

Name	Hooria Khan Orakzai
ID	14263
Department	BS(Software Engineering)
Submitted to	Sir Fazal-e-malik



Iqra National University Peshawar Pakistan Department of Computer Science

Summer Semester, Final Term Exam, September 2020

Paper :	Programming Fundamentals	Date and Starting Time:	29/September/2020, 9:00 am
Program:	BS (CS & SE)	Uploading Date and End Time:	29/September/2020, 3:00 pm
Teacher Name:	Dr. Fazal-e-Malik	Marks	50

Note: Attempt all Questions. Help can be taken from internet where ever is required.

Q.1

- a) What is the purpose of **if statement**? Discuss its two different forms with examples.

IF Statement:

The "if statement- is used to execute (or ignore a set of statements after testing a condition. The "if statement" evaluates a condition. If the given condition is true. The statement (or a set of statements) following the if statement.' is executed. If the given condition is false. the statement (or a set of statements) following the "if statement- condition is ignored and the control transfers to the next statement. The syntax of the "if statement" is:

if (condition)

statement-1;

statement-2;

Condition: specifies a condition or a relational expression.

When this condition is true, the statement following the "if statement" is executed. If the given condition is false. the statement following the "if statement" is ignored and the control transfers to the next statement.

In the above syntax, only statement-1 will he executed if the given condition is true otherwise the control shifts to Statement-2 that comes after the Statement-1. To execute a set of statements following the "if statement the set of statements is enclosed in curly braces. i.e. within { } the statements in braces are also known as compound statements.

The syntax of the “if statement” for executing a set of statements as if (condition)

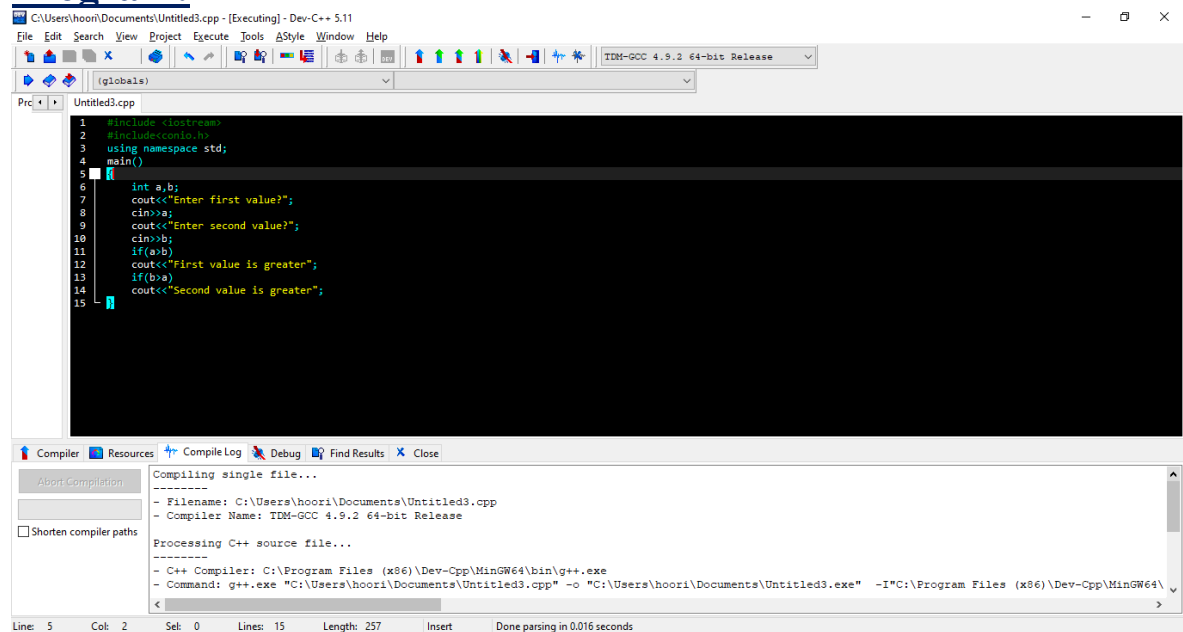
```
{  
    Statement-1  
    Statement-2  
    Statement-3  
    Statement-m  
}
```

Statement-n

In the above syntax. the set of statements (from Statement-1 to Statement-m) has been enclosed in curly braces. These are called compound statements. If the condition given in the “if statement” is true the compound statements given in the curly braces are executed. If the condition is false the control shifts to the Statement-n and the set of statements that follows the “if statement” is ignored.

- b) Write a C++ program to read two numbers from keyboard and then find the LARGEST number of them.

Program:



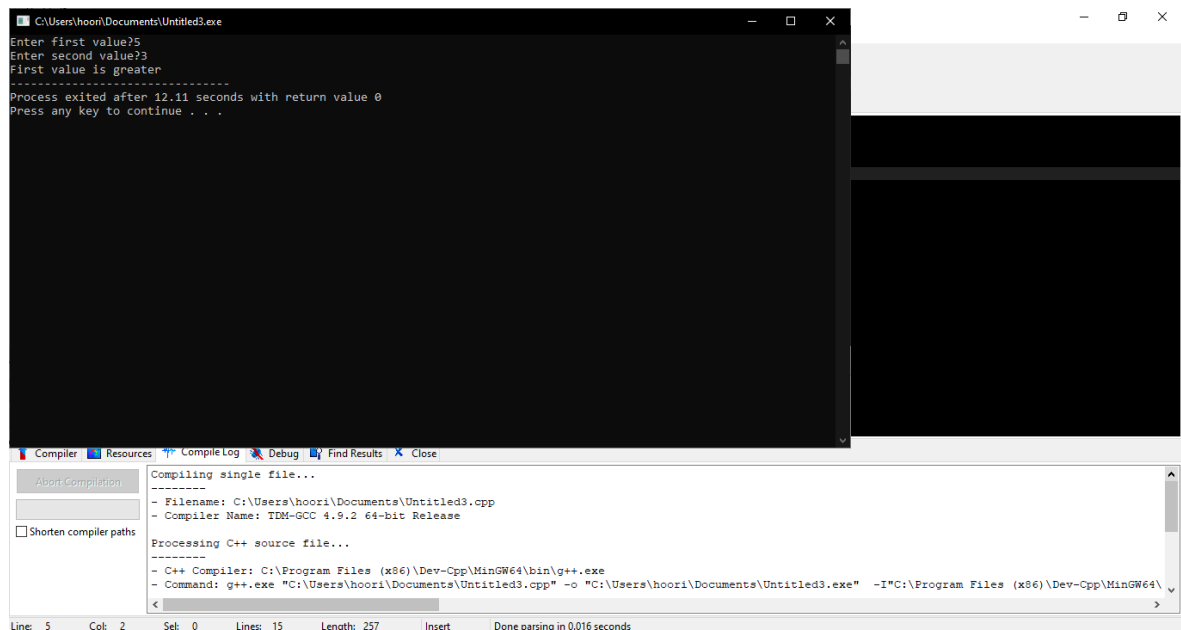
The screenshot shows a C++ IDE with a source code editor and a compiler output window. The source code is as follows:

```
1 #include <iostream>  
2 #include <conio.h>  
3 using namespace std;  
4 main()  
5 {  
6     int a,b;  
7     cout<<"Enter first value?";  
8     cin>>a;  
9     cout<<"Enter second value?";  
10    cin>>b;  
11    if(a>b)  
12        cout<<"First value is greater";  
13    if(b>a)  
14        cout<<"Second value is greater";  
15 }
```

The compiler output window shows the following text:

```
Compiling single file...  
-----  
- Filename: C:\Users\hoori\Documents\Untitled3.cpp  
- Compiler Name: TDM-GCC 4.9.2 64-bit Release  
-----  
Processing C++ source file...  
-----  
- C++ Compiler: C:\Program Files (x86)\Dev-Cpp\MinGW64\bin\g++.exe  
- Command: g++.exe "C:\Users\hoori\Documents\Untitled3.cpp" -o "C:\Users\hoori\Documents\Untitled3.exe" -I"C:\Program Files (x86)\Dev-Cpp\MinGW64\
```

Output



Q.2 a) What are the Logical Operators? Explain them

LOGICAL OPERATORS:

The logical operators are used to combine relational expressions or relational conditions. The expression containing logical operators is called logical expression or logical condition. It is also called compound condition or compound expression.

The output of a logical expression is also logical form. Its value is either true or false. In C++, following logical operators are used:

1. $\&\&$ \rightarrow AND Operator
2. $\|\|$ \rightarrow OR Operator
3. $!$ \rightarrow NOT Operator

1.The $\&\&$ (AND) Operator:

The $\&\&$ (AND) operator is used to combine two or more relational expressions. If all relational expressions are true then the output returned by the compound expression is true. If any one of relational expression in the compound expression is false, the output is false.

For example, if $x = 10$, $y = 15$, $z = 5$ then the compound expression $(x < y) \&\& (z = 5)$ returns true because both the expressions $(x < y)$ and $(z = 5)$ are true. Similarly, the compound expressions $(x > y) \&\& (z = 5)$ and $(x < y) \&\& (z > x)$ will return false values.

2. The $\|\|$ (OR) Operator

The (OR) operator is used to combine more than one relational expression. If any one of the given relational expressions is true, the output will be true otherwise the output will be false. For example, if $x = 0$, $y = 15$, $z = 5$ then the compound expression $(x > y) \|\| (z == 5)$ returns true because $(z == 5)$ is true.

Similarly, $(x != y) \|\| (x == y) \|\| (z > 10)$ returns true because $(x != y)$ is true. If all

the relational expressions in the compound condition are false then the output will be false.

3. The ! (Not) Operator

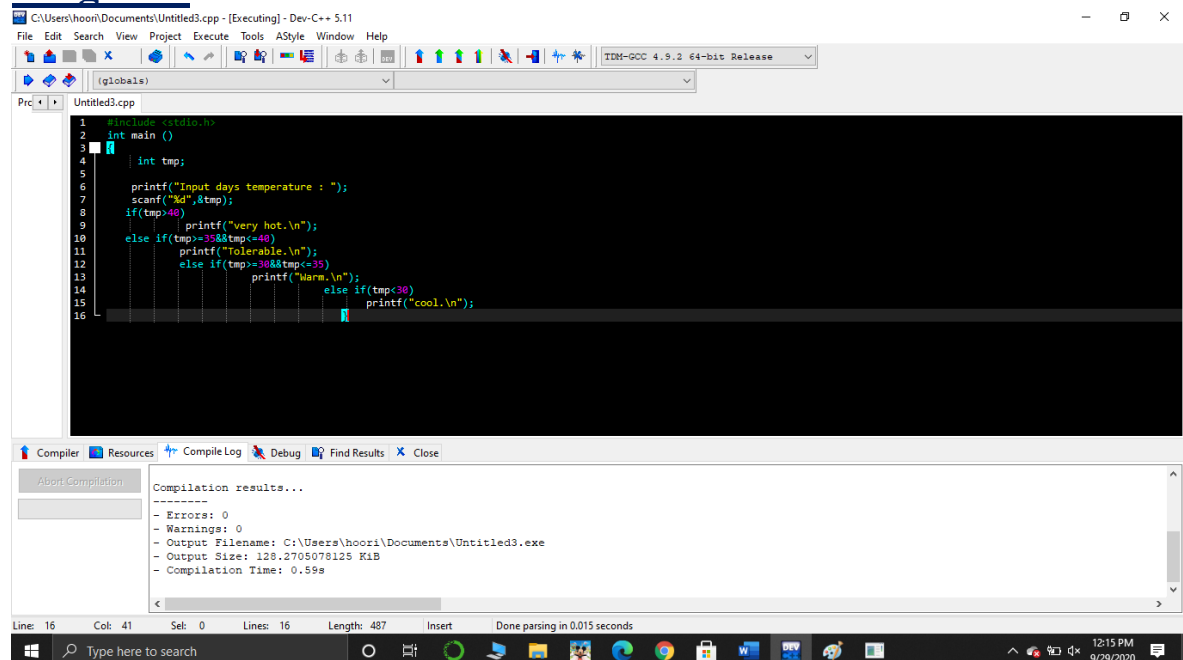
The ! (NOT) operator is also known as the unary operator. It inverts the value returned by the relational expression or the compound expression. For example, if $x = 5$ and $y = 10$ then the logical expression $(x > y)$ returns true. It is because $(x > y)$ returns false and the "!" (NOT) operator inverts the result into true.

b) Write a C++ program to get Temperature in Fahrenheit F and then find the Atmosphere according to the below rules:

- If temperature F is above 40 degree Fahrenheit then display.....Very Hot.
- If temperature F is between 35 & 40 degree Fahrenheit then display.....Tolerable.
- If temperature F is between 30 & 35 degree Fahrenheit then display.....Warm.
- If temperature F is less than 30 degree Fahrenheit then display.....Cool.

5

Program:



```
1 #include <iostream>
2 int main ()
3 {
4     int tmp;
5
6     printf("Input days temperature : ");
7     scanf("%d",&tmp);
8     if(tmp>40)
9         printf("very hot.\n");
10    else if(tmp>=35&&tmp<=40)
11        printf("Tolerable.\n");
12    else if(tmp>=30&&tmp<=35)
13        printf("Warm.\n");
14    else if(tmp<30)
15        printf("cool.\n");
16 }
```

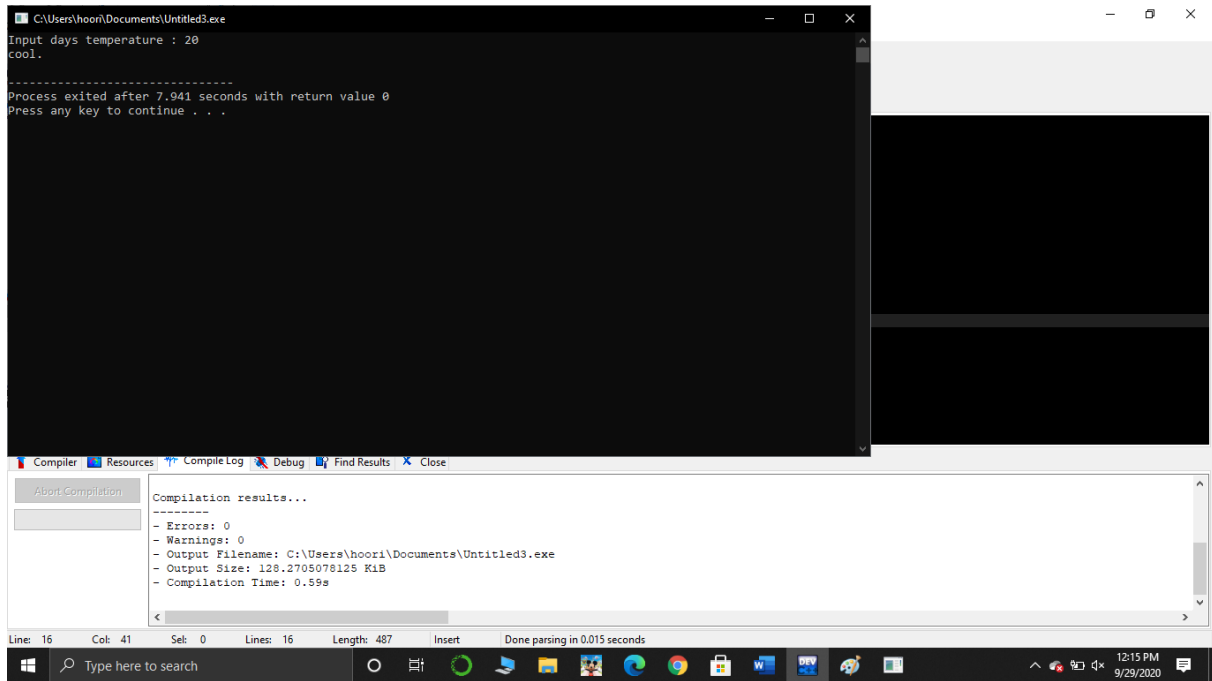
Compilation results...

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\hooori\Documents\Untitled3.exe
- Output Size: 128.2705079125 KiB
- Compilation Time: 0.59s

Output:

5

Q.3



What does **Looping** mean? Explain different loops in C++.

LOOPS

A statement or a set of statements that is executed repeatedly is called loop. The statement(s) in a loop are executed for a specified number of times or until some given condition remains true.

In C++ there are three kinds of loop statements. These are:

- The "-while" loop
- The "do-while" loop
- The "for" loop

The "while" Loop:

It is a conditional loop statement. It is used to execute a statement or a set of statements as long as the given condition remains true. The syntax of the - while" loop is:

while (condition)

Statement;

where

condition: It consists of a relational expression. If it is true, the statement or the set of statements given in the while loop is executed.

statement: It represents the body of loop. The compound statements or a set or statements are written in braces {}

The while loop for more than one statement is written as:

while (condition)

```
{  
statement (s);  
}
```

When "while" loop statement is executed. the computer first evaluates the given condition. If the given condition is true, the statement or-set of statements in the body of the "while" is executed. After executing the statements under "while". the control shifts back to "while" and the condition is again tested. if the given condition becomes false at any stage during execution, the execution 'of the body of loop is terminated and control shifts to the statement that comes immediately after the body of the loop.

The body of the loop must contain a statement so that the condition on the loop becomes false during execution of the loop. If the condition of the loop never becomes false, the loop never ends. It becomes an infinite loop.

The “do-while” Loop:

The do-while loop is also conditional loop statement. It is like a while loop but in this test the condition is tested after executing the statement of loop

The syntax of the “do-while loop” is:

```
do  
{  
    statements;  
}  
while condition
```

where

do is a keyword of C++. It indicates the starting of the do while loop

Statements enclosed in braces represent the body of the loop

Condition it is the condition that must remain true for the execution of the loop

In “do-while loop” the body of the loop is executed at least once before the condition is tested. It is repeatedly executed as long as the test condition remains true If the given condition becomes false at any stage during the program execution the loop terminates and control shifts to the statement that comes immediately after the keyword “while”.

The difference between the “while-loop” and do while loop is that

- In “while-loop” test condition comes before the body of the loop. First the condition is tested and then the body of the loop is executed
- In “do-while loop” the body of the loop comes before the test condition the body of the loop is executed and then the condition is tested.

The For-Loop:

The "for loop." statement is used to execute a set of statements repeatedly for a fixed number of times. It is also known as counter loop. The structure of this loop is different from both the "while" and the "do-while" loops. It has the following parts:

- i) Initialization
- ii) Condition
- iii) Increment or decrement
- iv) Body of the loop

The general syntax of the for loop is: for (initialization; condition; increment/decrement)

(i) **In initialization** part, the value in the variable (or a set of variables) is assigned when control enters into the loop first time. New variable(s) can also be declared and initialized in this part. For example:

```
for (int a=6, b = 10; a < 10; a++)
```

In the above example, the variables "a" and "b" are declared and initialized.

If the variables have already been declared, then they are not declared again. The initialization part of for loop" is optional. If this part is omitted then a semicolon is used in its place. The syntax of the for loop" without initialization part is:

```
for (; a<=10; a++)
```

(ii) **In condition** part, the test condition is given. The body of the loop executes as long as this condition remains true. It is like the test condition of "while" or "do-while" loop. In the above example "a<=10" represents the condition.

(iii) **In increment or decrement** part, the value in the variable(s) is incremented or decremented. To increment or decrement values in more than one variable, the variables are written, separated by commas. This part is executed after executing the body of the loop. Each time the body of the loop is executed, the value of the variable is increased or decreased. The use of this part is optional. If this part is not used then the control variable must be incremented or decremented inside the body of the loop.

(iv) **The statements** under the "for" loop are the body of loop. If more than one statement are to be executed, these are enclosed in braces.

- a) Write a C++ program to read a number from keyboard and then determine whether it is **Even or Odd** number?

Program:

```

1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6     int number, remainder;
7
8     cout << "Enter the number : ";
9     cin >> number;
10    remainder = number % 2;
11    if (remainder == 0)
12        cout << number << " is an even integer " << endl;
13    else
14        cout << number << " is an odd integer " << endl;
15
16    return 0;
17 }

```

Compilation results...

```

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\hoori\Documents\Untitled3.exe
- Output Size: 1.83260917663574 MiB
- Compilation Time: 2.38s

```

Output:

```

C:\Users\hoori\Documents\Untitled3.exe
Enter the number : 5
5 is an odd integer

-----
Process exited after 9.302 seconds with return value 0
Press any key to continue . . .

```

Compilation results...

```

-----
- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\hoori\Documents\Untitled3.exe
- Output Size: 1.83260917663574 MiB
- Compilation Time: 2.38s

```

Q.4 a) What is the purpose of using **break and continue statements**?

The "Continue" Statement:

The "continue statement shifts the control back to the beginning the loop. It is used inside, the body of the loop. When this statement is executed inside the body of the loop, the control shifts to the first statement of the body of the loop.

The syntax is:

Continue;

For example:


```
continue;
```

For example:

```
#include <iostream>
```

```
Using namespace std;
```

```
Int main()
```

```
{
```

```
Int c;
```

```
C=1;
```

```
while(c<=5)
```

```
{
```

```
    cout <<"Pakistan"<< endl;
```

```
    c++;
```

```
    continue;
```

```
    cout<<"Islamabad";
```

```
    cout<<"Peshawar";
```

```
}
```

```
}
```

In the above program only first three statements inside the while loop are executed 5 times. The two statements after "continue" statement are not executed because each time the "continue" statement returns the control back to the beginning of body of loop.

The "break" 'Statement:

The "break" statement terminates the execution of the loop when it is used inside the body of the loop. The difference between the "break" statement and the "continue" statement is that:

- The "break" statement terminates the loop during execution. The other statements of the body of the loop that comes under the "break" statement are not executed.
- The "continue" statement does not terminate the loop but it shifts the control at the beginning of the body of the loop. The statements of the body of the loop that come under the "continue" statement are not executed.

b) Write a C++ program to find the sum of the following numbers:

$$1+2+3+\dots+10$$

Program:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i,sum=0;
6     cout << "\n\n Find the sum of first 10 given numbers:\n";
7     cout << "-----\n";
8     cout << " The numbers are: \n";
9     for (i = 1; i <= 10; i++)
10    {
11        cout << i << " ";
12        sum=sum+i;
13    }
14    cout << "\n The sum of first 10 given numbers: "<<sum << endl;
15 }

```

Compilation results...

```

- Errors: 0
- Warnings: 0
- Output Filename: C:\Users\hoori\Documents\Untitled3.exe
- Output Size: 1.83260917663574 MIB
- Compilation Time: 1.17s

```

Output:

```

C:\Users\hoori\Documents\Untitled3.exe

Find the sum of first 10 given numbers:

The numbers are:
1 2 3 4 5 6 7 8 9 10
The sum of first 10 given numbers: 55

-----
Process exited after 0.1673 seconds with return value 0
Press any key to continue . . .

```

- Q.5 Explain the following with proper examples
- C++ Character set
 - Constants
 - Variables
 - Keywords
 - Relational Operators

1.C++ Character set:

In C++, character set is a set of all valid characters that can be used in a C++ Program. Characters set is used to specify the characters or symbols recognized by the language. Character set is a set of all valid characters that can be used to form words, numbers and expression's in source programs.

2.Constants:

Constants refer to fixed values that the program may not alter and they are called literals. Constants can be of any of the basic data types and can be divided into Integer Numerals, Floating-Point Numerals, Characters, Strings and

Boolean Values.

3.Variables:

Variables are containers for storing data values.

In C++, there are different types of variables (defined with different keywords), for example:

- **int** - stores integers (whole numbers), without decimals, such as 123 or -123
- **double** - stores floating point numbers, with decimals, such as 19.99 or -19.99
- **char** - stores single characters, such as 'a' or 'B'. Char values are surrounded by single quotes
- **string** - stores text, such as "Hello World". String values are surrounded by double quotes
- **bool** - stores values with two states: true or false

4.Keywords:

The words that are used by the language for special purposes are called keywords. These are also called reserved words. For example, in a C++ program, the word main is used to indicate the starting of program, include is used to add header files, int to declare an integer type variable. All these words are keywords of C++. The keywords cannot be used as variable names in a program.

5.Relational Operator:

In C++ Relational operators, two operators that is == (Is Equal to) and != (is Not Equal To), are used to check whether the two variables to be compared are equal or not. Let us take one example which demonstrate these two operators.

😊 *****😊

Note:

- 1. I have already uploaded a 20 Marks assignment. If a student does not upload then he/she will get Zero marks out of 20*
- 2. Write the answers in MS Word or hand written in scanned PDF Form in one documents and do not send in Zip Form*

3. Please write your Name and ID on top of your answer Paper otherwise you will get zero marks.