

CYCLE 4 : BOUCLE POUR


Répéter dans un programme avec une Instruction itérative

INTRODUCTION : Pourquoi les boucles ?

- 1- Comment écrire une instruction POUR
- 2- Comment le CPU exécute une boucle POUR
- 3- EXEMPLES :
 - Table de multiplication
 - Calcul cumulatif
 - Cas particulier du décomptage

Chapitre 4 : Répéter dans un programme en C : Instructions itératives (BOUCLES)

INTRODUCTION : Pourquoi les boucles ?

 Table de multiplication par 2 :

0 X 2 = 0
1 X 2 = 2
2 X 2 = 4
.
10 X 2 = 20

```
int main()
{
    int produit;

    // multiplication par 0
    produit= 0*2;
    printf("\n0 x 2 = %d",produit);

    // multiplication par 1
    produit= 1*2;
    printf("\n1 x 2 = %d",produit);

    ... 11 fois ... !!
}
```

Code plus synthétique:

Instructions à faire 11 fois:
i commence à 0 et finit à 10

produit= i*2;
printf("%dx2=%d",i,produit);



CYCLE 4 : BOUCLE POUR

Répéter dans un programme avec une Instruction itérative

1- Comment écrire une instruction POUR

for() { }

Instruction POUR



```
int main()
{
    /* Répétition avec comptage croissant :
        for ( i=valInit ; i<=valFin ; i=i+pas )
        {
            instructions;
        }
    */
    for ( i=0 ; i<=10 ; i=i+1 )
    {
        printf("%d\t",i);
    }
}
```

i:entier

À NOTER !!

```
int main()
{
    for ( i=valInit ; i<=valFin ; i=i+pas )
    {
        instructions;
    }
}
```

**PAS de ; à la fin du
for()**

Variable compteur
(variable de boucle) i
obligatoirement de
type entier



Règle de programmation : lisibilité code

Indentations dans les accolades du *for*



Le vocabulaire du POUR

```

int main()
{
  for ( i=valInit ; i<=valFin ; i=i+pas )
  {
    instructions;
  }


```

Initialisation :
i=valInit

Incrémentation :
i=i+pas

Condition de poursuite de boucle : **i<=valFin** (cas croissant)

Condition de fin de boucle : **i>valFin**






7


Précisions sur le PAS

```

int main()
{
  for ( i=valInit ; i<=valFin ; i=i+pas )
  {
    instructions;
  }
}

```

-  **Le PAS** : incrément entre 2 exécutions du POUR.
-  Si pas **positif** : compte, sens croissant
-  Si pas **négatif** : décompte, sens décroissant (+condition de poursuite inversée)



8

Rôle de l'instruction POUR



Le POUR **répète l'exécution** d'un bloc d'instructions **un nombre connu de fois**
(boucle avec compteur)



CYCLE 4 : BOUCLE POUR

Répéter dans un programme avec une Instruction itérative

2- Comment le CPU exécute une boucle POUR



Principe d'exécution de l'instruction POUR

```
for ( i=valInit ; i<=valFin ; i=i+pas )  
{  
    instructions;  
}
```



*Le CPU **recommence** l'exécution des instructions **tant que la variable compteur i ne dépasse pas valFin** :*

- Le compteur i démarre, la première fois, à valInit ; puis le CPU vérifie la condition de poursuite : $i \leq \text{valFin}$ (pour un comptage croissant ici)
- Si condition VRAIE, le CPU exécute la boucle, puis revient au FOR
- Lorsque le CPU revient au FOR : i est incrémenté de pas ; puis le CPU vérifie la condition de poursuite de boucle : $i \leq \text{valFin}$
- Quand la condition devient FAUSSE, le CPU sort du POUR.

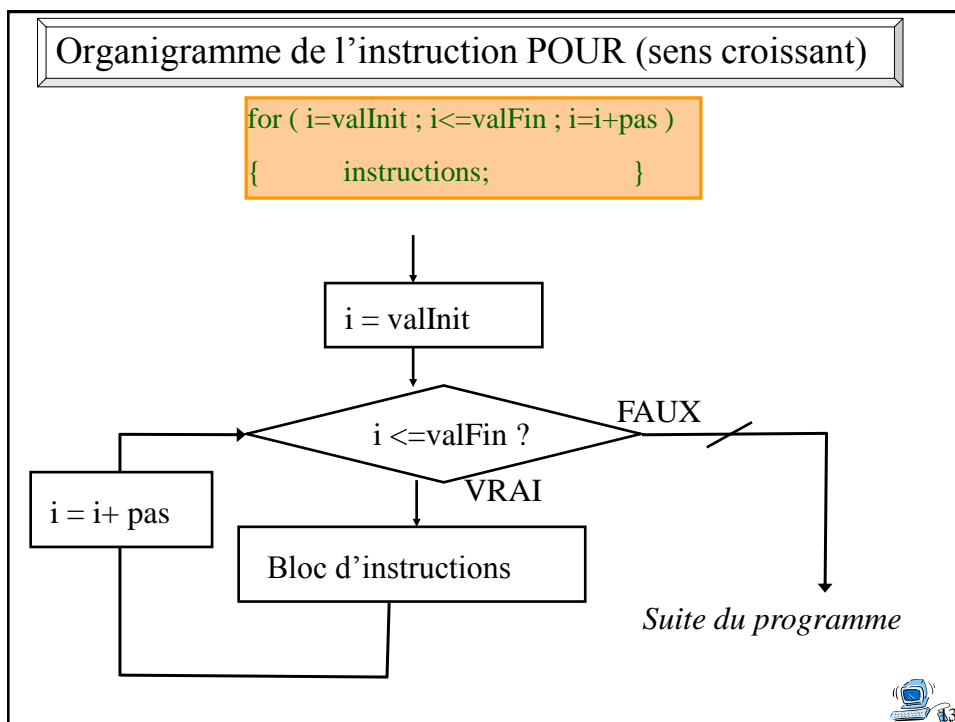


Précisions sur la condition de poursuite de boucle

```
for ( i=valInit ; i<=valFin ; i=i+pas )  
{  
    instructions;  
}
```

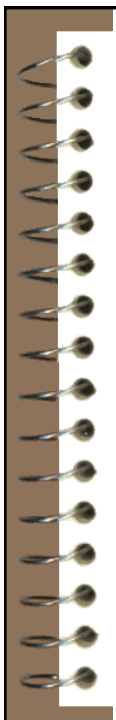
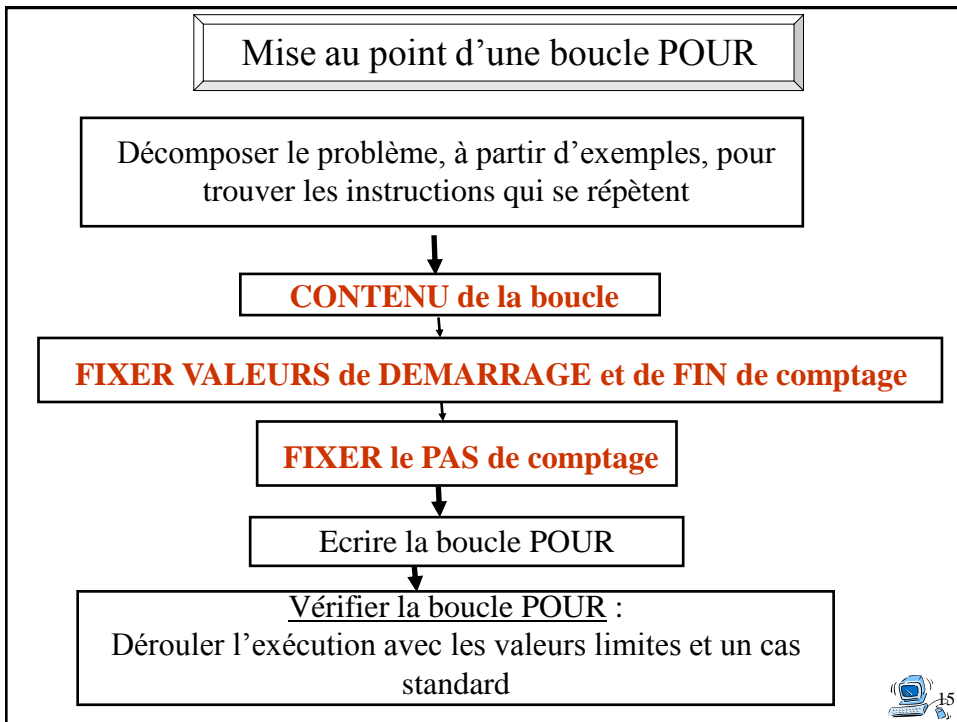
- ✚ Si condition de poursuite de boucle VRAIE (*i ne dépasse pas valFin*) : **la boucle continue.**
- ✚ Si condition de fin de boucle VRAIE (*i dépasse valFin*) : **la boucle s'arrête.**
- ✚ **Condition de poursuite = NON(Condition de fin) :**
conditions logiques inverses





Simulation d'exécution du POUR (sens croissant)

INSTRUCTION	PROCESSEUR
1- for	a- initialisation : i= valeurInit b- test poursuite de boucle i<=valFin : VRAI
2- {	→exécution des instructions
3- } : for	→(automatique) a- incrémentation : i= i + pas b- test poursuite boucle i<=valFin : VRAI
4- {	→exécution des instructions
5- } : for	→(automatique) a- incrémentation : i= i + pas b- test poursuite boucle i<=valFin : FAUX
6- }	→suite du programme après la fin du POUR




CYCLE 4 : BOUCLE POUR

Répéter dans un programme avec une Instruction itérative

3- EXEMPLES :

- Table de multiplication
- Calcul cumulatif
- Cas particulier du décomptage




Affichage de la table de multiplication par 2



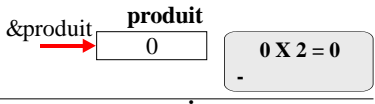
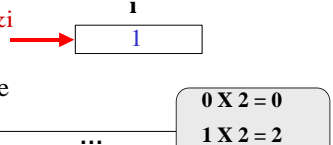
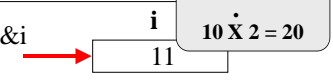
```


int main()
{
    int produit, i;

    /* parcours des valeurs à multiplier
    et affichage des produits */
    for ( i=0 ; i<=10 ; i=i+1 )
    {
        produit= i*2;
        printf("%d x 2 = %d\n",i,produit);
    }
}
    
```

incrémenter i = i+1 → for (i=0 ; i<=10 ; i=i+1)
exécution boucle → {
 produit= i*2;
 printf("%d x 2 = %d\n",i,produit);
rebouclage → }
Fin → }



Instruction	Processeur	Mémoire
1- Déclaration variables	Réservation mémoire	
2- for	a- initialisation i = 0 b- test poursuite boucle 0 ≤ 10 : VRAI	
3- DANS la boucle : -produit= i*2; -printf()	a- calcul: 0x2 b- affectation c- affichage	
4- } : for	a- incrémentation i = i + 1 b- test poursuite boucle 1 ≤ 10 : VRAI ...	
5- } : for	a- incrémentation i = i + 1 b- test poursuite boucle 11 ≤ 10 : FAUX	
6- }	suite après Fin de POUR	



CALCUL CUMULATIF

Le programme saisit 4 entier et les additionne au fur et à mesure

```
int main()
{
    int    a, somme, i;

    somme= 0;           // initialisation de la somme, alors 1° cumul sera: 0+1°entier
    for (i=1 ; i<=4 ; i++)           // comptage de 4 saisies
    {
        printf("Donner un entier : ");           // saisie d'un entier
        scanf("%d",&a);

        // à chaque passage: cumul de a+ valeur précédente de somme

        somme= somme + a;

    }

    printf("somme : %d",somme);           // affichage de la somme des 4 entiers
}
```



19

Précisions sur le CALCUL CUMULATIF

- ☐ Une boucle peut permettre de cumuler des quantités dans une même variable.
- ☐ Variable mise à jour à chaque passage dans la boucle : valeur précédente écrasée par nouvelle valeur calculée.



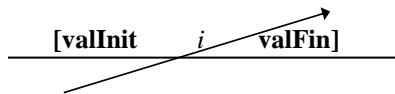
20

Précisions sur une boucle POUR qui décompte

Suivant signe du pas, ATTENTION à condition de poursuite

✚ Comptage : **pas > 0**

- $\text{valInit} \leq \text{valFin}$ (croissant)



✚ Condition poursuite de Boucle :

$$i \leq \text{valFin}$$

✚ Condition fin de Boucle :

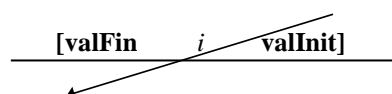
$$i > \text{valFin}$$

for(i=1;i<=10;i=i+2)

$i = \{1, 3, 5, 7, 9\}$

✚ Décomptage : **pas < 0**

- $\text{valInit} \geq \text{valFin}$ (décroissant)



$$i \geq \text{valFin}$$

$$i < \text{valFin}$$

for(i=50;i>=47;i=i-1)

$i = \{50, 49, 48, 47\}$

