1. Structure de programmation

1.1. Structure conditionnelle

1.1.1. Instruction simple, bloc d'instruction

Def :

Une expression simple est une expression suivie d'un point virgule ;

Attention : ; fait parti de l'instruction, ce n'est pas un séparateur

Def :

Un bloc d'instruction est une suite d'instructions délimitées par des {}

Syntaxe

```
{
    /* suite d'instructions en tout genre */
}
```

Rque : un bloc d'instruction peut remplacer une instruction simple

1.1.2. Instruction if

Syntaxe :

```
if (expression)
{
    /*bloc d'instruction 1*/
}
else
{
    /*bloc d'instruction 2*/
}
```

Si expression est vraie alors le bloc d'instruction 1 est exécuté sinon le bloc d'instruction 2 est exécuté

Exemple :

```
public class Max
{ public static void main (String args[])
  {
      float x,y;
      System.out.println("Donner deux flottants");
      x = Clavier.lireFloat();
      y = Clavier.lireFloat();

      if (x < y)
            System.out.println("max = " + y);
      else
            System.out.println("max = " + x);

  }
}
```

Imbrication des instructions if

Un exemple :

```
if (a<=b)  if (b<=c) System.out.println(“ordonne”);
else System.out.println(“non ordonne”);
```

Un else se rapporte toujours au dernier if rencontré auquel un else n’a pas encore été attribué.

Instruction switch

Le *switch* est une structure multibranchement dans laquelle une valeur est comparée à plusieurs constantes.

Syntaxe

```
switch (expression)
{
      case exp1 :
            BlocInstruction1;
            break;

      case exp2 :
            BlocInstruction2;
            break;

            .
            .
            .
            .
      case expN :
            BlocInstructionN;
```

```
                break;

        default
                BlocInstruction;
    }
```

Rque :

- L'instruction break est nécessaire pour un choix exclusif
- L'utilisation de default est optionnelle
- Tous les cas doivent être différents
- Expi doivent être des entiers (ie de type int ou char)

Exemple :

```
public class Default
{ public static void main (String[] args)
  {
    int n ;
    System.out.print ("donnez un nombre entier : ") ;
    n = Clavier.lireInt() ;
    switch (n)
    { case 0 : System.out.println ("nul") ;
               break ;
      case 1 : System.out.println ("un") ;
               break ;
      default : System.out.println ("grand") ;
    }
    System.out.println ("Au revoir");
  }
}
```

1.2. Structure répétitive

Ces structures permettent de répéter un certain nombre de fois un ensemble de commandes.

while

Syntaxe :

```
while (expression)
{
        /*bloc d'instruction*/
}
```

Le bloc d'instruction est exécuté tant que expression est vraie

Exemple :

```
public class While1
{ public static void main (String args[])
  {
    int n, som ;
    som = 0 ;
    while (som < 100)
     {
       System.out.print ("donnez un nombre : ") ;
       n = Clavier.lireInt() ;
       som += n ;
     }
    System.out.println ("Somme obtenue : " + som) ;
  }
}
```

do while

Syntaxe

```
do
{
        /*bloc d'instruction*/
} while (exp1) ;
```

Rque : Le bloc d'instruction est exécuté au moins une fois.

Exemple :

```
public class Do1
{
  public static void main (String args[])
  {
    int n ;
    do
     { System.out.print ("donnez un nombre >0 : ") ;
       n = Clavier.lireInt() ;
       System.out.println ("vous avez fourni " + n) ;
     }
    while (n <= 0) ;
    System.out.println ("reponse correcte") ;
  }
}
```

for

Syntaxe :

```
for(exp1,exp2,exp3)
{
        /* bloc d'instruction */
}
```

Où exp1 : initialisation d'une variable de contrôle appelée aussi index
exp2 : expression conditionnelle définissant la condition d'arrêt de la boucle
exp3 : une ou des expressions modifiant l'index

Rque :

- Cette syntaxe peut être équivalente dans certains cas à while
- Il est possible d'omettre exp1, exp2 ou exp3 mais les ; restent obligatoires.

Exemple :

```
public class For1
{
   public static void main (String args[])
  {
    int i ;
    for (i=1 ; i<=5 ; i++)
     {
      System.out.print ("bonjour ") ;
       System.out.println (i + " fois") ;
     }
  }
}
```