INFO-H-100
Séance d'exercices 3

Structures itératives

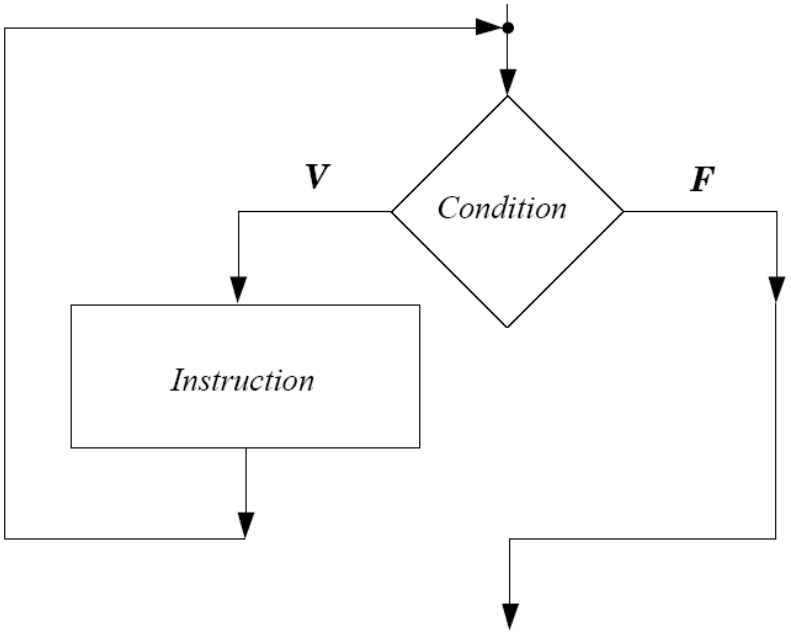
Boucles

- Une **boucle** permet de répéter une instruction donnée.

- Si plusieurs instructions à répéter : utiliser un **bloc**

- 3 types de boucles :
  - `while`
  - `do while`
  - `for`

La boucle « `while` »




```
while(condition)
      instruction
```

1. La *condition* est évaluée

2. Si la *condition* est vérifiée, on exécute l'*instruction* et on retourne au point 1.

   Si la *condition* n'est pas vérifiée, on « sort » de la boucle.

Rappel : Bloc

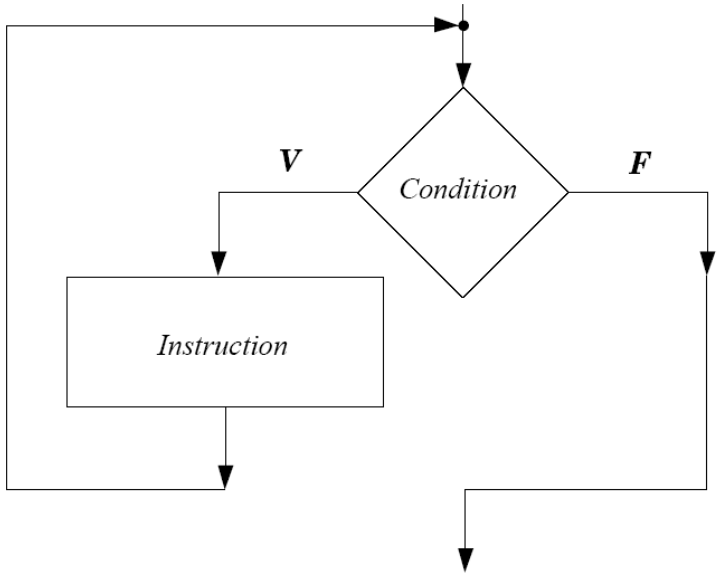
- Bloc = instruction composée
- Regroupement d'instructions

```
{
  x = 3 ;
  y = 7;
  z = x * y ;
}
```

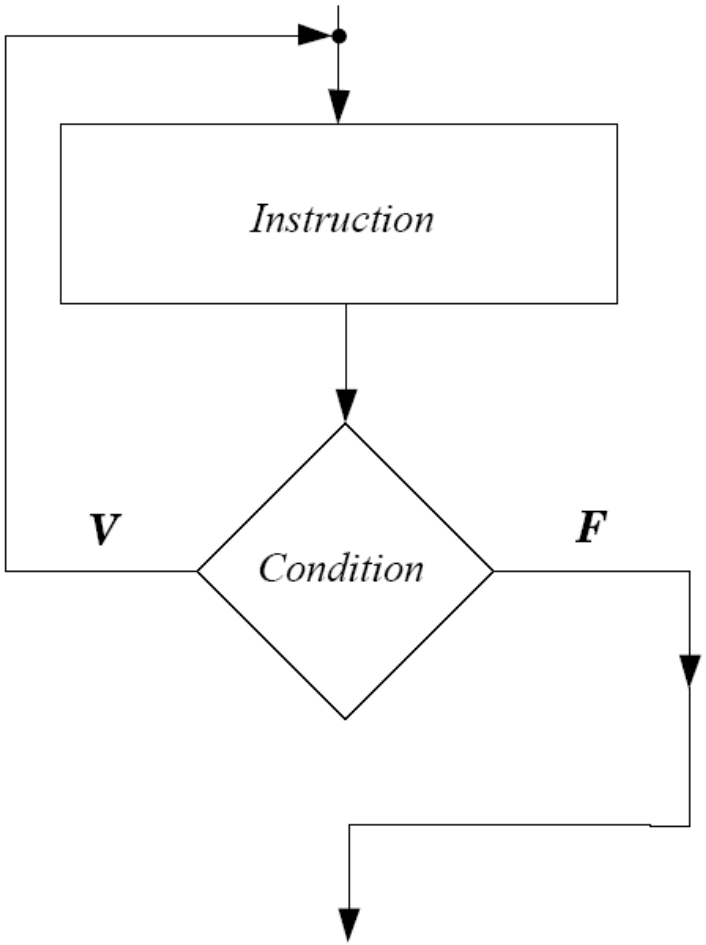
- Si il y a plusieurs instructions dans une boucle : utiliser un bloc.

Boucle « `while` » : exemple

```
cin >> x;
while(x < 8)
{
    x = x + 2;
    cout << x << ' ';
}
cout << endl;
```



La boucle « `do while` »



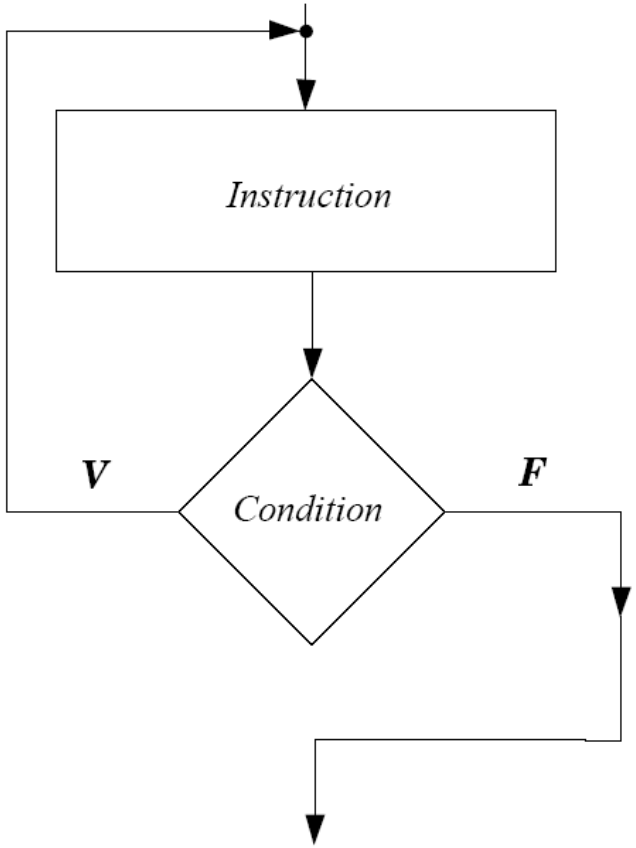

```
do
   instruction

while(condition);
```

1. L'*instruction* est exécutée
2. La *condition* est évaluée
3. Si la *condition* est vérifiée, on retourne au point 1, sinon on sort de la boucle.

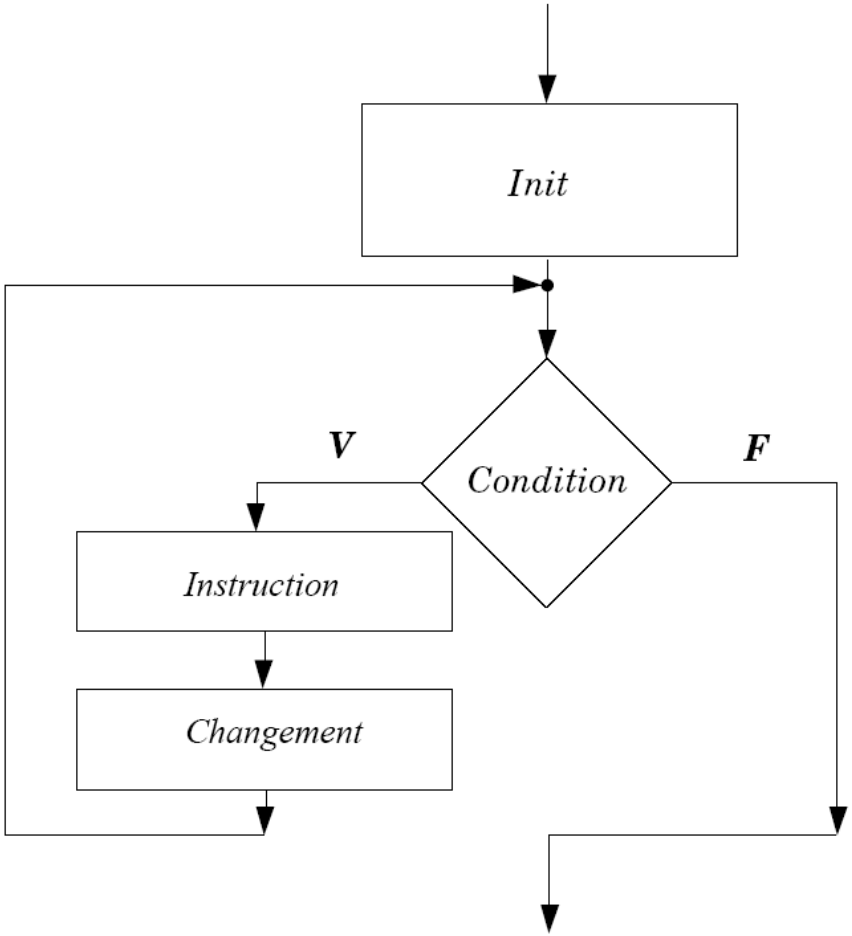
- **L'*instruction* est exécutée au moins une fois**

Boucle « `do while` » : exemple

```
cin >> x;
do
{
   x += 2;
   cout << x << ' ';
}
while (x <= 6);
```



La boucle « `for` »



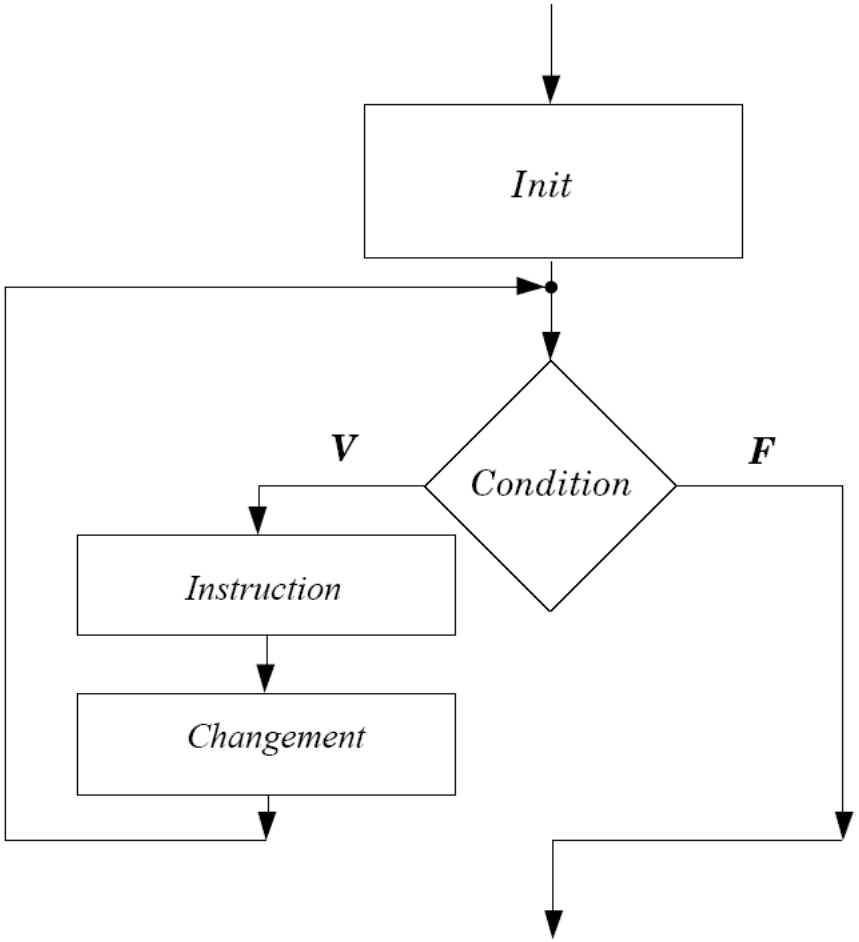

```
for(init; condition; changement)
   instruction
```

1. '*init*' est exécutée
2. La *condition* est évaluée.
3. Si la *condition* est vérifiée :
    1. L'*instruction* est exécutée
    2. Le *changement* est exécuté
    3. Retour au point 2.

   Sinon, on sort de la boucle.

Boucle « `for` » : exemple

```
for(int i=0; i<5; ++i)
{
   cout << i << ' ';
}
cout << endl;
```



Rappel :

- `++i` ajoute 1 à i (incrémente i)
- `--i` enlève 1 à i (décrémente i)

Boucles « `while` » et « `for` »

- Les deux boucles suivantes sont presque équivalentes :

```
for(int i=0; i<5; ++i)
{
  cout << i << ' ';
}
```

```
int i=0;
while(i<5)
{
  cout << i << ' ';
  ++i;
}
```

- Dans un `for`, l'initialisation et le changement sont optionnels et peuvent donc être vides :

```
int i=0;
for(; i<5; ++i)
{
  cout << i << ' ';
}
```

Remarques

- Attention aux boucles infinies.

- Attention à la visibilité :
  - Une variable déclarée dans la partie initialisation d'un `for` n'est visible que
    - dans les parties *condition* et *changement* du `for`
    - dans bloc attaché au `for`.

- Attention aux compteurs dans les boucles :
  - Si on veut faire un traitement complexe du compteur dans la boucle, on choisira plutôt un `while`.

Exercices

- 22, 23, 24, 25
- 26 (variantes 1, 2 croissante et décroissante)
- 27
- 31
- 34
- 35