Department of
Computer Science and Engineering

Title: Introduction to While and Do -While Loop Control Structure in C

Structured Programming Lab

CSE 104

Green University of Bangladesh

1 Introduction

A loop structure is used to execute a certain set of actions for a predefined number of times or until a particular condition is satisfied. There are 3 control statements available in C/C++ to implement loop structures.

- While statements
- Do while statements
- For statements

1.1 While Statements

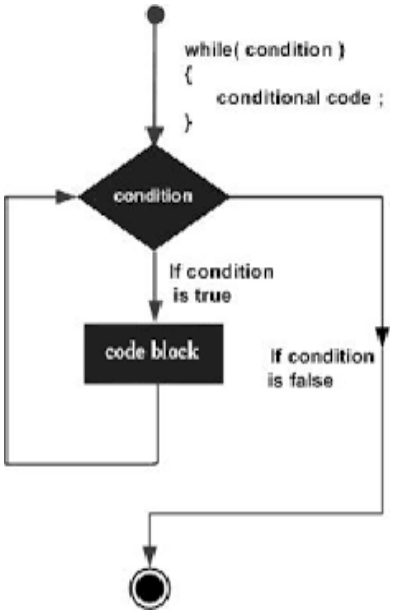
The while statement is used to carry out looping operations, in which a group of statements is executed repeatedly, until some condition has been satisfied. The conditional expression is checked for TRUE first. If it is TRUE then all statements inside curly braces are executed.Then program control comes back to check the condition has changed or to check if it is still TRUE. The statements inside braces are executed repeatedly, as long as the condition is TRUE. When the condition turns FALSE, program control exits from while loop.

General form of the While statement:

```
while (expression)
    statement;

        OR
while (expression)
{
    statement1;
    statement2;
}
```

Flowchart of While statement:



Example 1: Write a C program to print all natural numbers from 1 to n.

Algorithm 1 Steps in pseudo code:

1: step1: Start
2: step2: Declare variables count and n
3: step3: Display a message "Enter the value of n "
4: step4: Read n
5: step5: Display a message "n Natural numbers are: "
6: step6: Initialize count=1
7: step7: If the expression "count<=n" is evaluated to true, steps 8 and 9 are executed and If the expression "count<=n" is evaluated to false, go to step 10
8: step8: Display current value of count
9: step9: Increment count variable by 1
10: step10: End

Code:

```c
#include <stdio.h>
int main()
{
    int count,n;
    printf("Enter the value of n\n");
    scanf("%d",&n);
    printf("%d Natural numbers are:\n",n);

    count=1;
    while (count <= n)
        {
            printf("%d\n", count);
            count++;
        }
   return 0;
}
```

Output:

```
6
6 Natural numbers are:
1
2
3
4
5
6
```

Example 2: Write a C program to print all odd numbers between 1 to n

Algorithm 2 Steps in pseudo code:

1: step1: Start
2: step2: Declare variables count and n
3: step3: Display a message "Enter the value of n "
4: step4: Read n
5: step5: Display a message "All odd numbers between 1 to n are: "
6: step6: Initialize count=1
7: step7: If the expression "count<=n" is evaluated to true, steps 8 and 9 are executed and If the expression "count<=n" is evaluated to false, go to step 10
8: step8: If "count%2==1" is evaluated true Display the value of count
9: step9: Increment count variable by 1
10: step10: End

Code:

```c
#include <stdio.h>
int main()
{
    int count,n;
    printf("Enter the value of n\n");
    scanf("%d",&n);
    printf("All odd numbers between 1 to %d are:\n",n);

    count=1;
    while (count <= n)
        {
            if(count%2==1)
                printf("%d\t", count);
            count++;
        }
   return 0;
}
```

Output:

```
Enter the value of n
12
All odd numbers between 1 to 12 are:
1       3       5       7       9       11
```

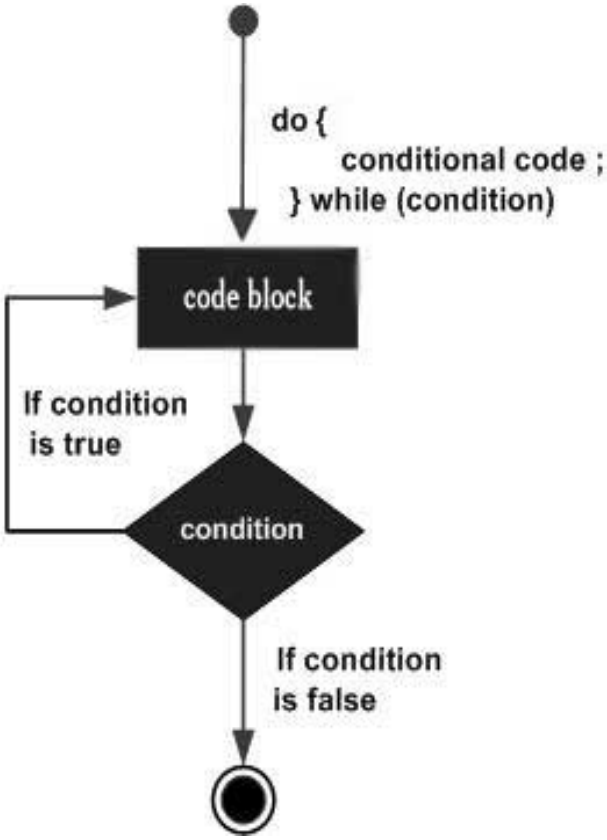
1.2 Do-While Statements

Unlike while, do while is an exit controlled loop. Here the set of statements inside braces are executed first. The condition inside while is checked only after finishing the first time execution of statements inside braces. If the condition is TRUE, then statements are executed again. This process continues as long as condition is TRUE. Program control exits the loop once the condition turns FALSE.

General form of the Do-While statement:

```
do
{
    Statement(s);
    Updation;
} while (condition);
```

Flowchart of Do-While statement:



Example 1: Write a C program to sum 10 integer numbers (take the numbers from user) and display the result.

Algorithm 3 Steps in pseudo code:

1: step1: Start
2: step2: Declare variables sum, number and count
3: step3: Initialize count=1 and sum=0
4: step4: Read number
5: step5: Add value of number variable to sum variable
6: step6: Increment count variable by 1
7: step7: If the expression "count<=10" is evaluated to true, repeat steps 4 to 6 and If the expression "count<=10" is evaluated to false, go to step 8
8: step8: Display the current value of sum
9: step9: End

Code:

```c
#include <stdio.h>
int main()
{
    int sum,number,count;
    count=1,sum=0;

        do{
            scanf("%d",&number);
            sum=sum+number;
            count++;
         } while (count <= 10);
    printf("Result = %d\n",sum);
   return 0;
}
```

Output:

```
23
10
43
-62
90
12
56
-34
49
10
Result = 197
```

Example 2: Write a C program to calculate sum of digits of any number.

Algorithm 4 Steps in pseudo code:

1: step1: Start
2: step2: declare variables sum, number and n
3: step3: Read number
4: step4: Initialize sum=0
5: step5: Divide number by 10 and assign the remainder to n
6: step6: Add the value of n to sum
7: step7: Update the number variable as number=number/10
8: step8: If the expression "number!=0" is evaluated to true, repeat steps 5 to 7 and If the expression "number!=0" is evaluated to false, go to step 9
9: step9: Display a message "Sum of digits of given number is = " with the value of sum
10: step10: End

Code:

```c
#include <stdio.h>
int main()
{
    int sum,number,n;
    scanf("%d",&number);
    sum=0;
        do{
            n=number%10;
            sum=sum+n;
            number=number/10;
        } while (number != 0);
    printf("Sum of digits of given number is = %d\n",sum);
   return 0;
}
```

Output:

```
10023
Sum of digits of given number is = 6
```

2 Discussion & Conclusion

Based on the focused objective(s) to understand about while and do-while statements in C program, the additional lab exercise made me more confident towards the fulfilment of the objectives(s).

3 Lab Task (Please implement yourself and show the output to the instructor)

1. Write a C program to print all alphabets from a to z.
2. Write a C program to print all even number 1 to 100.
3. Write a C program to enter a number and print its digit in reverse order.
4. Write a C program to find frequency of each digit in a given integer.

4 Lab Exercise (Submit as a report)

1. Write a C program to find sum of first and last digit of any number.
2. Write a C program to swap first and last digits of any number.
3. Write a C program to calculate product of digits of any number.
4. Write a program in C to find the sum of the series 1 +11 + 111 + 1111 + .. n terms.

5 Policy

Copying from internet, classmate, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.