LYCÉE de l'ÉLORN
le choix de votre réussite

Spécialité NSI

ALGORITHMIQUE ET PROGRAMMATION

I- Définition :

Un algorithme – ou un speudocode – est une suite d'opérations ou d'instructions permettant de résoudre un problème ou d'obtenir un résultat. Il est **indépendant du langage** utilisé et est censé être compréhensible par tous. L'avantage d'un tel langage est de pouvoir être facilement transcrit dans un autre langage de programmation structurée comme Python, Java, C++, PHP …

II- Structure d'un algorithme et d'un algorigramme

Les algorithmes auront un format uniquement textuel alors que l'algorigramme inclut une représentation graphique des structures.

Algorithme ou pseudocode

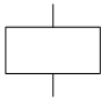
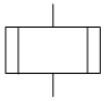
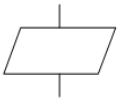
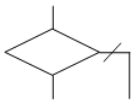
algorithme *nom de l'algorithme ;*

Const *liste des constantes ;*
Var *liste des variables ;*
Fonc *liste des fonctions ;*

Début
action 1;
action 2;
action 3;

Fin algorithme

Algorigramme

SYMBOLE	DESIGNATION	SYMBOLE	DESIGNATION
Symboles de traitement		Symboles auxiliaires	
	Symbole général Opération sur des données, instructions, ...		Renvoi Connecteur utilisé à la fin et en début de ligne pour en assurer la continuité
	Sous-programme Portion de programme		Début, fin ou interruption d'un algorithme
	Entrée-Sortie Mise à disposition ou enregistrement d'une information		Liaison Les différents symboles sont reliés entre eux par des lignes de liaison. Le cheminement va de haut en bas et de gauche à droite. Un cheminement différent est indiqué à l'aide d'une flèche.
Symbole de test			
	Branchement Décision d'un choix parmi d'autres en fonction des conditions		

Les instructions ou actions seront désignées par un verbe à l'infinitif, elles peuvent être :

- *afficher* ou *écrire* — *affiche à l'écran les termes entre guillemets ou le contenu des variables*
- *saisir* — *affecte la valeur saisie à la variable concernée*
- *imprimer* — *envoie l'information à l'imprimante*
- *lire* — *lit les informations existant dans un fichier*
- *ouvrir* — *ouvre le fichier sur lequel les données existent*

III- Déclaration des variables et des constantes.

a) Les constantes :

Elles représentent des nombres, des caractères, des chaînes de caractères … dont la valeur ne peut pas être modifiée au cours de l'exécution du programme.

Exemple : *Const PI = 3,14159265358979323846264338327 95*
Const LesProfs = " M. Salaun et M. Maléjacq"

b) Les variables :

Elles représentent des nombres, des états, des caractères, des chaînes de caractères dont la valeur peut être modifiée au cours de l'exécution du programme. Les variables sont définies par deux caractéristiques essentielles, à savoir :

L'identificateur : le nom de la variable.
Le type : la nature de la variable (entier, réel, booléen, caractère, chaîne de caractères …)

IV- Les types de bases

Nous pouvons considérer cinq types de bases :

a) L'entier	Mot clé : entier	octet, mot de 16, 32, 64 … bits, signé ou pas
b) Le réel	Mot clé : réel	nombre à virgule de dimension plus ou moins grande.
c) Le booléen	Mot clé : booléen	il ne peut prendre que deux valeurs : VRAI (1) ou FAUX (0)
d) Le caractère	Mot clé : car	un seul caractère : "a","A", "h" … *usuellement le code ascii*
e) La chaîne de caractère	Mot clé : chaîne	au moins deux caractères : "toto", "trucmuche" …

V - Les opérateurs

a) Opérateur d'affectation

L'affectation consiste à affecter une valeur à une variable (à ne pas confondre avec l'égalité).

Exemple : x = 3 x = x+2 *(l'on voit aussi x ← 3 x ← x+2)*

b) Opérateur sur les entiers et les réels

Arithmétiques : +, -, *, /, DIV *(division entière),* ↑*(puissance)*
Comparaison : >, <, ≥, ≤, =, ≠

c) Opérateur sur les entiers et les booléens

Fonctions logiques : et, ou, non, ouex, non et, non ou, << *(décalage à gauche)*, >> *(déc à droite)*

d) Opérateur sur les caractères et les chaînes

Fonctions de concaténation : + , & ex : Avec a = "Bonjour" et b = " Monsieur"
a+b donne "Bonjour Monsieur"

Fonctions de comparaison : =, ≠

VI- Les structures algorithmiques fondamentales

a) La structure "*SI ALORS SINON"* *(IF THEN ELSE)*

Dès que nous voulons exprimer une possibilité de choix simple à deux alternatives, nous utilisons cette structure. C'est une des opérations les plus utilisées. Pour ainsi dire, on pourrait tout décrire avec des opérations simples et des « SI ALORS SINON ».

Notation algorithmique

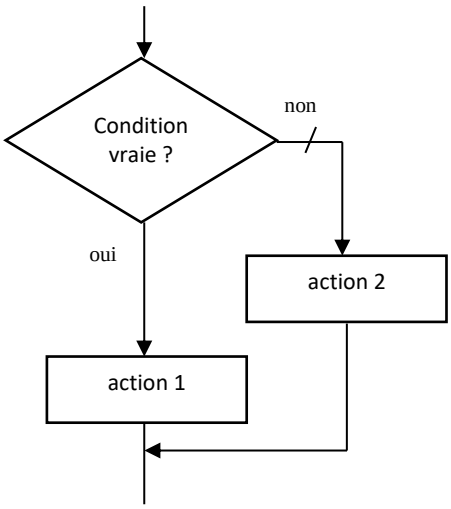
```
si condition alors
  action1 ;
sinon
  action2 ;
finsi ;
```

Exemple :

```
si (note_ds ≥ 10) alors
 écrire "vous avez la moyenne" ;
sinon
 écrire "vous n'avez pas la moyenne" ;
finsi ;
```

Dans cette structure l'exécution d'une des deux actions distinctes ne dépend que du résultat d'un test effectué sur la condition de nature booléenne. Notez que le sinon est optionnel.

Algorigramme équivalent



b) La structure conditionnelle "Selon" *ou* "Suivant Cas" *(SWITCH CASE)*

Cette structure conditionnelle est appelée aussi à choix multiple ou sélective. Elle sélectionne entre plusieurs choix à la fois et pas seulement un seul comme pour la structure « si alors sinon ».

Notation :

```
selon valeur faire
  valeur1 : action1 ;
  valeur2 : action2 ;
  valeur3 : action3 ;
  …
  sinon action4 ;
finselon ;
```

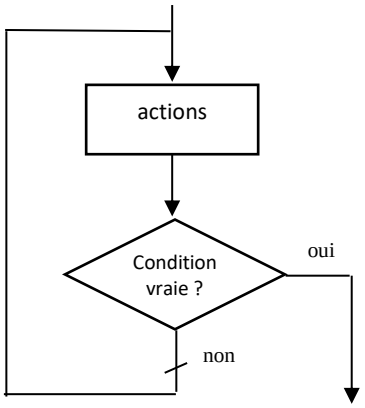
Exemple :

```
selon numéro_de_département faire
   22 : écrire " 22 est le département des Côtes-d'Armor " ;
   29 : écrire " 29 est le département du Finistère" ;
   35 : écrire " 35  est le département de l'Ille-et-Vilaine " ;
   56 : écrire " 56 est le département du Morbihan » ;
   Sinon : écrire " ce département n'est pas en Bretagne" ;
finselon ;
```

c) La structure de boucle "*REPETER JUSQU'À"* *(DO LOOP UNTIL)*

Cette structure répète l'exécution d'une opération ou d'un traitement qui sera ici exécuté au moins une fois :

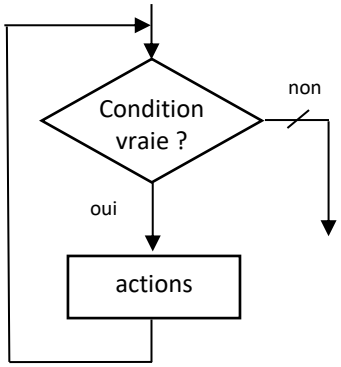
répéter
action1 ;
action2 ;
…
jusqu'à condition vraie ;



d) La structure de boucle *"REPETER TANT QUE" (LOOP WHILE)*

Dans cette structure on commence par tester la condition. Si elle est vérifiée, le traitement est exécuté. L'action peut donc ne jamais être exécutée dans cette boucle.

tant que condition **faire**
action1 ;
action2 ;
…
fintant que ;

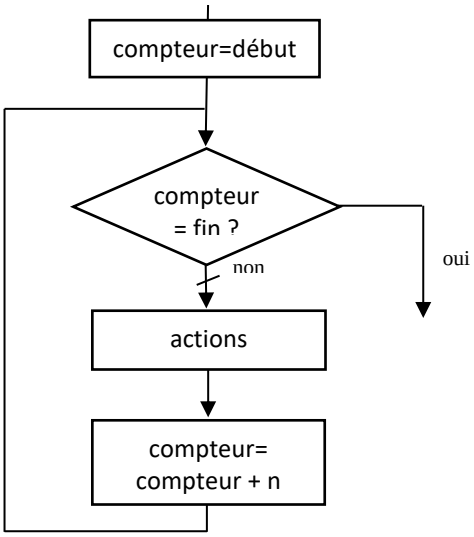


e) La structure de boucle *"POUR … DE … A … FAIRE" (FOR NEXT)*

C'est une boucle dont la sortie s'effectue lorsque le nombre souhaité de répétition est atteint. On utilise une variable de contrôle caractérisée par :

- *Sa valeur initiale*
- *Sa valeur finale*
- *Son pas de variation*

pour compteur **de** début **à** fin **pas** n **faire**
action1 ;
action2 ;
…
finpour ;



VII- Première application

Donner un algorithme puis un algorigramme d'une application permettant le calcul de la racine carrée d'un nombre réel saisie au clavier. Si le nombre saisi n'a pas de racine carrée, l'indiquer.