National University
of computer and emerging sciences

LAB 04

Summary

Items	Description
Course Title	Programming Fundamentals
Lab Title	Stream Insertion/Extraction Operations
Duration	3 Hours
Operating System/Tool/Language	Ubuntu/ g++/ C++
Objective	Input/ Output Operations in C++

Exercise 1: A Simple First Program

You need to perform the following to complete the task.

1. Open the Terminal (Ctrl + Alt + t)
2. Installation g++

lab@lab-OptiPlex-330:~$ **sudo apt install g++**

Note: g++ is already install on lab PCs

3. Create file of .cpp file extension using touch command

~$ **touch helloworld.cpp**

4. Now open the text editor using gedit command

~$ **gedit helloworld.cpp**

5. Write the following code in helloworld.cpp file.

```
#include<iostream>
int main()
{
 cout<<"hello world"<<endl<<"123";
 return 0;
}
```

(The line using namespace std; tells the compiler to use the std namespace. Namespaces are a relatively recent addition to C++.)

(The next line return 0; terminates main( )function and causes it to return the value 0 to the calling process.)

6. Save and close the file.
7. compile and execute it

~$ g++ -o hello helloworld.cpp

~$./hello

1. Standard output (cout)

cout is a C++ stream object, used for standard output by default is the screen. For formatted output operations, cout is used together with the insertion operator, which is written as << (i.e., two "less than" signs).

```
int a;
int b;
a=10;
b=20;
cout<<"A="<<a<<endl;
cout<<"B="<<b<<endl;
```

2. Standard input (cin)

In most program environments, the standard input by default is the keyboard, and the C++ stream object defined to access it is cin.
For formatted input operations, cin is used together with the extraction operator, which is written as >> (i.e., two "greater than" signs). This operator is then followed by the variable where the extracted data is stored. For example:

```
int height;
cin >> height;
```

Example: cin with extraction operator:

Program comments are explanatory statements that you can include in the C++ code. These comments help anyone reading the source code. All programming languages allow for some form of comments. C++ supports single-line and multi-line comments. C++ compiler ignores all characters available inside any comment. C++ comments start with /* and end with */. A comment can also start with //, extending to the end of the line.

```
#include <iostream> using namespace std; int main()
{ int a, b, c;
        /* For single input */ cout << "Enter a number: ";
        cin >> a;
        /* For multiple inputs*/ cout << "Enter 2 numbers: ";
        cin >> b >> c;
        cout << "Multiplication = " << (a*b*c);
        return 0;
}
```

When you run the program, a possible output will be:

```
Enter a number: 3
Enter 2 numbers: 3 3
Sum = 27
```

3. Constants and Variables:

Constants: A specific alphabetical and/or numeric value that is never changed.
For Ex. PI - 3.14159
Variables: The value that can be changed.
For Ex. ShoeCost = 56.00 and ShoeCost = 35.00

4. Data Types:

4.1 int - integer: a whole number.

This data type is used to define an integer number (-…. -3, -2,-1,0,1,2,3….). A single integer occupies 2 bytes. For example: int a; declares that you want to create an int variable called a. To assign a value to our integer variable we would use the following C statement: a=10;

4.2 float - floating point value: i.e. a number with a fractional part.

A float, or floating point, number has about seven digits of precision and a range of about 1.E-36 to E+36. A float takes four bytes to store.

4.3 double - a double-precision floating point value.

A double, or double precision, number has about 13 digits of precision and a range of about 1.E-303 to 1.E+303. A double takes eight bytes to store.
Note: Single precision and Double precision basically differs in the number of digits represented after the decimal point. Double precision number will represent more digits after the decimal point than a single precision number. Example: Single precision – 32.75 and double precision – 32.7543

4.4 char - a single character.

Used to define characters. A single character occupy 1 byte.
To assign, or store, a character value in a char data type is easy - a character variable is just a symbol enclosed by single quotes.

```
char a;
char a = '10';
```

5. Escape Sequences

Character combinations consisting of a backslash (\) followed by a letter or by a combination of digits are called "escape sequences." To represent a newline character, single quotation mark, or certain other characters in a character constant, you must use escape sequences. An escape sequence is regarded as a single character and is therefore valid as a character constant. Escape sequences are used to format our output. The following escape sequences can be used to print out special characters.

Escape Sequence	Description
\n	Newline
\t	Horizontal tab
****	Backslash
\'	Single quote
\"	Double quote

To insert a line break, a new-line character shall be inserted at the exact position the line should be broken. In C++, a new-line character can be specified as \n (i.e., a backslash character followed by a lowercase n). For example:

```
cout << "First sentence.\n";
cout << "Second sentence.\nThird sentence.";
```

This produces the following output:

```
First sentence.
Second sentence.
Third sentence.
```

Alternatively, the endl manipulator can also be used to break lines. For example:

```
cout << "First sentence." << endl;
cout << "Second sentence." << endl;
```

Output

```
First sentence.
Second sentence.
```

Example 2.1

Following program shows the use of Newline Escape Sequence (\n)

```cpp
#include <iostream>
using namespace std;

int main()
{
    cout << "This\nis\na\ntest\n\nHe said, How are you?\n";
    return 0;
}
```

Output

```
lab@lab-OptiPlex-330:~/Desktop/PF2019/Lab3$ g++ -o Q2_1 Q2_1.cpp
lab@lab-OptiPlex-330:~/Desktop/PF2019/Lab3$ ./Q2_1
This
is
a
test

He said, How are you?
```

Your turn: Edit above given code and use endl manipulator.

Example 2.2

This program shows the use of Horizontal tab Escape Sequence (\t)

```cpp
#include <iostream>
using namespace std;

int main()
{
    cout << "This is a test\t\tHe said, How are you?\n";
    return 0;
}
```

Output

```
lab@lab-OptiPlex-330:~/Desktop/PF2019/Lab3$ g++ -o Q2_2 Q2_2.cpp
lab@lab-OptiPlex-330:~/Desktop/PF2019/Lab3$ ./Q2_2
This is a test          He said, How are you?
```

Now try escape sequences **\\ , \' , \"** yourself.

Example 2.3

Program using multiple insertion operations (<<)

```
#include <iostream>
using namespace std;

int main()
{
    cout << "This is a test "<<"He said,\t\t"<<"How are you?\n";
    return 0;
}
```

Output

```
lab@lab-OptiPlex-330:~/Desktop/PF2019/Lab3$ g++ -o Q2_3 Q2_3.cpp
lab@lab-OptiPlex-330:~/Desktop/PF2019/Lab3$ ./Q2_3
This is a test He said,         How are you?
```

6. iomanip

iomanip is a library that is used to manipulate the output of C++ program.

Below are some Parametric manipulators

6.1 setw

It is used to sets the field width to be used on output operations

Example

```
    int  a,b;
    a = 200;
    b = 300;
    cout << setw (5) << a << setw (5) << b << endl;
    cout << setw (6) << a << setw (6) << b << endl;
    cout << setw (7) << a << setw (7) << b << endl;
    cout << setw (8) << a << setw (8) << b << endl;
}
```

Output of the above program

```
  200   300
   200   300
    200   300
     200   300
```

Now compile your code and see what the output is.

6.2 setprecision

It is used to sets the decimal precision to be used to format floating-point values on output operations.

Example

```cpp
#include <iostream>
#include <iomanip>

int main () {
   double f =3.14159;
   std::cout << std::setprecision(5) << f << '\n';
   std::cout << std::setprecision(9) << f << '\n';
   std::cout << std::fixed;
   std::cout << std::setprecision(5) << f << '\n';
   std::cout << std::setprecision(9) << f << '\n';
   return 0;
}
```

Output

```
3.1416
3.14159
3.14159
3.141590000
```

LAB TASKS

Task 1

Run all sample programs and note down the output of each program

Task 2

Write a C++ program to print the following lines:

Do not waste water even if you were at a running stream.

Task 3

Write a program which takes two integers from user and perform arithmetic operations on them (addition, subtraction, multiplication, division, mod). Display results on screen.

Sample Output:

Suppose A=50 B=50

Output
Addition of 50 and 50 = 100
Subtraction of A and B is 0
50*50=2500
Division of A and B = 1
50 mod 50 = 0

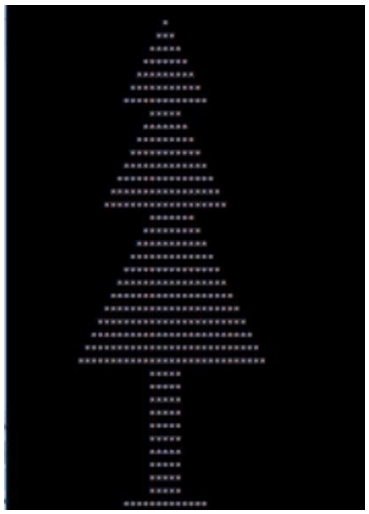
Task 4

Write a program to assign values to two variables by assignment statement. Swap the values of both the variables:
Hint: Use third variable
Sample Output:
A=100
B=200

After Swapping
A=200, B=100

Task 5

Write a program to print the following using just ONE COUT statement & setw function

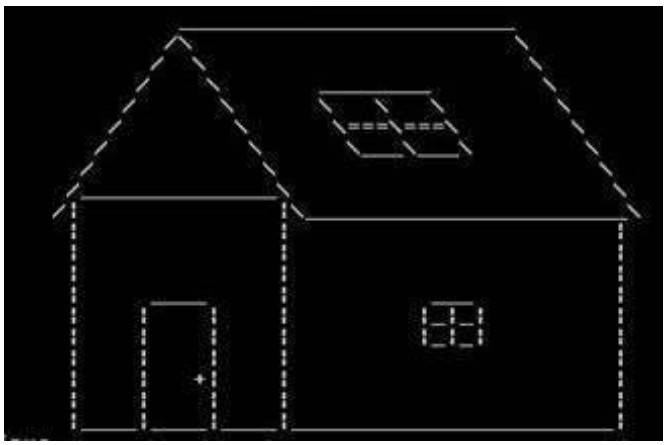


Note: Use **setw** function instead of space character

National University
of computer and emerging sciences

Task 6

Write a program that prints the following using COUT & setw statement.



Note: Use **setw** function instead of space character

Submission Instructions:

1. Save all .cpp files with your roll no and task number
   e.g. i21XXXX_Task01.cpp
2. Save all screenshots of terminal with your roll no and task number
3. Now create a new folder with name ROLLNO_LAB03 e.g. i21XXXX_LAB03
4. Move all your .cpp files to this newly created directory and compress it into .zip file.
5. Now you must submit this zipped file on Google Classroom.