|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Section  Date  Enseignant  Matière | **B1** |  |  |
| **M. MALDONADO**  **[EIAD101] ALGORITMIQUE** | | |

**ALGORITHME ET PROGRAMME**

### Algorithme

**Définition**

* Spécification d’un schéma de calcul, sous forme d’une suite d’opérations élémentaires obéissant à un enchaînement déterminé *(Encyclopedia Universalis).*
* Notion antérieure à celle des ordinteurs  
  - algorithme d’Euclide (calcul du PGCD, +200 JC)
* **Ethymologie**  
  Mathématicien perse du 9e siècle : *Abu Ja’Far Mohammed Ibn Mûsâ al-Khowâ-rismi*

**Caractéristiques**

* Destiné à une personne (le programmeur)  
  - lisible  
  - compréhensible  
  *🡺indentation, nommage des objets (variables),découpage en modules*
* Importance de l’**ordre** des actions. Un algorithme doit :  
  - être défini **avec** **précision** et **sans** **ambiguïté**  
  - se terminer après un **nombre fini d’opérations**  
  - manipuler des **objets bien définis**
* Ecrit en **langage algorithmique**  
  Pascal libre

**Exemple**

Pour téléphoner (depuis une cabine, si, si, ça existe encore !)

1. Chercher le numéro du correspondant
2. Décrocher
3. Introduire la carte de téléphone
4. Composer le numéro
5. **Si** l'on a son correspondant au bout du fil **alors** parler
6. Raccrocher
7. Retirer la carte

On peut affiner l'algorithme pour les étapes encore complexes :

Pour composer le numéro :

1. Composer chaque chiffre
2. Attendre la tonalité
3. **Si** la ligne est occupée **alors**
4. Raccrocher
5. recommencer en 1  
   **Sinon**
6. attendre que la personne décroche.

On remarque :

* Des instructions séquentielles
* Des instructions alternatives (si … alors … sinon)
* Des instructions répétitives (boucle)
* Des raffinements (sous-algorithmes)

### Programme

**Définition**

* Traduction d’un algorithme dans un certain **langage de programmation** et sur un certain **type d’ordinateur** (processeur, système d’exploitation)
* **Choix du langage** :  
  - libre *(cf Projets Personnels Encadrés)* varier les compétences m  
  - **existant** (programme précédent)  
   ex : reprise d’un site en PHP, correction d’une application ACCESS 2003, …  
  - environnement technique  
   ex : application IPhone (Objective C), site (CSS, Javascript, …)

**Caractéristiques**

* Destiné à une machine (le compilateur / l’interprete). Il doit donc :   
  - respecter strictement la **syntaxe** du langage utilisé  
  - ne rien sous-entendre
* Doit cependant être **lisible**  
  - faciliter les corrections futures (1 erreur incompressible toute les 200 lignes)  
  - faciliter les évolutions et extensions futures  
    
  *🡺 commentaires, indentation, nommage des objets, découpage en modules*
* Un algorithme peut donner plusieurs programmes (en fonction du langage choisi)

**Langages**

* Plusieurs milliers !
* Quelques styles (**paradigmes**) de programmation seulement
* Le premier : **langage machine**
* Les **premiers**  
  COBOL (Common Business Oriented Language – IBM 1957 ) : applications de gestion  
  FORTRAN (FORmula TRANslator – IBM 1956) : applications scientifiques  
  LISP (LISt Processing language – MIT 1958) : intelligence artificielle
* **Importance du C**  
  diffusion (années 70), liens avec UNIX, influence (C++, C#, Java, PHP, Objective C,

**Paradigmes de programmation**

***<(http://fr.wikipedia.org/wiki/Paradigme_(programmation)>***

* Programmation **impérative**   
  (presque) tous  
  - Programmation **structurée** (PASCAL, C, …   
  - Programmation **procédurale** (PASCAL, C, …
* Programmation **orientée objet**   
  classe, objet, méthodes,   
  C++, C#, JAVA, PHP, Eiffel, Python, Ruby, SmallTalk, VB.NET,
* Programmation **déclarative :**  
  quoi et non comment  
  - programmation descriptive (HTML, XML, …)  
  - programmation fonctionnelle (LISP, Caml, …)  
  - programmation logique (PROLOG, …)  
  - programmation par contrainte (extensions de PROLOG, JAVA,
* Programmation **fonctionnelle**  
  manipulation de fonctions   
  LISP, Linq
* Programmation **événementielle**programmation d’IHM (interface Homme-Machine)  
  langage traditionnel (C#, Java, Pascal, …) + bibliothèques graphiques
* Programmation **concurrente**gestion du multitâches (parallélisme)   
  ADA, JAVA
* Programmation **internet**sites internet  
  - HTML, XML,   
  - CSS,   
  - Javascript,   
  - PHP, ASP.NET  
  - SQL

**Traduction**

* **Compilation**traduction compléte (production d’un exécutable)  
  A/ exécution rapide, autonome,   
  I/ mise au point longue, non portable  
    
  C,
* **Interprétation**traduction et exécution instruction par instruction  
  A/ mise au point rapide, portable  
  I/ nécessité d’un interprète, exécutionplus lente  
    
  php, HTML
* Approche **mixte**  
  compilation + interprétation  
  code intermédiaire  
  machine virtuelle  
    
  .NET (C#, …), JAVA