NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES
CL 103 – OBJECT ORIENTED PROGRAMMING LAB

Instructors: Mr. Basit Ali, Ms. Farah Sadia, Mr. Syed Zain ul Hassani,
Mr Muhammad Fahim

Email: basit.jasani@nu.edu.pk , farah.sadia@nu.edu.pk, zain.hassan@nu.edu.pk, m.fahim@nu.edu.pk

Lab # 01

Outline
- Introduction to Object Oriented Programming
- Why Object Oriented Programming is used?
- Difference between C and C++
- Explanation of basic C++ program
- Common Escape Sequence
- Data types in C++
- Operators in C++ (*arithmetic operators*, *relational operators*, *logical operators*, *increment and decrement operators*, *assignment statement*)
- Decision Statement in C++ (*if statement*, *if else statement*, *nested if statement*, *else if statement*, *switch statement*)
- Iterative Statements in C++ (*the for loop*, *the while loop* and *the do-while loop*)
- Exercise

INTRODUCTION TO OBJECT ORIENTED PROGRAMMING

- Object Oriented Programming (OOP) is a programming concept used in several modern programming languages, like C++, Java and Python.
- Object Oriented Programming works on the principle that objects are the most important part of a program. Manipulating these objects to get results is the goal of Object Oriented Programming.
- In OOP, the data is grouped together with the methods that operate upon it, which makes it easy to maintain the integrity of the data and provide a structure to the program.

Question: So what is an object oriented program exactly?

Answer: It is a program that contains objects, of course, which have certain properties and have methods linked to them. These methods are used to obtain a programming result.

WHY IS OOP NEEDED?

Problems with Procedural Languages

- Functions have unrestricted access to global data

- Unrelated Functions and data.

Before Object Oriented Programming programs were viewed as procedures that accepted data and produced an output. There was little emphasis given on the data that went into those programs.

DIFFERENCE BETWEEN C AND C++

The key differences include:

C	C++
It is a structural or procedural programming language.	It is an object oriented programming language.
Emphasis is on procedure or steps to solve a problem	Emphasis is on objects rather than procedure
Functions are the fundamental building blocks.	Objects are the fundamental building blocks.
In C, the data is not secured.	Data is hidden and can't be accessed by external functions.
C uses scanf() and printf() functions for standard input and output.	C uses cin>> and cout<< functions for standard input and output.
In C, namespace feature is absent.	In C++, namespace feature is present.
C program file is saved with .C extension.	C++ program file is saved with .CPP extension.

Table 1: Difference between C and C++

EXPLANATION OF BASIC C++ PROGRAM

An Example C++ Program

/* Comments can also be written starting with a slash followed by a star, and ending with a star followed by a slash. As you can see, comments written in this way can span more than one line. */ /* Programs should ALWAYS include plenty of comments! */ /* This program prints the table of entered number */

```
#include <iostream>
using namespace std;
int main()
{
        int input_num;
        //the number whose
        cout<<"Enter number";
        cin>>input_num;
        for (int i=0;i<=10;i++)
        {
                int output = input_num*i;
                cout<<input_num<<"*"<<i<<"="<<output<<endl;
        }
        return 0;
}
```

Program Output if 3 is entered as input to input_num

```
Enter number 3
3*0=0
3*1=3
3*2=6
3*3=9
3*4=12
3*5=15
3*6=18
3*7=21
3*8=24
3*9=27
3*10=30

--------------------------------
Process exited after 3.253 seconds with return value 0
Press any key to continue . . .
```

> ### *The #include Directive*

The #include directive causes the contents of another file to be inserted into the program Preprocessor directives are not C++ statements and do not require semicolons at the end

> ### *Using namespace std;*

The names cout and endl belong to the std namespace. They can be referenced via fully qualified namestd::cout and std::endl, or simply as cout and endl with a "using namespace std;" statement.

> ### *return0;*

The return value of 0 indicates normal termination; while non-zero (typically 1) indicates abnormal termination. C++ compiler will automatically insert a "return 0;" at the end of the the main () function, thus, it statement can be omitted.

> ### *Output using cout*

- Cout is an object
- Corresponds to standard output stream
- << is called insertion or input operator

> ### *Input With cin*

- Cin is an object
- Corresponds to standard input stream
- >> is called extraction or get from operator

Character	Name	Description
//	double slash	Marks the beginning of a comment
#	Pound sign	Marks the beginning of a preprocessor directive
<>	Opening and closing brackets	Encloses a filename when used with the #include directive
( )	Opening and closing parenthesis	Used in naming a function, as in int main ()
{ }	Opening and closing braces	Encloses a group of statements, such as the contents of a function.
" "	Opening and closing	Encloses a string of characters, such as a message that

	quotation marks	is to be printed on the screen
;	Semicolon	Marks the end of a complete programming statement

Table 2: Mandatory symbols in basic program

COMMON ESCAPE SEQUENCES

Escape Sequence	Name	Description
\n	Newline	Causes the cursor to go to the next line for subsequent printing
\t	Horizontal tab	Causes the cursor to skip over to the next tab stop
\b	Backspace	Causes the cursor to back up, or move left one position
\r	Return	Causes the cursor to go to the beginning of the current line, not the next line
\\	Backslash	Causes a backslash to be printed
\'	Single quote	Causes a single quotation mark to be printed
\"	Double quote	Causes a double quotation mark to be printed

Table 3: Escape Sequence

DATATYPES

There are many different types of data.
Variables are classified according to their data type, which determines the kind of information that may be stored in them. Integer variables only hold whole numbers.

Data Type	Size	Range
short	2 bytes	-32,768 to +32.767
unsigned short	2 bytes	0 to +65,535
Int	4 bytes	-2,147,4833,648 to +2,147,4833,647
unsigned int	4 bytes	0 to 4,294,967,295
Long	4 bytes	-2,147,4833,648 to +2,147,4833,647
Unsigned long	4 bytes	0 to 4,294,967,295

Table 4: Data types and size

Other Data Types

- *Char Data Type*

- Usually 1 byte long
- Internally stored as an integer
- ASCII character set shows integer representation for each character
- 'A' == 65, 'B' == 66, 'C' == 67, etc
- Single quotes denote a character, double quotes denote a string

- *Boolean Data Type*

Boolean variables are set to either true or false

OPERATORS

There are many operators in C++ for manipulating data which include arithmetic Operators, Relational Operators, Logical operators and many more which will be discussed accordingly.

➤ *Arithmetic Operators*

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulo

Table 5: Arithmetic Operators

➤ *Relational Operators*

Operator	Description
= =	Equals to
!=	Not Equals to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

Table 6: Relational Operators

➤ *Logical Operators*

Operator	Description
&&	Logical AND
||	Logical OR
!	NOT

Table 7: Logical Operators

➤ *Increment and Decrement Operators*

C++ introduces increment and decrement operators which are ++ and – respectively. These operators increment/decrement 1 in the operand's value.

For example: x++ will be equivalent to x=x+1 or x+=1.

The special characteristic of these operators is that they can be used for pre-increment as well as post-increment. To understand, consider the following statements:

A=b++; *//The statement will assign the contents of b to A and then increments the value of b by 1*
A=++b; *//The statement will first increment the value of b by 1 and then assign the new value to A.*

➤ *Assignment statements*

value = 5; *//This line is an assignment statement*

The assignment statement evaluates the expression on the right of the equal sign then stores it into the variable named on the left of the equal sign. The data type of the variable was in integer, so the data type of the expression on the right should evaluate to an integer as well.

DECISIONS

Sometimes, we want a program to choose among several possible alternative courses of action.
This means that some statements in the program may not be executed.
The choice between alternatives is based on some
condition A condition is either true or false
Use relational and/or logical operators to express a condition
Following are the main types of decision statements:

Statement	Description
If statement	An if statement consists of a boolean expression followed by one or more statement
If… else statement	An if statement can be followed by an optional else statement, which executes when the boolean expression is false
nested if statements	You can use one if or else if statement inside another if or else if statement(s)
Switch statement	A switch statement allows a variable to be tested for equality against a list of values
Nested Switch statement	You can use one swicth statement inside another switch statement(s)

if statement

- *Single statement if condition*

if (expression)
          statement;

Statement will be executed only if expression is true

Sample Program

```
#include <iostream>
using namespace std;
int main()
{
        int x;
        cin >> x;
        if (x == 5)
```

```
            cout <<"Condition is true and the value of x is "<<x;
        return 0;
}
```

Program Output is 5 is enteredas input to x

```
5
Condition is true and the value of x is 5
----------------------------------
Process exited after 1.945 seconds with return value 0
Press any key to continue . . .
```

Program output if other than 5 is entered as input to x

```
3

----------------------------------
Process exited after 1.758 seconds with return value 0
Press any key to continue . . .
```

- *Compound statement if condition*

Often, we want to execute several statements if a condition is true. Use braces to indicate the block of statements to be executed.

if (expression)
{
*   statement1;*
*   statement2;*
}

Sample Program

```
#include <iostream>
using namespace std;
int main()
{
        int x;
        cin >> x;
        if (x == 5)
        {
                cout <<"Condition is true and the value of x is
                "<<x; cout <<"\nWelcome to NUCES-FAST";
        }

        return 0;
}
```

Program Output is 5 is enteredas input to x

```
5
Condition is true and the value of x is 5
Welcome to NUCES-FAST
----------------------------------
Process exited after 1.887 seconds with return value 0
Press any key to continue . . .
```

if else statement

if (expression)
        statement1;
else
        statement2;

Statement1 will be executed if expression is true Statement 2 will be executed if expression is false.
Both statements will never be executed.

Sample Program

```
#include <iostream>
using namespace std;
int main()
{
        int x;
        cin >> x;
        if (x == 5)
        {
                cout <<"Condition is true and the value of x is "<<x;
        }
        else
        {
                cout <<"Condition is false.";
        }
        return 0;
}
```

Program Output is 5 is enteredas input to x

```
5
Condition is true and the value of x is 5
----------------------------------
Process exited after 7.017 seconds with return value 0
Press any key to continue . . .
```

Program output if other than 5 is entered as input to x

```
3
Condition is false.
---------------------------------
Process exited after 1.671 seconds with return value 0
Press any key to continue . . .
```

Nested if statement

Possible to put one if or if-else statement inside another if or if-else statement

```
if (expression1)
{
        statement1;
        if (expression2)
        {
                statement2;
        }
}
```

Sample Program

```
#include <iostream>
using namespace std;
int main()
{
        int age;
        cout<<"Enter your age: ";
        cin >> age;
        char gender;
        cout<<"\nEnter your gender i.e, M for male and F for female: ";
        cin >> gender;
        if (age < 4)
        {
                if(gender=='M')
                cout <<"\nA baby boy";

                else
                cout <<"\nA baby girl";
        }
        return 0;
}
```

Program output if age is less than 4 and gender is male

```
Enter your gender i.e, M for male and F for female: M

A baby boy
---------------------------------
Process exited after 5.248 seconds with return value 0
Press any key to continue . . .
```

Else if statement

if (expression1)
          statement 1;
 else if (expression 2)
            statement 2;
...
else
          statement n;

Sample Program

```
#include <iostream>
using namespace std;
int main( )
{
        int percentage;
        cin >> percentage;
        if (percentage >= 50)
                cout << "You have passed";
        else if (percentage < 50)
                cout << "Try your best to clear the couse in next attempt."
        return 0;

}
```

Program output if percentage is greater than or equals to 50

```
55
You have passed
---------------------------------
Process exited after 3.106 seconds with return value 0
Press any key to continue . . . ▄
```

Multiple conditions can be written by making several else-is clauses. Once a condition is true, control will never go to other else-if conditions. You can also add an else clause after else if statements.

ITERATIVE STATEMENTS (LOOPS)

Loops repeat a statement a certain number of times, or while a condition is fulfilled. They are introduced by the keywords while, do, and for.

➢
   The for loop

The for loop is designed to iterate a number of times.

Its syntax is:

for (initialization; condition; increase)
statement;

Sample Program

```
#include <iostream>
using namespace std;
int main()
{
        for (int n=10; n>0; n--)
        {
                cout << n << ", ";
        }
        cout << "\nEnd of for loop\n";
}
```

Program output

```
10, 9, 8, 7, 6, 5, 4, 3, 2, 1,
End of for loop

------------------------------------
Process exited after 0.01807 seconds with return value 0
Press any key to continue . . . ▄
```

➢ ***The while loop***

The simplest kind of loop is the while-loop.

Its syntax is:

while (expression
statement;

The while-loop simply repeats statement while expression is true. If, after any execution of statement, expression is no longer true, the loop ends, and the program continues right after the loop.

Sample Program

```
#include <iostream>
using namespace std;
int main()
{
        int n = 10;
        while (n>0)
        {
                cout << n << ", ";
                --n;
        }
        cout << "\nEnd of while loop";
}
```

Program output

```
10, 9, 8, 7, 6, 5, 4, 3, 2, 1,
End of while loop
----------------------------------
Process exited after 0.0153 seconds with return value 0
Press any key to continue . . .
```

➢ *The do-while loop*

A very similar loop is the do-while loop, whose syntax is:

do
  statement;
while (condition);

It behaves like a while-loop, except that condition is evaluated after the execution of statement instead of before, guaranteeing at least one execution of statement, even if condition is never fulfilled.

Sample Program

```
#include <iostream>
#include <string>
using namespace std;
int main ()
{
        string str;
        do
        {
                cout << "Enter text: ";
                getline (cin,str);
                cout << "You entered: " << str << '\n';
        }
        while (str != "goodbye");
}
```

Program Output

```
Enter text: ITC Section D
You entered: ITC Section D
Enter text: ITC Section A
You entered: ITC Section A
Enter text: ITC Section B
You entered: ITC Section B
Enter text: ITC Section C
You entered: ITC Section C
Enter text: goodbye
You entered: goodbye


----------------------------------
Process exited after 46.12 seconds with return value 0
Press any key to continue . . .
```

Activities

Question # 01: Write a program to find a student's average marks. The program should ask the user to enter the number of courses he has taken and the total marks he has obtained thus far. It should then display his average to 4 decimal places.

Question # 02: Write a program that generates the following output using /= and += operators.

100
50
52
26
28

Question # 03: Write a function power to raise a number m to power n. The function takes a double value for m and int value for n. Use default value for n to make the function to calculate squares when this argument is omitted.

Question # 04: If you have two fractions, a/b and c/d, their sum can be obtained from the
formula a/b + c/d = a*d + b*c / b*d
For example, 1/4 + 2/3 is
1/4 + 2/3 = 1*3 + 4*2 / 4*3 = 3+8 / 12 = 11/12
Write a program that encourages the user to enter two fractions and then displays their sum in fractional form.
The interaction with the user might look like this:

Enter first fraction:
1/2   Enter   Second
Fraction: 2/5 Sum =
9/10

Question # 05: Write a program for multiplication of two matrices.

Question # 06: Write a function named "swap_floats" that takes two floating point arguments and interchanges the values that are stored in those arguments. The function should return no value. To take an example,
if the following code fragment is executed
float x = 5.8, y = 0.9;
swap_floats (x, y);
cout << x << " " << y << endl;
then the output will be 0.9 5.8

Question # 07: Write a statement (or comment) to accomplish each of the following (assume that using directives have been used for cin, cout and endl):

   a.   State that a program calculates the product of three integers.
   b.   Declare the variables x,y,z and result to be of type int (in separate statements).
   c.   Prompt the user to enter three integers.
   d.   Read three integers from the keyboard and store them in the variables x, y and z.
   e.   Compute the product of the three integers contained in variables x, y and z, and assign the result to the variable result.
   f.   Print "The product is" followed by the value of the variable result.

Question # 08: Create the equivalent of a four-function calculator. The program should ask the user to enter a number, an operator, and another number. (Use floating point.) It should then carry out the specified arithmetical operation: adding, subtracting, multiplying, or dividing the two numbers. Use a switch statement to select the operation. Finally, display the result.

```
                         CALCULATOR
1-     addition
2-     subtraction
3-     division
4-     multiplication
5-     modulo
6-     square
7-     square root
8-     cube root
9-     exponent
10-    logarithm
11-    sinusoidal
12-    co-sinusoidal
13-    tangent
14-    absolute
Enter option to perform operation: 8
Enter number to find cube root: 27
Cube root of 27 is: 3
```

Question # 09:

Using IF and Switch statement, write a program that displays the following menu for the food items available to take order from the customer:

B= Burger (Rs. 200)

F= French Fries (Rs. 50)

P= Pizza (Rs. 500)

S= Sandwiches (Rs. 150)

The costumer can order any combination of available food. The program first ask to enter the no of types of snacks i.e. 2, 3 or 4 then it ask to enter the choice i.e. B for Burger and then for quantity. The program should finally display the total charges for the order.

Question # 10:
Bob's Discount Bolts charges the following prices:
5 cents per bolt
3 cents per nut
1 cent per washer
Write a program that asks the user for the number of bolts, nuts, and washers in their purchase and then calculates and prints out the total. As an added feature, the program checks the order. A correct order must have at least as many nuts as bolts and at least twice as many washers as blots, otherwise the order has an error. For an error the program writes out "Check the Order: too few nuts" or "Check the Order: too few washers" as appropriate. Both error messages are written if the order has both errors. If there are no errors the program writes out "Order is OK."

Number of bolts:	**12**
Number of nuts:	**8**
Number of washers:	**24**
Check the Order:	**too few nuts**

Total cost: **108**

Question # 11: Suppose you give a dinner party for six guests, but your table seats only four. In how many ways can four of the six guests arrange themselves at the table? Any of the six guests can sit in the first chair. Any of the remaining five can sit in the second chair. Any of the remaining four can sit in the third chair, and any of the remaining three can sit in the fourth chair. (The last two will have to stand.) So the number of possible arrangements of six guests in four chairs is 6*5*4*3, which is 360.

Write a program that calculates the number of possible arrangements for any number of guests and any number of chairs. (Assume there will never be fewer guests than chairs.) Don't let this get too complicated. A simple for loop should do it.

Question # 12: Write a C program to read an amount (integer value) and break the amount into smallest possible number of bank notes. Note: The possible banknotes are 100, 50, 20, 10, 5, 2 and 1.

Question # 13: Sequence is defined as: 2, 6, 14, 30, 62.....

Write a program to build above sequence.

Home Work:

Question # 01:
Write a program in C++ to read any Month Number in integer and display the number of days for this month.

Question # 02:
write a program to convert a number into sentence e.g.

input : 123

Output : **One hundred and twenty three**

Question # 03
A certain grade of steel is graded according to the following conditions
1. Hardness must be greater than 50
2. Carbon content must be less than 70
3. Tensile strength must be greater than 5600

The grades are as follows:
- Grade is 10 if all three conditions are met.
- Grade is 9 if conditions (i) and (ii) are met.
- Grade is 8 if conditions (ii) and (iii) are met.
- Grade is 7 if conditions (i) and (iii) are met.
- Grade is 6 if only one condition is met.
- Grade is 5 if none of the conditions are met.

Write a program, which will require the user to give values of hardness, carbon content and tensile strength of the steel under consideration and output the grade of the steel.