East West University
Department of Computer Science & Engineering
Plot No-A/2, Main Road, Jahurul Islam City, Aftabnagar, Dhaka-1219

Lab Manual
Course : CSE -105
Credit Title : Structured Programming
Instructor : Md.Shamsujjoha (MSJ)

Lab-4: Basic Program Control: Loops.

Loop control statements are one of the most important parts of structured programming. The basis of programme control starts with loop. In this lab we will go to the detail of C's while loop and do while loop. A while statement is a C looping statement. It allows repeated execution of a statement or block of statements as long as the condition remains true (nonzero). If the condition is not true when the while command is first executed, the statement(s) is never executed. The form while Statement is following:

```
while (condition)
    statement
```

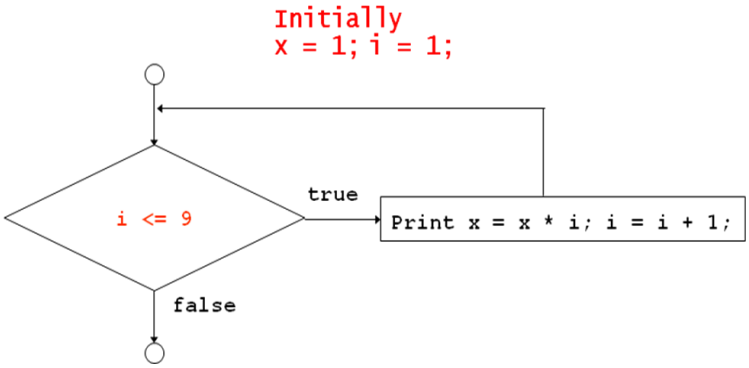
Here, ***condition*** is any valid C expression, usually a relational expression. When condition evaluates to false (zero), the while statement terminates, and execution passes to the statement following first statement; otherwise, the first C statement in the statements is executed. Statement is the C statement that is executed as long as condition remains true. Note that, the statement is singular here too, not plural alike ***if-else-if.*** To make a ***while*** statement control two or more statements, generally we use a compound statement. Compound statement has the form:

```
{

 Statement_1;
 Statement_2;
  …   …     …
 Statement_n;
}
```

Placing braces around a group of statements forces the compiler to treat it as a single statement, i.e., for the following C code, all n statements will be executed while the expression is true.

```
     while(condition)
          {
               Statement_1;
               Statement_2;
                 …   …     …
                Statement_n;
          }
```

Exercise: 1 Consider the following flowchart,



As you can imagine, this flow chart continuously print the value of ***x*** and ***i*** until ***i*** is becomes greater than 9. In other words, it allows repeated execution of a statements ***x=x*i,*** and ***i=i+1*** as long as the (***i<=9)*** remains true. The C code for this problem is as follows,

```
int main()
{
int x = 1, i =1;
while (i <= 9)
 {
   x = x*i;
   i = i+1;
   printf("\nThe value of x and i is %d, %d, respectively\n", x,i );
}
   return 0;
}
```

Output:

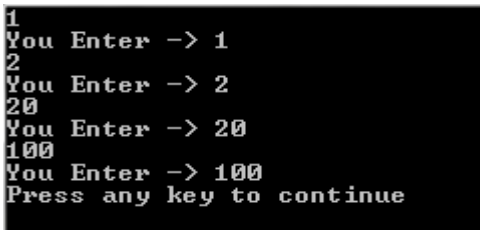
```
The value of x and i is 1, 2, respectively

The value of x and i is 2, 3, respectively

The value of x and i is 6, 4, respectively

The value of x and i is 24, 5, respectively

The value of x and i is 120, 6, respectively

The value of x and i is 720, 7, respectively

The value of x and i is 5040, 8, respectively

The value of x and i is 40320, 9, respectively

The value of x and i is 362880, 10, respectively
Press any key to continue_
```

Your 1st task is to write the above code and match your output will given one.
Can you code programme this without loop? Let's try it for 10 Mins,

Exercise: 2 In this programme we want to take input from command line until we get one greater than 99, i.e., our programme should repeatedly take inputs form keyboard and print it values. However, if the value of the input is greater than 99 we stop. Write the following code and check you output.

```
int nbr=0;
while (nbr <= 99)
{
   scanf("%d", &nbr );
   printf("You Enter -> %d\n", nbr );
}
```

Output:



Match your output with the given two on right. Next, your task to draw a Flowchart (in a paper) of the above programme. Can you solve this problem without loop. The ans is _____.

Exercise: 3 Nesting while Statements, Just like the if statements, while statements can also be nested. Following shows an example of nested while statements. Although this isn't the best use of a while statement, the example does present some new ideas.

```
3:   #include <stdio.h>
4:   int array[5];
6:   main()
8:   {
9:      int ctr = 0,
10:         nbr = 0;
11     printf("This program prompts you to enter 5 numbers\n");
13     printf("Each number should be from 1 to 10\n");
14     while ( ctr < 5 )
16:      {
17:         nbr = 0;
18:         while (nbr < 1 || nbr > 10)
19:         {
20:            printf("\nEnter number %d of 5: ", ctr + 1 );
21:            scanf("%d", &nbr );
22:         }
23:         array[ctr] = nbr;
25:         ctr++;
26:      }
27      for (ctr = 0; ctr < 5; ctr++)
29:         printf("Value %d is %d\n", ctr + 1, array[ctr] );
30:     return 0;
32:  }
```

Output : This program prompts you to enter 5 numbers, Each number should be from 1 to 10
Enter number 1 of 5: 3
Enter number 2 of 5: 6
Enter number 3 of 5: 3
Enter number 4 of 5: 9
Enter number 5 of 5: 2
Value 1 is 3
Value 2 is 6
Value 3 is 3
Value 4 is 9
Value 5 is 2

Forget about **array[5];** we will come to this very shortly. Check your output with the above outputs. For additional practice, there are two things you can change in this program. The first is the values that the program accepts. Instead of 1 to 10, try making it accept from 1 to 100. You can also change the number of values that it accepts. Currently, it allows for five numbers. Try making it accept 10.

Remarks : DON'T use the following convention if it isn't necessary:
while (x)
Instead, use this convention:
while (x != 0)

Although both work, the second is clearer when you're debugging (trying to find problems in) the code. When compiled, these produce virtually the same code.

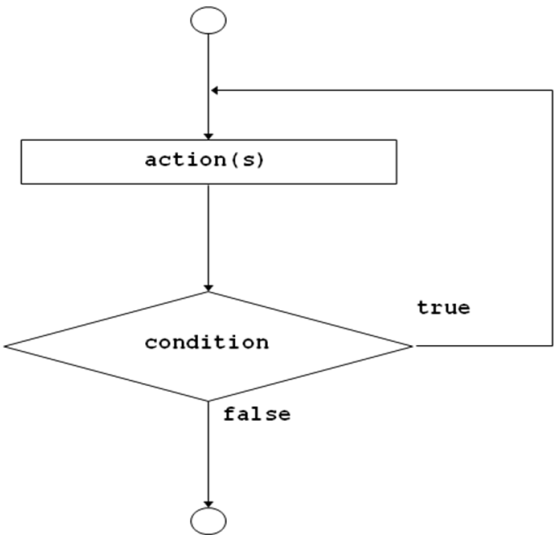
Exercise: 4 C's second loop construct is the do...while loop, which executes a block of statements as long as a specified condition is true. The do...while loop tests the condition at the end of the loop rather than at the beginning, as is done by the while loop. The structure of the do...while loop is as follows:

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

Here, condition is any C expression, and statement is a single or compound C statement. When program execution reaches a do...while statement, the following events occur:

1. The statements in statement are executed.
2. condition is evaluated.
3. If it's true, execution returns to step 1. If it's false, the loop terminates.

The operation of a do...while loop is shown in the following Figure,



The statements associated with a do...while loop are always executed at least once. This is because the test condition is evaluated at the end, instead of the beginning, of the loop. In contrast, while loops evaluate the test condition at the start of the loop, so the associated statements are not executed at all if the test condition is initially false. The do...while loop is used less frequently than while loops. It is most appropriate when the statement(s) associated with the loop must be executed at least once.

Now, your task is to Code the **Exercise 1 and 2 with do while loop**. The output should be identical as shown in that exercise.

Exercise: 5 Write a program that will take an integer number `n` as input and print the result of the following series:

a. 1+2+3+. . .+n
b. 1+1/2+1/3+. . . + n
c. 1.2+3.4+5.6+. . .+(n-1).n
d. 1.2.3+2.3.4+3.4.5+. . .+n.(n+1).(n+2)
e. 1+2+4+7+11+ . . . (up to `n` numbers)

Sample Input	Sample Output
4	a) 10 b) 2.08 c) 14 d) 210 e) 14

Exercise: 6 Write a program that reads two positive integers corresponding to two year values, ensures that the first year value is less than the second, and then determines and outputs all year values for leap years. A leap year is one that can be evenly divided by 4, unless it is a centennial, in which case it must be evenly divided by 400. For example, 1600 and 1992 are leap years, whereas 1700 and 1998 are not. Your programme should output all the leap years between this two input year.

Exercise: 7 Here, we will solve the Grading problem (LAB 2) in a more specific way. Now, The grading of each course is based on the following weighted scale:

- Term 1 – 20%
- Term 2 – 20%
- Final – 30%
- Attendance – 10%
- Class Tests – 20%

The letter grades are given based on the total marks obtained by a student and is shown below:

- A >= 90%
- B >= 80% & < 90%
- C >= 70% & < 80%
- D >= 60% & < 70%
- F < 60%

Term 1 and Term 2 exams are out of 20 each, Final is out of 30 and Attendance given is out of 10. Three class tests are taken per semester and the average of best two is counted towards the final grade. Every class test is out of 20. Example: Say Tara obtained marks of 15, 18, 25 and 8 in Term 1, Term 2, Final and Attendance respectively. Her 3 class test marks are 15, 12 and 17. Since average of best 2 will be counted, her class test mark will be equal to (15 + 17) / 2 = 16. Therefore, total marks = 15 + 18 + 25 + 8 + 16 = 82 and she will be getting a B.

Input : The first line of input is an integer T(T<100) that indicates the number of test cases. Each case contains 7 integers on a line in the order Term1 Term2 Final Attendance Class_Test1 Class_Test2 Class_Test3. All these integers will be in the range [0, total marks possible for that test].

Output : For each case, output the case number first followed by the letter grade {A B C D F}. Follow the sample for exact format.

Sample input	**Sample Output**
3 15 18 25 8 15 17 12 20 20 30 10 20 20 20 20 20 30 10 18 0 0	Case 1: B Case 2: A Case 3: B

Did you complete thee code?, if yes you haved solve you 1st ACM/UVA problem . Problem no 11777.

Exercise: 8 In this problem you will be analyzing a property of an algorithm whose lassification is not known for all possible inputs. The Problem Consider the following algorithm:

```
1.  input n
2.  print n
3.  if n = 1 then STOP
4.      if n is odd then n ⟵ 3n + 1
5.      else n ⟵ n/2
6.  GOTO 2
```

Given the input 22, the following sequence of numbers will be printed

22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

It is conjectured that the algorithm above will terminate (when a 1 is printed) for any integral input value. Despite the simplicity of the algorithm, it is unknown whether this conjecture is true. It has been verified, however, for all integers n such that 0 < n < 1,000,000 (and, in fact, for many more numbers than this.) Given an input n, it is possible to determine the number of numbers printed before and including the 1 is printed. For a given n this is called the cycle-length of n. In the example above, the cycle length of 22 is 16. For any two numbers i and j you are to determine the maximum cycle length over all numbers between and including both i and j. The Input The input will consist of a series of pairs of integers i and j, one pair of integers per line. All integers will be less than 10,000 and greater than 0.

You should process all pairs of integers and for each pair determine the maximum cycle length over all integers between and including i and j. The Output For each pair of input integers i and j you should output i, j, and the maximum cycle length for integers between and including i and j. These three numbers should be separated by at least one space with all three numbers on one line and with one line of output for each line of input. The integers i and j must appear in the output in the same order in which they appeared in the input and should be followed by the maximum cycle length (on the same line).

Sample input	**Sample Output**
1 10 100 200 201 210 900 1000	1 10 20 100 200 125 201 210 89 900 1000 174

Home Works :

1. Solve the ACM problem number 11777 and 100. That is you should get the acceptance for this two problem.
2. Write a program that will take a number `n` as input and print `n` lines of output according to the following sample output.

Sample Input	Sample Output
3	1 2 3 4 5 6
5	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

3. Write a program to read an integer number `n` and print the value of 3^n

Sample Input	Sample Output
2	9
4	81
0	1

Reminder: Next week no student will be allowed to enter into the lab without completion of the above home tasks. Submission of the home tasks can be either Hard Copy (Hand Written or, printed) or Soft Copy (Source Code). We also *vigorously* opposed to the academic dishonesties, as it seriously detracts from the education of honest students. In the completion of your home works, it is impermissible to discuss a general method of solution with other students, or to make use of online resources. In case of any problem, inconsistency and errors in coding knock your Instructors/teachers. We are here to help you and to remove your confusions.

Thanks
Maheen Islam, Dr. Taskeed Jabid and Md. Shamsujjoha
Faculty members, Department of Computer Science & Engineering, East West University
Email: maheen@ewubd.edu, tasked@ewubd.edu and dishacse@yahoo.com