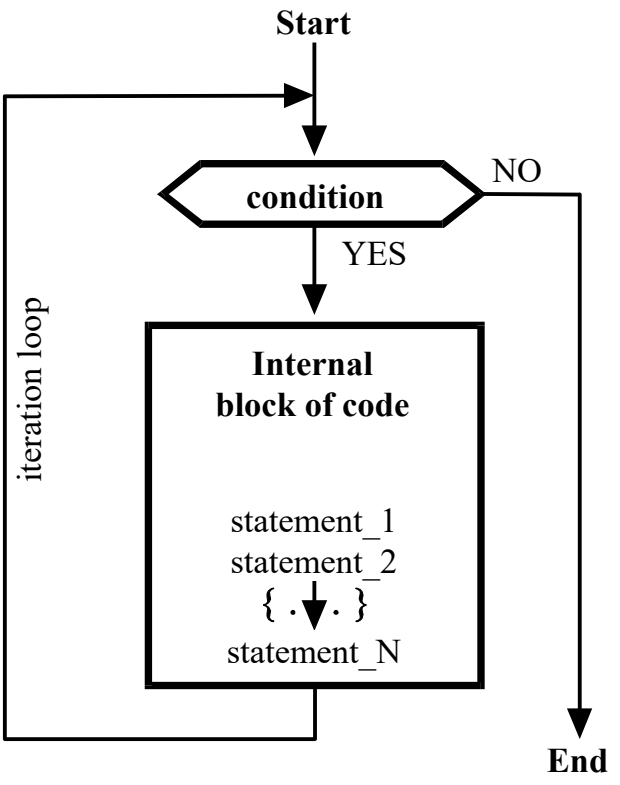
REPETITIVE STATEMENTS – LOOPS

- **while( )** loop




```
while ( condition )
    statement;
```

```
while ( condition )
 {
     statement_1;
     statement _2;
     . . .
     statement_N;
 }
```

The loop is executed as long as the value of the condition is **not zero**

Examples:

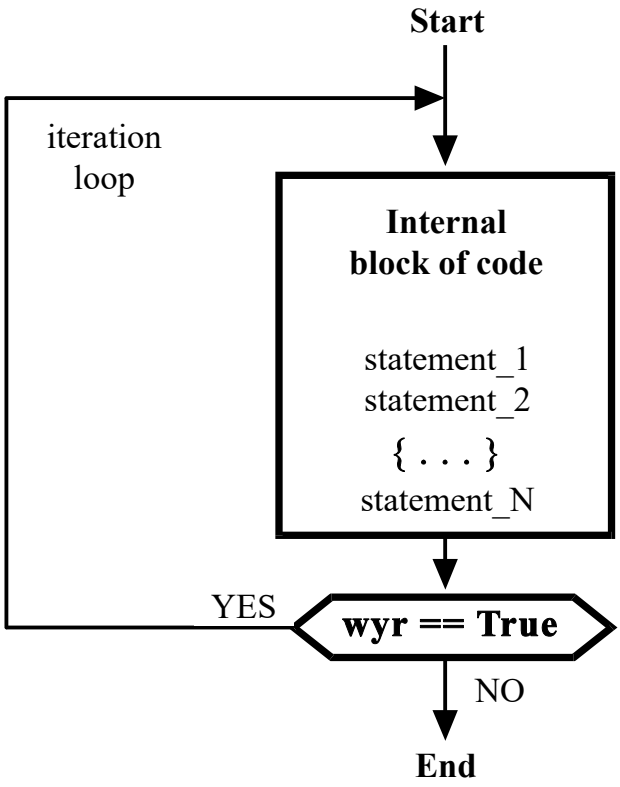
```
int i ;  // a loop displaying numbers 1,2,3 ...
i = 1;
while( i <=10 )
  {
    printf ( "%2d\n" , i );
    i = i + 1;
  }
```

```
int i = 1;  // 1, 2, 3, . . . in another notation
while( i<11 )
   printf ( "%2d\n" , i++ );
```

```
int i ;  // a loop displaying numbers 10,9,8
i = 10;
while( i != 0 )
  {
    printf ( "%2d\n" , i );
    i = i – 1;
  }
```

```
int i = 10;  //10, 9, 8, . . . in another notation
while( i )
   printf ( "%2d\n" , i –– );
```

- **do while( )** loop




```
do
    statement;
while ( condition );
```

```
do
    {
        statement_1;
        statement_2;
    . . .
        statement_N;
    }
while ( condition );
```

The loop is executed as long as the value of the condition is **not zero**

Examples:

```
int i ;  // a loop displaying numbers 1,2,3 ...
i = 1;
do
  {
    printf ( "%2d\n" , i );
    i = i + 1;
  }
while( i<=10);
```

```
int i = 1;  // 1, 2, 3, . . .  in another notation
do
   printf ( "%2d\n" , i );
while( ++i <11 );
```

```
int i ;  // a loop displaying numbers 10,9,8
i = 10;
do
  {
    printf ( "%2d\n" , i );
    i = i – 1;
  }
 while( i != 0 );
```

```
int i = 10;  // 10, 9, 8, . . . in another notation
do
   printf ( "%2d\n" , i );
while( – –i );
```

example 1: *Reading the keyboard keys until you press 'k' – with a **„while"** loop*

```
#include <stdio.h>                                   // implementation in C
#include <conio.h>                    // library <conio.h> containing function getch()

int main( void ) {
   char key = 'a';
   while( key != 'k' ) {
      printf( "\n press any key: " );
      key = getch( );
   }
   return 0;
}
```

example 2: *Reading the keyboard keys until you press ESC button – with a **„do while"** loop*

```
#include <stdio.h>                                   // implementation in C
#include <conio.h>

int main( void ) {
   char key;
   do {
      printf( " \n press any key: " );
      key = getche( );                          // the "e" echo variant of getch()
   } while(key != 27 );                          // 27 = the code of ESCape key
   return 0;
}
```

example 3: *Display a line constructed from 10 characters „minus"*

```
#include <stdio.h>                                   // implementation in C

int main( ) {
   int counter=0;  while( counter<10 ) {  printf( "-" );  counter++;  }
   return 0;
}
```

example 4: *Simulation of a dice: drawing numbers from range 1÷6, until hitting „six"*

```
#include <stdio.h>                                   // implementation in C
#include <stdlib.h>          // library containing functions „rand" i „srand"
#include <time.h>            // library containing function „time"

int main( )  {
   int  random_number;
   srand( time(0) );
   do {
        random_number = rand()%6 + 1;                        // simulate a dice roll
        printf("\n Selected random number: %d ", random_number );

   } while(random_number != 6 );
   printf( "\n\n Done! Press ENTER to finish the program" );  getchar();
   return 0;
}
```

- Loop **for(; ;)**

for(*initialization* **;** *condition* **;** *modification* **)**
    internal_statement ;

is equivalent of construction:

```
initialization ;
while( condition )
  {
    internal_statement ;
    modification ;
  }
```

```
int i ;
i = 10;
while( i != 0 )
  {
    printf ( "%2d\n" , i );
    i = i – 1;
  }
```

```
int i ;
for( i = 10; i != 0 ; i = i – 1 )
  printf( "%2d\n" , i );
```

or another notation:

```
for(int i = 10; i ; i––) printf("%2d\n" , i );
```

example 5: *Display a bar from 80 characters '#'*

```
#include <stdio.h>
int main( ) {
   for( int i=0 ; i<80 ; i++ )   printf( "# " );
   return 0;
}
```

example 6: *Program printing the table of selected ASCII codes*

```
#include <stdio.h>
int main( ) {
   for( int code=32; code <256; code++ )
      printf( "%4d = %c" , code , code );
   return 0;
}
```

example 7: *Primitive calculator for adding a numbers typed on the keyboard*

```
#include <stdio.h>
int main( ) {
   double sum=0, number;
   while( scanf( "%lf" , & number ) )
      printf( "\t%.2f\n" , sum+= number );
   return 0;
}
```

example 8: *Calculate the sum of **N** numbers entered from the console*

```
#include <iostream.h>                                   // implementation in C++
int main( ) {
   int i, N;
   float number, sum;
   cout << "How many numbers would you like to sum N = ";
   cin >> N;
   sum=0;
   for( i=1; i<=N; i++ )
      {
         cout << "Enter " << i << " number:  " ;
         cin >> number ;
         sum = sum+number;
      }
   cout << endl << endl<<"The sum of " << N << " entered numbers is: " << sum ;
   cin.ignore( cin.rdbuf()->in_avail() );   // another form of flushing the keyboard buffer
   cin.get( );
   return 0;
}
```

example 9: *Drawing on the screen, rectangular frame with given coordinates*

```
#include <stdio.h>                                      // implementation in C
      #include <windows.h>          // library available only in MS Windows operating system
      #include <conio.h>            // nonstandard conio library (console input / output) DOS / Windows

      void gotoxy(int x, int y)     // example implementation of the "cursor positioning" in the text console
      {
         HANDLE hConsole = GetStdHandle(STD_OUTPUT_HANDLE);
         COORD pos;    pos.X = x;     pos.Y = y;
         SetConsoleCursorPosition(hConsole, pos);
      }
int main()  {
   int x1, y1, x2, y2, x, y;
   printf("Podaj wspolrzedne 1 naroznika \n\r X1 = ");          scanf("%d", &x1);
   printf("\n Y1 = ");                                         scanf("%d", &y1);
   printf("\nPodaj wspolrzedne 2 naroznika \n\r X2 = ");       scanf("%d", &x2);
   printf("\n Y2 = ");                                         scanf("%d", &y2);
   for (gotoxy(x1, y1), x = x1; x <= x2; x++ )          // drawing the top border
      printf("-");
   for (gotoxy(x1, y2), x = x1; x <= x2; x++ )          // drawing the bottom border
      printf("-");
   for ( y = y1 + 1; y < y2; y++ )
   {
      gotoxy(x1, y);   printf("|");                          // drawing the left border
      gotoxy(x2, y);   printf("|");                          // drawing the left border
   }
   while ( kbhit() )  getch();  // the kbhit () checks, if there is anything in the keyboard buffer
   getch();                     // getch ()reads and removes one character from the keyboard buffer
   return 0;
}
```

example 10: *Displaying all numbers from range 1÷1000 divisible by 13*

```
#include <stdio.h>                                   // implementation in C
int main( ) {
   for( int number=13; number<=1000; number+=13 )
      printf( "%3d \n " , number );
   return 0;
}
```

example 11: *Display all pairs of numbers (2-tuples) x,y∈[1,100] satisfying the equation:* $x^2 + y^2 < 500$

```
#include <stdio.h>
int main( ) {
   for( int x=1; x<100; x++ )
      for( int y=1; y<100; y++ )
         if( x*x + y*y < 500 )
            printf( "\n x=%d  y=%d" , x, y);
   return 0;
}
```

example 12: *The program which classifies keystrokes*

```
#include <stdio.h>                                   // implementation in C
#include <conio.h>
#define  ESC  27                                     // definition of a keyboard code for «Escape»

int main(void) {
      int key=0;
      while( key != ESC )
         {
            printf( "\n\nPress any key (ESC->to finish): " );
            key = getch();
            if( 'a'<=key  &&  key<='z' )
               printf( "=> This is lowercase letter " );
            else if( 'A'<=key  &&  key<='Z' )
               printf( "=> This is uppercase letter " );
            else if( '0'<=key  &&  key<='9' )
               printf( "=> This is digit " );
            else if( key == 13 )
               printf( "=> This is ENTER key " );
            else if( key == ' ' )
               printf( "=> This is space key ");
            else
               printf( "=> Unrecognized key ");
         }
      return 0;
   }
```

```
#include <stdio.h>          // Example 13 : program which classifies "function" keys
#include <conio.h>
#include "my_key_definition.h"          // including the file with private definitions
int main( void )
  {
    int key, key2;
    do
      {
        printf( "\n\n Press any key:  " );
        key = getch( );
        switch( key )
          {
            case ENTER  :  printf( "This is ENTER" );    break;
            case ESC    :  printf( "This is ESCAPE" );   break;
            case PREFIX :  // is first code equal to prefix (224 or 0) ?
                  key2 = getch( );
                  switch( key2 )
                  {
                    case DELETE     :  printf( "Delete" );      break;
                    case UP_ARROW   :  printf( "Up arrow" );    break;
                    case DOWN_ARROW :  printf( "Down arrow" );  break;
                  }
                  break;
            case BACKSPACE :  printf( "This is BACKSPACE" );    break;
            default  : printf( "Another – unrecognized key " );   break;
          }
      }
    while( key != ESC );
    return 0;
  }
```

// File **«my_key_definitions.h»** *which contains my definitions of codes for selected keys*

```
#ifndef   MY_KEY_DEFINITIONS
    #define  MY_KEY_DEFINITIONS
    #define  PREFIX          224
    // single code keys
    #define  ESC              27
    #define  ENTER            13
    #define  BACKSPACE         8
    // additional "function" keys – encoded with two numbers
    #define  DELETE           83          // 224, 83
    #define  UP_ARROW         72          // 224, 72
    #define  DOWN_ARROW       80          // 224, 80
    #define  LEFT_ARROW       75          // 224, 75
    #define  RIGHT_ARROW      77          // 224, 77
    #define  HOME             71          // 224, 71
    #define  END              79          // 224, 79
#endif
```