

Chapitre 7:

Les instructions de contrôle

L'instruction de contrôle if

- Nous commencerons par apprendre à utiliser des structures de contrôle conditionnelles, aussi appelées alternatives, du fait que leur forme généralisée permet de faire un choix entre plusieurs portions de code à exécuter.
- La forme la plus simple d'une structure conditionnelle est d'exécuter quelque chose dans le cas où une condition est vraie.
- Pour cela, on utilisera la structure de contrôle **if**, dont la syntaxe est la suivante :

L'instruction de contrôle if

Syntaxe

```
if (expression)
{
    Instruction1;
    Instruction2;
}
```

```
if (expression)
{
    Instruction1;
    Instruction2;
}
else
{
    Instruction3;
    Instruction4;
}
```

N.B. Bien sur on peut faire des tests (**if**) imbriqués

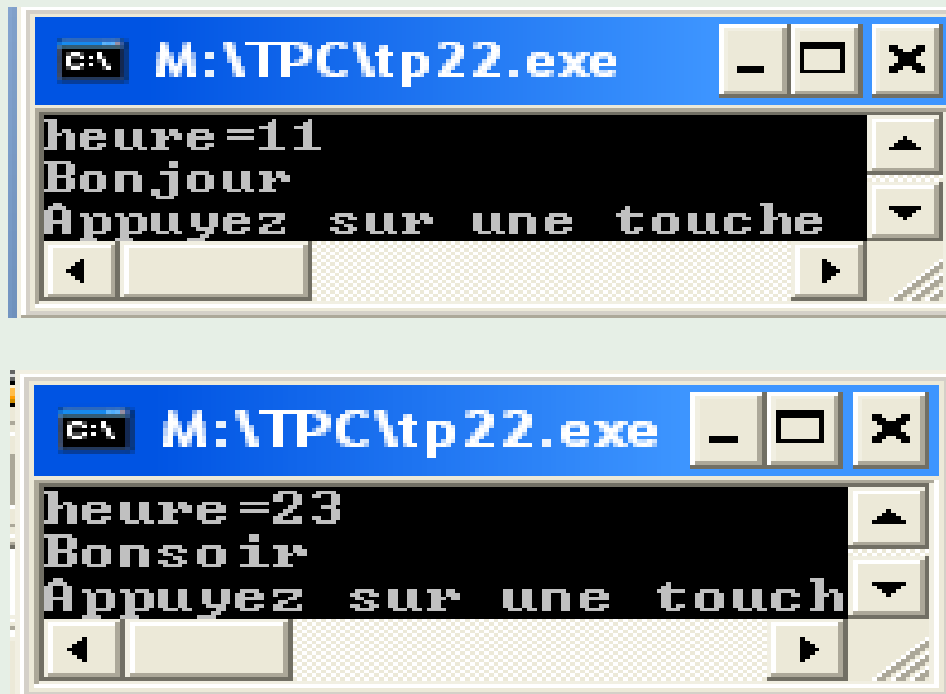
L'instruction de contrôle if

Exemple

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    int heure;
    printf("heure=");
    scanf("%d", &heure);
    if( heure <= 19)
    {
        printf("Bonjour\n");
    }
    else
    {
        printf("Bonsoir\n");
    }
    system("PAUSE");
    return 0;
}
```

L'instruction de contrôle if

Exemple



L'instruction de contrôle if

Remarques

1. La syntaxe de l'instruction `if` n'impose aucun point virgule.
2. Dans le cas d'imbrication des instructions `if`, un `else` se rapporte toujours au dernier `if` rencontré auquel un `else` n'a pas encore été attribué.
3. La condition doit être entre parenthèses.
4. Il est possible de définir plusieurs conditions à remplir avec les opérateurs `&&` et `||`.
5. La condition peut être une combinaison de plusieurs conditions.

Autre façon d'écriture du if..else

- Considérons l'instruction suivante :
if (expression)
instruction1;
else
instruction2;
- En langage C, il est possible, grâce à l'aide de **l'opérateur conditionnel**, de traduire cette instruction de la manière suivante:
expression ? instruction1 : instruction2;

Autre façon d'écriture du if..else

Exemple

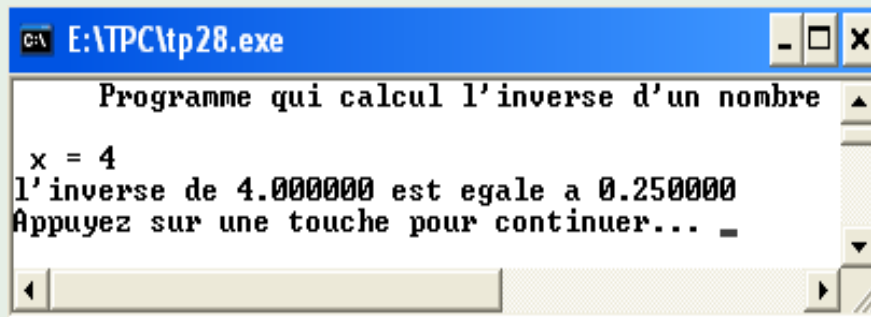
```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    float x;
    printf("    Programme qui calcul l'inverse d'un nombre \n\n ");
    printf("x = ");
    scanf("%f",&x);

    (x!=0) ? printf("l'inverse de %f est egale a %f\n",x,1/x) :
            puts("erreur division par zero");

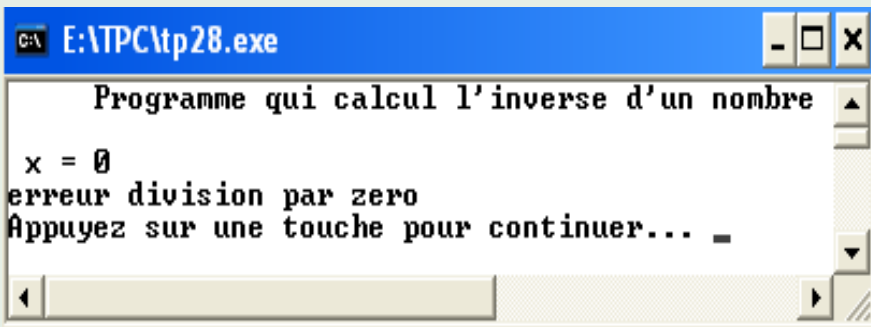
    /*if( x!=0)
    {
        printf("l'inverse de %f esi egale a %f\n",x,1/x);
    }
    else
    {
        puts("erreur division par zero");
    }*/
    system("PAUSE");
    return 0;
}
```


Autre façon d'écriture du if..else

Exemple



```
C:\ E:\TPC\tp28.exe
Programme qui calcul l'inverse d'un nombre
x = 4
l'inverse de 4.000000 est egale a 0.250000
Appuyez sur une touche pour continuer... _
```



```
C:\ E:\TPC\tp28.exe
Programme qui calcul l'inverse d'un nombre
x = 0
erreur division par zero
Appuyez sur une touche pour continuer... _
```

L'instruction de contrôle switch

- L'instruction **switch** permet de faire plusieurs tests de valeurs sur le contenu d'une même variable.
- Ce branchement conditionnel simplifie beaucoup le test de plusieurs valeurs d'une variable ou expression, car cette opération aurait compliquée avec des if imbriqués.
- La syntaxe de **switch** est la suivante:

L'instruction de contrôle switch

Type: char ou int

```
switch(expression)
{
    case valeur_1 : liste d'instruction_1 ;
                    break ;
    case valeur_2 : liste d'instruction_2 ;
                    break ;
    .
    .
    case valeur_n : liste d'instruction_n ;
                    break ;
    default :      liste d'instruction_Comp ;
}
```

Expression constante entière

L'instruction de contrôle switch

Exemple : système feu rouge

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    char feu;
    printf("      Programme qui gere le systeme feu rouge \n\n ");
    printf("      feu = r, v, ou j \n\n ");
    printf("feu=");
    feu=getchar();

    switch(feu)
    {
        case 'r': puts("STOP");
                 break ;
        case 'v': puts("AVANCEZ");
                 break ;
        case 'j': puts("ATTENTION");
                 break ;
        default : puts("Il faut taper r, v ou j !!!!?");
    }

    system("PAUSE");
    return 0;
}
```

Les boucles

- Les boucles sont des structures qui permettent d'exécuter plusieurs fois la même série d'instructions jusqu'à qu'une condition ne soit plus réalisable.
- En C, nous allons voir trois type de boucles:
 - ✓ la boucle **for** (pour),
 - ✓ la boucle **while** (tant que... faire...),
 - ✓ La boucle **do ... while** (répéter ... tant que).

La boucle for

- L'instruction **for** permet l'exécution répétitive d'une instruction ou d'un bloc d'instructions un nombre **bien déterminé** de fois et cela grâce à un **compteur**.
- Le **compteur** a une valeur de **départ**, une valeur **d'arrêt** et une **instruction** d'incrémentation ou décrémentation.
- La structure **for** est une boucle qui teste une condition avant d'exécuter les instructions. Ces instructions sont exécutées (répétées) tant que la condition est remplie (VRAI).

La boucle for

Syntaxe

```
for(expression_I ; expression_C ; expression_R)
{
    instruction_1 ;
    instruction_2 ;
    .....
}
```

- ✓ **expression_I** : Initialise le compteur.
- ✓ **expression_C**: Condition de bouclage.
- ✓ **expression_R**: Réinitialisation du compteur (incrémentement ou décrémentation).

La boucle for

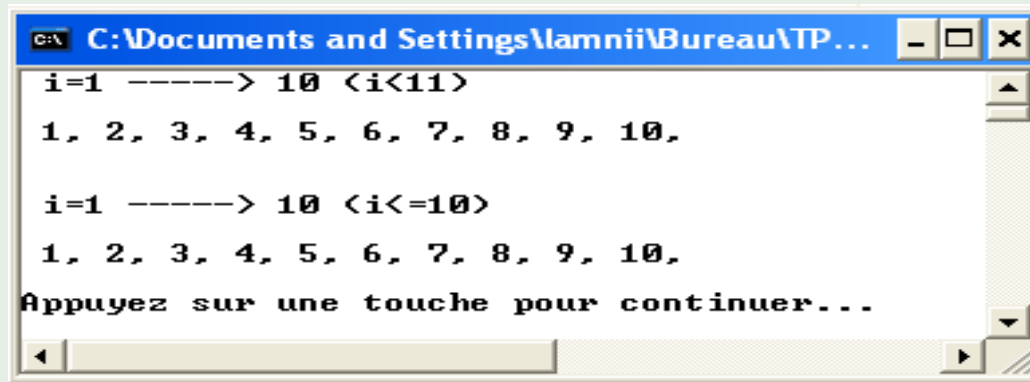
Exemple

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{ int i ;
printf(" i=1 -----> 10 (i<11)\n\n");
  for(i=1 ;i<11 ;i++ )
  {
    printf(" %d, ", i);
  }
printf("\n\n\n");
printf(" i=1 -----> 10 (i<=10)\n\n");
for(i=1 ;i<=10 ;i++ )
{
  printf(" %d, ", i);
}
printf("\n\n");

system("PAUSE");
return 0;
}
```


La boucle for

Exemple



```
C:\Documents and Settings\lamnii\Bureau\TP...
i=1 -----> 10 <i<11>
1, 2, 3, 4, 5, 6, 7, 8, 9, 10,

i=1 -----> 10 <i<=10>
1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
Appuyez sur une touche pour continuer...
```

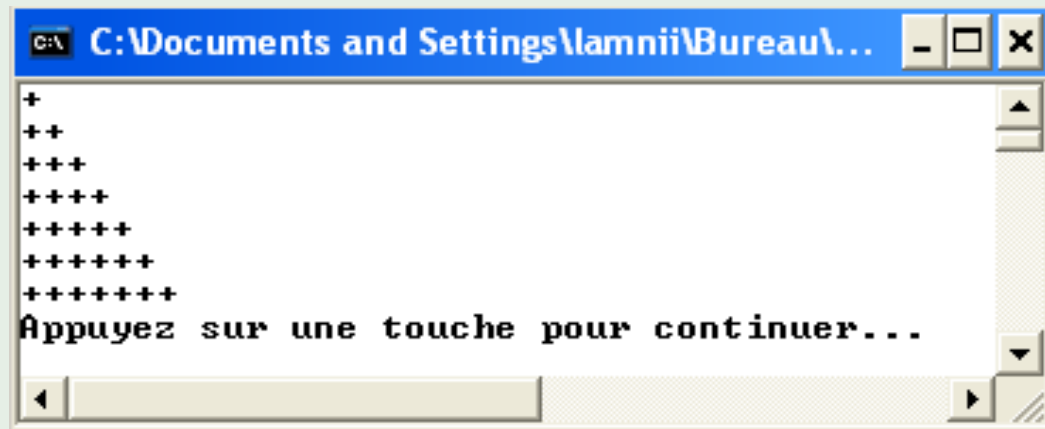
La boucle for

Exemple

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{ int i, j ;
  for(i=1 ; i<8 ; i++ )
  {
    for(j=1 ; j<=i ; j++ )
    {
      printf( "+" ) ;
    }
    printf( "\n" ) ;
  }
  system( "PAUSE" ) ;
  return 0 ;
}
```

La boucle for

Exemple



A screenshot of a Windows command prompt window. The title bar shows the path "C:\Documents and Settings\lamnii\Bureau\...". The command prompt displays a for loop that prints a pyramid of plus signs. The output is as follows:

```
+  
++  
+++  
++++  
+++++  
++++++  
+++++++  
+++++++  
Appuyez sur une touche pour continuer...
```

The window includes standard Windows controls (minimize, maximize, close) and a scrollbar on the right side.

La boucle for

Exemple

Programme qui calcul la somme:

$$som = \sum_{i=-2}^4 i^2$$

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main(int argc, char *argv[])
{ int i, som ;
  som=0;
  for(i=-2 ;i<=4 ;i++ )
  {
    som=som+(i*i);
  }
  printf("la somme est eegle a %d \n", som);
  system("PAUSE");
  return 0;
}
```

La boucle for

✓ Boucles infinie et boucles vide

```
for(    ;    ;    )  
{  
    printf("boucle infinie.\n");  
}
```

Ou bien

```
for(    ; -3 ;    )  
{  
    printf("boucle vide.\n");  
}
```

La boucle while

La structure **while** ou boucle **while** permet de faire répéter l'exécution d'instructions tant qu'une certaine condition est remplie (**vraie**). Avec la syntaxe:

```
while (expression)
{
    instruction_1 ;
    instruction_1 ;
    .....
    instruction_n ;
}
```

La boucle while

- Le mot clé **while** est suivi d'une expression entre parenthèses représentant la condition de la boucle (critère de bouclage). Viennent ensuite les instructions qui doivent être exécutées en fonction de cette condition. Si l'instruction est unique, les accolades sont facultatives.
- La portion: **while**(expression) est appelée en-tête de boucle. La ou les instructions venant après forment ce qu'on appelle le corps de la boucle.

La boucle while

Exemple

```
int a=3 ;  
while(a > 0)  
{  
    printf("%d \n", a) ;  
    a-- ;  
}
```



3
2
1

La boucle do...while

- Contrairement aux structures **for** et **while**, la boucle **do... while** teste sa condition après l'exécution des instructions du corps de la boucle.
- Syntaxe

```
do
{
    Instruction_1 ;
    Instruction_2 ;
    .....
} while( expression) ;
```

La boucle `do...while`

- La boucle `do ...while` ressemble à `while`, mais les instructions sont au moins exécutées une seule fois.
- Remarquez qu'il faut ajouter le point-virgule à la fin de la boucle.

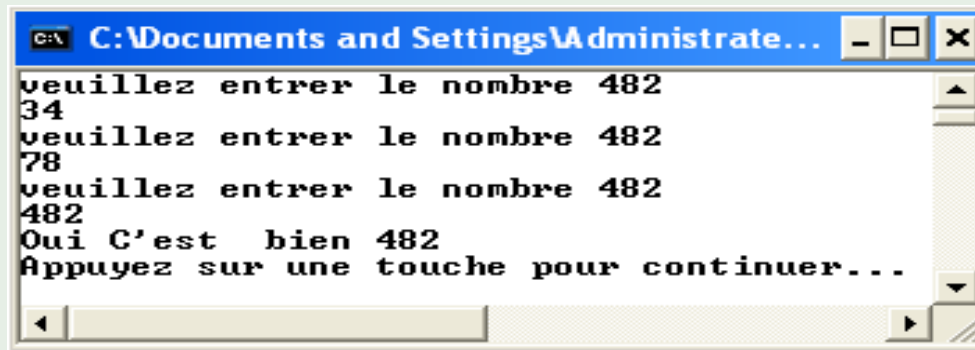
La boucle do...while

Exemple

```
#include <stdio.h>
int main(void)
{
    int a;
    do {
        puts("veuillez entrer le nombre 482");
        scanf("%d",&a);
    } while (a!=482);
    puts("Oui C'est bien 482");
    getch();
}
```

La boucle do..while

Exemple



```
C:\Documents and Settings\Administrate...  
veuillez entrer le nombre 482  
34  
veuillez entrer le nombre 482  
78  
veuillez entrer le nombre 482  
482  
Oui C'est bien 482  
Appuyez sur une touche pour continuer...
```

Instructions de branchement

- Les instructions de branchement transfèrent le contrôle du programme d'une instruction à une autre, cette dernière n'étant pas directement écrite après l'exécution précédemment effectuée. Nous examinerons trois instructions:
 - ✓ `break`
 - ✓ `continue`
 - ✓ `goto`
- Il existe encore l'instruction `return` que nous traiterons dans le chapitre des fonctions.

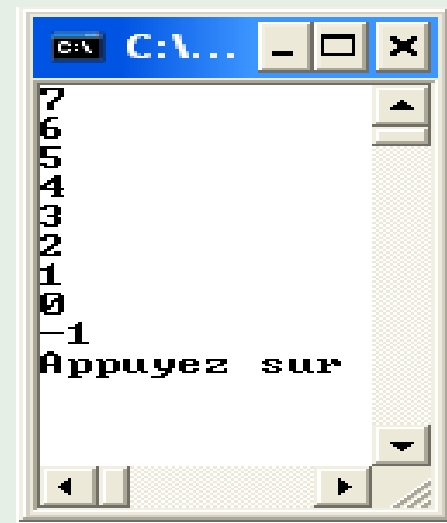
L'instruction break (interrompre)

- Elle ne peut s'utiliser qu'à l'intérieur d'une structure `for`, `while`, `do ... while` ou `switch`. Elle provoque l'arrêt avant de terminer l'exécution de toutes les instructions. Pour ce qui concerne l'instruction `switch`, nous avons déjà étudié ce mécanisme.
- Elle est limitée à un seul niveau d'imbrication.

L'instruction break (interrompre)

Exemple

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{ int a=7;
  do
  {
    if(a== -2) { break; }
    printf("%d\n",a);
    a--;
  }while (a!= -5);
  system("PAUSE");
  return 0;
}
```



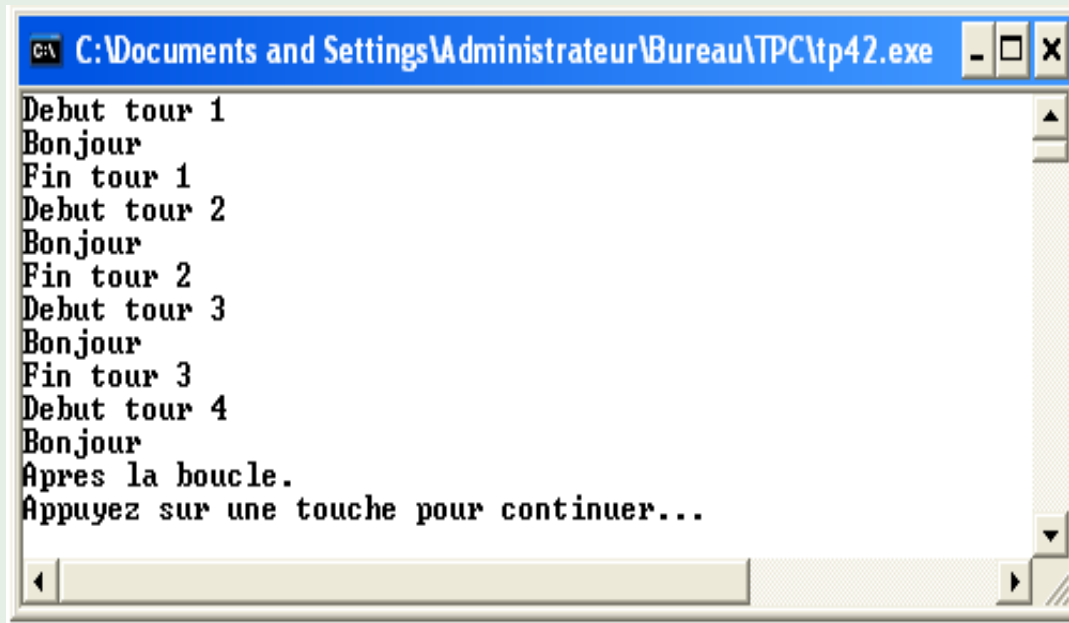
L'instruction break (interrompre)

Exemple

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    int i;
    for(i=1; i<=10; i++)
    {
        printf("Debut tour %d \n", i);
        printf("Bonjour\n");
        if(i==4)
        {
            break;
        }
        printf("Fin tour %d \n", i);
    }
    printf("Après la boucle.\n", i);
    system("PAUSE");
    return 0;
}
```


L'instruction break (interrompre)

Exemple



A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Documents and Settings\Administrateur\Bureau\TPC\tp42.exe. The window contains the following text:

```
Debut tour 1  
Bonjour  
Fin tour 1  
Debut tour 2  
Bonjour  
Fin tour 2  
Debut tour 3  
Bonjour  
Fin tour 3  
Debut tour 4  
Bonjour  
Après la boucle.  
Appuyez sur une touche pour continuer...
```

L'instruction continue (continuer)

- Cette instruction provoque le passage à la prochaine itération d'une boucle.
- Dans le cas d'un `white` ou `do ... white`, nous obtenons un saut vers l'évaluation du test de sortie de boucle.
- Dans le cas d'un `for` on passe à l'expression d'incrémentatation puis au test de bouclage.
- En cas de `boucles imbriquées`, permet uniquement de continuer la boucle la plus interne.

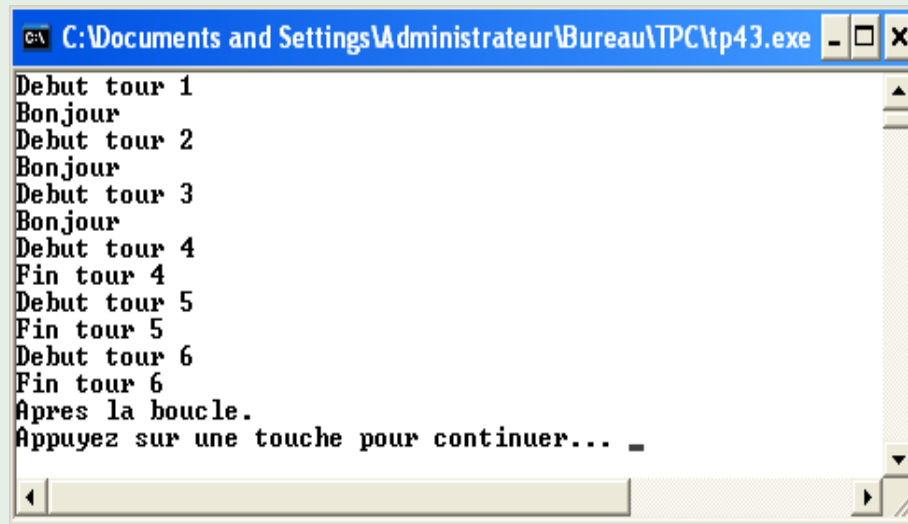
L'instruction continue (continuer)

Exemple

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{ int i;
  for (i=1; i<=6; i++)
  {
    printf("Debut tour %d \n", i);
    if (i<=3)
    {
      printf("Bonjour\n");
      continue;
    }
    printf("Fin tour %d \n", i);
  }
  printf("Après la boucle.\n", i);
  system("PAUSE");
  return 0;
}
```

L'instruction continue (continuer)

Exemple



```
C:\Documents and Settings\Administrateur\Bureau\TPC\tp43.exe
Debut tour 1
Bonjour
Debut tour 2
Bonjour
Debut tour 3
Bonjour
Debut tour 4
Fin tour 4
Debut tour 5
Fin tour 5
Debut tour 6
Fin tour 6
Apres la boucle.
Appuyez sur une touche pour continuer... _
```

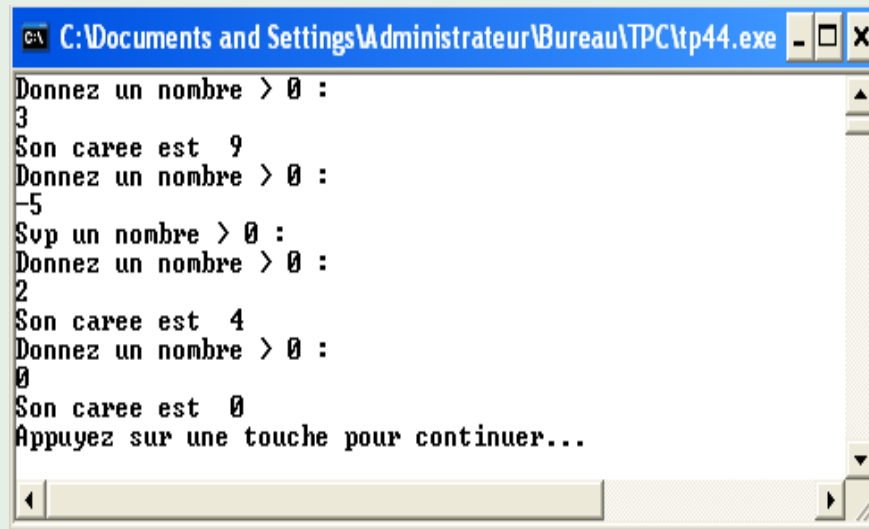
L'instruction continue (continuer)

Exemple

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main(int argc, char *argv[])
{ int n;
  do
  {
    printf("Donnez un nombre > 0 : \n");
    scanf("%d",&n);
    if(n<0)
    {
      printf("Svp un nombre > 0 : \n");
      continue;
    }
    printf("Son carree est  %d \n",n*n);
  }while (n);
  system("PAUSE");
  return 0;
}
```

L'instruction continue (continuer)

Exemple



```
C:\Documents and Settings\Administrateur\Bureau\TPC\tp44.exe
Donnez un nombre > 0 :
3
Son caree est 9
Donnez un nombre > 0 :
-5
Svp un nombre > 0 :
Donnez un nombre > 0 :
2
Son caree est 4
Donnez un nombre > 0 :
0
Son caree est 0
Appuyez sur une touche pour continuer...
```

Instruction goto (aller à)

- L'instruction **goto** provoque un saut à un endroit du programme repéré par une étiquette (label). Le programme continue alors à l'instruction qui se trouve à cet endroit-là. Voici la syntaxe de **goto**:

goto étiquette;

- **étiquette** peut être n'importe quel libellé par le langage (un mot par exemple). L'instruction identifier par l'**étiquette** doit avoir la forme syntaxique suivante.

Étiquette : instruction;

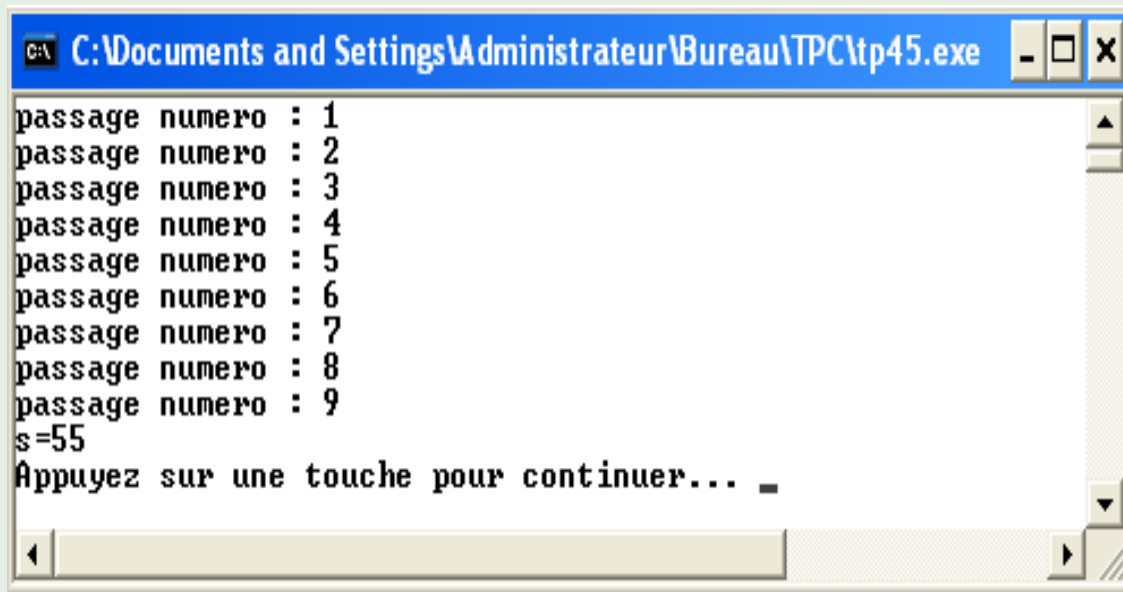
Instruction goto (aller à)

Exemple

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{ int a=1,s=0;
  Etiquette: s=s+a;
  if(a<10)
    { printf("passage numero : %d\n", a);
      a++;
      goto Etiquette;
    }
  printf("s=%d\n", s);
  system("PAUSE");
  return 0;
}
```


Instruction goto (aller à)

Exemple



```
C:\Documents and Settings\Administrateur\Bureau\TPC\tp45.exe
passage numero : 1
passage numero : 2
passage numero : 3
passage numero : 4
passage numero : 5
passage numero : 6
passage numero : 7
passage numero : 8
passage numero : 9
s=55
Appuyez sur une touche pour continuer...
```

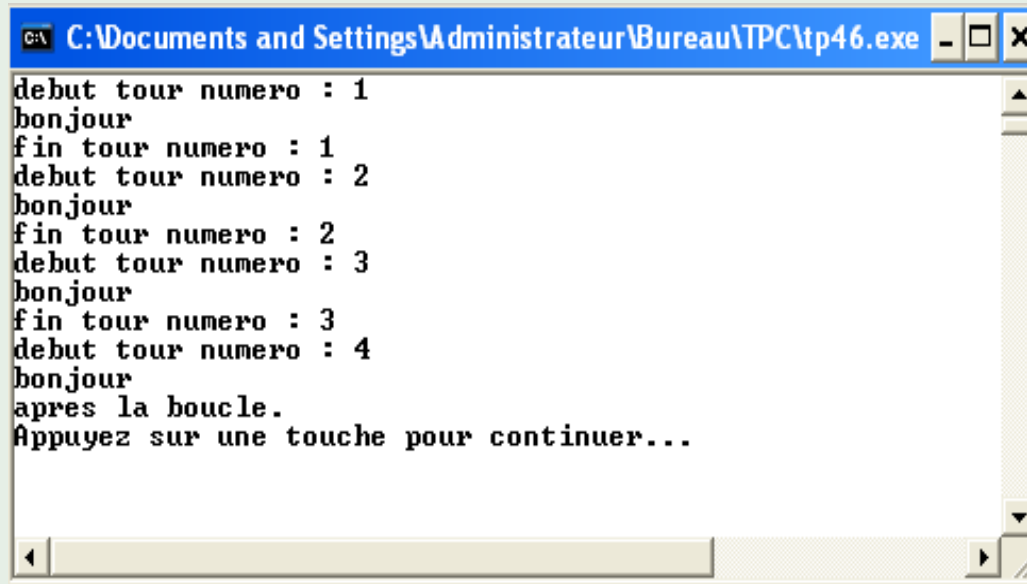
Instruction goto (aller à)

Exemple

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{ int i;
  for(i=1;i<=10;i++)
  { printf("debut tour numero : %d\n", i);
    printf("bonjour\n");
    if(i==4)
      goto sortie;
    printf("fin tour numero : %d\n", i);
  }
  sortie: printf("apres la boucle.\n");
  system("PAUSE");
  return 0;
}
```

Instruction goto (aller à)

Exemple



```
C:\Documents and Settings\Administrateur\Bureau\TPC\tp46.exe
debut tour numero : 1
bonjour
fin tour numero : 1
debut tour numero : 2
bonjour
fin tour numero : 2
debut tour numero : 3
bonjour
fin tour numero : 3
debut tour numero : 4
bonjour
apres la boucle.
Appuyez sur une touche pour continuer...
```