**Algorithmie Module 1 Partie 1**

**Objectif :**

* Aborder la prog. Présentation d’un langage universel
* Organiser un programme
* Utiliser des données élémentaires
* Structurer les étapes de résolution d’un problème

**Liens pédago :**

Prérequis : historique et modèle de von Neuman

Nécessaire pour : réaliser tout programme

**Définition :**

Algorithme : Suite finie de règles à appliquer dans un ordre déterminer à un nombre fini de données pour arriver, en un nombre finie d’étapes, a un certain résultat et cela indépendamment des données (source : E.Universalis).

Algorithmie>Nécessité :

* Les langages HL évoluent très vite et nous ne connaissons pas la syntaxe et les particularités des futurs langage, l’algo a traversé 40 ans de langages de haut nvx tout en s’adaptant aux nouveautés.
* L’écriture algo permet de prendre de la distance face aux spés des langages évolués pour lesquels la compréhension ainsi que la constru de nvx prog se sont considérablement alourdis
* La validation des structures algo est bien plus aisée que la validation de langage évolués.
* Le travail en équipe nécessite un langage commun, puis une répartition suivant le langage évolué approprié aux spécificités du moment.

Algo>Historique

* Science very ancienne visant à transmettre des moyens efficaces pour avoir des résultats en partant d’éléments donnés
* Désigne par la suite tout procéder de calcul systématique
* En informatique moderne : procédé auto et effectif comportant une description (finie) des entrées, des sortis et des taches élémentaires à effectuer

**Organigramme :**

Procédure de réalisation d’un prog informatique :

L’écriture algorithmique est une phase intermédiaire et indispensable pour réaliser un programme

La qualité du dev final dépend aussi de cette phase cruciale.

La suppression de cette phase ctrairemen à une idée courante de « gain de temps » qui en découlerais, est généralement la cause de retard de livraison finale du prds logiciel et parfois à l’origine de l’accroissement de la difficulté de résolution d’un programme complexe.

Une variable aura TOUJOURS un type de donnés.

« nom\_de\_la\_variable : [TYPE] » est la syntaxe pour déclarer une variable

**Les types élémentaires possibles :**

* Les entiers
* Les réels
* Les caractères
* Les chaines de caractères
* Les booléens

**Exemples :**

n\_nombre :Entier

c\_caractere :caractère

S\_chaine : chaine

note\_sur\_20 : Entier

Si on a plusieurs vars du même type on peut faire une syntaxe comme ceci : nom\_var1, var2, var3 : [TYPE]

Une variable ne peut avoir qu’un seul et unique [TYPE] élémentaire.

**Affectation :**

C’est le fais d’attribuer une valeur a ma variable

nom\_de\_la\_variable ←valeur

Flèche vers la gauche est le symbole d’affectation

note\_sur\_20←15

1 : déclaration et en suite en 2 : affectation de valeur

On peut affecter à une variable la valeur contenue dans une autre variable :

Nom\_de\_la\_variable←nom\_variable2

Note\_sur\_100←note\_sur\_20\*5

**Réaffectation :**

Note\_sur\_100←note\_sur\_100+5

**Notion de variable :**

En informatique une variable possède à un moment donné une valeur et une seule. A la rigueur elle ne peut pas avoir de valeur du tout (une fois qu’elle a été déclarée, et tant qu’on ne l’a pas affectée. A signaler que dans certains langages…)

**Notion de constante :**

* Les Variables : VAR
  + Déclaration
  + Affectation
  + Utilisation
  + Réaffectation
* Les constantes : CONST
  + Initialisation (avec déclaration)
  + Utilisation

Déclaration de constante : nom\_de\_la\_constate← valeur [TYPE]

Exemple :

Pi←3.14 : REEL

Pourquoi utiliser une constante plutôt qu’une variable ?

* Optimisation
* Fiabilité
* Souplesse
* Facilité de relecture

**Présentation des notions de données** :

* Les données manipulées par les instructions sont :
  + Des variables proprement dites
  + Des constantes
  + Des valeurs littérales (« Zombie U »,45, VRAI)
  + Des expressions complexes (combinaisons de variable, constantes et valeurs littérales avec des opérateurs)

**Primitives :**

Une instruction est un ordre élémentaire -au sens algo, que peut exécuter un prog

**Deux instructions d’entrée-sortie :**

* Lire : lecture de la frappe (au clavier)
* Ecrire : affichage (à l’écran)

**Notion de séquences :**

Suite d’opération élémentaire (ou ACTION) dont l’exécution est séquentielle.

La lecture d’un algo s’effectue de haut en bas.

**Structure de l’algorithme :**

Nom de l’algo

Spécifications de l’algo :

* // BUT : cet algo effectue
* //ENTREE : une liste de M noms
* //SORTIE : une liste triée par ordre alphabétique

Déclaration :

//CONSTANTES : les constates

Corps de l’algorithme

**Présentation des conventions d’écriture :**

* Les commentaires :
  + Sont précédé de : commentaire ligne 🡪// ou \*/xxxx/\*🡪pour commentaire paragraphe
* Les mots clés primitives :
  + Sont écrits en majuscule : VAR :
* Les chaînes de caractères sont encadrées par : « »
* Les caractères sont encadrés par : simple guillemet
* Les accents sont seulement autorisés :
  + Dans les lignes de commentaires
  + Dans les chaînes de caractères : « élèves »
  + Pour les caractères uniques : ‘à’ (alt+145/146)

**Valeurs intermédiaires :**

* On ne connait plus la valeur initiale entrée par l’utilisateur
* Il serait plus « user friendly » d’utiliser l’instruction :

ECRIRE « le carré de »

ECRIRE valeur

ECRIRE « est »

ECRIRE produit

* Est-on vraiment à un entier prêt ?

Exo : écrire un algo complet qui demande 3 nombres entiers à un utilisateur et qui affiche leur somme.

Algorithme : Calcul

//BUT : cet algo fait la somme de 3 entiers

//ENTREE : trois entier saisis par l’utilisateur

//SOORTIE :la somme

VAR : nb1, nb2, nb3, somme : ENTIER

DEBUT

Ecrire « Veuillez entrer 3 entiers »

LIRE nb1, nb2, nb3

Somme ←nb1, nb2, nb3

ECRIRE « la somme est » + somme

FIN

Ecrire un algorithme complet qui demande deux nombres entiers à un utilisateur et qui affiche la somme le produit et le quotient de ces valeurs.

Algorithme : Calcul2

//BUT : cet algo fait la somme de 2 entiers

//ENTREE : 2 entier saisis par l’utilisateur

//SOORTIE :la somme,le produit,le quotient

VAR : nb1, nb2, somme,produit : ENTIER

Quotient : REEL

Début

Ecrire « Veuillez entrer 2 nombres »

Lire nb1

Lire nb2

Somme←nb1+nb2

Produit←nb1\*nb2

Quotient←nb1/nb2

Ecrire somme+produit+quotient

FIN

**Structures :**

* Structure algorithmique
  + Séquence
  + Ruptures de séquence
* Structure de données :
  + Structures élémentaires
  + Structures avancées
* Structure de programmes