ITERATION OR LOOPING CONTROL STRUCTURES

One of the fundamental properties of a computer is its ability to repetitively execute a set of statements. These *looping* capabilities enable you to develop concise programs containing repetitive processes that could otherwise require thousands or even millions of program statements to perform. The C programming language contains three different program statements for program looping. They are known as the "**for statement"**, the "**while" statement**, and the "**do while" statement.**

The '**while'** loop

The while statement is used to carry out looping instructions where a group of instructions are executed repeatedly until some conditions are satisfied.

General form

while (*expression*)
program statement

The *expression* specified inside the parentheses is evaluated. If the result of the expression evaluation is TRUE, the *program statement* that immediately follows is executed. After execution of this statement (or statements if enclosed in braces), the *expression* is once again evaluated. If the result of the evaluation is TRUE, the *program statement* is once again executed. This process continues until the *expression* finally evaluates as FALSE, at which point the loop is terminated. Execution of the program then continues with the statement that follows the *program statement*.

Example 1:

```
//display the digits 1 through 9
#include <stdio.h>
Int main ( )
{
Int digit;
digit=0;
While (digit<9)
{
Printf("%d",digit);
Digit++;
}
Return 0;
}
```

Example 2: calculating the average of n numbers

```
#include <stdio.h>
Int main ( )
{
Int n, count =1;
Float x, average, sum=0.0;
Printf("how many numbers do you want to input :");
Scanf("%d",&n);
While(count<=n)
{
Printf("enter number:");
Scanf("%f",&x);
Sum=sum+x;
Count++;
}
Average=sum/count;
Printf("the average of the %d values is %f",n,average);
Return 0;
}
```

Task 1: write a program that can display even numbers between 1 and 20. The program should also compute the sum and average of the even numbers.

Task 2: write a program that can display odd numbers between 1 and 50. The program should also compute the sum and average of the odd numbers.

The '**do…while**' loop

It is used when a loop condition is tested at the end of each loop pass.

General form

do
program statement
while (*loop_expression*);

The statement (simple or compound) will be executed repeatedly as long as the value of the expression is true.

NB since the test comes at the end, the loop body must be executed at least once.

Example: rewriting the program that counts from 0 to 9, using the do while loop:

//displays the digits 1 through 9

```
#include <stdio.h>
Int main ( )
{
Int digit;
digit=0;
do
{
Printf("%d",digit);
Digit++;
}while(digit<9);
Return 0;
}
```

Task: rewrite the program that computes the average of n numbers using the do..while loop

THE 'FOR' LOOP

This is the most commonly used looping statement in C programming.

General form

for (initialization; expression; increment) statement;

For repeating a block, the general form is for initialization; expression; increment)

{

statement sequence

}

Statement

Where:

The initialization is an assignment statement that sets the initial value of the loop control variable, which acts as the counter that controls the loop.

The expression is a conditional expression that determines whether the loop will repeat.

The increment defines the amount by which the loop control variable will change each time the loop is repeated.

Example: counting 0 to 9 using a 'for' loop

```
//displays the digits 1 through 9

#include <stdio.h>

Int main( )

{

Int digit;

For (digit=1;digit<=9;digit++)

{

Printf("%d",digit);

}
```

Example 2: averaging a set of numbers using the FOR loop

```
#include <stdio.h>
Int main ( )
{
Int n;
Float x, average, sum=0.0;
Printf("how many numbers do you want to input :");
Scanf("%d",&n);
For(count=1;count<=n; count++)
{
Printf("x=");
Scanf("%d",&x);
Sum+=x;
}
Average=sum/n;
Printf("the average of the %d values is %f",n,average);
Return 0;
}
```

NB:

- **You can count down using the decrement operator**
- **You can count by any number you wish; twos, threes, etc**

Task: using the for loop find the square root of numbers between 1 and 99

```
//program to determine the square root of numbers
#include <stdio.h>
#include <math.h>
Int main ( )
```

```
{
Int n;
Float x;
For (n=1; n<=99; n++)
{
X=sqrt(n);
Printf("the root of %d is %f",n,x);
}
Return 0;
}
```

NB

The conditional expression in a FOR loop is always tested at the top of the loop. This means that the code inside the loop may not be executed at all if the condition is false to begin with.

Example: for (count=10; count < 5; count++)

```
{
printf("%d",count);
}
```

This loop will never execute, because its control variable, count, is greater than 5 when the loop is first entered. This makes the conditional expression, count<5, false from the outset; thus, not even one iteration of the loop will occur.

Missing Pieces (for loop)

Another aspect of the "**for loop"** that is different in C than in many computer languages is that pieces of the loop definition need not be there. For example, if you want to write a loop that runs until the number 123 is typed in at the keyboard, it could look like this:

```
#include <stdio.h>
```

```
Int main ( )
{
Int n;
For (n=1; n!=5;)
{
Printf(“enter the value of n:”);
Scanf(“%d”,&n);
}
```

You can also move the initialization part out of the for loop as shown below

```
#include <stdio.h>
Int main ( )
{
Int n=1;
For (; n!=5;)
{
Printf(“enter the value of n:”);
Scanf(“%d”,&n);
}
```

Task: using the for loop determine 6! (six factorial)