Chapitre 4

Les structures de contrôle itératives

Leçon 1 :

Les Structures de contrôle itératives complètes

I-Définition itérative complète :

Un résultat a une définition itérative complète s'il est la répétition d'une suite d'instructions, un **nombre fini de fois connu à l'avance.**

1- Parcours croissant :

Vocabulaire et syntaxe:

Algorithme	Pascal

[Init] **Pour** c **de** 1 **à** n **faire**
Instruction 1
Instruction 2
….
Instruction p
Traitement à Répéter
FinPour
	……. ; {Init}
	FOR c:=1 **TO** n **DO**
	Begin
	Instruction_1;
	Instruction_2;
	……..;
	Instruction_p;
	End;

R : la répétition de p instructions (n fois), n nombre de répétition

Remarques :

- ✓ La partie Init contient les éventuelles initialisations des variables qui seront mis à jour au niveau de traitement répétitif.
- ✓ Le **compteur** doit être de type scalaire. (Entier, caractère, booléen.)
- ✓ L'initialisation et l'avancement du compteur C est faite automatiquement. (Incrémentation par défaut par un pas=1)
- ✓ Le traitement répétitif de la boucle POUR peut s'exécuter 0 ou n fois (n≥1)
- ✓ Lorsque le traitement répétitif est composé de plusieurs instructions, les expressions Begin et End sont nécessaires.

2- Parcours décroissant :

L'avancement du compteur se fait par un pas=-1

Algorithme	Pascal

[inst1, inst2, ...instm] **Pour** i **de** n **à** 1(pas=-1) faire
Instruction 1
Instruction 2
….
Instruction p
Traitement à Répéter
FinPour
	……. ; {Init}
	FOR I:=n **downTO** 1 **DO**
	Begin
	Instruction_1;
	Instruction_2;
	……..;
	Instruction_p;
	End;

Décrémentation automatique du compteur (passage au prédécesseur de la valeur en cours).

Activités :

*Afficher le message bonjour 10 fois

Pour i **de** 1 **à** 10 **faire**
écrire (''Bonjour'')
FinPour

en Pascal :

```
program bonjour ;
uses wincrt ;
     var   i:integer ;
Begin
     For i:=1 to 10 do
     writeln('Bonjour') ;
End.
```

*Afficher les entiers de 1 à 10 :

Pour i **de** 1 **à** 10 **faire**
écrire (i)
FinPour

en Pascal :

```
For i:=1 to 10 do
writeln(i) ;
```

*Afficher les entiers de 1 à 10 en ordre inverse:

Pour i **de** 10 **à** 1 (pas=-1) **faire**
écrire (i)
FinPour

en Pascal :

```
For i:=10 downto 1 do
writeln(i) ;
```

*Afficher les lettres de A à K :

Pour i **de** "A" **à** "K" **faire**
écrire (i)
FinPour

en Pascal :

```
Var   i :char ;
....
For i:='A' to 'K' do
writeln(i) ;
```

Exercice 1:

Écrire un programme qui permet de faire la somme de 5 réels.

Algorithme :

0) Début Somme
1) [S ← 0] pour i de 1 à 5 faire
   Ecrire(" Donner un réel : "), lire(x)
   S ← S + x
   FinPour
2) Ecrire("La somme est ", S)
3) Fin Somme

Trace d'exécution

S=0

i	1	2	3	4	5
x	10	3	22	9	14
S	10	13	35	44	58

s=58

T.D.O

Objet	Type/Nature	Rôle
i	Entier	compteur
X	Réel	Réels à saisir
S	Réel	Somme des réels

Traduction en Pascal :

```
program somme ;
uses wincrt ;
var
     x,s : real ;
     i:integer ;
begin
s:=0 ;
     for i:=1 to 5 do
     begin
        write(' Donner un réel : ') ;
        readln(x);
        s:= s + x ;
      end ;
writeln('La somme est ', s:5:2) ;
End.
```

Exercice 2 :

Écrire un programme qui permet de faire le factoriel d'un entier n donné.

Exemples : 3 ! = 1 x 2 x 3 = 6 6!= 1 x 2 x 3 x 4 x 5 x 6 = 720

Algorithme :

0) Début Factoriel
1) Ecrire("Donner un entier") , lire(n)
2) [f← 1] pour i de 2 à n faire
   f ← f * i
   FinPour
3) Ecrire("Le factoriel est ", f)
4) Fin Factoriel

Trace d'exécution

f=**1**

i	**2**	**3**	**4**	**5**	**6**
f	2	6	24	120	720

f=720

Traduction en Pascal :

```
program factoriel ;
uses wincrt ;
var
     i,n : integer ;
     f : longint ;
begin
   write(' Donner un entier : ') ;
        readln(n);
f:=1 ;
     for i:=2 to n do
         f:= f * i ;

writeln('Le factoriel est ', f) ;
End.
```

Applications : (Série)

Exercice 4 :

Ecrire un programme qui permet d'afficher la table de multiplication d'un entier donné **n** .

Exemple : pour n=8

Table de multiplication pour : 8
1 x 8 = 8
2 x 8 = 16
3 x 8 = 24
...
10 x 8 = 80

Corrigé :

```
program multiplication;
uses wincrt;
var i,n: integer;
begin
Writeln('Donner un entier='); readln(n);

Writeln('Table de multiplication pour ',n);
for i:=1 to 10 do
writeln(i ,' x ', n, '= ', i*n);

end.
```

Exercice 5 :

Ecrire un programme qui permet d'afficher un nombre x à la puissance n. (avec x et n à saisir)

Exemple : X=5 n=3 donne $p=x^n=5^3=5*5*5$

Corrigé :

```
program puissance;
uses wincrt;

var  n,x,p,i:integer;
begin
write ('donner x= '); readln(x);
write ('donner n= '); readln(n);

p:=1;
for i:=1 to n do
p:=p*x;

writeln(x , ' à la puissace ', n ,' = ', p);
end.
```

Exercice7 :
Ecrire un programme qui permet de saisir un mot en majuscule puis le convertir en minuscule sauf le premier caractère puis l'affiche.

Corrigé :

0) Début conversion
1) Ecrire("Donner un mot en majuscule "), lire(mot)
2) Pour i de 2 à long(mot) faire
   Mot[i]←chr(ord(mot[i])+32)
   Finpour
3) Ecrire("Le mot après conversion", mot)
4) Fin Conversion

T.D.O

Objet	Type/Nature	Rôle
Mot	Chaine	Chaine à saisir
i	entier	compteur

```
Program conversion ;
Uses wincrt ;
Var  Mot : string ;
     I :integer ;
Begin
Writeln('Donner un mot en majuscule');
Readln(mot) ;
For i :=2 to length(mot) do
  Mot[i] :=chr( ord(mot[i])+32) ;
Writeln('Le mot après conversion=', mot) ;
End.
```

Remarque :
Pour ne convertir que les lettres majuscules, on pourra modifier le programme comme suit :

```
For i :=2 to length(mot) do
  If mot[i] in ['A'..'Z'] then Mot[i] :=chr( ord(mot[i])+32) ;
```

Exercice8 :
Ecrire un programme qui permet de saisir une chaine Ch puis une lettre L ensuite calcul et affiche le nombre d'occurrence de L dans Ch

Corrigé :

```
Program occurrence ;
uses wincrt ;
var   i, occ:integer ;   ch:string ;  L:char ;
begin
writeln('Donner un mot') ;   readln(Ch) ;
writeln('donner une lettre') ;  readln(L) ;
occ:=0 ;
for i:=1 to length(Ch) do
     if ch[i]=L then occ:= occ+1 ;

writeln('Le nombre d''occurrence est ', occ) ;
end.
```

Exercice 14 :

Écrire un programme qui permet de vérifier si un entier nb est premier.

NB : Un nombre premier est un entier naturel qui admet exactement deux diviseurs distincts entiers et positifs (qui sont 1 et lui-même). Cette définition exclut 1 et 0.

Exemples : Les nombres premiers inférieurs à 100 sont :

2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89 et 97.

Algorithme :

0) Début nombre_premier

1) Écrire("Donner un entier ") , lire(n)

2) [nb ← 0] **pour i de 1 à n faire**

Si n mod i=0 Alors nb ← nb+1

FinSi

FinPour

3) Si nb=2 alors Écrire("Premier")

sinon Écrire("Non premier")

FinSi

4) Fin nombre_premier

T.D.O

Objets	Type/nature	Rôle
i,	Entier	Compteur
nb	Entier	Nombre de diviseurs de n
n	Entier	Nombre à vérifier

Pascal :

```
Program nombre_premier ;
uses wincrt ;
var   i , n , nb : integer ;
begin
Writeln('Donner un entier = ' ) ;
Readln(n) ;
nb:= 0 ;
for i:=1 to n do
  if n mod i=0 then nb:= nb+1 ;

if nb=2 then  Writeln('Premier')
else  Write('Non premier') ;

end.
```

Exercice 15 :

Écrire un programme qui permet d'afficher tous les entiers premier entre 1 et 100 .

Pascal :

```
program nombre_premier2 ;
uses wincrt ;
var  i,nb,n : integer ;
begin
    for n:=1 to 100 do
      begin
           nb:= 0 ;
           for i:=1 to n do
            if n mod i=0 then nb:= nb+1 ;

           if nb=2 then  Writeln(n) ;
       end ;
end.
```

Rappel :

Remplissage d'un tableau :	Affichage des éléments d'un tableau :
pour i de 1 à n faire Écrire("donner l'élément n° ", i), lire (T[i]) FinPour	pour i de 1 à n faire Écrire(" l'élément n° ", i, "est",T[i]) FinPour

Déclaration d'un tableau en pascal :

```
Var t : array [1..20] of integer ;
```

Ou

```
Type
     Tab=array[1..20]of integer;
Var
      t : tab;
```

Exercice 9 :
Remplissage et affichage d'un tableau d'entiers de taille n à saisir.

Corrigé :

```
program tableau;
uses wincrt;
var t:array[1..20]of integer;
I,n : integer;
Begin
Writeln('Donner la taille du tableau') ;
Readln(n) ;
For i :=1 to n do
  Begin
  Writeln('Donner l''élément n°',i) ;
  Readln(t[i]) ;
  End;
For i :=1 to n do
Writeln('l''élément n°', i , 'est', t[i] ) ;
End.
```

Exercice 10 :

Écrire un programme qui permet de remplir un tableau T par n entiers puis détermine le maximum de ce tableau.

Corrigé :

Analyse :

Nom : Maximum
Résultat=Ecrire("Le maximum est",max)
Max=[max ← T[1]] Pour i de 2 à n faire
Si T[i]>max alors max ← T[i]
FinSi
FinPour
T= Pour i de 1 à n faire
T[i]=Donnée("Donner l'entier n°",i)
FinPour
n=Donnée("Donner le nombre d'entiers : ")
Fin Maximum

T.D.O

Objet	Type/Nature	Rôle
T	Tableau de 20 entiers	Tableau d'entiers
I	Entier	Compteur
max	entier	Le max du tableau

Algorithme

0) Début maximum
1) Ecrire("Donner le nombre d'entiers : ") , lire(n)
2) Pour i de 1 à n faire
Ecrire("Donner l'entier n°", i), lire(t[i])
FinPour
3) max ← T[1]
4) Pour i de 2 à n faire
si T[i] >max alors max ← T[i]
FinSi
FinPour
5) Ecrire("Le maximum est ", max)
6) Fin Maximum

Programme Pascal :

```
program maximum ;
uses wincrt ;
type
tab=Array [1..20] of integer ;
var
t:tab ;
n,i,max : integer ;
begin
writeln('Donner le nombre d''entiers :') ;
readln(n) ;

for i:=1 to n do
begin
writeln('Donner l''entier n°', i) ;
readln( T[i] ) ;
end ;
max:=T[1] ;
for i:=2 to n do
  if t[i]>max then max:=T[i] ;
writeln('Le maximum est ', max) ;
end.
```

Activité1 :

Ajouter un contrôle de saisie sur n (la taille du tableau) pour que la valeur de n soit dans l'intervalle [1..20]

Corrigé :

répéter Ecrire("Donner le nombre d"entiers : ") Lire(n) jusqu'à n dans [1..20]	en Pascal : **repeat** writeln('Donner le nombre d''entiers :') ; readln(n) ; **until** n in [1..20] ;

Leçon 2

Structures de contrôle itératives à conditions d'arrêt

I-Définition itérative à condition d'arrêt :

Un résultat a une définition itérative à condition d'arrêt s'il est la répétition d'une suite d'instruction et l'arrêt est géré par une condition.

1- La structure Répéter ...Jusqu'a :

Algorithme	Pascal
.....[Init] **Répéter** Instruction 1 Instruction 2 .... Instruction N **Traitement à Répéter** **Jusqu'à** (condition d'arrêt)	.... ; {Init} **Repeat** Instruction 1; Instruction 2; .... Instruction N; **Until** (condition d'arrêt);

Remarques:

- ✓ S'il y a un éventuel compteur, il faut l'initialiser avant la boucle; de même on doit assurer son avancement au sein de la boucle.
- ✓ Le traitement répétitif de la boucle répéter peut s'exécuter 1 ou n fois (n≥2).
- ✓ La condition à vérifier à chaque fois est considérée comme une condition de sortie car elle nous permet de quitter la boucle.
- ✓ Même si le traitement répétitif est composé de plusieurs instructions, on a jamais besoin des expressions Begin et End.
- ✓ La boucle répéter est utilisée entre autres dans le contrôle des données saisies.

*Les problèmes récurrents : voir exemple

Activité 1:

Écrire un programme qui permet de saisir une moyenne entre 0 et 20.

Algorithme :

0) Début saisie
1) **Répéter**
Ecrire("Donner une moyenne entre 0 et 20 "), lire(moy)
Jusqu'a (moy>=0) et (moy<=20)
2) Ecrire("La moyenne est ", moy)
3) Fin saisie

T.D.O

Objet	Type/Nature	Rôle
Moy	Réel	Moyenne à saisir

Programme Pascal :

```
Program saisie ;
Uses wincrt ;
Var moy :real ;
Begin
Repeat
writeln('Donner une moyenne entre 0 et 20 ');
readln(moy) ;
Until (moy>=0) AND (moy<=20) ;
Writeln('La moyenne est ',moy :5:2);
End.
```

2- La boucle Tant que :

Algorithme	Pascal

.....[Init]
Tant que (condition d'entrée) **Faire**
Instruction 1
Instruction 2
....
Instruction N
Fin TantQue
(Instruction 1 … Instruction N : **Traitement à Répéter**)
	.... ; {Init}
	While (Condition) **Do**
	Begin
	Instruction 1;
	Instruction 2;
	....
	Instruction N;
	End;

Remarques:

- ✓ Le traitement répétitif de la boucle Tant que peut s'exécuter 0 ou n fois. (0 fois dès le début si la condition n'est pas vérifiée).
- ✓ La condition à vérifier à chaque fois est considérée comme une **condition d'entrée** car elle nous permet **d'accéder au corps de la boucle**.
- ✓ Lorsque le traitement répétitif est composé de plusieurs instructions, les expressions Begin et End sont nécessaires.
- ✓ **Condition d'entrée = NON (Condition de sortie)**

Activité 2:

Écrire un programme qui permet de calculer le pgcd de 2 entiers a et b par la méthode de différence.

Exemple : a=15 b =27

pgcd(15,27) =pgcd(15 , 27-15)=pgcd(15 , 12)
=pgcd(15-12 ,12)=pgcd(3 , 12)
=pgcd(3 , 12-3)=pgcd(3 , 9)
=pgcd(3 , 9-3)=pgcd(3 , 6)
=pgcd(3 , 6-3)=pgcd(3 , 3) → a=b donc pgcd(a,b)=a=b=3

Algorithme :

0) Début pgcd
1) Ecrire("Donner a="),lire(a)
2) Ecrire("Donner b="),lire(b)
3) Tant que a< >b faire
Si a>b Alors a ← a-b
Sinon b ← b-a
FinSi
FinTantque
4) Ecrire("Le pgcd est ", a)
5) Fin pgcd

T.D.O

Objets	Type/nature	Rôle
a,b	entier	Entiers à saisir

En Pascal :

```
program pgcd ;
uses wincrt ;
var    a , b : integer ;
begin
 write('Donner a= '); readln(a);
 write('Donner b= '); readln(b) ;
     while a<>b do
         begin
         if a>b then a:=a-b
                    else b:=b-a ;
         end ;

writeln('Le pgcd est ', a) ;
end.
```

Applications : (Série)

Exercice 13 :

Ecrire un programme qui permet de saisir un mot composé de 10 caractères au maximum puis de vérifier si ce mot est palindrome (se lit dans les deux sens)

Exemples : ELLE, RADAR, REVER, DVD,...

Analyse :

Nom: Palindrome
Résultat= Si (pal=mot) alors écrire("palindrome ")
sinon Ecrire(" Non palindrome ")
FinSi
pal= [pal ← ""]Pour i de long(mot) à 1 (pas=-1) faire
pal:=pal+mot[i]
FinPour
mot=Répéter
mot=Donnée("Donner un mot ")
Jusqu'a (long(mot)<=10)
Fin Palindrome

T.D.O

Objets	Type/nature
i mot, pal	Entier chaîne de caractères

Algorithme :

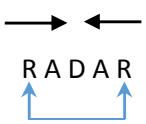
0) Début palindrome
1) Répéter
Écrire("Donner un mot "), lire(mot)
Jusqu'a (long(mot)<=10)
2) pal ← ""
3) Pour i de long(mot) à 1 (pas=-1) faire
pal ← pal+mot[i]
FinPour
4) Si (pal=mot) alors écrire("palindrome ")
sinon Ecrire(" Non palindrome ")
FinSi
5) Fin Palindrome

Traduction en Pascal :

```
Program plindrome ;
uses wincrt ;
var  i:integer ;
     mot,pal : string ;
begin
repeat
Write('Donner un mot ') ;
Readln(mot) ;
until (length(mot)<=10) ;
pal:= '' ;
For i:=length(mot) DownTo 1 do
pal:=pal+mot[i] ;

if (pal=mot) then
     writeln('palindrome ')
     else
   writeln(' Non palindrome ') ;
End.
```

2ème méthode :



0) Début palindrome2
1) Répéter
Écrire("Donner un mot "), lire(mot)
Jusqu'a (long(mot) dans [2..10])
2) L ← long(mot)
3) i ← 0
4) p ← vrai
5) **Répéter**
i ← i+1
si mot[i]<>mot[l-i+1] alors p ← faux Finsi
jusqu'à (i=L div 2) ou (p=faux)
6) Si (p=vrai) alors écrire("palindrome ") sinon Écrire(" Non palindrome ") FinSi
7) Fin Palindrôme2

T.D.O

Objet	Type/Nature	Rôle
Mot	Chaine	Chaine à saisir
L	Entier	Longueur du mot
i	Entier	Compteur
p	Booléen	Indiquer si la chaine est palindrome ou non

3ème méthode :

```
program palindrome3;
uses wincrt;
var
     mot:String;
     L, i:integer;
Begin
Repeat
   write('Donner un mot: ');
   readln(mot);
until length(mot)in [2..10];

L:=length(mot);
i:=1;
while (mot[i]=mot[L-i+1]) and (i<>L div 2) do
i:=i+1;

if  (mot[i]=mot[L-i+1]) then writeln('Palindrome')
 else writeln('Non palindrome');
end.
```

Exercice 6 :

Ecrire un programme pascal pour chaque figure suivante : pour n=5

```
*********  *        *          *
*       *  *       **         ***
*       *  *      ***        *****
*       *  *     ****       *******
*********  *    *****     *********
```

Corrigé :

```
program cadre;
uses wincrt;
var  i,n:integer;
begin
write('donner le nombre de lignes =');
readln(n);
for i:=1 to n do
if (i=1) or (i=n) then
writeln('**********')
else
writeln('*        *');
end.
```

```
program triangle;
uses wincrt;
var  I,j,n:integer;
begin
write('donner le nombre de lignes =');
readln(n);
for i:=1 to n do
begin
     for j:=1 to i do
        write('*');
   writeln;
end;

end.
```

```
program isocele;
uses wincrt;
var  I,j,k,n:integer;
begin
write('donner le nombre de lignes =');
readln(n);
for i:=1 to n do
begin
for j:=1 to (n-i) do
   write(' ');
for k:=1 to (2*i)-1 do
   write('*');
writeln;
end;
end.
```

3- Cas général (boucle pour):

Il y a des fois ou le compteur entre dans le calcul fait par le module à répéter ; en plus les opérations de calcul exigent des valeurs non entières et progressant avec un pas p non entier.

L'astuce consiste à chercher par division entière le nombre d'itération à accomplir et avec une expression généralement linaire revenir au compteur dont on a besoin.

Cas général Algorithme	Pascal

……..[iinit]
Pour i **de** d **à** f (pas=p) faire
Instruction 1
Instruction 2
….
Instruction m } **Traitement à Répéter**
FinPour
	……. ; {Init}
	n:=1+round((f-d)/p);
	FOR I:=1 **TO** n **DO**
	Begin
	c :=i * p ;
	Instruction_1;
	Instruction_2;
	…….;
	Instruction_m;
	End;

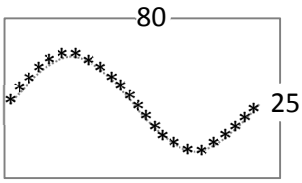
Si p est positif, le parcours est ascendant et si p est négatif, le parcours est descendant.

Le nombre de répétition est **n=1+((E((f-d)/p)** et dans ce cas le compteur effectif est **c =i*p**

Remarques : n est toujours positif, c'est le signe de p qui détermine le compteur c.

Voir >Exercice corrigé page 114

Exercice :



On se propose de dessiner la courbe de sin(t) par des " * " et pour qu'il soit visible à l'écran on va utiliser les équations suivantes : x(t)=arrondi(t*10) y(t)=12 - arrondi(sin(t)*10)

Avec t un angle en radian parcourant l'intervalle [0..2π] avec un pas=0.1

Soit l'analyse suivante :

Nom: Courbe_sin
Résultat= courbe
Courbe=[ ] **pour** t de 0 **à** 2π (pas = p) **faire**
Cx ← arrondi(t*10)
Cy←12 - arrondi(sin(t)*10)
Écrire(TAB(cx,cy), "* ")
FinPour
t = Compteur
p =0.1(constante)
Fin courbe_sin

T.D.O

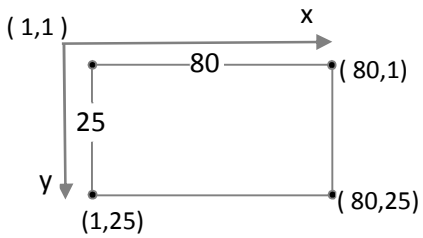
Objets	Type/Nature
i,n,cx,cy	Entier
t	réel
pi	constante=3.14
p	constante=0.1

Traduire ce programme en pascal en effectuant les transformations qu'il faut :

```
program courbe_sin;
 uses wincrt;
 const   pi=3.14;
         p=0.1;
 var i,n,cx,cy:integer;
     t:real;
 begin
 n:=.....................
 t:=.....................
 for i:=1 to n do
        begin
              .....................
              cx:=.....................;
              cy:=.....................;
              .....................
              write('*');
        end;
 end.
```

Remarque :

L'écran d'affichage est de dimension suivante :



gotoxy(x,y)

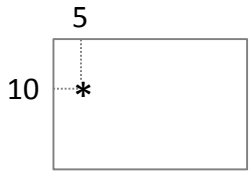
amène le curseur aux cordonnées spécifiques au sein de l'écran virtuel.

avec x et y entiers.

Exemple :

gotoxy(5,10) ;

write('*') ;



Corrigé :

```
program courbe_sin;
 uses wincrt;
 const   pi=3.14;
         p=0.1;
 var i,n,cx,cy:integer;
      t:real;
 begin
 n:=1+round(2*pi/p) ;
 t:=-p ;
 for i:=1 to n do
        begin
           t:=t+p ;
           cx:=round(t*10);
           cy:=12-round(sin(t)*10);
        gotoxy(cx,cy) ;
           write('*');
        end;
 end.
```

II-Les itérations complètes récurrentes :

Le résultat se forme au fur et à mesure et à une étape donnée, il dépend d'un certain nombre de résultats précédents. Si une relation lie deux éléments successifs (récurrence d'ordre 1) si elle lie trois éléments successifs (récurrence d'ordre 2). (voir exemple factoriel)