Cours d’Algorithmie

Programme

- Déclaration d’une application
- Les Variables
- Affichage/Saisie
- Les Procédures et Fonctions
- Les Conditions
- Les Boucles
- Les Tableaux

1) Déclaration d'une application

Voici les instructions pour déclarer une application

Pour déclarer une application en C, il est conseillé de retourner un entier en fin de traitement pour notifier au système la fin du programme.
En Java, on peut ne rien retourner (void)

Pseudo Code	C	Java

Début
 	
//Traitement//
 	
Fin
	Int main(void)
	{
	
	//Traitement//
	
	return 0;
	}
		public static void main(String[] args) {
		
		//Traitement//
		
		}

2) Les Variables

Voici les instructions pour déclarer des variables et leur affecter une valeur

Ici, on déclare deux variables de type Entier.
var2 est initialisé dès sa déclaration.
On affecte 12 à var dans le code.

Pseudo Code	C	Java

Variables :
var1,var2=0 de type Entier
Début
 	
var1<- 12
 	
Fin
	Int main(void)
	{
	int var1,var2=0;
	var1=12;
	
	return 0;
	}
		public static void main(String[] args)
		{
		int var1,var2=0;
		var1=12;
		
		}

3) Affichage/Saisie

Voici les instructions permettant d'afficher des éléments à l'écran puis de stocker une saisie au clavier

Ici, on affiche à l'écran « Saisir votre Age »
On récupère la prochaine saisie au clavier dans la variable age de type Entier
On affiche ensuite à l'écran « Vous avez 12 ans ! »

En C, à chaque affichage, il est conseillé de faire un retour à la ligne (\n)
On doit spécifier le type de donnée saisie au clavier ou afficher à l'écran (%i pour int)

En Java, le retour à la ligne est automatique avec la fonction println.
Comme en C, on doit spécifier le type de saisie au clavier (nextInt/nextLine)

Pseudo Code	C	Java

Variables :
age de type Entier
Début
 	
Afficher « Saisir votre Age »
Saisir age
Afficher « Vous avez », age, « ans! »
 	
Fin
	Int main(void)
	{
	int age;
	printf(''Saisir votre Age \n'');
	scanf(''%i'',&age);
	printf(''Vous avez %i ans !'',age);
	return 0;
	}
		public static void main(String[] args)
		{
		int age;
		System.out.println(''Saisir votre Age'');
		Scanner sc=new Scanner();
		age=sc.nextInt();
		System.out.println(''Vous avez ''+age+'' ans !'');
		
		}

4) Les Procédures et Fonctions

Voici les instructions pour coder une fonction, puis l'appeler dans le code principal.

Ici, on affecte 20 à la variable âge, on appel la méthode afficheAge qui affiche « Vous avez 20 ans », on appel ensuite la méthode agePlusDix qui affecte 30 à la variable âge puis on appel de nouveau la méthode afficheAge qui affiche « Vous avez 30 ans »

Pseudo Code

```
Variables :
age de type Entier

Procedure afficheAge(age Entier)
   Afficher « Vous avez   » ,age,  « ans »
Fin Procedure


Fonction agePlusDix(age Entier) en Entier
  Renvoyer age+10
Fin Fonction


Début

age<- 20
afficheAge(age)
age<-agePlusDix(age)
AfficheAge(age)

Fin
```

C

```
void afficheAge(int age)
{
  printf(''Vous avez %i ans \n'',age);
}

int agePlusDix(int age)
{
    return age+10;
}


Int main(void)
{
  age=20;
  afficheAge(age);
  age=agePlusDix(age);
 afficheAge(age);

return 0;
}
```

Java

```
public static void afficheAge(int age)
{
   System.out.println(''Vous avez %i ans'',age);
}

public static int agePlusDix(int age)
{
    return age+10;
}


public static void main(String[] args)
{
    age=20;
    afficheAge(age);
    age=agePlusDix(age);
   afficheAge(age);

}
```

5) Les Conditions (If Else)

Voici les instructions pour déclarer une condition

Ici, on demande à l'utilisateur de saisir son âge.
S'il entre un entier négatif, on lui affiche un message d'erreur et lui demande de saisir à nouveau
On vérifie si l'âge est inférieur à 18, si c'est le cas, on affiche « Vous êtes mineur »
Dans le cas contraire, on affiche « Vous êtes majeur »

Pseudo Code	C	Java

Variables :
age de type Entier
Début
 	
Afficher « Saisir votre age »
Saisir age
 	
Si age<1
Afficher « Votre age ne peut être negatif »
Saisir Age
Fin Si
 	
Si age<18
Afficher « Vous êtes mineur »
Sinon
Afficher « Vous êtes majeur»
Fin Si
 	
Fin
	Int main(void)
	{
	int age;
	printf(''Saisir votre age \n'');
	scanf(''%i'',&age);
	
	if(age<1)
	{
	printf (''Saisir votre Age \n'');
	scanf(''%i'',&age);
	}
	If(age<18)
	{
	printf ('Vous êtes majeur \n'');
	}
	else
	{
	printf ('Vous êtes mineur \n'');
	}
	
	return 0;
	}
		public static void main(String[] args)
		{
		Int age;
		System.out.println(''Saisir votre Age'');
		Scanner sc=new Scanner();
		age=sc.nextInt();
		
		if(age<1)
		{
		System.out.println(''Saisir votre Age'');
		age=sc.nextInt();
		}
		If(age<18)
		{
		System.out.println('Vous êtes majeur'');
		}
		else
		{
		System.out.println('Vous êtes mineur'');
		}
		}

5) Les Conditions (Operateurs)

En Algorithmie, la plupart des opérateurs que vous connaissez sont utilisables.

Voici les principaux que l'on utilisera en cours

Opérateur	Description
+ - * / ()	Opérateurs classiques, la notion de priorité se fait avec des parenthèses
< ou >	Permet de vérifier si une valeur est inférieur/supérieur/
<= ou >=	Permet de vérifier si une valeur est inférieur ou égale / supérieur ou égale
!= ou == (Attention, un seul = affecte une valeur !)	Permet de vérifier si une valeur est Différent de / Egale à
++	Incrémente de 1 la variable. (Age=17; Age++; => Age vaut 18)
--	décrémente de 1 la variable. (Age=17; Age--; => Age vaut 16)
&&	**et** logique
||	**ou** logique
!	**Non** logique
%	Modulo, retourne le reste d'une division (10%3 retourne 1)

5) Les Conditions (Switch)

La condition if... else que l'on vient de voir est le type de condition le plus souvent utilisé.

En fait, il n'y a pas 36 façons de faire une condition en C ou en Java.

Le if... else permet de gérer tous les cas.

Toutefois, il peut s'avérer quelque peu… répétitif. Nous allons donc découvrir les conditions ternaires ainsi que les switch

Voyons l'exemple suivant :

```
if(age==10){printf(''Vous êtes enfant'');}
else if(age==15){System.out.println(''Vous êtes ado'');}
else if(age==18){System.out.println(''Vous êtes majeur'');}
else if(age==30){System.out.println(''Vous êtes adulte'');}
else if(age==90){System.out.println(''Vous êtes vieux'');}
else {System.out.println('Aucune catégorie'');}
```

```
switch(age)
{
 case 10 :  System.out.println (''Vous êtes enfant'');break;
 case 15 :  System.out.println (''Vous êtes enfant'');break;
 case 18 :  System.out.println (''Vous êtes enfant'');break;
 case 30 :  System.out.println (''Vous êtes enfant'');break;
 case 90 :  System.out.println (''Vous êtes enfant'');break;
default :  System.out.println (''Aucune catégorie'');break;
}
```

Vous devez mettre une instruction break; obligatoirement à la fin de chaque cas. Si vous ne le faites pas, alors l'ordinateur ira lire les instructions en dessous censées être réservées aux autres cas ! L'instruction break; commande en fait à l'ordinateur de « sortir » des accolades.

5) Les Conditions (Ternaire)

Une autre façon d'affecter une valeur à la suite d'un test.

Imaginons qu'on veuille affecter à une variable testMajeur la valeur «MINEUR» si l'age est <18, dans le cas contraire on affecte « MAJEUR».

Voyons l'exemple suivant :

```
int age;
String testMajeur;

{…}//Saisie de l'age
If(age<18)
{
  testMajeur=« MINEUR»;
}
else
{
 testMajeur=«  MAJEUR »;
}
```

```
int age;
String testMajeur;

{…}//Saisie de l'age

testMajeur=(age<18) ? « MINEUR» : « MAJEUR»;
```

6) Les Boucles (While)

Nous avons rencontré un problème avec les conditions, lorsque l'âge saisi est négatif, on demande à l'utilisateur. S'il se trompe de nouveau, le programme continue le traitement au lieu de redemander un âge correcte.

Grace à une boucle while, nous allons pouvoir corriger ça

Pseudo Code	C	Java

Variables :
age de type Entier
Début
 	
{…}//Saisie de l'age
TantQue age<1
Afficher «L'age doit être positif »
Saisir age
FinTantQue
 	
Fin
	Int main(void)
	{
	int age;
	{…}//Saisie de l'age
	
	while(age<1)
	{
	printf(''L'age doit être positif \n'');
	scanf(''%i'',&age);
	}
	
	return 0;
	}
		public static void main(String[] args)
		{
		int age;
		Scanner sc=new Scanner();
		
		{…}//Saisie de l'age
		
		while(age<1)
		{
		System.out.printl(''L'age doit être positif'');
		age=sc.nextInt();
		}
		
		}

Il existe deux variantes de la boucle while :

- *while(condition){traitement}*
- *do{traitement} while(condtion)*

Le premier effectue le test avant de rentrer et faire son traitement. Le second test après le traitement, il entre au minimum une fois dans la boucle

6) Les Boucles (For)

Lorsque l'utilisateur connait à l'avance le nombre de tours de boucle nécessaire, on utilisera la boucle « For » qui est un cas particulier de « TantQue »

Voyons l'exemple suivant ou l'on voudrait afficher 10 fois « Bonjour » à l'écran

Pseudo Code	C	Java

Variables :
i de type Entier
Début
 	
Pour i <- 1 à 10
Afficher « Bonjour »
i Suivant
FinPour
 	
Fin
	Int main(void)
	{
	
	for(int i=1;i<=10;i++)
	{
	printf(''Bonjour \n'');
	}
	return 0;
	}
		public static void main(String[] args)
		{
		for(int i=1;i<=10;i++)
		{
		System.out.printl(''Bonjour'');
		}
		}

On pourrait très bien demander à l'utilisateur de saisir le nombre d'occurrence dans la boucle et remplacer le 10 par cette variable.

Il est également possible de manipuler le compteur, il suffit ici d'utiliser la variable i. On pourrait ainsi afficher « Bonjour n°10»

7) Les Tableaux

Imaginons que dans un programme, nous ayons besoin simultanément de 12 valeurs (par exemple, des notes pour calculer une moyenne). Evidemment, la seule solution dont nous disposons à l'heure actuelle consiste à déclarer douze variables, appelées par exemple Note1, Note2, Note3, etc.
si en plus on est dans une situation on l'on ne peut pas savoir d'avance combien il y aura de valeurs à traiter, là on est carrément cuits.
C'est pourquoi la programmation nous permet de rassembler toutes ces variables en une seule, au sein de laquelle chaque valeur sera désignée par un numéro. En bon français, cela donnerait donc quelque chose du genre « la note numéro 1 », « la note numéro 2 », « la note numéro 8 ».

Pseudo Code	C	Java

Tableau Note[12] en Numerique
Variables :
Début
  Pour i<– 0 à 11
  Afficher « Saisir la note : », (i+1)
  Saisir Note[i]
  FinPour
 	
Fin
	Int main(void)
	{
	  float Note[12];
	
	  for(int i=0;i<=11;i++)
	  {
	  printf(''Saisir la note %i'',i+1);
	  scanf(''%lf'',&Note[i]);
	  }
	
	  return 0;
	}
		public static void main(String[] args)
		  {
		  float Note[12];
		  Scanner sc=new Scanner();
		
		  for(int i=0;i<=11;i++)
		  {
		  System.out.println (''Saisir la note %i'',i+1);
		  Note[i]=sc.nextInt();
		  }
		
		}

Dans un tableau, la valeur d'un indice doit toujours :
être égale au moins à 0 (dans quelques rares langages, le premier élément d'un tableau porte l'indice 1). Note(6) est le septième élément du tableau Note !
être un nombre entier Quel que soit le langage, l'élément Note(3,1416) n'existe jamais.
être inférieure ou égale au nombre d'éléments du tableau (moins 1, si l'on commence la numérotation à zéro).

Ne pas confondre l'**indice** d'un élément d'un tableau avec le **contenu** de cet élément. La troisième maison de la rue n'a pas forcément trois habitants, et la vingtième vingt habitants. En notation algorithmique, il n'y a aucun rapport entre i et tab(i).