académie Guyane

MINISTÈRE
DE L'ÉDUCATION NATIONALE,
DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

TIC EN MATHÉMATIQUES-ALGORITHMIQUE
Première journée

Christophe AUBRY - Alaeddine BEN RHOUMA - Ghislain ROYER

Inspection Pédagogique Régionale de Mathématiques
Académie de la Guyane

Novembre et Décembre 2015

Plan

Plan

Plan

Plan

Plan

Plan

Plan

1. Algorithme et programmation dans les nouveaux programmes
2. Les algorithmes : étymologie, définitions et histoire
   - Étymologie
   - Des définitions
   - Algorithme de Babylone
   - Algorithme d'Euclide
   - Les algorithmes d'Al-Khawarizmi : résolution d'équations de premier et second degré

Plan

1. Algorithme et programmation dans les nouveaux programmes
2. Les algorithmes : étymologie, définitions et histoire
   - Étymologie
   - Des définitions
   - Algorithme de Babylone
   - Algorithme d'Euclide
   - Les algorithmes d'Al-Khawarizmi : résolution d'équations de premier et second degré
   - Algorithme de Syracuse-Collatz

Plan

Plan

Plan

Plan

Plan

Plan

Plan

Algorithmique et programmation dans les nouveaux programmes

Algorithmique et programmation dans les nouveaux programmes

Ce programme est ancré dans les cinq domaines du socle, et il est structuré selon les quatre thèmes classiques : nombres et calculs ; organisation et gestion de données, fonctions ; grandeurs et mesures ; espace et géométrie.

Algorithmique et programmation dans les nouveaux programmes

Ce programme est ancré dans les cinq domaines du socle, et il est structuré selon les quatre thèmes classiques : nombres et calculs ; organisation et gestion de données, fonctions ; grandeurs et mesures ; espace et géométrie. En outre, **un enseignement de l'informatique** est dispensé conjointement en mathématiques et en technologie.

Algorithmique et programmation dans les nouveaux programmes

Ce programme est ancré dans les cinq domaines du socle, et il est structuré selon les quatre thèmes classiques : nombres et calculs ; organisation et gestion de données, fonctions ; grandeurs et mesures ; espace et géométrie. En outre, **un enseignement de l'informatique** est dispensé conjointement en mathématiques et en technologie.
Ces domaines du socle et ces thèmes du programme **ne sont évidemment pas étanches**.

Algorithmique et programmation dans les nouveaux programmes

Ce programme est ancré dans les cinq domaines du socle, et il est structuré selon les quatre thèmes classiques : nombres et calculs ; organisation et gestion de données, fonctions ; grandeurs et mesures ; espace et géométrie. En outre, **un enseignement de l'informatique** est dispensé conjointement en mathématiques et en technologie.
Ces domaines du socle et ces thèmes du programme **ne sont évidemment pas étanches**.

[...]. Le raisonnement, au coeur de l'activité mathématique, doit prendre appui sur des situations variées (par exemple problèmes de nature arithmétique ou géométrique,

Algorithmique et programmation dans les nouveaux programmes

Ce programme est ancré dans les cinq domaines du socle, et il est structuré selon les quatre thèmes classiques : nombres et calculs ; organisation et gestion de données, fonctions ; grandeurs et mesures ; espace et géométrie. En outre, **un enseignement de l'informatique** est dispensé conjointement en mathématiques et en technologie.
Ces domaines du socle et ces thèmes du programme **ne sont évidemment pas étanches**.

[...]. Le raisonnement, au coeur de l'activité mathématique, doit prendre appui sur des situations variées (par exemple problèmes de nature arithmétique ou géométrique, mais également mise au point d'un programme qui doit tourner sur un ordinateur

Algorithmique et programmation dans les nouveaux programmes

Ce programme est ancré dans les cinq domaines du socle, et il est structuré selon les quatre thèmes classiques : nombres et calculs ; organisation et gestion de données, fonctions ; grandeurs et mesures ; espace et géométrie. En outre, **un enseignement de l'informatique** est dispensé conjointement en mathématiques et en technologie.
Ces domaines du socle et ces thèmes du programme **ne sont évidemment pas étanches**.

[...]. Le raisonnement, au coeur de l'activité mathématique, doit prendre appui sur des situations variées (par exemple problèmes de nature arithmétique ou géométrique, mais également mise au point d'un programme qui doit tourner sur un ordinateur ou pratique de jeux pour lesquels il faut développer une stratégie gagnante, individuelle ou collective, ou maximiser ses chances).

Algorithmique et programmation dans les nouveaux programmes

Ce programme est ancré dans les cinq domaines du socle, et il est structuré selon les quatre thèmes classiques : nombres et calculs ; organisation et gestion de données, fonctions ; grandeurs et mesures ; espace et géométrie. En outre, **un enseignement de l'informatique** est dispensé conjointement en mathématiques et en technologie.
Ces domaines du socle et ces thèmes du programme **ne sont évidemment pas étanches**.

[...]. Le raisonnement, au coeur de l'activité mathématique, doit prendre appui sur des situations variées (par exemple problèmes de nature arithmétique ou géométrique, mais également mise au point d'un programme qui doit tourner sur un ordinateur ou pratique de jeux pour lesquels il faut développer une stratégie gagnante, individuelle ou collective, ou maximiser ses chances).

Projet de programmes pour les cycles 2, 3 et 4 p. 356

Algorithmique et programmation dans les nouveaux programmes

L'enseignement de l'informatique au cycle 4 n'a pas pour objectif de former des élèves experts, mais de leur apporter des clés de décryptage d'un monde numérique en évolution constante.

Algorithmique et programmation dans les nouveaux programmes

L'enseignement de l'informatique au cycle 4 n'a pas pour objectif de former des élèves experts, mais de leur apporter des clés de décryptage d'un monde numérique en évolution constante. Il permet d'acquérir des méthodes qui construisent **la pensée algorithmique**

Algorithmique et programmation dans les nouveaux programmes

L'enseignement de l'informatique au cycle 4 n'a pas pour objectif de former des élèves experts, mais de leur apporter des clés de décryptage d'un monde numérique en évolution constante. Il permet d'acquérir des méthodes qui construisent **la pensée algorithmique** et développe des compétences dans **la représentation de l'information et de son traitement**,

Algorithmique et programmation dans les nouveaux programmes

L'enseignement de l'informatique au cycle 4 n'a pas pour objectif de former des élèves experts, mais de leur apporter des clés de décryptage d'un monde numérique en évolution constante. Il permet d'acquérir des méthodes qui construisent **la pensée algorithmique** et développe des compétences dans **la représentation de l'information et de son traitement**, la résolution de problèmes, le contrôle des résultats.

Algorithmique et programmation dans les nouveaux programmes

L'enseignement de l'informatique au cycle 4 n'a pas pour objectif de former des élèves experts, mais de leur apporter des clés de décryptage d'un monde numérique en évolution constante. Il permet d'acquérir des méthodes qui construisent **la pensée algorithmique** et développe des compétences dans **la représentation de l'information et de son traitement**, la résolution de problèmes, le contrôle des résultats.
Pour donner du sens aux apprentissages et valoriser le travail des élèves, cet enseignement doit se traduire par la réalisation de productions collectives (programme, application, animation, sites, etc.) dans le cadre d'activités de création numérique, au cours desquelles les élèves développent leur autonomie, mais aussi le sens du travail collaboratif.

Algorithmique et programmation dans les nouveaux programmes

L'enseignement de l'informatique au cycle 4 n'a pas pour objectif de former des élèves experts, mais de leur apporter des clés de décryptage d'un monde numérique en évolution constante. Il permet d'acquérir des méthodes qui construisent **la pensée algorithmique** et développe des compétences dans **la représentation de l'information et de son traitement**, la résolution de problèmes, le contrôle des résultats.
Pour donner du sens aux apprentissages et valoriser le travail des élèves, cet enseignement doit se traduire par la réalisation de productions collectives (programme, application, animation, sites, etc.) dans le cadre d'activités de création numérique, au cours desquelles les élèves développent leur autonomie, mais aussi le sens du travail collaboratif.

Projet de programmes pour les cycles 2, 3 et 4 p. 357

Algorithmique et programmation dans les nouveaux programmes

Au cycle 4, les élèves **s'initient à la programmation**,

Algorithmique et programmation dans les nouveaux programmes

Au cycle 4, les élèves **s'initient à la programmation**, en développant dans une démarche de projet quelques programmes simples,

Algorithmique et programmation dans les nouveaux programmes

Au cycle 4, les élèves **s'initient à la programmation**, en développant dans une démarche de projet quelques programmes simples, sans viser une connaissance experte et exhaustive d'un langage ou d'un logiciel particulier.

Algorithmique et programmation dans les nouveaux programmes

Au cycle 4, les élèves **s'initient à la programmation**, en développant dans une démarche de projet quelques programmes simples, sans viser une connaissance experte et exhaustive d'un langage ou d'un logiciel particulier. En créant un programme, ils développent des méthodes de programmation,

Algorithmique et programmation dans les nouveaux programmes

Au cycle 4, les élèves **s'initient à la programmation**, en développant dans une démarche de projet quelques programmes simples, sans viser une connaissance experte et exhaustive d'un langage ou d'un logiciel particulier. En créant un programme, ils développent des méthodes de programmation, revisitent les notions de variables et de fonctions sous une forme différente,

Algorithmique et programmation dans les nouveaux programmes

Au cycle 4, les élèves **s'initient à la programmation**, en développant dans une démarche de projet quelques programmes simples, sans viser une connaissance experte et exhaustive d'un langage ou d'un logiciel particulier. En créant un programme, ils développent des méthodes de programmation, revisitent les notions de variables et de fonctions sous une forme différente, et s'entraînent au raisonnement.

Algorithmique et programmation dans les nouveaux programmes

Au cycle 4, les élèves **s'initient à la programmation**, en développant dans une démarche de projet quelques programmes simples, sans viser une connaissance experte et exhaustive d'un langage ou d'un logiciel particulier. En créant un programme, ils développent des méthodes de programmation, revisitent les notions de variables et de fonctions sous une forme différente, et s'entraînent au raisonnement.

Projet de programmes pour les cycles 2, 3 et 4 p. 368

Attendus de fin de cycle

Attendus de fin de cycle

Écrire,

Attendus de fin de cycle

Écrire, mettre au point

Attendus de fin de cycle

Écrire, mettre au point et exécuter un programme simple.

Projet de programmes pour les cycles 2, 3 et 4 p. 368

Attendus de fin de cycle

Connaissances et compétences associées	Exemples de situations, d'activités et de ressources pour l'élève
Décomposer un problème en sous-problèmes afin de structurer un programme ; reconnaître des schémas. Écrire, mettre au point (tester, corriger) et exécuter un programme en réponse à un problème donné. Écrire un programme dans lequel des actions sont déclenchées par des événements extérieurs. Programmer des scripts se déroulant en parallèle. » Notions d'algorithme et de programme. » Notion de variable informatique. » Déclenchement d'une action par un évènement, séquences d'instructions, boucles, instructions conditionnelles.	Jeux dans un labyrinthe, jeu de Pong, bataille navale, jeu de nim, tic tac toe. Réalisation de figure à l'aide d'un logiciel de programmation pour consolider les notions de longueur et d'angle. Initiation au chiffrement (Morse, chiffre de César, code ASCII…). Construction de tables de conjugaison, de pluriels, jeu du cadavre exquis… Calculs simples de calendrier. Calculs de répertoire (recherche, recherche inversée...). Calculs de fréquences d'apparition de chaque lettre dans un texte pour distinguer sa langue d'origine : français, anglais, italien, etc.

Repères de progressivité

En 5ème, les élèves s'initient à la programmation évènementielle.

Repères de progressivité

En 5ème, les élèves s'initient à la programmation évènementielle. Progressivement, ils développent de nouvelles compétences, en programmant des actions en parallèle,

Repères de progressivité

En 5ème, les élèves s'initient à la programmation évènementielle. Progressivement, ils développent de nouvelles compétences, en programmant des actions en parallèle, en utilisant la notion de variable informatique,

Repères de progressivité

En 5ème, les élèves s'initient à la programmation évènementielle. Progressivement, ils développent de nouvelles compétences, en programmant des actions en parallèle, en utilisant la notion de variable informatique, en découvrant les boucles et les instructions conditionnelles qui complètent les structures de contrôle liées aux événements.

Repères de progressivité

En 5ème, les élèves s'initient à la programmation évènementielle. Progressivement, ils développent de nouvelles compétences, en programmant des actions en parallèle, en utilisant la notion de variable informatique, en découvrant les boucles et les instructions conditionnelles qui complètent les structures de contrôle liées aux événements. En 3ème, ils abordent la gestion des objets, en leur faisant échanger des messages.

Les algorithmes racontés par les Sépas

Les algorithmes racontés par les Sépas

Question : comment les Sépas définissent-ils un algorithme ?

Étymologie

Étymologie

Le mot algorithme vient du nom latinisé du mathématicien perse Al-Khawarizmi (780-850), écrivant en langue arabe, surnommé « le père de l'algèbre ».

Des définitions

Des définitions

Il n'existe pas de définition universellement admise du mot « algorithme ».

Des définitions

Il n'existe pas de définition universellement admise du mot « algorithme ».

Définition selon Wikipédia

Des définitions

Il n'existe pas de définition universellement admise du mot « algorithme ».

Définition selon Wikipédia

Un algorithme est une suite **finie** et **non ambiguë** d'opérations ou d'**instructions** permettant de résoudre un problème ou d'obtenir un résultat donné.

Des définitions

Définition selon Donald Ervin Knuth

Des définitions

Définition selon Donald Ervin Knuth

Donald Ervin Knuth est un informaticien et mathématicien américain de renom et professeur émérite en informatique à l'université Stanford. Il est un des pionniers de l'algorithmique et a fait de nombreuses contributions dans plusieurs branches de l'informatique théorique.

Des définitions

Définition selon Donald Ervin Knuth

Donald Ervin Knuth est un informaticien et mathématicien américain de renom et professeur émérite en informatique à l'université Stanford. Il est un des pionniers de l'algorithmique et a fait de nombreuses contributions dans plusieurs branches de l'informatique théorique.

Des définitions

Définition selon Donald Ervin Knuth

Donald Ervin Knuth est un informaticien et mathématicien américain de renom et professeur émérite en informatique à l'université Stanford. Il est un des pionniers de l'algorithmique et a fait de nombreuses contributions dans plusieurs branches de l'informatique théorique.

Donald Knuth lista les cinq propriétés suivantes comme étant les prérequis d'un algorithme :

Des définitions

1. **la finitude :** Un algorithme doit toujours se terminer après un nombre fini d'étapes.

Des définitions

1. **la finitude :** Un algorithme doit toujours se terminer après un nombre fini d'étapes.

2. **définition précise :** Chaque étape d'un algorithme doit être définie précisément, les actions à transposer doivent être spécifiées rigoureusement et sans ambiguïté pour chaque cas.

Des définitions

1. **la finitude :** Un algorithme doit toujours se terminer après un nombre fini d'étapes.

2. **définition précise :** Chaque étape d'un algorithme doit être définie précisément, les actions à transposer doivent être spécifiées rigoureusement et sans ambiguïté pour chaque cas.

3. **entrées :** . . . des quantités qui lui sont données avant qu'un algorithme ne commence. Ces entrées sont prises dans un ensemble d'objets spécifié.

Des définitions

1. **la finitude :** Un algorithme doit toujours se terminer après un nombre fini d'étapes.

2. **définition précise :** Chaque étape d'un algorithme doit être définie précisément, les actions à transposer doivent être spécifiées rigoureusement et sans ambiguïté pour chaque cas.

3. **entrées :** . . . des quantités qui lui sont données avant qu'un algorithme ne commence. Ces entrées sont prises dans un ensemble d'objets spécifié.

4. **sorties :** . . . des quantités ayant une relation spécifiées avec les entrées.

Des définitions

1. **la finitude :** Un algorithme doit toujours se terminer après un nombre fini d'étapes.

2. **définition précise :** Chaque étape d'un algorithme doit être définie précisément, les actions à transposer doivent être spécifiées rigoureusement et sans ambiguïté pour chaque cas.

3. **entrées :** . . . des quantités qui lui sont données avant qu'un algorithme ne commence. Ces entrées sont prises dans un ensemble d'objets spécifié.

4. **sorties :** . . . des quantités ayant une relation spécifiées avec les entrées.

5. **rendement :** . . . toutes les opérations que l'algorithme doit accomplir doivent être suffisamment basiques pour pouvoir être en principe réalisées dans une durée finie par un homme utilisant un papier et un crayon.

Des définitions

Définition selon Gérard Berry

Des définitions

Définition selon Gérard Berry

Gérard Berry est un informaticien français, professeur au Collège de France, membre de l'Académie des sciences française (depuis 2002), de l'Académie des technologies (depuis 2005), et de l'Academia Europaea (depuis 1993).

Des définitions

Définition selon Gérard Berry

Gérard Berry est un informaticien français, professeur au Collège de France, membre de l'Académie des sciences française (depuis 2002), de l'Académie des technologies (depuis 2005), et de l'Academia Europaea (depuis 1993).

Des définitions

Définition selon Gérard Berry

Gérard Berry est un informaticien français, professeur au Collège de France, membre de l'Académie des sciences française (depuis 2002), de l'Académie des technologies (depuis 2005), et de l'Academia Europaea (depuis 1993).

Gérard Berry donne la définition grand public de l'algorithme suivante :

Des définitions

Définition selon Gérard Berry

« Un **algorithme**, c'est tout simplement une façon de décrire dans ses moindres détails comment procéder pour faire quelque chose. Il se trouve que beaucoup d'actions mécaniques (. . .) se prêtent bien à une telle décortication.

Des définitions

Définition selon Gérard Berry

« Un **algorithme**, c'est tout simplement une façon de décrire dans ses moindres détails comment procéder pour faire quelque chose. Il se trouve que beaucoup d'actions mécaniques (...) se prêtent bien à une telle décortication. Le but est d'évacuer la pensée du calcul, afin de le rendre exécutable par une machine numérique (ordinateur, ...).

Des définitions

Définition selon Gérard Berry

« Un **algorithme**, c'est tout simplement une façon de décrire dans ses moindres détails comment procéder pour faire quelque chose. Il se trouve que beaucoup d'actions mécaniques (...) se prêtent bien à une telle décortication. Le but est d'évacuer la pensée du calcul, afin de le rendre exécutable par une machine numérique (ordinateur, ...). On ne travaille donc qu'avec un reflet numérique du système réel avec qui l'algorithme interagit. »

Algorithme dans la vie quotidienne

Algorithme dans la vie quotidienne

Une recette de cuisine peut être réduite à un algorithme, si on peut réduire sa spécification aux éléments constitutifs :

Algorithme dans la vie quotidienne

Une recette de cuisine peut être réduite à un algorithme, si on peut réduire sa spécification aux éléments constitutifs :

- des entrées (les ingrédients, le matériel utilisé) ;

Algorithme dans la vie quotidienne

Une recette de cuisine peut être réduite à un algorithme, si on peut réduire sa spécification aux éléments constitutifs :

- des entrées (les ingrédients, le matériel utilisé) ;
- des instructions élémentaires simples, dont l'exécution amène au résultat voulu ;

Algorithme dans la vie quotidienne

Une recette de cuisine peut être réduite à un algorithme, si on peut réduire sa spécification aux éléments constitutifs :

- des entrées (les ingrédients, le matériel utilisé) ;
- des instructions élémentaires simples, dont l'exécution amène au résultat voulu ;
- un résultat : le plat préparé.

Algorithme dans la vie quotidienne

Une recette de cuisine peut être réduite à un algorithme, si on peut réduire sa spécification aux éléments constitutifs :

- des entrées (les ingrédients, le matériel utilisé) ;
- des instructions élémentaires simples, dont l'exécution amène au résultat voulu ;
- un résultat : le plat préparé.

Cependant, les recettes de cuisine ne sont en général pas présentées rigoureusement sous forme non ambiguë : il est d'usage d'y employer des termes vagues laissant une liberté d'appréciation à l'exécutant alors qu'un algorithme stricto sensu doit être précis et sans ambiguïté.

Algorithme de Babylone

Algorithme de Babylone

Si A est un entier naturel positif, $\sqrt{A}$ est le plus souvent un nombre irrationnel.

Algorithme de Babylone

Si A est un entier naturel positif, $\sqrt{A}$ est le plus souvent un nombre irrationnel. Se pose alors un problème de calcul :

Algorithme de Babylone

Si A est un entier naturel positif, $\sqrt{A}$ est le plus souvent un nombre irrationnel. Se pose alors un problème de calcul : comment trouver une valeur approchée de $\sqrt{A}$ qui soit aussi proche qu'on veut de ce nombre ?

Algorithme de Babylone

Si A est un entier naturel positif, $\sqrt{A}$ est le plus souvent un nombre irrationnel. Se pose alors un problème de calcul : comment trouver une valeur approchée de $\sqrt{A}$ qui soit aussi proche qu'on veut de ce nombre ?

Cette question a été résolue il y a longtemps grâce à l'algorithme de Babylone, appelé aussi algorithme de Héron.

L'algorithme de Babylone : calcul de racines carrées

Il y a 4000 ans, les mathématiciens babyloniens utilisaient un système de numération sexagésimal.

L'algorithme de Babylone : calcul de racines carrées

Il y a 4000 ans, les mathématiciens babyloniens utilisaient un système de numération sexagésimal. Après transcription en base 10, on a découvert que la tablette YBC 7289 portait la valeur $\sqrt{2} = 1,414222$.

L'algorithme de Babylone : calcul de racines carrées

Il y a 4000 ans, les mathématiciens babyloniens utilisaient un système de numération sexagésimal. Après transcription en base 10, on a découvert que la tablette YBC 7289 portait la valeur $\sqrt{2} = 1,414222$.

L'algorithme de Babylone : calcul de racines carrées

Il y a 4000 ans, les mathématiciens babyloniens utilisaient un système de numération sexagésimal. Après transcription en base 10, on a découvert que la tablette YBC 7289 portait la valeur $\sqrt{2} = 1,414222$.

Beaucoup plus tard, au premier siècle de notre ère, la méthode de calcul des Babyloniens a été reprise par Héron d'Alexandrie dans son principal ouvrage *Les Métriques*.

L'algorithme de Babylone : calcul de racines carrées

Il y a 4000 ans, les mathématiciens babyloniens utilisaient un système de numération sexagésimal. Après transcription en base 10, on a découvert que la tablette YBC 7289 portait la valeur $\sqrt{2} = 1,414222$.

Beaucoup plus tard, au premier siècle de notre ère, la méthode de calcul des Babyloniens a été reprise par Héron d'Alexandrie dans son principal ouvrage *Les Métriques*. Avec les notations qui sont les nôtres, on peut décrire de manière simple l'algorithme utilisé par aussi bien les Babyloniens que par Héron pour calculer le nombre $\sqrt{A}$:

L'algorithme de Babylone : calcul de racines carrées

- Choisir une première estimation e de $\sqrt{A}$;

L'algorithme de Babylone : calcul de racines carrées

- Choisir une première estimation e de $\sqrt{A}$;

- Calculer une deuxième estimation e' en faisant la moyenne arithmétique de e et de $\frac{A}{e}$;

L'algorithme de Babylone : calcul de racines carrées

- Choisir une première estimation e de $\sqrt{A}$;

- Calculer une deuxième estimation e' en faisant la moyenne arithmétique de e et de $\frac{A}{e}$;

- Remplacer e par e' et recommencer pour avoir une troisième estimation e'' ;

L'algorithme de Babylone : calcul de racines carrées

- Choisir une première estimation e de $\sqrt{A}$;

- Calculer une deuxième estimation e' en faisant la moyenne arithmétique de e et de $\frac{A}{e}$;

- Remplacer e par e' et recommencer pour avoir une troisième estimation e'' ;

- Continuer ainsi autant de fois que nécessaire pour obtenir la précision souhaitée.

L'algorithme de Babylone : calcul de racines carrées

- Choisir une première estimation e de $\sqrt{A}$;

- Calculer une deuxième estimation e' en faisant la moyenne arithmétique de e et de $\frac{A}{e}$;

- Remplacer e par e' et recommencer pour avoir une troisième estimation e'' ;

- Continuer ainsi autant de fois que nécessaire pour obtenir la précision souhaitée.

Algorithme d'Héron avec tableur

Algorithme d'Euclide

Algorithme d'Euclide

Au III^{e} siècle avant J.C, en écrivant ses *Éléments*, Euclide a voulu exposer la totalité des mathématiques de son temps.

Algorithme d'Euclide

Au III^{e} siècle avant J.C, en écrivant ses *Éléments*, Euclide a voulu exposer la totalité des mathématiques de son temps. Bien qu'il ne l'appelle pas ainsi, il traite du PGCD dans le livre VII.

Algorithme d'Euclide

Au III^{e} siècle avant J.C, en écrivant ses *Éléments*, Euclide a voulu exposer la totalité des mathématiques de son temps. Bien qu'il ne l'appelle pas ainsi, il traite du PGCD dans le livre VII. Il commence par définir deux entiers qui sont premiers entre eux : ce sont des entiers dont *la plus grande mesure vaut 1*.

Algorithme d'Euclide

Au III^{e} siècle avant J.C, en écrivant ses *Éléments*, Euclide a voulu exposer la totalité des mathématiques de son temps. Bien qu'il ne l'appelle pas ainsi, il traite du PGCD dans le livre VII. Il commence par définir deux entiers qui sont premiers entre eux : ce sont des entiers dont *la plus grande mesure vaut 1*. Dans une proposition, il expose la façon de rechercher cette commune mesure :

« Deux nombres inégaux étant proposés, le plus petit étant toujours retranché du plus grand, si le reste ne mesure celui qui est avant lui que lorsqu'on a pris l'unité, les nombres proposés seront premiers entre eux. »

Algorithme d'Euclide

Au III^{e} siècle avant J.C, en écrivant ses *Éléments*, Euclide a voulu exposer la totalité des mathématiques de son temps. Bien qu'il ne l'appelle pas ainsi, il traite du PGCD dans le livre VII. Il commence par définir deux entiers qui sont premiers entre eux : ce sont des entiers dont *la plus grande mesure vaut 1*. Dans une proposition, il expose la façon de rechercher cette commune mesure :

« Deux nombres inégaux étant proposés, le plus petit étant toujours retranché du plus grand, si le reste ne mesure celui qui est avant lui que lorsqu'on a pris l'unité, les nombres proposés seront premiers entre eux. »

Il affirme ensuite que si les deux entiers ne sont pas premiers entre eux, ils ont alors une commune mesure (un PGCD) supérieure à 1.

Algorithme d'Euclide : organigramme

Organigramme de l'algorithme d'Euclide

dixit Algorizmi

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré.

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré. Il distingue alors six cas et les traite sur des exemples qui se généralise sans difficulté sur toute équation de même type.

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré. Il distingue alors six cas et les traite sur des exemples qui se généralise sans difficulté sur toute équation de même type. Il considère ainsi :

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré. Il distingue alors six cas et les traite sur des exemples qui se généralise sans difficulté sur toute équation de même type. Il considère ainsi :

- Carrés égaux aux racines : $ax^2 = bx$;

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré. Il distingue alors six cas et les traite sur des exemples qui se généralise sans difficulté sur toute équation de même type. Il considère ainsi :

- Carrés égaux aux racines : $ax^2 = bx$;
- Carrés égaux à un nombre : $ax^2 = c$;

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré. Il distingue alors six cas et les traite sur des exemples qui se généralise sans difficulté sur toute équation de même type. Il considère ainsi :

- Carrés égaux aux racines : $ax^2 = bx$;
- Carrés égaux à un nombre : $ax^2 = c$;
- Racines égales à un nombre : $bx = c$;

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré. Il distingue alors six cas et les traite sur des exemples qui se généralise sans difficulté sur toute équation de même type. Il considère ainsi :

- Carrés égaux aux racines : $ax^2 = bx$;
- Carrés égaux à un nombre : $ax^2 = c$;
- Racines égales à un nombre : $bx = c$;
- Carrés et racines égaux à un nombre : $ax^2 + bx = c$;

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré. Il distingue alors six cas et les traite sur des exemples qui se généralise sans difficulté sur toute équation de même type. Il considère ainsi :

- Carrés égaux aux racines : $ax^2 = bx$;
- Carrés égaux à un nombre : $ax^2 = c$;
- Racines égales à un nombre : $bx = c$;
- Carrés et racines égaux à un nombre : $ax^2 + bx = c$;
- Carrés et nombres égaux égaux aux racines : $ax^2 + c = bx$;

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré. Il distingue alors six cas et les traite sur des exemples qui se généralise sans difficulté sur toute équation de même type. Il considère ainsi :

- Carrés égaux aux racines : $ax^2 = bx$;
- Carrés égaux à un nombre : $ax^2 = c$;
- Racines égales à un nombre : $bx = c$;
- Carrés et racines égaux à un nombre : $ax^2 + bx = c$;
- Carrés et nombres égaux égaux aux racines : $ax^2 + c = bx$;
- Racines et nombres égaux aux carrés : $bx + c = ax^2$;

dixit Algorizmi

Contrairement aux mathématiciens grecs, Al-Khawarizmi détaille des méthodes effectives de résolutions d'équations. Il les traite avec des exemples tirés d'expériences pratiques.
Al-Khawarizmi nous présente une exposition complète de la résolution des équations du premier et second degré. Il distingue alors six cas et les traite sur des exemples qui se généralise sans difficulté sur toute équation de même type. Il considère ainsi :

- Carrés égaux aux racines : $ax^2 = bx$;
- Carrés égaux à un nombre : $ax^2 = c$;
- Racines égales à un nombre : $bx = c$;
- Carrés et racines égaux à un nombre : $ax^2 + bx = c$;
- Carrés et nombres égaux égaux aux racines : $ax^2 + c = bx$;
- Racines et nombres égaux aux carrés : $bx + c = ax^2$;

où a, b et c désignent des nombres positifs.

Résoudre $x^2 + 10x = 39$ avec Al-Khawarizmi

Résoudre $x^2 + 10x = 39$ avec Al-Khawarizmi

Les carrés plus les racines égaux à un nombre, c'est par exemple lorsque tu dis : un carré plus dix racines sont égaux à trente-neuf dirhams, c'est-à-dire que si on ajoute à un carré quelconque « une quantité » égale à dix racines, le tout sera trente-neuf.

texte traduit par Rosdi RACHED. Al Khawarizmi, le commencement de l'algèbre

Résoudre $x^2 + 10x = 39$ avec Al-Khawarizmi

Les carrés plus les racines égaux à un nombre, c'est par exemple lorsque tu dis : un carré plus dix racines sont égaux à trente-neuf dirhams, c'est-à-dire que si on ajoute à un carré quelconque « une quantité » égale à dix racines, le tout sera trente-neuf.

texte traduit par Rosdi RACHED. Al Khawarizmi, le commencement de l'algèbre

illustration

La conjecture de Syracuse (Problème de Collatz)

Présentation du problème

La conjecture de Syracuse (Problème de Collatz)

Présentation du problème

Le problème de la conjecture de Syracuse, également connue sous les noms de problème de Collatz, Kakutani, Ulam, ou $3x + 1$ se présente de manière très simple.

Présentation du problème

Le problème de la conjecture de Syracuse, également connue sous les noms de problème de Collatz, Kakutani, Ulam, ou $3x + 1$ se présente de manière très simple.

On se donne une entier naturel n plus grand que 1.

Présentation du problème

Le problème de la conjecture de Syracuse, également connue sous les noms de problème de Collatz, Kakutani, Ulam, ou $3x + 1$ se présente de manière très simple.

On se donne une entier naturel n plus grand que 1. S'il est pair, on le divise par deux,

Présentation du problème

Le problème de la conjecture de Syracuse, également connue sous les noms de problème de Collatz, Kakutani, Ulam, ou $3x + 1$ se présente de manière très simple.

On se donne une entier naturel n plus grand que 1. S'il est pair, on le divise par deux, s'il est impair, on le multiplie par 3 et on lui ajoute 1.

La conjecture de Syracuse (Problème de Collatz)

Présentation du problème

Le problème de la conjecture de Syracuse, également connue sous les noms de problème de Collatz, Kakutani, Ulam, ou $3x + 1$ se présente de manière très simple.

On se donne une entier naturel n plus grand que 1. S'il est pair, on le divise par deux, s'il est impair, on le multiplie par 3 et on lui ajoute 1.

Organigramme de l'algorithme

La conjecture de Syracuse (Problème de Collatz)

Présentation du problème

Le problème de la conjecture de Syracuse, également connue sous les noms de problème de Collatz, Kakutani, Ulam, ou $3x + 1$ se présente de manière très simple.

On se donne une entier naturel n plus grand que 1. S'il est pair, on le divise par deux, s'il est impair, on le multiplie par 3 et on lui ajoute 1.

Organigramme de l'algorithme

Algorithme réalisé avec un tableur

Conjecture de Syracuse-Collatz

On conjecture que l'on finit toujours par trouver la valeur 1 au fil des calculs,

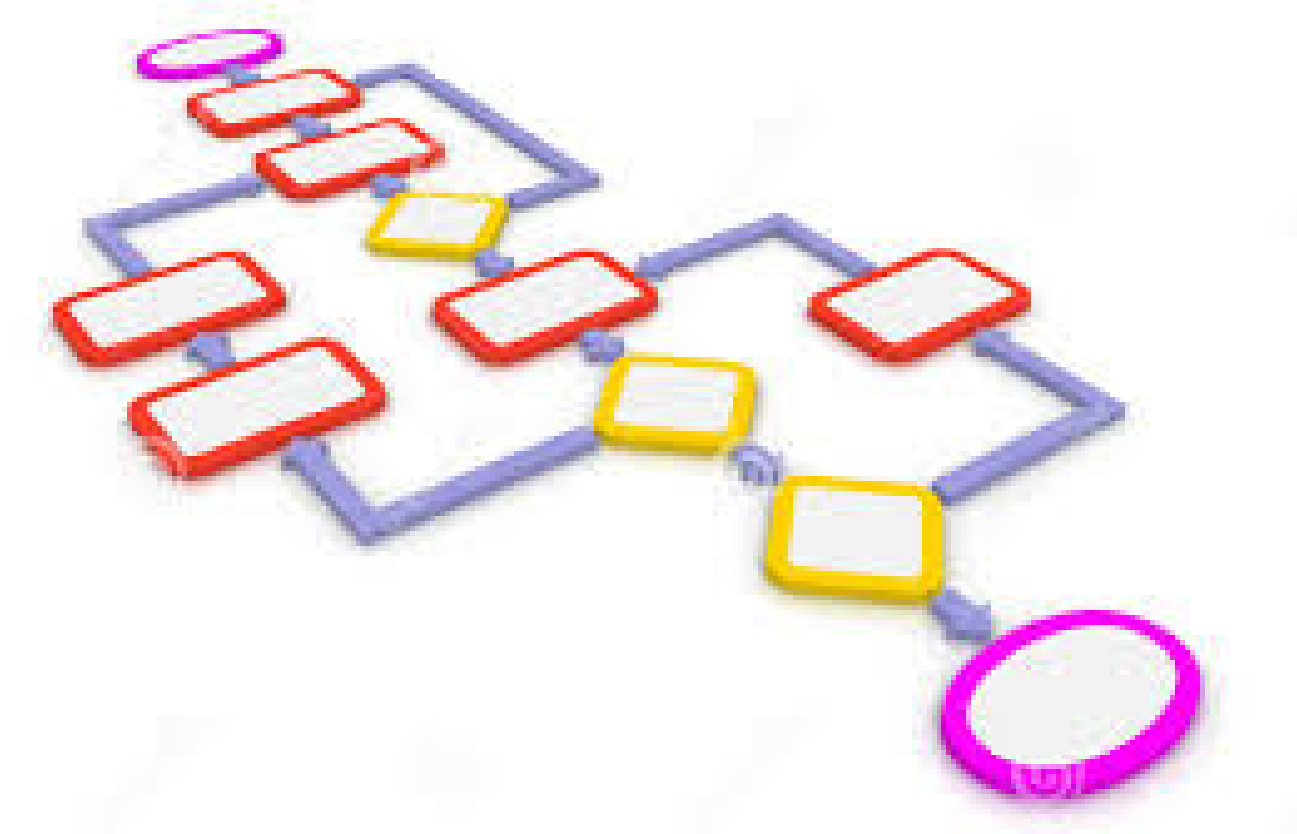
Conjecture de Syracuse-Collatz

On conjecture que l'on finit toujours par trouver la valeur 1 au fil des calculs, valeur à partir de laquelle on restera bloqué dans le cycle 1-4-2-1-. . .

Algorithme et programmation

Algorithme ou programme ?

Algorithme ou programme ?



Algorithme / Programme : Quelle différence ?

Algorithme ou programme ?

Algorithme ou programme ?

L'algorithme est exprimé dans un modèle théorique de machine virtuelle (Von Neumann) qui **ne dépend pas de la machine réelle** sur laquelle on va l'utiliser.

Algorithme ou programme ?

L'algorithme est exprimé dans un modèle théorique de machine virtuelle (Von Neumann) qui **ne dépend pas de la machine réelle** sur laquelle on va l'utiliser.

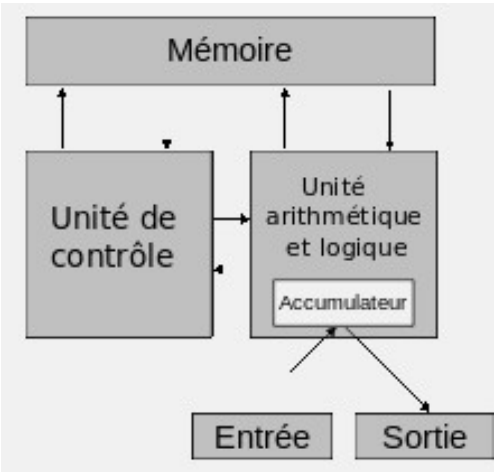
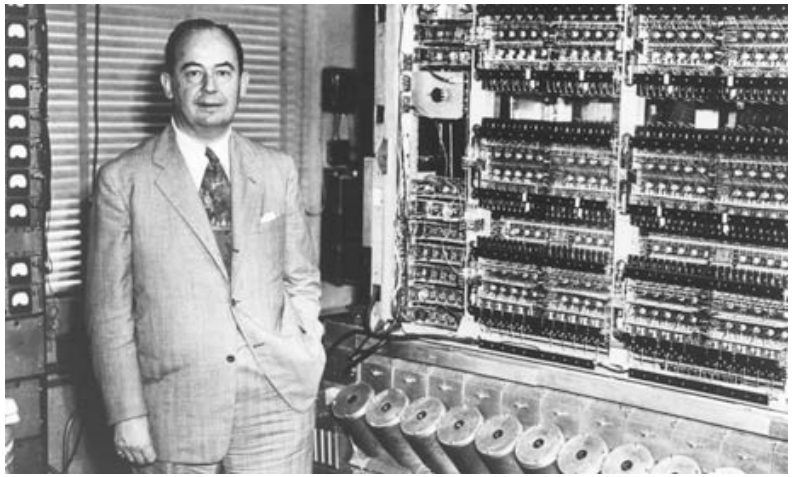


Schéma de l'architecture de von Neumann.



John von Neumann dans les années 1940.

Algorithme ou programme ?

Il peut être **écrit en langage naturel**, mais pour être lisible par tous, on utilise un langage algorithmique plus restreint qui comporte tous les concepts de base de fonctionnnement d'une machine.

Algorithme ou programme ?

Il peut être **écrit en langage naturel**, mais pour être lisible par tous, on utilise un langage algorithmique plus restreint qui comporte tous les concepts de base de fonctionnnement d'une machine.

Le **langage de programmation** est l'intermédiaire entre l'humain et la machine. Les langages de programmation servent avant tout à transcrire les algorithmes sous une forme qui puisse être traitée par l'ordinateur tout en restant intelligible par l'humain.

Algorithme ou programme ?

Il peut être **écrit en langage naturel**, mais pour être lisible par tous, on utilise un langage algorithmique plus restreint qui comporte tous les concepts de base de fonctionnnement d'une machine.

Le **langage de programmation** est l'intermédiaire entre l'humain et la machine. Les langages de programmation servent avant tout à transcrire les algorithmes sous une forme qui puisse être traitée par l'ordinateur tout en restant intelligible par l'humain.

Étant donné que le langage de programmation est destiné à l'ordinateur, il doit donc respecter une syntaxe stricte. Un algorithme peut toutefois aboutir à plusieurs programmes.

Algorithme ou programme ?

Un programme est donc un algorithme transcrit dans un langage de programmation particulier.

Algorithme ou programme ?

Un programme est donc un algorithme transcrit dans un langage de programmation particulier.

Une autre différence entre algorithme et programme est que l'exécution d'un algorithme doit toujours se terminer avec un résultat, alors que celle d'un programme peut conduire à une boucle infinie (ne jamais s'arrêter).

Quelques algorithmes

Variables

Variables

Dans un programme, les données sont manipulées via des variables :

Variables

Dans un programme, les données sont manipulées via des variables :

- une variable est une case mémoire ;

Variables

Dans un programme, les données sont manipulées via des variables :

- une variable est une case mémoire ;
- une variable est désignée par un nom (identifiant) ;

Variables

Dans un programme, les données sont manipulées via des variables :

- une variable est une case mémoire ;
- une variable est désignée par un nom (identifiant) ;
- une variable a un type de donnée (implicite dans certains langages) ;

Variables

Dans un programme, les données sont manipulées via des variables :

- une variable est une case mémoire ;
- une variable est désignée par un nom (identifiant) ;
- une variable a un type de donnée (implicite dans certains langages) ;
- une variable contient une valeur du type et cette valeur peut varier.

Variables

Dans un programme, les données sont manipulées via des variables :

- une variable est une case mémoire ;
- une variable est désignée par un nom (identifiant) ;
- une variable a un type de donnée (implicite dans certains langages) ;
- une variable contient une valeur du type et cette valeur peut varier.

Cycle de vie d'une variable :

Variables

Dans un programme, les données sont manipulées via des variables :

- une variable est une case mémoire ;
- une variable est désignée par un nom (identifiant) ;
- une variable a un type de donnée (implicite dans certains langages) ;
- une variable contient une valeur du type et cette valeur peut varier.

Cycle de vie d'une variable :

- déclaration de la variable (nom et type) ;

Variables

Dans un programme, les données sont manipulées via des variables :

- une variable est une case mémoire ;
- une variable est désignée par un nom (identifiant) ;
- une variable a un type de donnée (implicite dans certains langages) ;
- une variable contient une valeur du type et cette valeur peut varier.

Cycle de vie d'une variable :

- déclaration de la variable (nom et type) ;
- affectations de valeurs à la variable ;

Variables

Dans un programme, les données sont manipulées via des variables :

- une variable est une case mémoire ;
- une variable est désignée par un nom (identifiant) ;
- une variable a un type de donnée (implicite dans certains langages) ;
- une variable contient une valeur du type et cette valeur peut varier.

Cycle de vie d'une variable :

- déclaration de la variable (nom et type) ;
- affectations de valeurs à la variable ;
- suppression de la variable (souvent automatique) ;

Variables

Que valent les variables a et b à la fin de l'algorithme ?

```
DEBUT
   a ← 0
   b ← 1
   a ← a + b
   b ← a * b
   a ← a - 2
FIN
```

Entrées et sorties

Entrées et sorties

Quelles sont les entrées et les sorties de l'algorithme suivant ?

```
DEBUT
   AFFICHER "Entrez deux nombres"
   DEMANDER X1, X2
   AFFICHER "La somme de ",X1," et ",X2," est ",X1 + X2
FIN
```

Entrées et sorties

Quelles sont les entrées et les sorties de l'algorithme suivant ?

```
DEBUT
   AFFICHER "Entrez deux nombres"
   DEMANDER X1, X2
   AFFICHER "La somme de ",X1," et ",X2," est ",X1 + X2
FIN
```

Analysez la ligne 4.

Quelques algorithmes

Écrire un algorithme qui prend en entrée deux nombres a et b et permute leurs valeurs respectives.

Quelques algorithmes

Écrire un algorithme qui prend en entrée deux nombres a et b et permute leurs valeurs respectives.

DEBUT

Quelques algorithmes

Écrire un algorithme qui prend en entrée deux nombres a et b et permute leurs valeurs respectives.

DEBUT
  DEMANDER a

Quelques algorithmes

Écrire un algorithme qui prend en entrée deux nombres a et b et permute leurs valeurs respectives.

```
DEBUT
   DEMANDER a
   DEMANDER b
```

Quelques algorithmes

Écrire un algorithme qui prend en entrée deux nombres a et b et permute leurs valeurs respectives.

DEBUT
  DEMANDER a
  DEMANDER b
  c $\leftarrow$ a

Quelques algorithmes

Écrire un algorithme qui prend en entrée deux nombres a et b et permute leurs valeurs respectives.

DEBUT
  DEMANDER a
  DEMANDER b
  c $\leftarrow$ a
  a $\leftarrow$ b

Quelques algorithmes

Écrire un algorithme qui prend en entrée deux nombres a et b et permute leurs valeurs respectives.

DEBUT
  DEMANDER a
  DEMANDER b
  c $\leftarrow$ a
  a $\leftarrow$ b
  b $\leftarrow$ c

Quelques algorithmes

Écrire un algorithme qui prend en entrée deux nombres a et b et permute leurs valeurs respectives.

DEBUT
  DEMANDER a
  DEMANDER b
  c $\leftarrow$ a
  a $\leftarrow$ b
  b $\leftarrow$ c
FIN

Structures conditionnelles

Structures conditionnelles

- Qu'affiche l'algorithme suivant, pour $a = 5$?

```
DEBUT
   DEMANDER a
   SI a < 0 ALORS
      a ← - a
   FIN SI
   AFFICHER a
FIN
```

Structures conditionnelles

- Qu'affiche l'algorithme suivant, pour $a = 5$?

```
DEBUT
   DEMANDER a
   SI a < 0 ALORS
      a ← - a
   FIN SI
   AFFICHER a
FIN
```

- Qu'affiche l'algorithme pour $a = -4$?

Structures conditionnelles

- Qu'affiche l'algorithme suivant, pour $a = 5$?

```
DEBUT
   DEMANDER a
   SI a < 0 ALORS
      a ← - a
   FIN SI
   AFFICHER a
FIN
```

- Qu'affiche l'algorithme pour $a = -4$?
- Plus généralement, si a est un nombre décimal, que fait cet algorithme ?

Structures conditionnelles

Structures conditionnelles

- Qu'affiche l'algorithme suivant pour $a = 7$ et $b = 21$?

```
DEBUT
   DEMANDER a et b
   SI a < b ALORS
      AFFICHER b
   SINON
      AFFICHER a
   FIN SI
FIN
```

Quelques algorithmes

Structures conditionnelles

- Qu'affiche l'algorithme suivant pour $a = 7$ et $b = 21$?

```
DEBUT
   DEMANDER a et b
   SI a < b ALORS
      AFFICHER b
   SINON
      AFFICHER a
   FIN SI
FIN
```

- Qu'affiche l'algorithme pour $a = 13$ et $b = 3$?

Structures conditionnelles

- Qu'affiche l'algorithme suivant pour $a = 7$ et $b = 21$?

```
DEBUT
   DEMANDER a et b
   SI a < b ALORS
      AFFICHER b
   SINON
      AFFICHER a
   FIN SI
FIN
```

- Qu'affiche l'algorithme pour $a = 13$ et $b = 3$?
- Plus généralement, si a et b sont deux nombres décimaux, que fait cet algorithme ?

Quelques algorithmes

Structures répétitives

Structures répétitives

- Qu'affiche l'algorithme suivant, si on entre 14 pour a et 3 pour b ?

```
DEBUT
   DEMANDER a et b
   TANT QUE b <= a
      a ← a - b
   FIN TANT QUE
   AFFICHER a
FIN
```

Structures répétitives

- Qu'affiche l'algorithme suivant, si on entre 14 pour a et 3 pour b ?

```
DEBUT
   DEMANDER a et b
   TANT QUE b <= a
      a ← a - b
   FIN TANT QUE
   AFFICHER a
FIN
```

- Plus généralement, si a et b sont deux entiers strictement positifs, que fait cet algorithme ?

Structures répétitives

Structures répétitives

Que fait l'algorithme suivant pour $n = 3$?

```
DEBUT
   DEMANDER n
   somme ← 0
   POUR i ALLANT DE 1 à n
      somme ← somme + i
   FIN POUR
   AFFICHER somme
FIN
```

Quelques algorithmes

Structures répétitives

Que fait l'algorithme suivant pour $n = 3$?

```
DEBUT
   DEMANDER n
   somme ← 0
   POUR i ALLANT DE 1 à n
      somme ← somme + i
   FIN POUR
   AFFICHER somme
FIN
```

- Que fait l'algorithme pour $n = 6$?

Quelques algorithmes

Structures répétitives

Que fait l'algorithme suivant pour $n = 3$?

```
DEBUT
   DEMANDER n
   somme ← 0
   POUR i ALLANT DE 1 à n
      somme ← somme + i
   FIN POUR
   AFFICHER somme
FIN
```

- Que fait l'algorithme pour $n = 6$?
- Plus généralement, si n est un entier positif, que fait cet algorithme ?

Structures répétitives

Structures répétitives

Écrire un algorithme qui calcule par la méthode d'Euclide le PGCD de deux entiers strictement positifs.

On pourra calculer « à la main » le PGCD de 123 *et* 27 *avant d'écrire l'algorithme.*

On écrira a mod b *l'opération qui retourne le reste de la division euclidienne de a par b.*

Les fonctions

Les fonctions

Une **fonction** est un bloc d'instructions qui peut être appelé dans un autre bloc.

Les fonctions

Une **fonction** est un bloc d'instructions qui peut être appelé dans un autre bloc.

- La fonction a un nom (pour pouvoir être appelée) ;

Les fonctions

Une **fonction** est un bloc d'instructions qui peut être appelé dans un autre bloc.

- La fonction a un nom (pour pouvoir être appelée) ;
- La fonction a des paramètres contenant des valeurs ;

Les fonctions

Une **fonction** est un bloc d'instructions qui peut être appelé dans un autre bloc.

- La fonction a un nom (pour pouvoir être appelée) ;
- La fonction a des paramètres contenant des valeurs ;
- Une fonction peut renvoyer une valeur au code qui l'a appelé. Une fonction peut ne rien renvoyer (on parle alors de procédure).

Exemple

Exemple

Au combat de dés, chaque joueur lance un dé. On veut créer une fonction appelée `gagnant` qui renvoie 1 si le joueur 1 gagne, 2 si le joueur 2 gagne et 0 en cas d'égalité.

Exemple

Au combat de dés, chaque joueur lance un dé. On veut créer une fonction appelée `gagnant` qui renvoie 1 si le joueur 1 gagne, 2 si le joueur 2 gagne et 0 en cas d'égalité. *Un joueur est gagnant si son numéro tiré est plus grand que celui de l'autre joueur.*

Exemple

Au combat de dés, chaque joueur lance un dé. On veut créer une fonction appelée `gagnant` qui renvoie 1 si le joueur 1 gagne, 2 si le joueur 2 gagne et 0 en cas d'égalité. *Un joueur est gagnant si son numéro tiré est plus grand que celui de l'autre joueur.*

On notera `aleaent(1,6)` *la fonction qui génère un entier aléatoire compris entre* 1 *et* 6.

```
DEBUT
  FONCTION gagnant (de1 , de2)
    SI de1 > de2 ALORS
      RETOURNER 1
    SINON
      SI de1 < de2 ALORS
        RETOURNER 2
      SINON
        RETOURNER 0
      FIN SI
    FIN SI
  FIN FONCTION
  de1 ← aleaent(1, 6)
  AFFICHER "Le joueur 1 a tiré le numéro ",de1
  de2 ← aleaent (1,6)
  AFFICHER "Le joueur 2 a tiré le numéro ",de2
  g ← gagnant (de1 , de2)
  SI g = 0 ALORS
    AFFICHER "Match nul !"
  SINON
    AFFICHER "Le joueur ",g," a gagné."
  FIN SI
FIN
```

Les tableaux ou listes

Les tableaux ou listes

Un **tableau** (ou une liste) est un type composé de **plusieurs valeurs** et **indexé par un type discret ordonné**.

Les tableaux ou listes

Un **tableau** (ou une liste) est un type composé de **plusieurs valeurs** et **indexé par un type discret ordonné**. En fonction du langage de programmation utilisé, les indices sont souvent des entiers allant de 0 à la longueur du tableau - 1 ou de 1 à la longueur du tableau.

Les tableaux ou listes

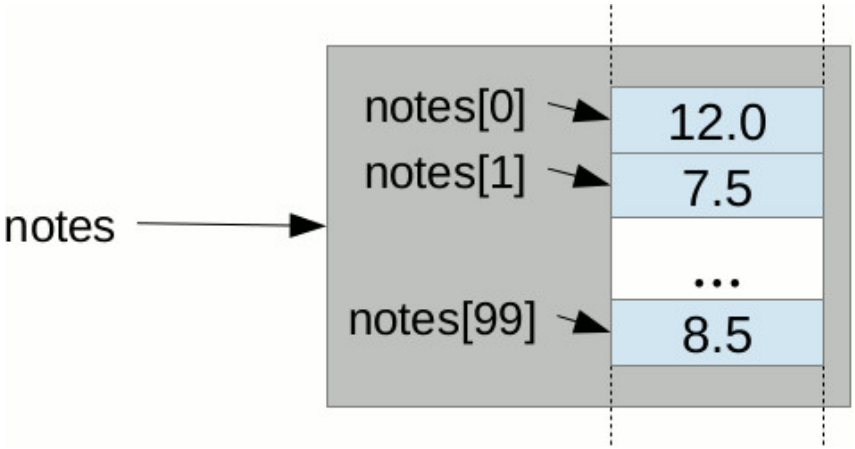
Un **tableau** (ou une liste) est un type composé de **plusieurs valeurs** et **indexé par un type discret ordonné**. En fonction du langage de programmation utilisé, les indices sont souvent des entiers allant de 0 à la longueur du tableau - 1 ou de 1 à la longueur du tableau.

Voici un exemple d'un tableau `notes` contenant 100 décimaux.

Les tableaux ou listes

Un **tableau** (ou une liste) est un type composé de **plusieurs valeurs** et **indexé par un type discret ordonné**. En fonction du langage de programmation utilisé, les indices sont souvent des entiers allant de 0 à la longueur du tableau - 1 ou de 1 à la longueur du tableau.

Voici un exemple d'un tableau `notes` contenant 100 décimaux.




```
notes[0] ← 12.0
notes[1] ← 7.5
⋮
notes[99] ← 8.5
```

Les tableaux ou listes

Les tableaux ou listes

- On accède aux cases d'un tableau en précisant son indice entre crochets.

Les tableaux ou listes

- On accède aux cases d'un tableau en précisant son indice entre crochets.
- L'opérateur `longueur` permet de connaître la taille d'un tableau.

Les tableaux ou listes

- On accède aux cases d'un tableau en précisant son indice entre crochets.
- L'opérateur `longueur` permet de connaître la taille d'un tableau.
- En programmation, un tableau est de taille fixe, alors qu'une liste est de taille dynamique.

Les tableaux ou listes

- On accède aux cases d'un tableau en précisant son indice entre crochets.
- L'opérateur `longueur` permet de connaître la taille d'un tableau.
- En programmation, un tableau est de taille fixe, alors qu'une liste est de taille dynamique.
- Dans un algorithme, un tableau sera assimilé à une liste.

On suppose désormais que les indices des tableaux ou listes commencent à partir de 1*.*

Les tableaux ou listes

Les tableaux ou listes

Imaginons un jeu de 4 joueurs identifiés par leurs noms dans lequel le joueur qui commence est choisi au hasard.

Les tableaux ou listes

Imaginons un jeu de 4 joueurs identifiés par leurs noms dans lequel le joueur qui commence est choisi au hasard.

```
DEBUT
   noms = ["Luc","Marc","Yann","Loïc"]
   joueur ← aleaent (1,4)
   AFFICHER "C'est à ",noms[joueur]," de commencer"
FIN
```

Les tableaux ou listes

Imaginons un jeu de 4 joueurs identifiés par leurs noms dans lequel le joueur qui commence est choisi au hasard.

DEBUT
  noms = ["Luc","Marc","Yann","Loïc"]
  joueur ← `aleaent` (1,4)
  AFFICHER "C'est à ",noms[joueur]," de commencer"
FIN

On crée une liste `noms` de quatre éléments. Chaque élément est une chaîne de caractères contenant le nom du joueur.

Les tableaux ou listes

Imaginons un jeu de 4 joueurs identifiés par leurs noms dans lequel le joueur qui commence est choisi au hasard.

```
DEBUT
   noms = ["Luc","Marc","Yann","Loïc"]
   joueur ← aleaent (1,4)
   AFFICHER "C'est à ",noms[joueur]," de commencer"
FIN
```

On crée une liste `noms` de quatre éléments. Chaque élément est une chaîne de caractères contenant le nom du joueur.
La numérotation commence à 1. Le nom du troisième joueur est donc stocké dans la variables `noms`[3].

Les tableaux ou listes

Imaginons un jeu de 4 joueurs identifiés par leurs noms dans lequel le joueur qui commence est choisi au hasard.

DEBUT
  noms = ["Luc","Marc","Yann","Loïc"]
  joueur ← `aleaent` (1,4)
  AFFICHER "C'est à ",noms[joueur]," de commencer"
FIN

On crée une liste `noms` de quatre éléments. Chaque élément est une chaîne de caractères contenant le nom du joueur.
La numérotation commence à 1. Le nom du troisième joueur est donc stocké dans la variables `noms`[3].
Pour initialiser une liste, il suffit donc de placer entre crochets les éléments en les séparant par des virgules.

Les tableaux ou listes

Imaginons un jeu de 4 joueurs identifiés par leurs noms dans lequel le joueur qui commence est choisi au hasard.

DEBUT
  noms = ["Luc","Marc","Yann","Loïc"]
  joueur ← `aleaent` (1,4)
  AFFICHER "C'est à ",noms[joueur]," de commencer"
FIN

On crée une liste `noms` de quatre éléments. Chaque élément est une chaîne de caractères contenant le nom du joueur.
La numérotation commence à 1. Le nom du troisième joueur est donc stocké dans la variables `noms`[3].
Pour initialiser une liste, il suffit donc de placer entre crochets les éléments en les séparant par des virgules.
On peut placer ce que l'on veut dans une liste : du texte, des nombres, des images, . . . , ou même des listes.

Les tableaux ou listes

Les tableaux ou listes

Voici les opérations élémentaires sur les listes :

Les tableaux ou listes

Voici les opérations élémentaires sur les listes :

- *liste*1 = *liste*2 : Teste l'égalité de deux listes *liste*1 et *liste*2 et renvoie VRAI si dans les cases de même indice des deux listes les éléments sont égaux. Sinon renvoie FAUX.

Les tableaux ou listes

Voici les opérations élémentaires sur les listes :

- *liste*1 = *liste*2 : Teste l'égalité de deux listes *liste*1 et *liste*2 et renvoie VRAI si dans les cases de même indice des deux listes les éléments sont égaux. Sinon renvoie FAUX.
- $e \in$ *liste* : Teste si un élément e est dans la liste *liste* et renvoie VRAI ou FAUX.

Les tableaux ou listes

Voici les opérations élémentaires sur les listes :

- *liste*1 = *liste*2 : Teste l'égalité de deux listes *liste*1 et *liste*2 et renvoie VRAI si dans les cases de même indice des deux listes les éléments sont égaux. Sinon renvoie FAUX.
- $e \in$ *liste* : Teste si un élément e est dans la liste *liste* et renvoie VRAI ou FAUX.
- *liste*.**AJOUTER**(e) : Ajoute l'élément e à la fin de la liste *liste*.

Les tableaux ou listes

Voici les opérations élémentaires sur les listes :

- *liste*1 = *liste*2 : Teste l'égalité de deux listes *liste*1 et *liste*2 et renvoie VRAI si dans les cases de même indice des deux listes les éléments sont égaux. Sinon renvoie FAUX.
- $e \in$ *liste* : Teste si un élément e est dans la liste *liste* et renvoie VRAI ou FAUX.
- *liste*.**AJOUTER**(e) : Ajoute l'élément e à la fin de la liste *liste*.
- *liste*.**INSÉRER**(i, e) : Insère l'élément e au rang i de la liste *liste*.

Les tableaux ou listes

Voici les opérations élémentaires sur les listes :

- *liste*1 = *liste*2 : Teste l'égalité de deux listes *liste*1 et *liste*2 et renvoie VRAI si dans les cases de même indice des deux listes les éléments sont égaux. Sinon renvoie FAUX.
- $e \in$ *liste* : Teste si un élément e est dans la liste *liste* et renvoie VRAI ou FAUX.
- *liste*.**AJOUTER**(e) : Ajoute l'élément e à la fin de la liste *liste*.
- *liste*.**INSÉRER**(i, e) : Insère l'élément e au rang i de la liste *liste*.
- *liste*.**SUPPRIMER**(e) : Supprime la $1^{\text{ère}}$ occurence de l'élément e dans la liste *liste*.

Les tableaux ou listes

Voici les opérations élémentaires sur les listes :

- *liste*1 = *liste*2 : Teste l'égalité de deux listes *liste*1 et *liste*2 et renvoie VRAI si dans les cases de même indice des deux listes les éléments sont égaux. Sinon renvoie FAUX.
- $e \in$ *liste* : Teste si un élément e est dans la liste *liste* et renvoie VRAI ou FAUX.
- *liste*.**AJOUTER**(e) : Ajoute l'élément e à la fin de la liste *liste*.
- *liste*.**INSÉRER**(i, e) : Insère l'élément e au rang i de la liste *liste*.
- *liste*.**SUPPRIMER**(e) : Supprime la $1^{\text{ère}}$ occurence de l'élément e dans la liste *liste*.
- *liste*.**EXTRAIRE**(i) : Supprime l'élément d'indice i da le liste *liste*.

Les tableaux ou listes : exercice

Les tableaux ou listes : exercice

LOTO : Réalisez une simulation de tirage du Loto : 5 boules parmi 49 (numérotées de 1 à 49) sont tirées et un numéro chance entre 1 et 10.

On admettra l'existence d'une fonction `choisir(liste)` *qui tire au hasard et renvoie un élément de la liste* `liste`.

Les tableaux ou listes : solution de l'exercice

Les tableaux ou listes : solution de l'exercice

```
DEBUT
   // On crée une liste des 49 numéros du loto.
   POUR i ALLANT DE 1 à 49
      boules.AJOUTER(i)
   FIN POUR
   // On tire 5 des 49 numéros.
   POUR i ALLANT DE 1 à 5
      b = choisir(boules)
// À chaque fois qu'on tire un numéro, on le supprime de la liste pour ne pas le retirer.
      boules.SUPPRIMER(b)
      AFFICHER "La boule numéro ",i," est ",b
   FIN POUR
   // On tire un numéro chance.
   AFFICHER "Le numéro chance est ",aleaent(1,10)
FIN
```

L'algorithmique au collège

Par le biais de la géométrie

Programmes de constructions

Construire un segment de longueur $\sqrt{n}$, où n est un entier.

Programmes de constructions

Construire un segment de longueur $\sqrt{n}$, où n est un entier.

Spirale de Théodore de Cyrène pour construire la racine carrée d'un entier

Programmes de constructions

Construire un segment de longueur $\sqrt{n}$, où n est un entier.

Spirale de Théodore de Cyrène pour construire la racine carrée d'un entier

(Théorème de Pythagore)

Quadrature du rectangle

Construire un segment de longueur $\sqrt{x \times y}$ à partir de deux segments de longueurs respectives x et y.

Quadrature du rectangle

Construire un segment de longueur $\sqrt{x \times y}$ à partir de deux segments de longueurs respectives x et y.

La quadrature du rectangle

Quadrature du rectangle

Construire un segment de longueur $\sqrt{x \times y}$ à partir de deux segments de longueurs respectives x et y.

La quadrature du rectangle

(Calcul littéral)

Racine carrée d'un côté

Construire un segment de longueur $\sqrt{l}$ à partir d'un segment de longueur l.

Racine carrée d'un côté

Construire un segment de longueur $\sqrt{l}$ à partir d'un segment de longueur l.

Racine carrée d'un segment

Racine carrée d'un côté

Construire un segment de longueur $\sqrt{l}$ à partir d'un segment de longueur l.

Racine carrée d'un segment

(Trigonométrie : le cosinus)

Carré d'un côté

Construire un segment de longueur l^2 à partir d'un segment de longueur l.

Carré d'un côté

Construire un segment de longueur l^2 à partir d'un segment de longueur l.

Le carré d'un segment

Carré d'un côté

Construire un segment de longueur l^2 à partir d'un segment de longueur l.

Le carré d'un segment

(Trigonométrie : la tangente)

Produit de deux segments

Construire un segment de longueur $x \times y$ à partir de deux segments de longueurs respectives x et y.

Produit de deux segments

Construire un segment de longueur $x \times y$ à partir de deux segments de longueurs respectives x et y.

Le produit de deux segments

Produit de deux segments

Construire un segment de longueur $x \times y$ à partir de deux segments de longueurs respectives x et y.

Le produit de deux segments

(Théorème de Thalès)

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.
Les principales instructions utilisées pour la tortue sont :

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.
Les principales instructions utilisées pour la tortue sont :
`tortue[ ]` : pour la création d'une tortue ;

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.
Les principales instructions utilisées pour la tortue sont :
`tortue[ ]` : pour la création d'une tortue ;
`tortueAvance[<tortue>,<distance>]` ;

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.
Les principales instructions utilisées pour la tortue sont :
`tortue[ ]` : pour la création d'une tortue ;
`tortueAvance[<tortue>,<distance>]` ;
`tortueRecule[<tortue>,<distance>]` ;

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.
Les principales instructions utilisées pour la tortue sont :
`tortue[ ]` : pour la création d'une tortue ;
`tortueAvance[<tortue>,<distance>]` ;
`tortueRecule[<tortue>,<distance>]` ;
`tortueADroite[<tortue>,<angle>]` ;

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.
Les principales instructions utilisées pour la tortue sont :
`tortue[ ]` : pour la création d'une tortue ;
`tortueAvance[<tortue>,<distance>]` ;
`tortueRecule[<tortue>,<distance>]` ;
`tortueADroite[<tortue>,<angle>]` ;
`<tortueAGauche[<tortue>,<angle>]` ;

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.
Les principales instructions utilisées pour la tortue sont :
`tortue[ ]` : pour la création d'une tortue ;
`tortueAvance[<tortue>,<distance>]` ;
`tortueRecule[<tortue>,<distance>]` ;
`tortueADroite[<tortue>,<angle>]` ;
`<tortueAGauche[<tortue>,<angle>]` ;
`tortueBC[<tortue>]` et `tortueLC[<tortue>]`.

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.
Les principales instructions utilisées pour la tortue sont :
`tortue[ ]` : pour la création d'une tortue ;
`tortueAvance[<tortue>,<distance>]` ;
`tortueRecule[<tortue>,<distance>]` ;
`tortueADroite[<tortue>,<angle>]` ;
`<tortueAGauche[<tortue>,<angle>]` ;
`tortueBC[<tortue>]` et `tortueLC[<tortue>]`.

Pour répéter plusieurs fois la même séquence de déplacement, on utilise `Répéter[ <Nombre>, <Script>, <Script>, ...]`.

La tortue de GeoGebra

Grâce à la tortue de GeoGebra, on peut concevoir des exercices de construction où les élèves vont manipuler des translations et des rotations afin de réaliser des figures ou des motifs géométriques.
Les principales instructions utilisées pour la tortue sont :
`tortue[ ]` : pour la création d'une tortue ;
`tortueAvance[<tortue>,<distance>]` ;
`tortueRecule[<tortue>,<distance>]` ;
`tortueADroite[<tortue>,<angle>]` ;
`<tortueAGauche[<tortue>,<angle>]` ;
`tortueBC[<tortue>]` et `tortueLC[<tortue>]`.

Pour répéter plusieurs fois la même séquence de déplacement, on utilise `Répéter[ <Nombre>, <Script>, <Script>, ...]`.

Construction de plusieurs carrés à l'aide d'une procédure avec la tortue de GeoGebra

Construction d'un cercle avec la tortue de GeoGebra

Par le biais de l'arithmétique

Crible d'Ératosthène

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120

Crible d'Ératosthène

	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110
111	112	113	114	115	116	117	118	119	120

Les nombres parfaits

Les nombres parfaits

Définition : Un entier naturel n **parfait** est un entier dont la somme de ses diviseurs propres est égale à l'entier lui-même.

Les nombres parfaits

Définition : Un entier naturel n **parfait** est un entier dont la somme de ses diviseurs propres est égale à l'entier lui-même.

Question 1 : Trouver un nombre parfait.

Les nombres parfaits

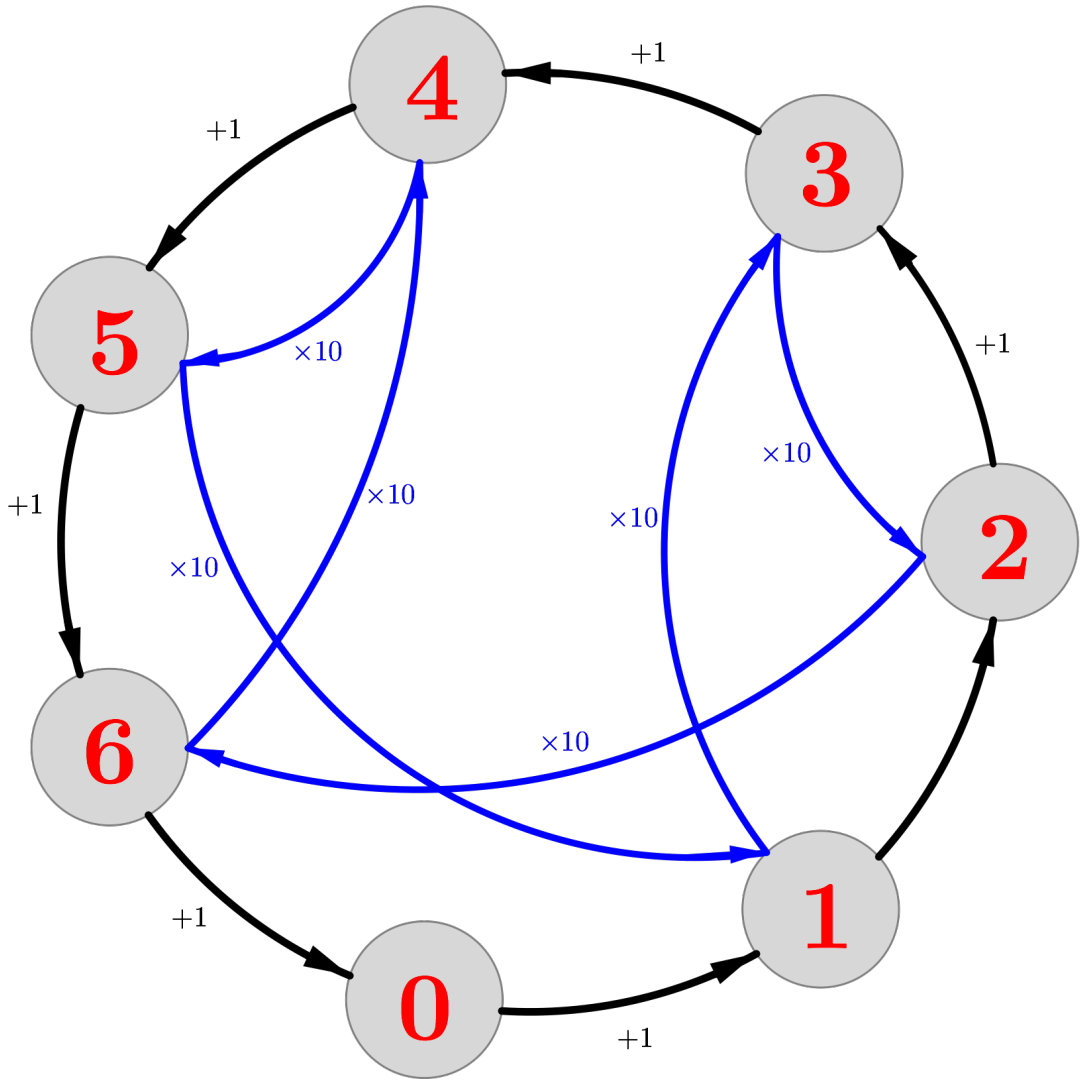
Définition : Un entier naturel n **parfait** est un entier dont la somme de ses diviseurs propres est égale à l'entier lui-même.

Question 1 : Trouver un nombre parfait.

Question 2 : Écrire un algorithme pour tester la « perfection » d'un entier naturel donné.

Critères de divisibilité : automate fini

Critères de divisibilité : automate fini



Par les jeux

Le robot de Lightbot

Le robot de Lightbot

lightbot™

Algorithmes par les jeux

Jeu de Nim

Jeu de Nim

On dispose sur une table *16 objets*. Chacun leur tour, les deux joueurs ramassent *un, deux ou trois objets* sur la table. Le joueur qui **ramasse le dernier objet** remporte la partie.

Jeu de Nim

On dispose sur une table *16 objets*. Chacun leur tour, les deux joueurs ramassent *un, deux ou trois objets* sur la table. Le joueur qui **ramasse le dernier objet** remporte la partie.

Jouer au Nim (deux joueurs)

Jeu de Nim

On dispose sur une table *16 objets*. Chacun leur tour, les deux joueurs ramassent *un, deux ou trois objets* sur la table. Le joueur qui **ramasse le dernier objet** remporte la partie.

Jouer au Nim (deux joueurs)

Jouer au Nim (contre l'ordinateur)

Stratégie gagnante

Stratégie gagnante

Le premier à jouer perd, car il existe une astuce pour que le deuxième joueur gagne à tous les coups.

Stratégie gagnante

Le premier à jouer perd, car il existe une astuce pour que le deuxième joueur gagne à tous les coups. La stratégie gagnante est de laisser 4, 8, 12 ou 16 objets à l'adversaire (un multiple de 4).

Stratégie gagnante

Le premier à jouer perd, car il existe une astuce pour que le deuxième joueur gagne à tous les coups. La stratégie gagnante est de laisser 4, 8, 12 ou 16 objets à l'adversaire (un multiple de 4).

Prenons le dernier tour comme exemple. Il reste 4 objets, et J1 joue :

Stratégie gagnante

Le premier à jouer perd, car il existe une astuce pour que le deuxième joueur gagne à tous les coups. La stratégie gagnante est de laisser 4, 8, 12 ou 16 objets à l'adversaire (un multiple de 4).

Prenons le dernier tour comme exemple. Il reste 4 objets, et J1 joue :

- si J1 prend 1 objet, J2 en prend 3 (dont le dernier) ;

Stratégie gagnante

Le premier à jouer perd, car il existe une astuce pour que le deuxième joueur gagne à tous les coups. La stratégie gagnante est de laisser 4, 8, 12 ou 16 objets à l'adversaire (un multiple de 4).

Prenons le dernier tour comme exemple. Il reste 4 objets, et J1 joue :

- si J1 prend 1 objet, J2 en prend 3 (dont le dernier) ;
- si J1 prend 2 objets, J2 en prend 2 (dont le dernier) ;

Stratégie gagnante

Le premier à jouer perd, car il existe une astuce pour que le deuxième joueur gagne à tous les coups. La stratégie gagnante est de laisser 4, 8, 12 ou 16 objets à l'adversaire (un multiple de 4).

Prenons le dernier tour comme exemple. Il reste 4 objets, et J1 joue :

- si J1 prend 1 objet, J2 en prend 3 (dont le dernier) ;
- si J1 prend 2 objets, J2 en prend 2 (dont le dernier) ;
- si J1 prend 3 objets, J2 en prend 1 (le dernier).

Stratégie gagnante

Le premier à jouer perd, car il existe une astuce pour que le deuxième joueur gagne à tous les coups. La stratégie gagnante est de laisser 4, 8, 12 ou 16 objets à l'adversaire (un multiple de 4).

Prenons le dernier tour comme exemple. Il reste 4 objets, et J1 joue :

- si J1 prend 1 objet, J2 en prend 3 (dont le dernier) ;
- si J1 prend 2 objets, J2 en prend 2 (dont le dernier) ;
- si J1 prend 3 objets, J2 en prend 1 (le dernier).

Dans ce cas, si J2 sait jouer, J1 perd à tous les coups. En appliquant la même méthode, J2 peut guider le jeu de manière à passer de 16 objets à 12, puis 8 et enfin 4.

Stratégie gagnante

Le premier à jouer perd, car il existe une astuce pour que le deuxième joueur gagne à tous les coups. La stratégie gagnante est de laisser 4, 8, 12 ou 16 objets à l'adversaire (un multiple de 4).

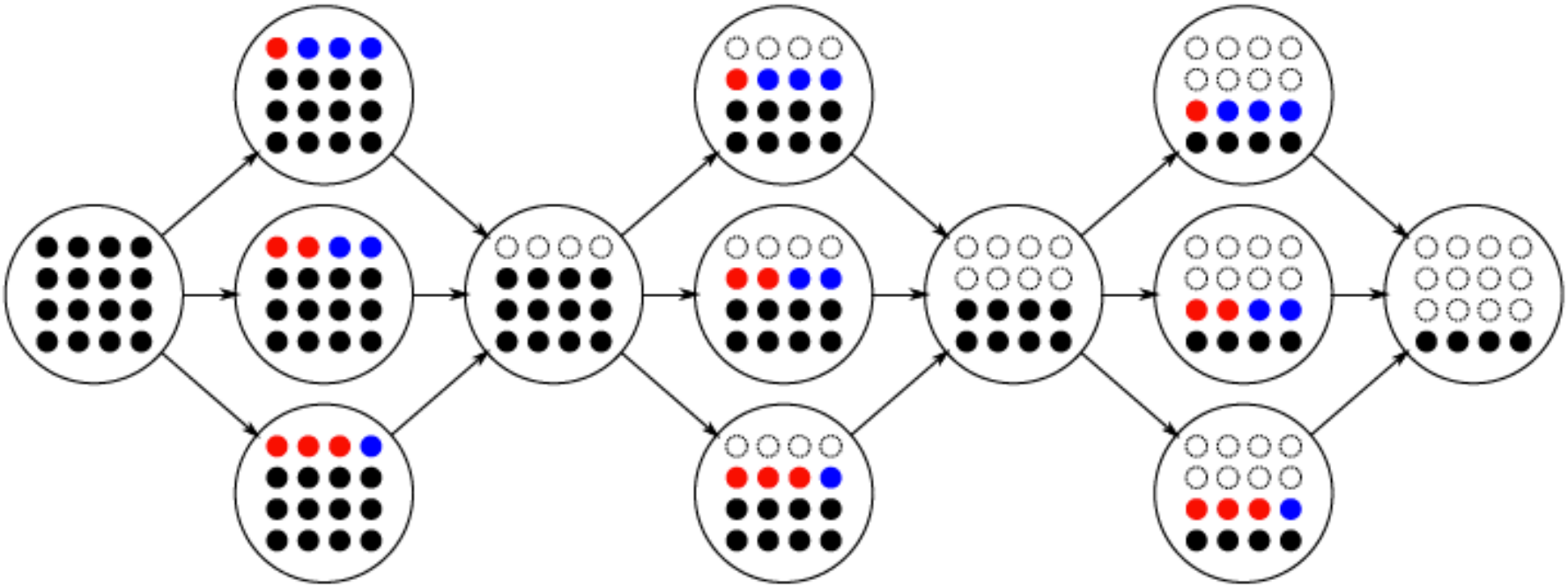
Prenons le dernier tour comme exemple. Il reste 4 objets, et J1 joue :

- si J1 prend 1 objet, J2 en prend 3 (dont le dernier) ;
- si J1 prend 2 objets, J2 en prend 2 (dont le dernier) ;
- si J1 prend 3 objets, J2 en prend 1 (le dernier).

Dans ce cas, si J2 sait jouer, J1 perd à tous les coups. En appliquant la même méthode, J2 peut guider le jeu de manière à passer de 16 objets à 12, puis 8 et enfin 4. Donc, si J2 sait jouer, J1 a perdu la partie avant même de commencer.

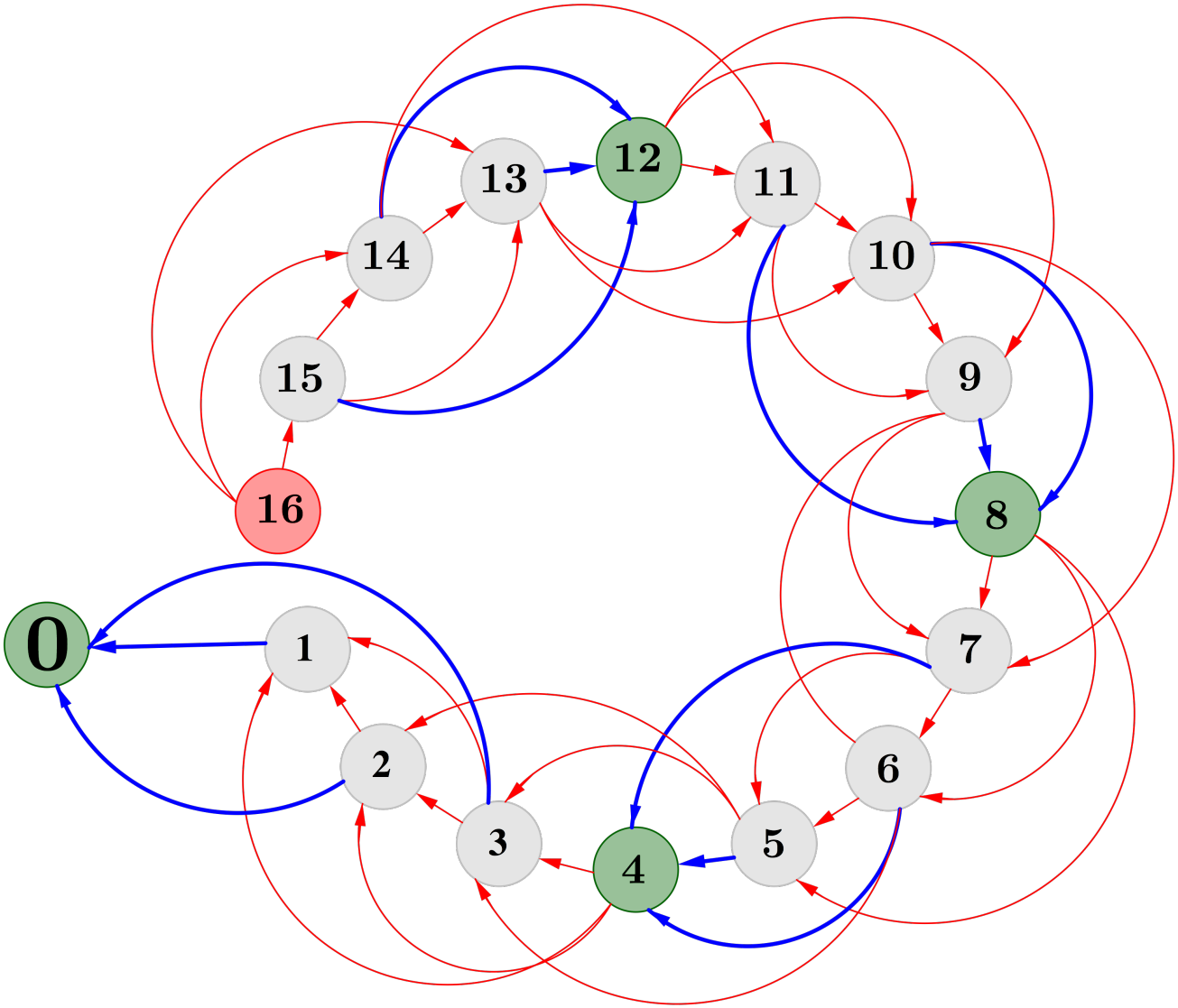
L'algorithme de la stratégie gagnante

L'algorithme de la stratégie gagnante



Noyau d'un graphe

Noyau d'un graphe



Choisir un **nombre entier** entre 1 et 31.

16- 17 - 18 - 19 - 20 - 21 - 22 - 23

16- 17 - 18 - 19 - 20 - 21 - 22 - 23

24 - 25 - 26 - 27 - 28 - 29 - 30 - 31

8 - 9 - 10 - 11 - 12 - 13 - 14 - 15

8 - 9 - 10 - 11 - 12 - 13 - 14 - 15

24 - 25 - 26 - 27 - 28 - 29 - 30 - 31

4 - 5 - 6 - 7 - 12 - 13 - 14 - 15

4 - 5 - 6 - 7 - 12 - 13 - 14 - 15

20 - 21 - 22 - 23 - 28 - 29 - 30 - 31

2 - 3 - 6 - 7 - 10 - 11 - 14 - 15

2 - 3 - 6 - 7 - 10 - 11 - 14 - 15

18 - 19 - 22 - 23 - 26 - 27 - 30 - 31

1 - 3 - 5 - 7 - 9 - 11 - 13 - 15

1 - 3 - 5 - 7 - 9 - 11 - 13 - 15

17 - 19 - 21 - 23 - 25 - 27 - 29 - 31

Cartes binaires

Cartes binaires : version 1

Cartes binaires : version 2

Interface de conversion binaire avec GéoGebra

Recherche dans un dictionnaire

Recherche dans un dictionnaire

Chercher le mot « mathématiques » dans un dictionnaire et compter le nombre d'étapes pour trouver sa définition.

Recherche dans un dictionnaire

Chercher le mot « mathématiques » dans un dictionnaire et compter le nombre d'étapes pour trouver sa définition.

Dictionnaire de l'académie française. 5ème édition

Le tri

Le tri

Trier une main de 6 cartes mélangées aléatoirement.

Le tri

Trier une main de 6 cartes mélangées aléatoirement.

Illustration d'un tri de cartes par sélection

Le tri

Trier une main de 6 cartes mélangées aléatoirement.

Illustration d'un tri de cartes par sélection

Illustration d'un tri à bulles pour un jeu de cartes

Le pile-face par téléphone

Les équipes de football de Maripasoula et de Cayenne doivent décider laquelle accueillera le match du championnat.

Le pile-face par téléphone

Les équipes de football de Maripasoula et de Cayenne doivent décider laquelle accueillera le match du championnat. Le plus simple serait de jouer à pile ou face,

Le pile-face par téléphone

Les équipes de football de Maripasoula et de Cayenne doivent décider laquelle accueillera le match du championnat. Le plus simple serait de jouer à pile ou face, mais les deux villes sont éloignées et les représentants des équipes, Alicia et Bernardo, ne peuvent pas se permettre de perdre du temps et de l'argent avec un voyage en avion.

Le pile-face par téléphone

Les équipes de football de Maripasoula et de Cayenne doivent décider laquelle accueillera le match du championnat. Le plus simple serait de jouer à pile ou face, mais les deux villes sont éloignées et les représentants des équipes, Alicia et Bernardo, ne peuvent pas se permettre de perdre du temps et de l'argent avec un voyage en avion.

Peuvent-ils le faire par téléphone ?

Le pile-face par téléphone

Les équipes de football de Maripasoula et de Cayenne doivent décider laquelle accueillera le match du championnat. Le plus simple serait de jouer à pile ou face, mais les deux villes sont éloignées et les représentants des équipes, Alicia et Bernardo, ne peuvent pas se permettre de perdre du temps et de l'argent avec un voyage en avion.

Peuvent-ils le faire par téléphone ? Disons qu'Alicia lance la pièce et que Bernardo choisisse pile ou face :

Le pile-face par téléphone

Les équipes de football de Maripasoula et de Cayenne doivent décider laquelle accueillera le match du championnat. Le plus simple serait de jouer à pile ou face, mais les deux villes sont éloignées et les représentants des équipes, Alicia et Bernardo, ne peuvent pas se permettre de perdre du temps et de l'argent avec un voyage en avion.

Peuvent-ils le faire par téléphone ? Disons qu'Alicia lance la pièce et que Bernardo choisisse pile ou face : cela ne fonctionnerait pas car si Bernardo choisissait face, Alicia pourrait toujours lui dire : « Désolée, mais c'est pile », et il serait incapable de savoir si c'est vrai.

Le pile-face par téléphone

Les équipes de football de Maripasoula et de Cayenne doivent décider laquelle accueillera le match du championnat. Le plus simple serait de jouer à pile ou face, mais les deux villes sont éloignées et les représentants des équipes, Alicia et Bernardo, ne peuvent pas se permettre de perdre du temps et de l'argent avec un voyage en avion.

Peuvent-ils le faire par téléphone ? Disons qu'Alicia lance la pièce et que Bernardo choisisse pile ou face : cela ne fonctionnerait pas car si Bernardo choisissait face, Alicia pourrait toujours lui dire : « Désolée, mais c'est pile », et il serait incapable de savoir si c'est vrai. Alicia n'est pas une menteuse, mais c'est tout de même un match important et la tentation de tricher est grande.

Le pile-face par téléphone

Les équipes de football de Maripasoula et de Cayenne doivent décider laquelle accueillera le match du championnat. Le plus simple serait de jouer à pile ou face, mais les deux villes sont éloignées et les représentants des équipes, Alicia et Bernardo, ne peuvent pas se permettre de perdre du temps et de l'argent avec un voyage en avion.

Peuvent-ils le faire par téléphone ? Disons qu'Alicia lance la pièce et que Bernardo choisisse pile ou face : cela ne fonctionnerait pas car si Bernardo choisissait face, Alicia pourrait toujours lui dire : « Désolée, mais c'est pile », et il serait incapable de savoir si c'est vrai. Alicia n'est pas une menteuse, mais c'est tout de même un match important et la tentation de tricher est grande. D'ailleurs, même si Alicia se montrait honnête, comment Bernardo pourrait-il la croire s'il perdait ?

Le pile-face par téléphone : circuit logique

Voici donc comment ils décident de procéder.

Le pile-face par téléphone : circuit logique

Voici donc comment ils décident de procéder. Ils conçoivent ensemble un circuit composé de portes logiques ET et de portes logiques OU.

Le pile-face par téléphone : circuit logique

Voici donc comment ils décident de procéder. Ils conçoivent ensemble un circuit composé de portes logiques ET et de portes logiques OU.

ET	0	1
0	0	0
1	0	1

OU	0	1
0	0	1
1	1	1

Le pile-face par téléphone : circuit logique

Voici donc comment ils décident de procéder. Ils conçoivent ensemble un circuit composé de portes logiques ET et de portes logiques OU.

ET	0	1
0	0	0
1	0	1

OU	0	1
0	0	1
1	1	1

Le réseau du pile-face par téléphone

Le pile-face par téléphone : le protocole

L'utilisation du circuit pour jouer à pile ou face par téléphone est la suivante :

Le pile-face par téléphone : le protocole

L'utilisation du circuit pour jouer à pile ou face par téléphone est la suivante : Alicia choisit des entrées aléatoires, soit une séquence de six chiffres binaires (des zéros et des uns, également appelés « bits ») qu'elle garde secrète.

Le pile-face par téléphone : le protocole

L'utilisation du circuit pour jouer à pile ou face par téléphone est la suivante : Alicia choisit des entrées aléatoires, soit une séquence de six chiffres binaires (des zéros et des uns, également appelés « bits ») qu'elle garde secrète. Elle introduit les six chiffres dans le circuit et envoie à Bernardo les six bits de sortie.

Le pile-face par téléphone : le protocole

L'utilisation du circuit pour jouer à pile ou face par téléphone est la suivante : Alicia choisit des entrées aléatoires, soit une séquence de six chiffres binaires (des zéros et des uns, également appelés « bits ») qu'elle garde secrète. Elle introduit les six chiffres dans le circuit et envoie à Bernardo les six bits de sortie. Celui-ci doit alors deviner si l'entrée d'Alicia est composée d'un nombre pair ou impair de uns

Le pile-face par téléphone : le protocole

L'utilisation du circuit pour jouer à pile ou face par téléphone est la suivante : Alicia choisit des entrées aléatoires, soit une séquence de six chiffres binaires (des zéros et des uns, également appelés « bits ») qu'elle garde secrète. Elle introduit les six chiffres dans le circuit et envoie à Bernardo les six bits de sortie. Celui-ci doit alors deviner si l'entrée d'Alicia est composée d'un nombre pair ou impair de uns – autrement dit, il doit deviner la parité de l'entrée.

Le pile-face par téléphone : le protocole

L'utilisation du circuit pour jouer à pile ou face par téléphone est la suivante : Alicia choisit des entrées aléatoires, soit une séquence de six chiffres binaires (des zéros et des uns, également appelés « bits ») qu'elle garde secrète. Elle introduit les six chiffres dans le circuit et envoie à Bernardo les six bits de sortie. Celui-ci doit alors deviner si l'entrée d'Alicia est composée d'un nombre pair ou impair de uns – autrement dit, il doit deviner la parité de l'entrée. Si le circuit est suffisamment complexe, Bernardo sera incapable de trouver la solution et devra donc répondre au hasard (à vrai dire, il pourrait même jouer à pile ou face pour choisir !).

Le pile-face par téléphone : le protocole

L'utilisation du circuit pour jouer à pile ou face par téléphone est la suivante : Alicia choisit des entrées aléatoires, soit une séquence de six chiffres binaires (des zéros et des uns, également appelés « bits ») qu'elle garde secrète. Elle introduit les six chiffres dans le circuit et envoie à Bernardo les six bits de sortie. Celui-ci doit alors deviner si l'entrée d'Alicia est composée d'un nombre pair ou impair de uns – autrement dit, il doit deviner la parité de l'entrée. Si le circuit est suffisamment complexe, Bernardo sera incapable de trouver la solution et devra donc répondre au hasard (à vrai dire, il pourrait même jouer à pile ou face pour choisir !). Si sa réponse est correcte, Bernardo gagne et le match décisif aura lieu à Cayenne.

Le pile-face par téléphone : le protocole

L'utilisation du circuit pour jouer à pile ou face par téléphone est la suivante : Alicia choisit des entrées aléatoires, soit une séquence de six chiffres binaires (des zéros et des uns, également appelés « bits ») qu'elle garde secrète. Elle introduit les six chiffres dans le circuit et envoie à Bernardo les six bits de sortie. Celui-ci doit alors deviner si l'entrée d'Alicia est composée d'un nombre pair ou impair de uns – autrement dit, il doit deviner la parité de l'entrée. Si le circuit est suffisamment complexe, Bernardo sera incapable de trouver la solution et devra donc répondre au hasard (à vrai dire, il pourrait même jouer à pile ou face pour choisir !). Si sa réponse est correcte, Bernardo gagne et le match décisif aura lieu à Cayenne. Si Bernardo se trompe, Alicia gagne et le match aura lieu à Maripasoula.

Le pile-face par téléphone : le protocole

L'utilisation du circuit pour jouer à pile ou face par téléphone est la suivante : Alicia choisit des entrées aléatoires, soit une séquence de six chiffres binaires (des zéros et des uns, également appelés « bits ») qu'elle garde secrète. Elle introduit les six chiffres dans le circuit et envoie à Bernardo les six bits de sortie. Celui-ci doit alors deviner si l'entrée d'Alicia est composée d'un nombre pair ou impair de uns – autrement dit, il doit deviner la parité de l'entrée. Si le circuit est suffisamment complexe, Bernardo sera incapable de trouver la solution et devra donc répondre au hasard (à vrai dire, il pourrait même jouer à pile ou face pour choisir !). Si sa réponse est correcte, Bernardo gagne et le match décisif aura lieu à Cayenne. Si Bernardo se trompe, Alicia gagne et le match aura lieu à Maripasoula. Une fois que Bernardo a fait part de sa réponse à Alicia, celle-ci révèle son entrée secrète pour que Bernardo vérifie s'il produit bien la sortie annoncée.

Le tic-tac-toe

Le tic-tac-toe

Intérêt :

Le tic-tac-toe

Intérêt :

- Utilisation des symétries et rotations pour réduire les cas à étudier.

Le tic-tac-toe

Intérêt :

- Utilisation des symétries et rotations pour réduire les cas à étudier.
- Utilisation d'arbre de décision pour élaborer une stratégie gagnante.

Le tic-tac-toe

Intérêt :
- Utilisation des symétries et rotations pour réduire les cas à étudier.
- Utilisation d'arbre de décision pour élaborer une stratégie gagnante.

Illustration du jeu

Le tic-tac-toe

Intérêt :
- Utilisation des symétries et rotations pour réduire les cas à étudier.
- Utilisation d'arbre de décision pour élaborer une stratégie gagnante.

Illustration du jeu

Arbre de décision du jeu de morpion

Merci de votre attention et à bientôt pour une deuxième journée algorithmique