Algorithmique 01

L'algorithme : L'algorithmique est une notion ancienne (apparue bien avant les premiers ordinateurs) qui peut se défnir comme :
« une méthode de résolution d'un problème sous la forme d'une suite d'opérations élémentaires obéissant à un enchaînement déterminé. »

Le programme : l'expression d'un algorithme dans un langage qu'un ordinateur donné comprend.

Les Variables :

Une VARIABLE possède **:**

Un nom, Un type,Une valeur, Une adresse.

Les Types de Variable en langage C :

En langage C les principaux types de variables sont les suivants :

Booléen :

Ensemble de définition : **{FAUX , VRAI}**

Déclaration algorithmique : **a , b : booléen**

Déclaration C : **bool a ,b ;**

Entier :

Ensemble de définition : **9**

Déclaration algorithmique **i , j : entier**

Déclaration C : **int i , j ;**

Flottant :

Déclaration algorithmique : **x,y : flottant**

Enseignant:BEZZA Youcef

Déclaration C **: float x , y ; ou double x , y ;**

Caractère :

Ensemble de définition : **La table ASCII**

Déclaration algorithmique : **c, g : caractère**

Déclaration C : **char c , g ;**

LES OPERATEURS EN C

Dans tout programme on manipule les **valeurs** et les **variables** qui les référencent, en les combinant avec des **opérateurs** pour former des **expressions**.

Exemple :

y= 3*a + b/5.

Opérateur	Notation algorithmique	Notation C
addition +	**a+b**	**a+b**
soustraction -	**a-b**	**a-b**
opposé -	**-a**	**-a**
produit *	**a * b**	**a * b**
division /	**a / b**	**a / b** (division entière ou décimale selon les types de variables)
modulo %	**a%b**	**a%b** (variables de type entier)

Opérateurs booléens en langage C:

Opérateur	**Notation algorithmique**	**Notation C**
négation	**NON(a)**	**!a**
ou	**a OU b**	**a || b**
et	**a ET b**	**a && b**

Enseignant:BEZZA Youcef

Les Boucles :

Une boucle à quoi ca sert ?

Une boucle permet de répéter une instruction (ou une liste d'instructions) plusieurs fois.

Boucle pour :

Definition :
Les boucles pour permettent de répéter une instruction un nombre donnée de fois. Elle se caractérisent par le fait que l'on connait à l'avance le nombre d'itérations que l'on va devoir effectuer.

Syntaxe :

Pour (variable) **de** (debut) **à** (fin) **faire**

instruction

Fin Pour

Exemple :

Si on suppose qu'une variable i entier a été declarée

Pour i **de** 0 **à** 10 **faire**

afficher(i)

Fin Pour

En langage C :

```
for(expression1;expression2;expression3){
bloc_d'instructions;
}
```

Boucle Tant que :

Les boucles tant que permettent d'effectuer des itérations tant qu'une certaine condition est verifiée.
On ne connait pas le nombre d'itérations à éffectuer, mais à chaque itération,

Enseignant:BEZZA Youcef

on vérifie si la condition est vraie ou fausse. dés que cette condition est fausse, on sort de la boucle

Syntaxe :

Tant que condition **faire**
bloc_d'instructions
Fin tant que

Exemple :

---

i :=1 ;
tantque(i<=n)faire
ecrire(i) ;
i :=i+1 ;
fin tantque

En langage C :

***while** (condition) {*
bloc_d'instructions;
}

Boucle REPETER :

Définition

Est une instruction itérative composée d'un bloc d'instructions (corps de la boucle) et d'une condition, le bloc d'instruction est exécuté une ou plusieurs fois et leur exécution est répétée jusqu'à ce que la condition soit vraie.

Remarques

La condition est évaluée après l'exécution du bloc d'instruction de la boucle " **REPETER** "

Syntaxe :

REPETER
bloc_d'instructions
JUSQU'A Condition

En langage C :

Do {
Bloc_d'instruction ;
}**While**(condition)

Enseignant:BEZZA Youcef