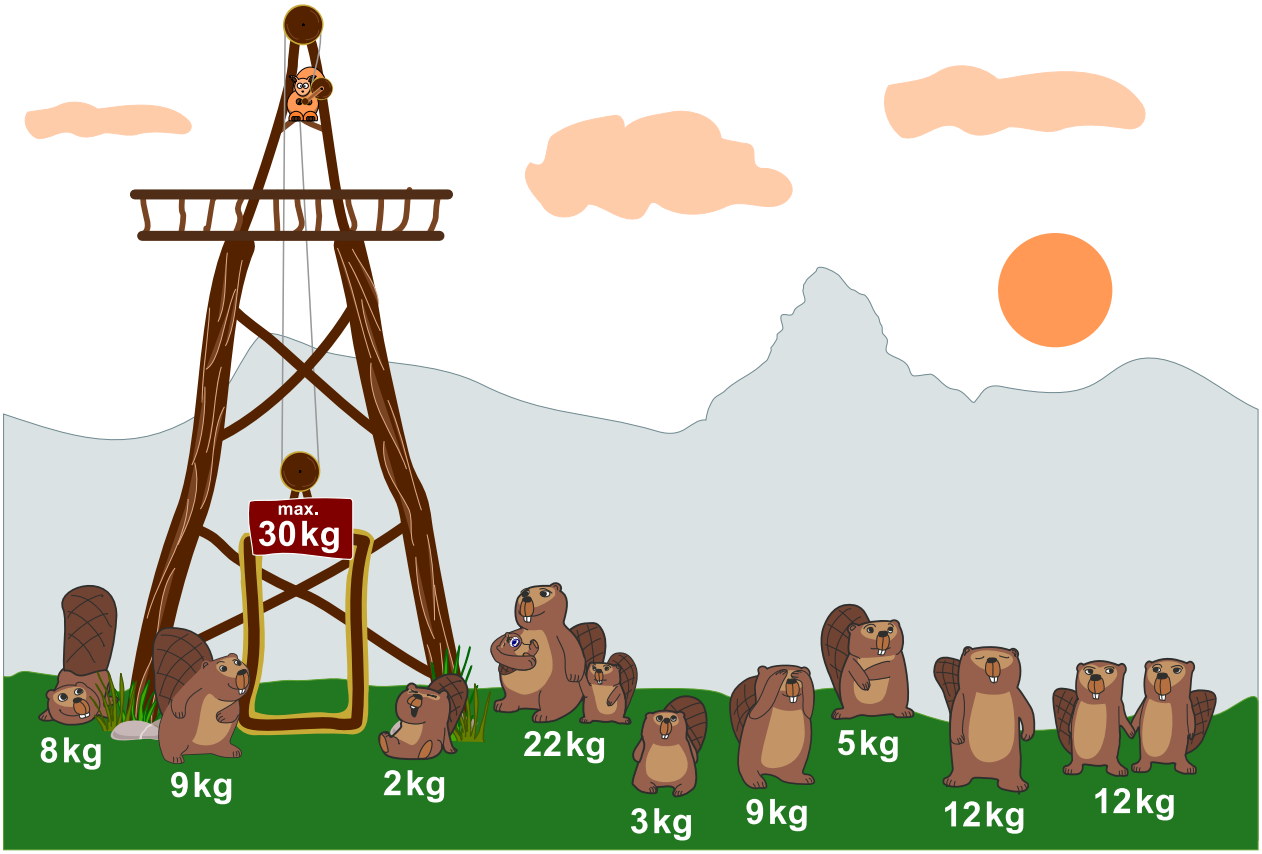
Lors de cet exercice, il a fallu trouver un chemin par le labyrinthe fléché en ne modifiant qu'une seule flèche. Comment trouve-t-on la solution d'un tel problème ? Comment un ordinateur pourrait-il procéder ? Un procédé appelé *retour sur trace* a été utilisé pour la démonstration du paragraphe précédent. Cela se fait grosso modo de la manière suivante : On suit une trace (depuis le départ ou depuis la fin) jusqu'à ce que l'on soit bloqué. On revient ensuite à l'étape précédente et on cherche une autre trace depuis là. De cette manière, on peut exclure les chemins impossibles dès le départ et être sûr de trouver la solution, étant donné que l'on essaie chaque possibilité.

Mots clés et sites web

Labyrinthe, retour sur trace
— `https://fr.wikipedia.org/wiki/Retour_sur_trace`

6. Excursion avec vue

Une famille de castors fait une excursion jusqu'à une tour panoramique. Ils sont en retard. L'ascenseur ne monte plus que deux fois avant la fermeture, et ne peut pas transporter plus de 30 kg d'un coup. Les jumeaux ne veulent monter qu'ensemble sur la tour. La maman castor porte le bébé dans ses bras et tient la main d'un petit castor. Et pourtant, on aimerait faire monter le plus grand nombre possible de castors au sommet de la tour.



Il faut se décider rapidement et seules les cinq options suivantes sont possibles. Qui doit rester en bas pour que le plus de castors possible puissent atteindre le sommet de la tour panoramique ?

A) Tout le monde peut monter.

B) La maman castor avec le bébé et le petit castor.

C) Les jumeaux et le castor de 5 kg.

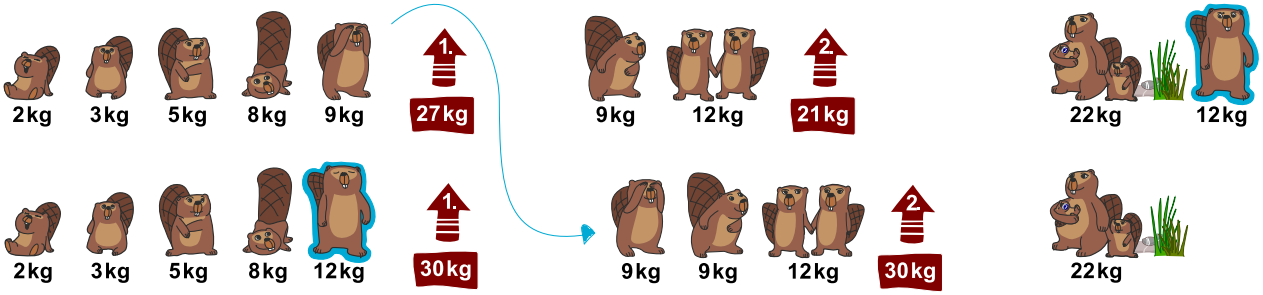
D) Les jumeaux et la maman castor avec le bébé et le petit castor.

E) La maman castor avec le bébé et le petit castor et le castor de 12 kg.

Solution

B) est l'une des bonnes solutions :

Une solution pourrait être de charger la cabine avec les castors les plus légers en premier : 2 kg + 3 kg + 5 kg + 8 kg + 9 kg = 27 kg lors de la première montée, et 9 kg + 12 kg = 21 kg lors de la seconde montée (cela fait 8 castors). Mais il y a encore de la place pour un castor supplémentaire dans l'ascenseur :



Cette stratégie permet d'utiliser les deux trajets en ascenseur de manière optimale : le castor de 9 kg est remplacé par celui de 12 kg lors du premier trajet. Le poids maximal de 30 kg est ainsi atteint. Le castor de 9 kg peut monter dans la cabine pour le second trajet, qui transporte également le poids maximal de 30 kg.
Les autres solutions proposées ne sont pas possibles : soit elles dépassent la charge maximale (le poids de la famille de castors entière dépasse 60 kg et même sans les jumeaux et le castor de 5 kg, elle atteint encore 65 kg), soit elles sont moins bonnes (si la maman avec les petits castors monte, ni les jumeaux, ni le castor de 9 kg, ni celui de 12 kg ne peuvent l'accompagner).

C'est de l'informatique !

Un des problèmes classiques en informatique est de trouver la combinaison optimale comme solution d'un problème. C'est souvent impossible de trouver un telle solution assez rapidement, ou dans un temps réaliste, parce qu'il y a trop de solutions possibles qui doivent être examinées. En informatique, on appelle cela un problème insoluble en pratique.
Ce problème est du même type que le *problème du sac à dos* lors duquel il faut ranger le plus d'objets possible dans un sac sans dépasser un poids maximal. Ce problème et les problèmes similaires sont appelés *NP-complets*. Ils ne peuvent être résolus que de manière approximative, ce qui veut dire que l'on peut trouver une bonne solution, mais pas nécessairement la solution optimale. On y parvient en réfléchissant à une stratégie judicieuse (« heuristique ») pour résoudre le problème.

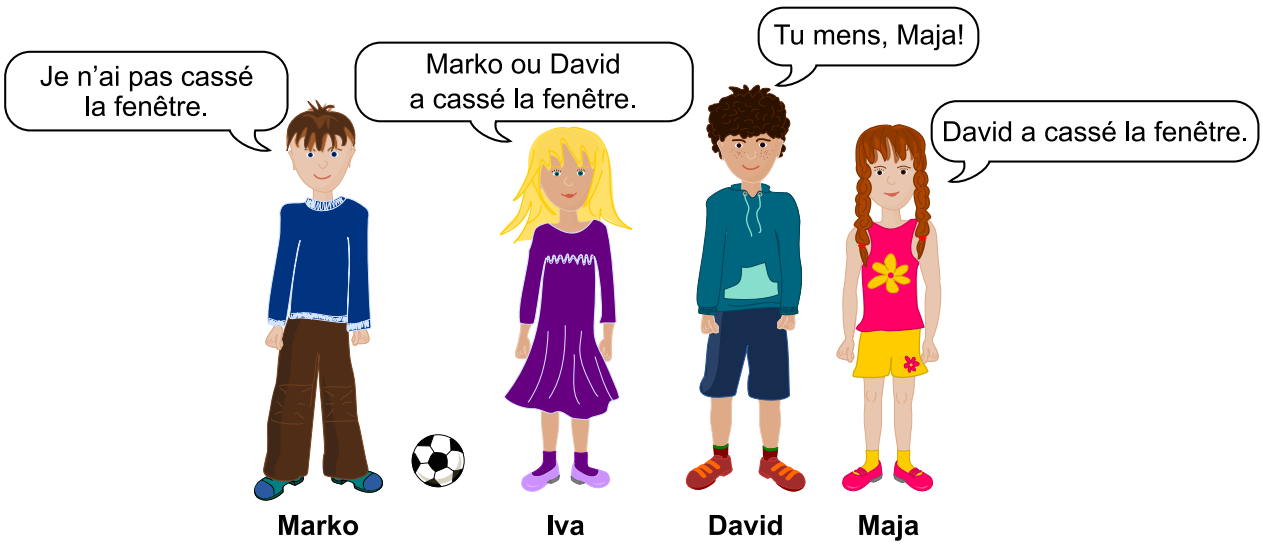
Mots clés et sites web

Optimisation, problème du sac à dos

- `https://fr.wikipedia.org/wiki/Optimisation_combinatoire`
- `https://fr.wikipedia.org/wiki/Problème_du_sac_à_dos`

7. Les mensonges ne mènent pas loin

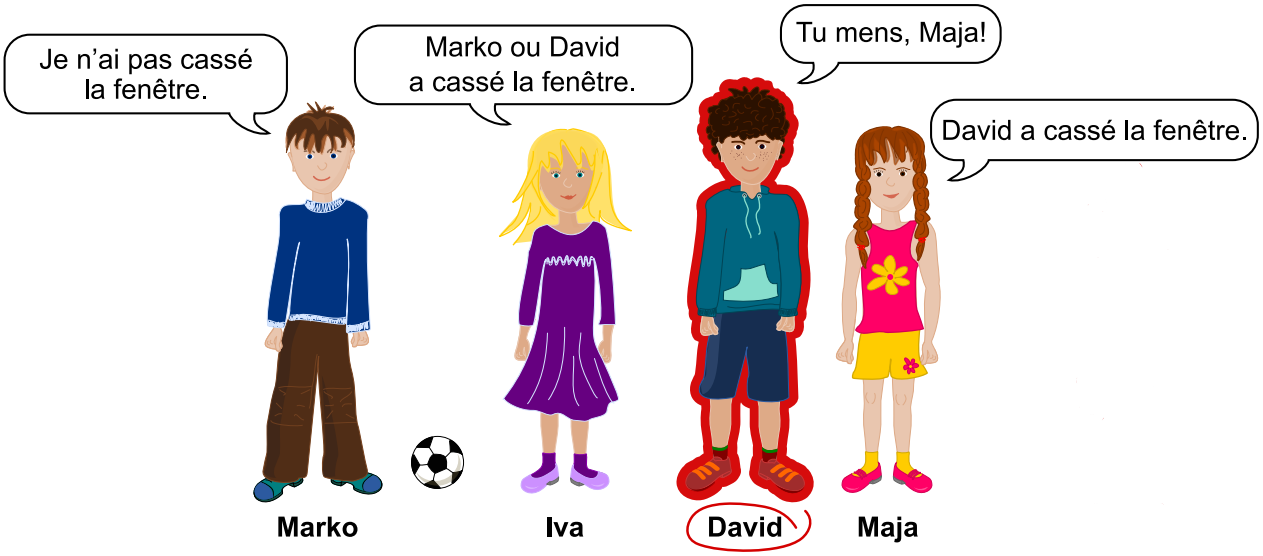
Par un jour de beau temps, Maja, David, Iva et Marko jouent au football près de la maison d'Anna. Tout à coup, une des fenêtres se casse et Anna cherche à savoir qui est responsable. Elle connaît les quatre enfants et sait que trois d'entre eux disent toujours la vérité ; elle ne sait pas ce qu'il en est du quatrième.
Les quatre enfants disent :



Clique sur l'enfant qui a cassé la fenêtre.

Solution

David a cassé la fenêtre.



Les déclarations de Maja et David ne peuvent pas être vraies toutes les deux ; l'un d'entre eux doit donc mentir. Si Maja disait la vérité, David mentirait et les déclarations d'Iva et de Marko seraient vraies. Par contre, si David disait la vérité, cela voudrait dire que Maja ment, mais aussi que soit Marko, soit Iva ment également, ce qui n'est pas possible étant donné que trois des enfants disent toujours la vérité.

C'est de l'informatique !

Tu dois raisonner de manière logique afin de résoudre ce problème. Le raisonnement se base sur la logique formulée en 1854 par George Boole (1815 – 1864), qui a décrit de manière formelle les bases d'énoncés logiques.
D'après lui, une déclaration (ou *assertion*) est soit *vraie* soit *fausse* (principe du tiers exclu). Plusieurs assertions peuvent être combinées à l'aide d'*opérateurs*. Des opérateurs logiques simples comme *ET* et *OU* lient deux assertions pour en former une nouvelle. Il existe aussi des opérateurs comme *NON* qui ne modifient qu'une seule assertion. La véracité de telles assertions combinées peut être déterminée à l'aide de *tables de vérité*.
L'un des opérateur qui représente la déduction « SI » → « ALORS » est l'implication. On parle alors de « tirer des conclusions logiques », c'est ce qui est nécessaire à la résolution de cet exercice.
Les ordinateurs fonctionnent également sur la base d'assertions booléennes et d'opérateurs logiques simples, car de tels ordinateurs peuvent être produits facilement en grandes quantités. Il existe quelques ordinateurs basés sur d'autres systèmes (par exemple les ordinateurs ternaires de la fin des années 50 en Union soviétique), mais soit ils sont restés au stade expérimental, soit ils n'ont jamais atteint une production de masse.

Mots clés et sites web

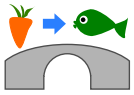
Logisches Schliessen

- `https://fr.wikipedia.org/wiki/Principe_du_tiers_exclu`
- `https://fr.wikipedia.org/wiki/George_Boole`
- `https://fr.wikipedia.org/wiki/Algèbre_de_Boole_(logique)`

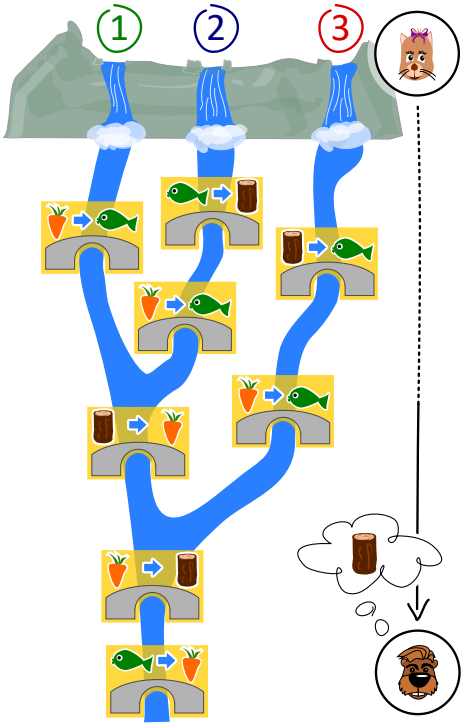
— `https://fr.wikipedia.org/wiki/Implication_(logique)`
— `https://fr.wikipedia.org/wiki/Ordinateur_ternaire`
— `https://it.wikipedia.org/wiki/Calcolatore_ternario`

8. Chutes d'eau

Katja est au sommet d'une montagne. Cette montagne a trois chutes d'eau qui se rejoignent dans une rivière en bas de la vallée.
Katja peut lancer un poisson ou une carotte dans l'une des trois chutes d'eau. Les cours d'eau sont enjambés par plusieurs ponts sous lesquels vivent des trolls. Les trolls remplacent les objets passant sous les ponts par d'autres objets.



Par exemple, si une carotte passe sous un pont comme celui ci-dessus, les trolls la remplacent par un poisson.



Justus est au bord de la rivière en bas de la vallée.
Justus a besoin de bois. Quel objet Katja doit-elle lancer dans quelle chute d'eau afin que Justus reçoive du bois ?

A) Elle lance un poisson dans la chute d'eau numéro 1.

B) Elle lance un poisson dans la chute d'eau numéro 2.

C) Elle lance une carotte dans la chute d'eau numéro 2.

D) Elle lance une carotte dans la chute d'eau numéro 3.

Solution

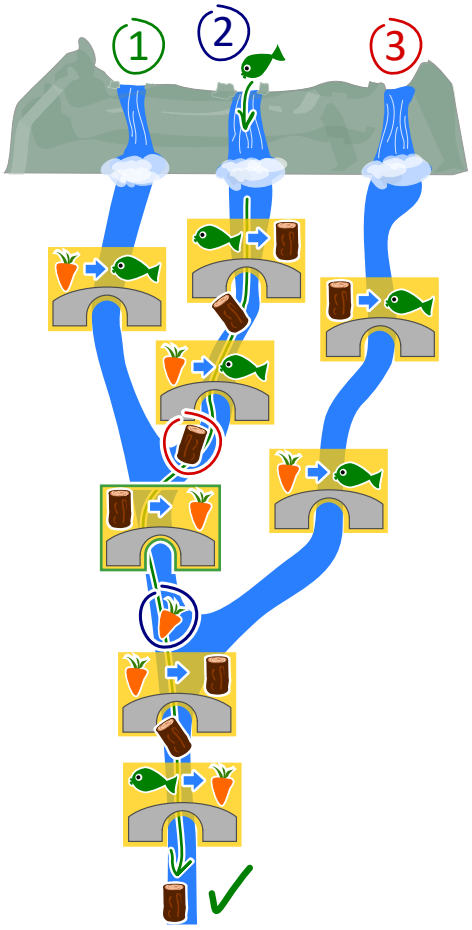
La bonne réponse est B). Elle lance un poisson dans la chute d'eau numéro 2.
Voilà ce qu'il se passe lors des différentes solutions proposées :

A) Un poisson lancé dans la chute d'eau numéro 1 ne sera remplacé que sous le dernier pont. Justus reçoit donc une carotte.

B) Un poisson lancé dans la chute d'eau numéro 2 est remplacé par du bois, qui est remplacé par une carotte, elle-même ensuite remplacée par du bois. Justus reçoit donc du bois. C'est la bonne solution.

C) Une carotte lancée dans la chute d'eau numéro 2 est remplacée par un poisson, qui est ensuite remplacé par une carotte. Justus reçoit donc une carotte.

D) Une carotte lancée dans la chute d'eau numéro 3 est remplacée par un poisson, qui est ensuite remplacé par une carotte. Justus reçoit donc une carotte.



Une autre manière de résoudre cet exercice consiste à commencer par la fin : Pour obtenir du bois en bas de la montagne, l'objet flottant doit être une carotte quand il passe sous l'avant-dernier pont. La seule possibilité d'avoir une carotte à cette étape est que du bois passe sous le pont commun des chutes numéro 1 et 2 (et pas 3) . La seule possibilité d'avoir du bois à cet endroit est de lancer un poisson dans la chute d'eau numéro 2.

C'est de l'informatique !

On peut se représenter un ordinateur comme une machine qui lit des données, les traite, puis écrit des données sortantes. Mais comment l'ordinateur « sait »-il comment traiter les données ? Il a reçu des ordres concernant les tâches à accomplir. Ceci se fait en écrivant des programmes.
Il existe beaucoup de langages de programmation qui fonctionnent selon différents paradigmes. La programmation fonctionnelle est un tel paradigme. Ce style de programmation est lui-même comme un petit ordinateur : il est composé de beaucoup de fonctions (ou routines) qui traitent des données et retournent des valeurs sortantes. Les ponts de cet exercice sont comme de petites fonctions, et le système complet comme un programme écrit dans un langage de programmation fonctionnel.

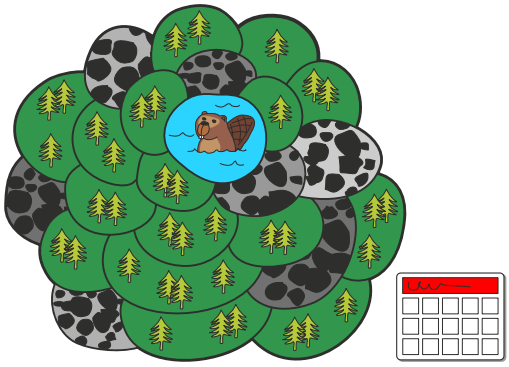
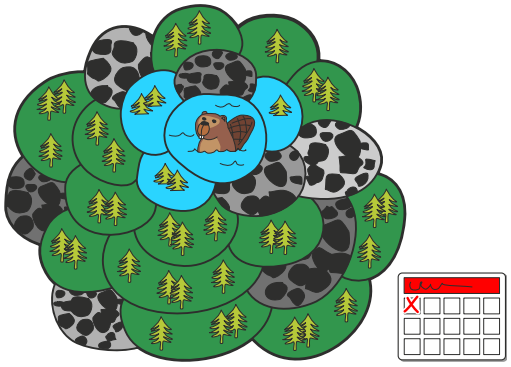
Mots clés et sites web

Paradigme de programmation, programmation fonctionnelle, fonctions et paramètres

- `https://fr.wikipedia.org/wiki/Test_(informatique)`
- `https://fr.wikipedia.org/wiki/Boîte_blanche`
- `https://fr.wikipedia.org/wiki/Test_de_la_boîte_noire`
- `https://fr.wikipedia.org/wiki/Paradigme_(programmation)`
- `https://fr.wikipedia.org/wiki/Programmation_fonctionnelle`
- `https://fr.wikipedia.org/wiki/Routine_(informatique)`

9. L'étang des castors

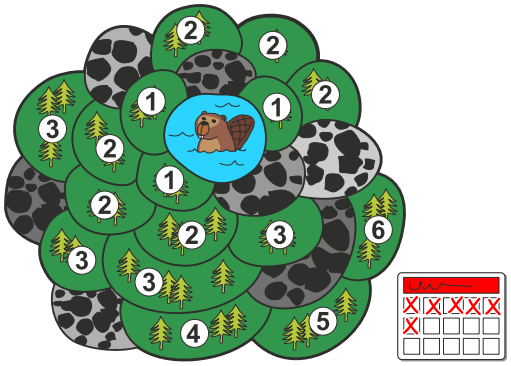
Il y a un petit étang dans une vallée. Il est entouré de parcelles de terrain forestier ou rocailleux. Plusieurs castors vivent dans l'étang.
Il vient un jour où les castors trouvent l'étang trop petit et décident d'inonder des parcelles de forêt. Chaque jour, ils inondent toutes les parcelles de forêt partageant une bordure avec une parcelle déjà inondée. Trois parcelles de forêt sont inondées le premier jour.

L'étang dans la vallée	Après un jour
	

Après combien de jours en tout (y compris le premier jour représenté plus haut) les parcelles forestières sont-elles toutes inondées ?

Solution

Toutes les parcelles de forêt sont inondées en six jours.
L'image ci-dessous montre au combientième jour chaque parcelle a été inondée. Les parcelles voisines de l'étang sont inondées après le premier jour et donc marquées du chiffre 1. Les parcelles voisines de ces dernières sont marquées du chiffre 2 ; elles sont inondées après le deuxième jour, et ainsi de suite. La dernière parcelle est marquée du chiffre six et est inondée le sixième jour – toutes les parcelles forestières sont donc inondées à ce moment-là.



C'est de l'informatique !

Dans cet exercice, les castors inondent un espace forestier connexe qui est composé, en plus de l'étang, de parcelles séparées. L'espace est connexe car l'on peut atteindre chaque parcelle de forêt en passant par d'autres parcelles sans sortir de l'espace forestier.
Il existe également en dehors de la vallée de l'étang des castors des espaces connexes devant être inondés. Une zone de couleur unie sur une image n'est finalement rien d'autre qu'un espace connexe de pixels de la même couleur. Un groupe d'adolescents, dans lequel chacun est relié à chacun par des liens d'amitiés directs ou par l'ami d'un ami d'un ami, est également un « espace connexe », si l'on considère un lien d'amitié entre deux personnes comme du voisinage.
En informatique, il y a des méthodes permettant de découvrir et d'investiguer les espaces connexes, comme les algorithmes de parcours en largeur ou en profondeur. Ces méthodes permettent par exemple de changer la couleur d'une zone sur une image ou de découvrir des groupements sur les réseaux sociaux.

Mots clés et sites web

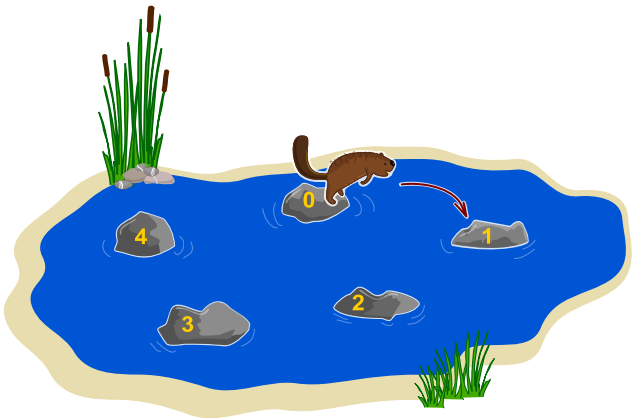
Algorithme à front d'onde, algorithme de parcours en largeur

- `https://fr.wikipedia.org/wiki/Graphe_connexe`
- `https://fr.wikipedia.org/wiki/Algorithme_de_parcours_en_largeur`

10. Compétition des castors

Plusieurs castors suivent un entraînement intensif en préparation à la compétition annuelle des castors. L'entraînement du jour consiste en un parcours de saut de pierre en pierre, dans le sens des aiguilles d'une montre, comme indiqué par la flèche. Si le castor saute 8 fois, il termine son parcours sur la pierre numéro 3 :

$$0 \to 1 \to 2 \to 3 \to 4 \to 0 \to 1 \to 2 \to 3.$$



Le castor le plus fort a sauté 129 *fois aujourd'hui. Sur quelle pierre se trouvait-il lorsqu'il s'est arrêté ?*

Solution

5 sauts amènent le castor sur la pierre de laquelle il est parti. Ces 5 sauts constituent un tour. Pour déterminer où le castor arrive après 129 sauts, nous devons déterminer combien de tours il fait, ainsi que combien de sauts il fait après le dernier tour complet. Ce sont dans ce cas-là $129 = 25 \cdot 5 + 4$ sauts (25 tours et 4 sauts), ce qui veut dire que le castor termine son parcours sur la même pierre que s'il n'avait sauté que 4 fois, donc sur la pierre numéro 4.

C'est de l'informatique !

Tu as peut-être déjà rencontré une opération semblable en cours de mathématique ; il s'agit d'une *division avec reste*, appelée aussi division euclidienne ou division posée.
Dans cet exercice, il s'agit d'effectuer la division euclidienne de $129 : 5 = 25$ reste 4. En informatique, le calcul du reste est fréquemment utilisé et a donc un nom spécifique : le modulo. Les signes « % » ou « mod » sont habituellement utilisés comme opérateurs, on peut donc écrire $129\%5 = 4$.
Cet opérateur est par exemple utilisé dans des boucles (comme le castor qui fait des tours en sautant) lorsque que des variables débordent et dans un procédé de cryptographie très répandu appelé « chiffrement RSA ».

Mots clés et sites web

Modulo-Operation

- `https://fr.wikipedia.org/wiki/Modulo_(opération)`
- `https://fr.wikipedia.org/wiki/Division_posée`
- `https://fr.wikipedia.org/wiki/Division_euclidienne`
- `https://fr.wikipedia.org/wiki/Chiffrement_RSA`

11. Maison numéro 29

Milo fait un stage dans un lotissement de maisons de vacances. Aujourd'hui, il doit fixer des plaques numérotées aux maisons de vacances. Certaines maisons sont déjà numérotées. Il commence par la maison numéro 50. Depuis là, il doit :

— aller à gauche si le nouveau numéro est plus petit que celui de la maison devant laquelle il se trouve,
— aller à droite si le nouveau numéro est plus grand que celui de la maison devant laquelle il se trouve,
— fixer la plaque numérotée à la maison devant laquelle il se trouve si celle-ci n'est pas encore numérotée.

A quelle maison de vacances Milo doit-il fixer le numéro 29 *?*

Solution

La maison de vacances correcte est la troisième depuis la gauche :

Le nouveau numéro 29 est plus petit que le numéro de la maison 50, donc Milo doit commencer par aller à gauche. Il arrive à la maison 24, dont le numéro est plus petit que 29, et doit donc aller à droite. Le numéro 29 est plus petit que celui de la maison 34 devant laquelle il arrive, donc il va à gauche. La maison suivante n'a pas encore de numéro, donc il y fixe la plaque numéro 29.

C'est de l'informatique !

La numérotation des maisons de vacances correspond à un *arbre binaire de recherche*, une structure de données souvent utilisée en informatique. Un arbre binaire de recherche permet de retrouver rapidement des données enregistrées.
Un arbre binaire de recherche est construit de telle façon qu'un « *élément* » est enregistré à chaque intersection (« *nœuds* »). Après chaque intersection, il y a au maximum deux chemins (« *arêtes* ») menant à d'autres intersections. Lors de l'enregistrement de nouvelles données, le chemin de gauche (par exemple) est toujours choisi lorsque le nouvel élément a une valeur plus petite que l'élément situé à l'intersection, et sinon le chemin de droite est suivi. Le nouvel élément est enregistré à la première intersection de libre.
Lorsque l'on cherche un élément précis, on peut ainsi facilement savoir quel chemin suivre à chaque intersection. Si l'arbre binaire de recherche est « *équilibré* » (appelé alors un *arbre AVL*), chaque étape de la recherche réduit de moitié le nombre d'intersections devant encore être au maximum parcourues avant de trouver l'élément recherché. Cela veut dire qu'un arbre de 1000 éléments peut être exploré en seulement 10 étapes, un arbre de 1'000'000 éléments en 20 étapes et un arbre de 1'000'000'000 éléments en 30 étapes (donc, pour n éléments, $\log_2(n)$ étapes).

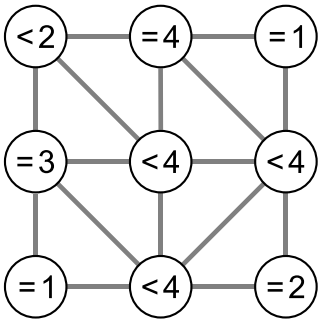
Mots clés et sites web

Arbre binaire de recherche, arbre AVL

- `https://fr.wikipedia.org/wiki/Arbre_binaire_de_recherche`
- `https://fr.wikipedia.org/wiki/Arbre_AVL`

12. Voisins

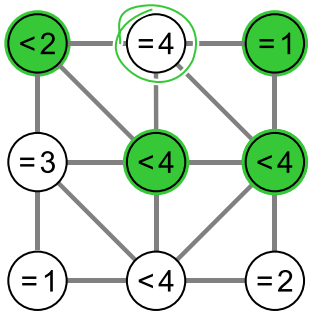
L'image ci-dessous montre neuf cercles partiellement connectés les uns aux autres. Une connexion entre deux cercles en fait des voisins. Les cercles peuvent être sélectionnés par un clic ; ils sont alors colorés en vert, alors que les cercles non-sélectionnés sont blancs.
Dans chaque cercle, une expression indique combien de cercles doivent être sélectionnés parmi les cercles voisins. Par exemple, le cercle portant l'expression « $= 3$ » doit avoir trois de ses quatre voisins sélectionnés, et les cercles portant l'expression « < 4 » peuvent en avoir au maximum trois qui sont sélectionnés.



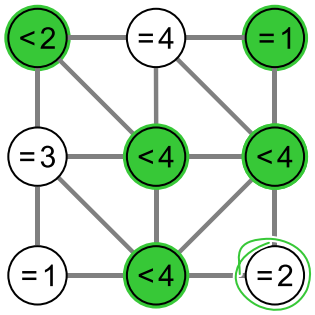
Sélectionne les cercles de manière à ce que toutes les conditions soient remplies.

Solution

Le cercle central de la rangée du haut porte l'expression « = 4 » et a quatre voisins, ce qui fait que chacun de ses quatre voisins doit être sélectionné :



Similairement, le cercle en bas à droite porte l'expression « = 2 » et possède deux voisins, qui doivent donc être les deux sélectionnés (le cercle du centre à droite est déjà sélectionné) :



Toutes les conditions sont ainsi remplies.
Si l'on sélectionnait un cercle supplémentaire, certaines conditions seraient violées :

- Si le cercle « = 4 » au centre de la rangée du haut était sélectionné, la condition « = 1 » du cercle en haut à droite ne serait plus remplie.
- Si le cercle « = 3 » à gauche de la rangée centrale était sélectionné, la condition « < 2 » du cercle en haut à gauche ne serait plus remplie.
- Si le cercle « = 1 » en bas à gauche était sélectionné, la condition « = 3 » du cercle à gauche de la rangée centrale ne serait plus remplie.
- Si le cercle « = 2 » en bas à droite était sélectionné, la condition « < 4 » du cercle à droite de la rangée centrale ne serait plus remplie.

C'est de l'informatique !

De combien d'essais a-t-on besoin pour résoudre le problème ? Si l'on essaie simplement chaque solution possible, on a deux réglages différents pour chacun des neuf cercles, ce qui fait $2^9 = 512$ possibilités différentes. On appelle cette méthode *recherche exhaustive* ou *recherche par force brute*. Pour chacune de ces 512 possibilités, il faudrait vérifier si toutes les conditions sont remplies.
Dans ce cas-là, il est plus judicieux de procéder de manière logique et cohérente. On cherche d'abord les cercles ayant des conditions ne pouvant être remplies que de manière unique. Il s'agit par exemple des cercles ayant exactement n connexions, donc n voisins, et la condition « = n ». On peut continuer à partir de là de manière logique en regardant s'il reste des conditions que l'on ne peut remplir que d'une seule façon. On peut ainsi trouver la bonne solution à moindre effort. De manière générale, on peut trouver de cette manière qu'il n'existe pas de solution, ou trouver au moins une solution parmi

plusieurs. On parle de démarche analytique lorsque seule une solution prometteuse parmi toutes les possibilités est considérée « *heuristique* ».

Mots clés et sites web

Voisinage dans la théorie de graphes, approche logique

- `https://fr.wikipedia.org/wiki/Voisinage_(théorie_des_graphes)`
- `https://fr.wikipedia.org/wiki/Heuristique_(mathématiques)`
- `https://fr.wikipedia.org/wiki/Recherche_exhaustive`

13. Jeu vidéo

Andrea a programmé un jeu vidéo à l'école. Les règles sont simples :
Le jeu se joue en plusieurs tours. Une feuille tombe lors de chaque tour. Le castor essaie d'attraper la feuille avant qu'elle ne touche le sol. Le castor gagne s'il attrape 15 feuilles avant que 4 feuilles ne touchent le sol.
La durée du jeu est égale au nombre de tours (et donc au nombre de feuilles tombées en tout).
Dans l'exemple suivant, le castor perd après 6 tours, car il a atteint le maximum de 4 feuilles touchant le sol. La durée du jeu dans cet exemple est de 6 tours.

Tour	Résultat	Score – nombre total de feuilles	
		Attrapées	Pas attrapées
1	Attrapée	1	0
2	Pas attrapée	1	1
3	Attrapée	2	1
4	Pas attrapée	2	2
5	Pas attrapée	2	3
6	Pas attrapée	2	4

Quelle est la durée maximale d'un jeu ?

A) 4 tours

B) 15 tours

C) 18 tours

D) 19 tours

E) 20 tours

F) La durée du jeu est illimitée.

Solution

Afin de déterminer la durée maximale d'un jeu, nous devons combiner toutes les situations lors desquelles le jeu continue. Pour ceci, nous combinons le nombre maximal de feuilles attrapées avant la fin du jeu (14 tours) avec le nombre maximal de feuilles touchant le sol avant la fin du jeu (3 tours). Au tour suivant, on peut soit attraper une 15^e feuille, soit en laisser tomber une 4^e. La durée maximale est donc $15 + 3 = 14 + 4 = 18$ tours et la bonne réponse est C).
La réponse A) 4 tours est la durée minimale du jeu si aucune feuille n'est attrapée.
La réponse B) 15 tours est la durée minimale du jeu si toutes les feuilles sont attrapées.
Les réponses D), E) et F) sont fausses, car le nombre maximal de feuilles attrapées ou pas attrapées est atteint avant.

C'est de l'informatique !

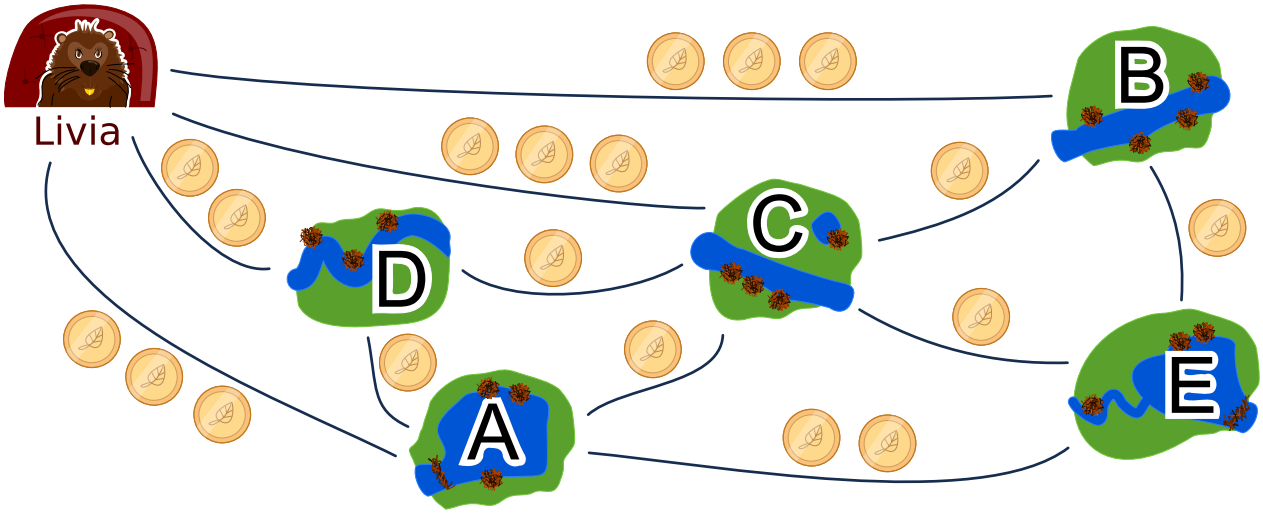
Lors de la programmation d'un jeu, les règles doivent être clairement définies. Les conséquences des règles doivent être bien comprises afin que le jeu permette de gagner et de perdre (le nombre de feuilles doit être suffisant) et que le jeu ne dure ni trop longtemps, ni pas assez.
Un jeu consistant en plusieurs tours est un processus, c'est-à-dire une suite d'opérations ordonnées. Les informaticiens sont des spécialistes de la modélisation et description de processus. Une des tâches principales est de déterminer tout ce qui peut se passer lors du déroulement d'un processus et combien de temps celui-ci peut durer.

Mots clés et sites web

Analyse, vérification et validation de logiciel
- `https://en.wikipedia.org/wiki/Software_verification`
- `https://en.wikipedia.org/wiki/Verification_and_validation`

14. Tournée des castors

Livia aimerait rendre visite à chacun de ses amis dans les villages A, B, C, D et E en transports publics. Elle fait la tournée de tous ses amis lors d'un seul voyage, sans passer deux fois par le même village. Elle rentre chez elle à la fin de sa tournée de visites. Le prix de transport de chaque ligne est affiché ci-dessous.



Une des routes possible pour voir ses amis est :

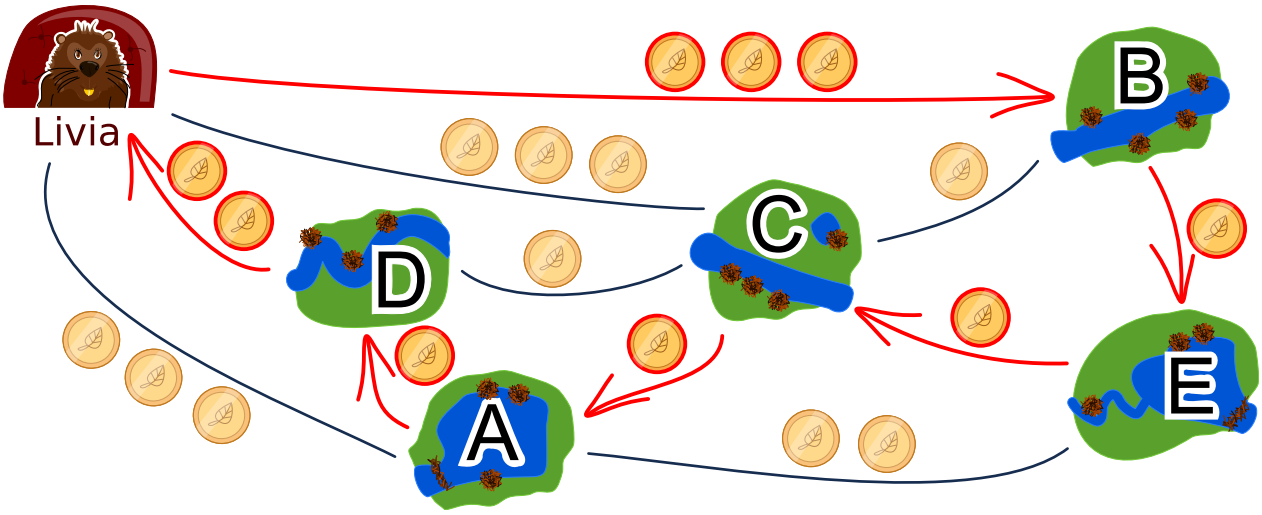
$$\text{départ} \to \text{B} \to \text{E} \to \text{A} \to \text{D} \to \text{C} \to \text{départ}.$$

Cette route coûte $3 + 1 + 2 + 1 + 1 + 3 = 11$ francs castor.
Dans quelle ordre Livia doit-elle rendre visite à ses amis ?

Solution

Il existe deux solutions optimales :

— départ → B → E → C → A → D → départ
— départ → D → A → C → E → B → départ



Les deux solutions sont semblables, mis à part le sens de trajet, et coûtent 9 francs castor. Il n'y a pas de meilleure solution. Depuis chez elle, Livia peut prendre une fois le chemin à 2 francs castor puis un chemin à 3 francs castor dans l'autre sens. Les quatre autres nœuds à visiter correspondent à quatre autres chemins coûtant au moins un franc castor chacun, ce qui fait déjà 9 francs castor au total.

Toutes les autres solutions sont plus chères :

— Coût de 10 francs castor : départ → A → D → C → E → B → départ
— Coût de 10 francs castor : départ → A → E → B → C → D → départ
— Coût de 10 francs castor : départ → B → C → E → A → D → départ
— Coût de 10 francs castor : départ → B → E → A → C → D → départ
— Coût de 10 francs castor : départ → B → E → C → D → A → départ
— Coût de 10 francs castor : départ → C → B → E → A → D → départ
— Coût de 10 francs castor : départ → D → A → E → B → C → départ
— Coût de 10 francs castor : départ → D → A → E → C → B → départ
— Coût de 10 francs castor : départ → D → C → A → E → B → départ
— Coût de 10 francs castor : départ → D → C → B → E → A → départ
— Coût de 11 francs castor : départ → B → E → A → D → C → départ
— Coût de 11 francs castor : départ → C → D → A → E → B → départ

Une des méthodes pour trouver le parcours le moins cher consiste à emprunter le chemin le moins cher, puis à chercher un solution à partir de là.

C'est de l'informatique !

La recherche de bonnes solutions, voire de solutions optimales, est l'un des problèmes fondamentaux de l'informatique. La description de notre problème d'optimisation peut être visualisée dans un diagramme ayant les amis comme nœuds et les chemins comme arêtes. La tâche consiste à passer par chacun des nœuds tout en minimisant la somme des poids des arêtes (le coût en francs castor). C'est un exercice semblable au célèbre problème du voyageur de commerce (Travelling Salesman Problem, TSP).

Ce type de problème est habituellement très difficile à résoudre de manière computationnelle. Afin d'éviter de devoir essayer chaque solution possible, on peut utiliser une bonne heuristique (un exemple d'heuristique est de commencer par le chemin le plus court) puis éliminer toutes les solutions qui sont moins bonnes. Dans ce cas, nous ne permettons qu'un passage par nœud. Si plusieurs passages par nœud étaient permis, le problème deviendrait plus complexe car il faudrait prendre beaucoup plus de possibilités en considération.

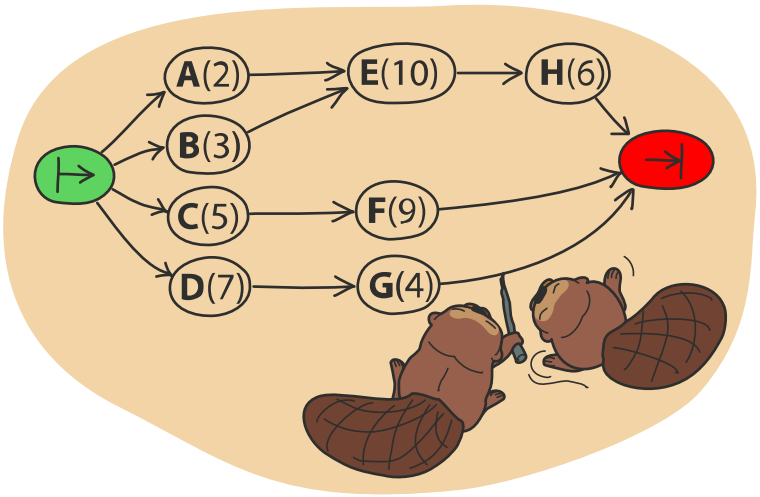
Mots clés et sites web

Optimisation, problème du voyageur de commerce

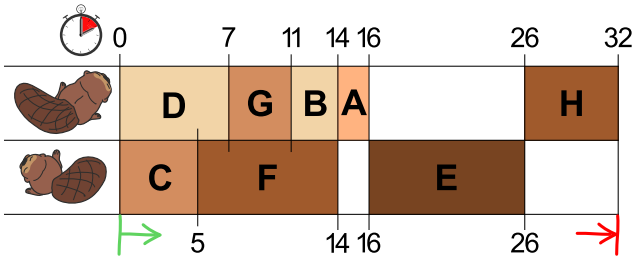
- `https://fr.wikipedia.org/wiki/Problème_du_voyageur_de_commerce`
- `https://en.wikipedia.org/wiki/Optimization_problem`
- `https://fr.wikipedia.org/wiki/Optimisation_(mathématiques)`

15. Deux castors au travail

Deux castors construisent un barrage et doivent pour cela réaliser huit tâches : abattre des arbres, enlever les branches des troncs, amener les troncs dans l'eau, et ainsi de suite. Chaque tâche est définie par une lettre (son nom) et un chiffre entre parenthèses qui donne le nombre d'heures de travail nécessaire à la réalisation de la tâche.
Certaines tâches ne peuvent être commencées que lorsque certaines autres sont terminées. Ce déroulement est représenté par des flèches dans le schéma ci-dessous. Les deux castors peuvent travailler en même temps à différentes tâches, mais ils ne peuvent pas travailler ensemble à la même tâche.



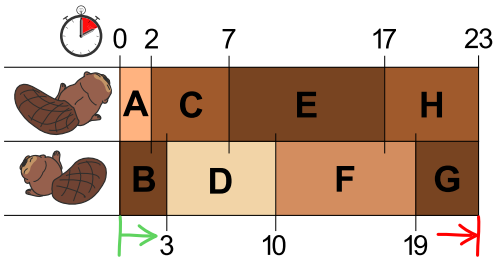
L'image ci-dessous montre un plan de travail possible pour les deux castors qui prévoit 32 heures de travail en tout, mais c'est possible de réaliser le barrage plus rapidement !



De combien de temps les castors ont-ils au minimum besoin pour construire le barrage ?

Solution

Au moins 23 heures sont nécessaires à la construction.
L'image dans la donnée montre un plan de travail possible. Dans celui-ci, le premier castor a une longue pause de 10 heures et le deuxième deux pauses de huit heures en tout. S'ils travaillaient sans pause, ils finiraient plus rapidement.



Si l'on veille à ce que les deux longues tâches E(10) et F(9) ne soient pas réalisées par le même castor, on trouve facilement un plan de travail qui prévoit 23 heures en tout. Ce n'est pas possible de construire le barrage plus rapidement, car les deux castors travaillent déjà sans interruption.

C'est de l'informatique !

Une possibilité pour trouver un des plans de travail les plus courts serait de suivre la règle suivante : « Choisis parmi les tâches à réaliser celle qui nécessite le plus d'heures de travail ». En informatique, on parle alors de stratégie *gloutonne*. On commence par réaliser les tâches qui nous font progresser le plus vers la solution finale du problème.
Les stratégies gloutonnes fonctionnent bien dans beaucoup de cas, mais parfois – comme dans cet exercice – elles ne sont pas adaptées. Cet exercice a été développé de manière à ce que la stratégie gloutonne ne fonctionne pas. Le fait de trouver de tels problèmes peu pratiques à résoudre est une tâche importante : en informatique théorique, par exemple, on analyse de manière ciblée la pire des situations (« worst case ») pour un programme afin de pouvoir mieux estimer le temps d'exécution d'un algorithme.
Il n'existe qu'une manière sûre de trouver la meilleure solution à ce problème, c'est d'essayer tous les plans de travail possibles qui respectent les règles données. Mais lors de grands projets, le nombre de possibilités peut être tellement grand que l'élaboration d'un plan de travail durerait trop longtemps. C'est dans ces cas-là qu'une stratégie gloutonne entre en jeu, car elle permet de trouver relativement rapidement une solution suffisamment bonne, même si ce n'est pas la solution optimale.

Mots clés et sites web

Ordonnancement, algorithme glouton

- `https://fr.wikipedia.org/wiki/Ordonnancement_de_travaux_informatiques`
- `https://fr.wikipedia.org/wiki/Tri_topologique`
- `https://fr.wikipedia.org/wiki/Algorithme_glouton`

A. Auteurs des exercices

Andrea Adamoli
Jared Asuncion
Javier Bilbao
Lucia Budinská
Špela Cerar
Kris Coolsaet
Valentina Dagienė
Darija Dasović Rakijašić
Christian Datzko
Susanne Datzko
Marissa Engels
Hanspeter Erni
Georgios Fessakis
Gerald Futschek
Martin Guggisberg
Bent Halden
Urs Hauser
Wei-fu Hou
Juraj Hromkovič
Takeharu Ishizuka
Svetlana Jakšić
Dong Yoon Kim
Vaidotas Kinčius
Jia-Ling Koh
Regula Lacher
Dan Lessner
Dimitris Mavrovouniotis
Karolína Mayerová
Samart Moodleah
Tom Naughton
Sanja Pavlovic Šijanović
Péter Piltmann
Zsuzsa Pluhár
Wolfgang Pohl
Ilya Posov
Nol Premasathian
J.P. Pretti
Doris Reck
Kirsten Schlüter
Andrea Maria Schmid
Mohamed El-Sherif
Jacqueline Staub
Allira Storey
Peter Tomcsányi
Willem van der Vegt
Jiří Vaníček
Troy Vasiga
Michael Weigend
Magdalena Zarach

B. Sponsoring : Concours 2018

http://www.haslerstiftung.ch/

http://www.roborobo.ch/

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

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

Standortförderung beim Amt für Wirtschaft und Arbeit Kanton Zürich

i-factory (Musée des transports, Lucerne)

http://www.ubs.com/

http://www.bbv.ch/

http://www.presentex.ch/

http://www.zubler.ch/
Zubler & Partner AG Informatik

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

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

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

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

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

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

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

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

C. Offres ultérieures

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