INFO 2

M. Pétréolle

06 Septembre 2015

Algorithmique

Langages

Programme

Définitions

- Informatique : science du traitement automatisé de l'information

Définitions

- Informatique : science du traitement automatisé de l'information
- L'algorithmique : ensemble des règles et des techniques qui sont impliquées dans la définition et la conception d'algorithmes

Définitions

- Informatique : science du traitement automatisé de l'information
- L'algorithmique : ensemble des règles et des techniques qui sont impliquées dans la définition et la conception d'algorithmes
- Algorithme : suite non-ambiguë d'instructions permettant de donner, à coup sur, et en nombre fini d'étapes, la réponse à un problème

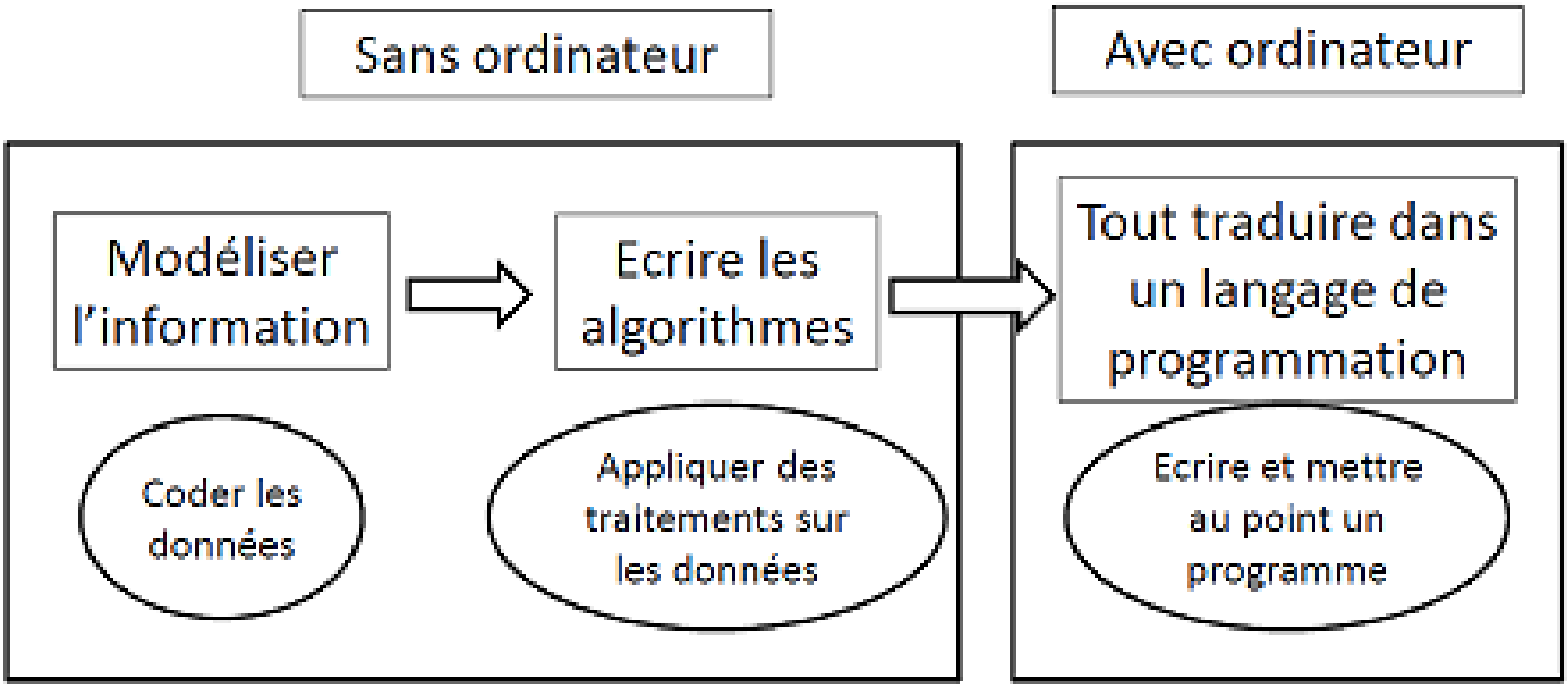
Définitions

- Informatique : science du traitement automatisé de l'information
- L'algorithmique : ensemble des règles et des techniques qui sont impliquées dans la définition et la conception d'algorithmes
- Algorithme : suite non-ambiguë d'instructions permettant de donner, à coup sur, et en nombre fini d'étapes, la réponse à un problème
- Langage : notation conventionnelle formelle destinée à traduire des algorithmes en programmes (logiciels)

Définitions

- Informatique : science du traitement automatisé de l'information
- L'algorithmique : ensemble des règles et des techniques qui sont impliquées dans la définition et la conception d'algorithmes
- Algorithme : suite non-ambiguë d'instructions permettant de donner, à coup sur, et en nombre fini d'étapes, la réponse à un problème
- Langage : notation conventionnelle formelle destinée à traduire des algorithmes en programmes (logiciels)
- Programme : représentation d'un algorithme dans un langage précis, en vue de l'utilisation sur une machine précise (Système d'exploitation précis)

Processus de création d'un programme informatique



Modéliser (Coder) l'information

Coder l'information

- Dans un ordinateur l'unité de base de la RAM est le bit, contraction de binary digit, qui signifie chiffre binaire

Coder l'information

- Dans un ordinateur l'unité de base de la RAM est le bit, contraction de binary digit, qui signifie chiffre binaire
- Un bit, par définition, est un composant quelconque ne pouvant se trouver que dans deux états possibles, exclusifs l'un de l'autre ex : 0 ou 1

Coder l'information

- Dans un ordinateur l'unité de base de la RAM est le bit, contraction de binary digit, qui signifie chiffre binaire
- Un bit, par définition, est un composant quelconque ne pouvant se trouver que dans deux états possibles, exclusifs l'un de l'autre ex : 0 ou 1
- Que faire avec un composant aussi élémentaire ?

Coder l'information

- Dans un ordinateur l'unité de base de la RAM est le bit, contraction de binary digit, qui signifie chiffre binaire
- Un bit, par définition, est un composant quelconque ne pouvant se trouver que dans deux états possibles, exclusifs l'un de l'autre ex : 0 ou 1
- Que faire avec un composant aussi élémentaire ?
- Avec un seul, pas grand chose,

Coder l'information

- Dans un ordinateur l'unité de base de la RAM est le bit, contraction de binary digit, qui signifie chiffre binaire
- Un bit, par définition, est un composant quelconque ne pouvant se trouver que dans deux états possibles, exclusifs l'un de l'autre ex : 0 ou 1
- Que faire avec un composant aussi élémentaire ?
- Avec un seul, pas grand chose,
- Avec plusieurs, beaucoup de choses . . . 8,16,32,64 bits

Coder l'information

- Dans un ordinateur l'unité de base de la RAM est le bit, contraction de binary digit, qui signifie chiffre binaire
- Un bit, par définition, est un composant quelconque ne pouvant se trouver que dans deux états possibles, exclusifs l'un de l'autre ex : 0 ou 1
- Que faire avec un composant aussi élémentaire ?
- Avec un seul, pas grand chose,
- Avec plusieurs, beaucoup de choses . . . 8,16,32,64 bits
- Coder des données discrètes ou continues

Coder l'information (2)

Nombres entiers

- 2013 en base 10

Coder l'information (2)

Nombres entiers

- 2013 en base 10

  4 digits décimaux suffisent

  $2013 = 2 \times 10^3 + 0 \times 10^2 + 1 \times 10^1 + 3 \times 10^0$

Coder l'information (2)

Nombres entiers

- 2013 en base 10
  4 digits décimaux suffisent
  $2013 = 2 \times 10^3 + 0 \times 10^2 + 1 \times 10^1 + 3 \times 10^0$
- 2013 en base 2 ou binaire
  11 digits binaires (bits) sont nécessaires
  $2013 = 1x2^{10} + 1x2^9 + 1x2^8 + 1x2^7 + 1x2^6 + 0x2^5 + 1x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1x2^0$

Coder l'information (2)

Nombres entiers

- 2013 en base 10
  4 digits décimaux suffisent
  $2013 = 2 \times 10^3 + 0 \times 10^2 + 1 \times 10^1 + 3 \times 10^0$
- 2013 en base 2 ou binaire
  11 digits binaires (bits) sont nécessaires
  $2013 = 1x2^{10} + 1x2^9 + 1x2^8 + 1x2^7 + 1x2^6 + 0x2^5 + 1x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1x2^0$
  Que faire avec le signe?

Coder l'information (2)

Nombres entiers

- 2013 en base 10
  4 digits décimaux suffisent
  $2013 = 2 \times 10^3 + 0 \times 10^2 + 1 \times 10^1 + 3 \times 10^0$
- 2013 en base 2 ou binaire
  11 digits binaires (bits) sont nécessaires
  $2013 = 1x2^{10} + 1x2^9 + 1x2^8 + 1x2^7 + 1x2^6 + 0x2^5 + 1x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1x2^0$
  Que faire avec le signe?
  Pourquoi utiliser le binaire ?

Coder l'information (3)

Nombres Réels en virgule flottante

- $27 = 0.27x10^2$, décaler de 2 digits sur la gauche, principe de la virgule flottante

Coder l'information (3)

Nombres Réels en virgule flottante

- $27 = 0.27x10^2$, décaler de 2 digits sur la gauche, principe de la virgule flottante
- En binaire $27 = (11011)_2$

Coder l'information (3)

Nombres Réels en virgule flottante

- $27 = 0.27x10^2$, décaler de 2 digits sur la gauche, principe de la virgule flottante
- En binaire $27 = (11011)_2$
- Autre écriture : $(0,11011x2^{101})_2$, décaler de $5 = (101)_2$ digits sur la gauche :
- 0,11011 est la mantisse
- 101 est l'exposant

Coder l'information (3)

Nombres Réels en virgule flottante

- $27 = 0.27x10^2$, décaler de 2 digits sur la gauche, principe de la virgule flottante
- En binaire $27 = (11011)_2$
- Autre écriture : $(0,11011x2^{101})_2$, décaler de $5 = (101)_2$ digits sur la gauche :
- 0,11011 est la mantisse
- 101 est l'exposant
- Représentation d'un réel : Une partie de la mémoire est réservée pour coder la mantisse, et l'autre pour coder l'exposant

Coder l'information (4)

- Caractères alphanumériques : Alphabets maj + min + Caractères nombres + Caractères spéciaux

Coder l'information (4)

- Caractères alphanumériques : Alphabets maj + min + Caractères nombres + Caractères spéciaux
- Code ASCII (American Standard for Communication and International Interchange) Sur 1 octet = 8 bits soit 28 = 256 positions dans la table

Coder l'information (4)

- Caractères alphanumériques : Alphabets maj + min + Caractères nombres + Caractères spéciaux
- Code ASCII (American Standard for Communication and International Interchange) Sur 1 octet = 8 bits soit 28 = 256 positions dans la table
- Unicode 250 000 caractères différents

Coder l'information (4)

- Caractères alphanumériques : Alphabets maj + min + Caractères nombres + Caractères spéciaux
- Code ASCII (American Standard for Communication and International Interchange) Sur 1 octet = 8 bits soit 28 = 256 positions dans la table
- Unicode 250 000 caractères différents
- En pratique : on utilise juste les caractères du clavier

Coder l'information (5)

Codage des sons

- Un son réel est une superposition de signaux

Coder l'information (5)

Codage des sons

- Un son réel est une superposition de signaux
- Le principe du codage MP3 est de ne coder que la partie du son que l'oreille humaine perçoit

Coder l'information (5)

Codage des sons

- Un son réel est une superposition de signaux
- Le principe du codage MP3 est de ne coder que la partie du son que l'oreille humaine perçoit
- On échantillonne le temps. La fréquence d'échantillonnage est le nombre de découpages par seconde.

Coder l'information (5)

Codage des sons

- Un son réel est une superposition de signaux
- Le principe du codage MP3 est de ne coder que la partie du son que l'oreille humaine perçoit
- On échantillonne le temps. La fréquence d'échantillonnage est le nombre de découpages par seconde. On discrétise l'amplitude : pour chaque intervalle de temps, on prend le nombre qui permet de «s'approcher le plus possible» de la courbe réelle

Coder l'information (5)

Codage des sons

- Un son réel est une superposition de signaux
- Le principe du codage MP3 est de ne coder que la partie du son que l'oreille humaine perçoit
- On échantillonne le temps. La fréquence d'échantillonnage est le nombre de découpages par seconde. On discrétise l'amplitude : pour chaque intervalle de temps, on prend le nombre qui permet de «s'approcher le plus possible» de la courbe réelle
- Pour coder cette courbe, il suffit maintenant de coder successivement les valeurs correspondant à chaque échantillon de temps

Coder l'information (6)

Codage des images

- BitMap
- JPEG
- GIF ...

Coder l'information (6)

Codage des images

- BitMap
- JPEG
- GIF ...

Codage vectoriel

- On code des primitives graphiques : ellipse , rectangle, lignes, sphères, cylindres, parallélépipèdes ...
- Utilisé dans les logiciels de CAO, de réalité virtuelle, ou pour les effets spéciaux numériques

Algorithmique

L'origine du mot

- Il vient du nom du mathématicien perse du 9^e siècle Abu Abdullah Muhammad ibn Musa al-Khwarizmi
- Le mot algorisme se référait à l'origine uniquement aux règles d'arithmétique utilisant les chiffres indo-arabes
- Au 18ème siècle, la traduction du nom Al-Khwarizmi a donné algorithme
- L'utilisation du mot a évolué pour inclure toutes les procédures définies pour résoudre un problème ou accomplir une tâche

Résolution de problèmes complexes à l'aide d'algorithmes

- Principe de l'analyse structurée : Pour résoudre un problème complexe on le découpe en une série de problèmes plus simple

$$\longrightarrow \text{Modularité}$$

Résolution de problèmes complexes à l'aide d'algorithmes

- Principe de l'analyse structurée : Pour résoudre un problème complexe on le découpe en une série de problèmes plus simple

$$\longrightarrow \text{Modularité}$$

- Chaque sous problème est exprimé sous forme d'un ensemble d'action appelé algorithme

Mettre en place un algorithme

- Rechercher les données

Mettre en place un algorithme

- Rechercher les données
- Rechercher les méthodes de traitement à appliquer aux données pour obtenir les résultats

Mettre en place un algorithme

- Rechercher les données
- Rechercher les méthodes de traitement à appliquer aux données pour obtenir les résultats
- Écrire la suite non-ambiguë d'instructions permettant de donner, à coup sur, et en un nombre fini d'étapes, la réponse au problème

Mettre en place un algorithme

- Rechercher les données
- Rechercher les méthodes de traitement à appliquer aux données pour obtenir les résultats
- Écrire la suite non-ambiguë d'instructions permettant de donner, à coup sur, et en un nombre fini d'étapes, la réponse au problème
- Faire appel à la logique

Écrire un algorithme

- Existe t'il un unique langage algorithmique ?

Écrire un algorithme

- Existe t'il un unique langage algorithmique ?
- Non, il n'y a pas de norme

Écrire un algorithme

- Existe t'il un unique langage algorithmique ?
- Non, il n'y a pas de norme
- Mais il doit contenir une syntaxe compréhensible par celui qui devra le lire (l'exécuter)

Écrire un algorithme

- Existe t'il un unique langage algorithmique ?
- Non, il n'y a pas de norme
- Mais il doit contenir une syntaxe compréhensible par celui qui devra le lire (l'exécuter)
- La langue est donc un problème potentiel

Écrire un algorithme (2)

Que doit contenir cette syntaxe ?

Écrire un algorithme (2)

Que doit contenir cette syntaxe ?

Besoins	Réponses
Manipuler des données	Les variables et les constantes
Contrôler le déroulement	Les structures de contrôle
Entrer les données Sortir les résultats	Les flux E/S
Effectuer des calculs	Les opérateurs

Les outils de l'algorithmique

Écriture d'un algorithme

Chaque algorithme devra être

- Nommé : on lui donne un titre (entête)

Écriture d'un algorithme

Chaque algorithme devra être

- Nommé : on lui donne un titre (entête)
- Délimité dans l'espace (sur le papier) : ce qui impose des marques (balises) de début et de fin d'algorithme

Écriture d'un algorithme

Chaque algorithme devra être

- Nommé : on lui donne un titre (entête)
- Délimité dans l'espace (sur le papier) : ce qui impose des marques (balises) de début et de fin d'algorithme
- Bien structuré :

```
Titre de l'algorithme
    Début
        déclarations des variables
        instructions ;
    Fin
```

Les variables

- Transportent les informations pendant le déroulement de l'algorithme

Les variables

- Transportent les informations pendant le déroulement de l'algorithme
- Identifiées de manière unique et claire.On nomme ainsi chaque information : donnée, résultat intermédiaire, résultat final...

Les variables

- Transportent les informations pendant le déroulement de l'algorithme
- Identifiées de manière unique et claire.On nomme ainsi chaque information : donnée, résultat intermédiaire, résultat final...
- Typée : on précise quelle genre de donnée peut être stockée dans chaque variable avec un déclarateur de type

Les variables (2)

Calcul du périmètre d'un cercle
    Début
        Entier compteur ;
        Réél diamètre ;
        Tableau vitesse ;
        ...

Les variables (2)

Calcul du périmètre d'un cercle

Début

Entier compteur ;

Réél diamètre ;

Tableau vitesse ;

...

Fin

Les opérateurs

- L'affectation : Opération qui consiste à écrire une information dans une variable

Les opérateurs

- L'affectation : Opération qui consiste à écrire une information dans une variable

$$\text{diamètre} \leftarrow 6.25;$$

- La comparaison logique : Opération qui consiste à comparer deux quantités

$$\text{longueur} = 2.5;$$

  Ici le résultat est VRAI ou FAUX

Les opérateurs

- L'affectation : Opération qui consiste à écrire une information dans une variable

$$\text{diamètre} \leftarrow 6.25;$$

- La comparaison logique : Opération qui consiste à comparer deux quantités

$$\text{longueur} = 2.5;$$

  Ici le résultat est VRAI ou FAUX
- Ne pas confondre ces deux opérations

Les opérateurs (2)

- Les opérateurs arithmétiques : tout opérateur mathématique

$$+,\ -,\ *,\ /,\ \div,\ \ldots$$

Les opérateurs (2)

- Les opérateurs arithmétiques : tout opérateur mathématique

$$+,\ -,\ *,\ /,\ \div,\ \ldots$$

- Les opérateurs logiques

$$>,\ <,\ \leq,\ \geq,\ =, \neq,\ \text{non, ou, et, ...}$$

Les structures de contrôle

Elles contrôlent la manière dont les instructions s'enchainent :

- Faire une action

Les structures de contrôle

Elles contrôlent la manière dont les instructions s'enchainent :

- Faire une action
- Faire un branchement conditionnel

Les structures de contrôle

Elles contrôlent la manière dont les instructions s'enchainent :

- Faire une action
- Faire un branchement conditionnel
- Faire une répétition

Les actions élémentaires

- Suite d'actions élémentaires en séquence :
  action1 ;
  action2 ; action3 ;
  action4 ;

Les actions élémentaires

- Suite d'actions élémentaires en séquence :
  action1 ;
  action2 ; action3 ;
  action4 ;
- Une action est délimitée par le « ; »

Les actions élémentaires

- Suite d'actions élémentaires en séquence :
  action1 ;
  action2 ; action3 ;
  action4 ;
- Une action est délimitée par le « ; »
- Par défaut les actions s'exécutent du haut en bas et de gauche à droite

Bloc d'actions en-capsulées

- Un bloc d'actions encapsulées est un ensemble d'actions délimités par les balises Début et Fin

```
Début
    action1 ;
    action2 ; action3 ;
    action4 ;
Fin
```

Bloc d'actions en-capsulées

- Un bloc d'actions encapsulées est un ensemble d'actions délimités par les balises Début et Fin

```
Début
    action1 ;
    action2 ; action3 ;
    action4 ;
Fin
```

- Tout le bloc est traité comme l'équivalent d'une seule et unique action (on ne « voit » pas les détails internes)

Structures conditionnelles : les TESTS

- Sans alternative :

  si (condition) action ;

  Il n'y a donc pas de sinon, donc la main passe à l'instruction suivante (après le « ; »)

Structures conditionnelles : les TESTS

- Sans alternative :

  si (condition) action ;

  Il n'y a donc pas de sinon, donc la main passe à l'instruction suivante (après le « ; »)

  Exemple : Si ($var \geq diametre$) $rayon \leftarrow 2.5$;

Structures conditionnelles : les TESTS (2)

- Avec une alternative :

si (condition) alors action 1 ;
sinon action 2 ;

Structures conditionnelles : les TESTS (2)

- Avec une alternative :

  si (condition) alors action 1 ;
  sinon action 2 ;

  Exemple : Si ($var \geq diametre$) *alors* $rayon \leftarrow 2.5$;
  sinon $rayon \leftarrow 42 * var$;

Structures conditionnelles : les TESTS (3)

Les tests en chaîne :

si (var = valeur 1) alors action1 ; action3; . . . ;
    sinon
        si (var = valeur 2) alors action2; . . . ;
    sinon
        si (var = valeur n) alors action n; . . . ;
    sinon
        action n+1 ;

Structure sélective

Remplace les tests multiples.
Exemple :
Suivant (choix)

Début

$1 : diametre \leftarrow 2 * rayon;$

$2 : surface \leftarrow 3.14 * rayon * rayon;$

$3 : perimetre \leftarrow 2 * 3.14 * rayon;$

$autre : afficher\ "choix\ inconnu\ !";$

Fin

Structures de répétition : les BOUCLES

La boucle POUR/FOR :

- Pour *variablecompteur* de *valeurdébut* à *valeurfin* par pas de *valeurpas* faire *action* ;

Structures de répétition : les BOUCLES

La boucle POUR/FOR :

- Pour *variablecompteur* de *valeurdébut* à *valeurfin* par pas de *valeurpas* faire *action* ;
- L'action est exécutée un nombre de fois CONNU avant de commencer la boucle. Le nombre de « tours » dépend des valeurs : valeurdebut, valeurfin et valeurpas

Structures de répétition : les BOUCLES

La boucle POUR/FOR :

- Pour *variablecompteur* de *valeurdébut* à *valeurfin* par pas de *valeurpas* faire *action* ;
- L'action est exécutée un nombre de fois CONNU avant de commencer la boucle. Le nombre de « tours » dépend des valeurs : valeurdebut, valeurfin et valeurpas
- Cette forme est particulièrement adaptée au traitement des tableaux (nombre de lignes et de colonnes connues)

Structures de répétition : les BOUCLES (2)

Exemple :

$n \leftarrow 10$

Pour i de 1 à n par pas de 2

    afficher i ;

Structures de répétition : les BOUCLES (3)

Les boucles événementielles :

- Dans certains cas on ne peut pas savoir quand la boucle va s'arrêter

Structures de répétition : les BOUCLES (3)

Les boucles événementielles :

- Dans certains cas on ne peut pas savoir quand la boucle va s'arrêter
- L'arrêt dépend d'un événement non déterminé à l'avance dans le temps

Structures de répétition : les BOUCLES (3)

Les boucles événementielles :

- Dans certains cas on ne peut pas savoir quand la boucle va s'arrêter
- L'arrêt dépend d'un événement non déterminé à l'avance dans le temps
- Exemple : remplir un verre.
  On doit laisser l'eau couler *jusqu'à* ce que le verre soit plein ou on doit laisser l'eau couler *tant que* le verre n'est pas plein

Structures de répétition : les BOUCLES (3)

4 boucles événementielles existent :

- répéter action jusqu'à condition

Structures de répétition : les BOUCLES (3)

4 boucles événementielles existent :

- répéter action jusqu'à condition
- jusqu'à condition répéter action

Structures de répétition : les BOUCLES (3)

4 boucles événementielles existent :

- répéter action jusqu'à condition
- jusqu'à condition répéter action
- faire action tant que condition
- tant que condition faire action

Les flux d'entrées et de sorties

- obtenir longueur ;
  donner depuis le flux d'entrée de données une valeur à la variable de nom longueur

Les flux d'entrées et de sorties

- obtenir longueur ;
  donner depuis le flux d'entrée de données une valeur à la variable de nom longueur
- afficher longueur ;
  restituer dans le flux de sortie de données la valeur contenue dans variable de nom longueur

Langages de programmation

Définition

On appelle « langage informatique » un langage destiné à décrire l'ensemble des actions consécutives qu'un ordinateur doit exécuter

Définition

On appelle « langage informatique » un langage destiné à décrire l'ensemble des actions consécutives qu'un ordinateur doit exécuter

Exemples : C, C++, Python, Caml, Assembleur, Brainfuck

Langage informatique

- Les langages "machine" : le programmeur écrit directement en binaires les instructions processeur. Avantage : au niveau du matériel, très rapide, compact. Inconvénient : connaissance du processeur nécessaire, pas du tout portable, très difficile à coder

Langage informatique

- Les langages "machine" : le programmeur écrit directement en binaires les instructions processeur. Avantage : au niveau du matériel, très rapide, compact. Inconvénient : connaissance du processeur nécessaire, pas du tout portable, très difficile à coder
- Les langages "assembleur" : le code est très proche du processeur mais est lisible, et compréhensible par un plus grand nombre d'initiés. Avantage : au niveau du matériel, très rapide, compact. Inconvénient : connaissance du processeur nécessaire, peu portable, assez difficile à coder

Langage informatique (2)

- Les langages de haut nivaux, ont ouvert la programmation au plus grand nombre en proposant une syntaxe proche de l'anglais
  Exemples : Fortran, Cobol, Lisp et Algol, Pascal, C, C++, Java, Perl, Python ...
  Avantages : facilité de portage, fonctions élaborées déjà disponibles
  Inconvénients : difficultés d'accès directs au matériel, gourmand en ressources.

Langage informatique (3)

- Contrairement au pseudo-langage utilisé en algorithmique, un langage informatique est extrêmement rigoureux. L'oubli d'un ";" empêche souvent le programme de fonctionner.

Langage informatique (3)

- Contrairement au pseudo-langage utilisé en algorithmique, un langage informatique est extrêmement rigoureux. L'oubli d'un ";" empêche souvent le programme de fonctionner.
- Il impose une syntaxe stricte et normalisée mais qui répond aux mêmes besoins qu'en algorithmique.

Mettre en place un programme en langage C

- Édition, écriture des instructions dans des ≪fichiers source ≫ : fichier de type texte ordinaire ayant l'extension ≪ .C ≫ ou ≪ .CPP ≫

Mettre en place un programme en langage C

- Édition, écriture des instructions dans des «fichiers source » : fichier de type texte ordinaire ayant l'extension « .C » ou « .CPP »
- Compilation des fichiers Srcs : obtention des «fichiers objets » contenant la traduction en langage machine ayant l'extension « .OBJ »

Mettre en place un programme en langage C

- Édition, écriture des instructions dans des «fichiers source » : fichier de type texte ordinaire ayant l'extension « .C » ou « .CPP »
- Compilation des fichiers Srcs : obtention des «fichiers objets » contenant la traduction en langage machine ayant l'extension « .OBJ »
- Éditions de liens : les fichiers objets sont liés aux bibliothèques du langage pour constituer le « fichier exécutable » ayant l'extension « .EXE »