**OOP Lab List**

**Unit 2:**

1. Write a program in C++ to print “Hello World”
2. Write a program in C++ that reads a numbers and then find its reverse.
3. Write a C++ program that reads radius of circle and finds area and circumference.
4. Write a program in C++ that reads a number and then check whether is prime or composite
5. Write a program in C++ that prints prime numbers from 300 to 500.
6. Write a program in C++ that reads two numbers **n1** and **n2** and then print all prime numbers between **n1** and **n2**
7. Write a C++ program that computer simple interest using default arguments function
8. Write a C++ program that read that finds cube of a number using inline function
9. Write a program in C++ to find product of two 3\*3 matrices.
10. Write a program in C++ to find product of two numbers using recursive function
11. Write a program in C++ to find sum of digits of a number using recursive function.
12. Write a C++ program to read three numbers and print the following results
13. Sum of the values.
14. Average of three values.
15. Largest of the three.
16. Smallest of the three.
17. Write a C++ program to read two integer values **m** and **n** and to decide and print whether m is a multiple of n or not.
18. Write a program in C++to compute the area of circle and area of triangle by overloading the area () function.
19. Write a program in C++ to read two matrices of size m\*n from the keyboard and then find sum and product of these two matrices. Use function to read, to find sum and to find product.
20. Write a function called zeroSmaller () that is passed two int arguments by reference and then sets the smaller of two numbers to 0. Write a main () program to exercise this function.
21. Write a function **power ()** to raise a number **m** to a power **n**. The function takes a double value for **m** and int value for **n**, and returns the result correctly. Use a default value of 2 for n to make the function to calculate squares when this argument is omitted. Write a main that gets the value of **m** and **n** from the user to test the function.

**Unit 3**

1. Create a class called Distance (meter, cm). Add suitable constructors. Add two member functions display() that prints value of attributes of Distance object and compare(Distance ) that compared two Distance objects. In side main() function create two objects of Distance and them compare them to print larger one
2. Write a C++ program to add two complex objects by using suitable constructors
3. Create a class **Number** with two int instance variable x and y. The class will have one constructor. The class also will contain member function **getMax ()** that will return larger number. From main () function create an object of **Number** and print the larger number.
4. Create a class called **employee** that contains a name (an array of characters) and an employee number (type long). Include the member function called **getData ()** to read data from the user for insertion into object, and another function called **putData()** to display the data. Assume the name has no embedded spaces.

Write a main () program to exercise this class. It should create an array of type employee and then invite the user to input data for up to 100 employees. Finally, it should print out the data for all employees

1. Create class named ‘student’ with member ‘rollno’, ‘name’ and ‘marks’. Write a C++ program that reads records of three students and then display that information on monitor.
2. Write a program that calculates the average of up to 100 English distance input by the user. Create an array of objects of the Distance class.
3. Write a program in C++ to demonstrate dynamic constructor.
4. Create two classes **DM** and **DB** which store the value of distances. **DM** stores distances in meters and centimeters and **DB** in feet and inches. Write a program that can read values for the class objects and add one object of DM with another object of **DB.** Use friend function to carry out the addition operation. The object that stores the results may be a DM object or DB object, depending on the units in which the results are required. The display should in the format of feet and inches or meters and centimeters depending on the object on display.
5. Create a **USMoney** class with two integer instance variables **dollars** and **cents.** Add a constructor with two parameters for initializing a **USMoney** object. The constructor should check that the cent value is between 0 and 99 and, if not, transfer some cents to the dollars variables to make it between 0 and 99. For example, if **x** is a **USMoney** object with 5 dollars and 80 cents, and if **y** is a **USMoney** object with 1 dollar and 90 cents, then **x.plus (y)** will return a new USMoney object with 7 dollars and 70 cents. From a **main ()** function create two objects of **USMoney** class and add them.
6. Create a class **Box** with fields width, height and depth. Add methods **getArea ()** and **getVolume ()**. Use suitable constructors. From main () function create an object of **Box** class and find its area as volume.

Unit 4

1. Create a class **Room** with instance variables **length** and **breadth**. Add one function **getArea ()** that returns the area of the room. Create a subclass **MyRoom** and add one instance variable **height**. Add one function **getVolume ()** that returns the volume. From **main ()** function create two **MyRoom** objects and find area and volumes of both rooms.
2. Define a **Shape** class (with necessary constructors and member functions) in Object-Oriented Programming (abstract necessary attributes and their types). (Write a complete code in C++ programming language)

Derive **Triangle and Rectangle** classes from Shape class adding necessary attributes.

Use these classes in a main function and display the area of triangle and rectangle.

1. Create a class **Box** with instance variables **length**, **breadth** and **height**. Add one function **getVolume ()** to compute the volume of box. Use suitable constructors. Create a subclass **BoxWeight** and add one variable **weight**. Add one function **getWeight ()** that displays the weight of box to this class. Add suitable constructors. Create one more subclass class **Shipment** from **BoxWeight**. Add one function **getCost ()** that displays the cost of the box. Add suitable constructors. From **main ()** function create an object of **Shipment** that initializes the instance variables through constructor.
2. Create a class called lecture that stores ID and name of lectures. From this class derive two classes: part time, which adds payperhr (type float): and full time, which adds paypermonth(type float). Each of these three classes should have a readdata () function to get its data from user at the key board and printdata() function to display the data.

Write a main() program to test the Full time and Part time classes by creating instance of them asking the user to fill their data with readdata () and display the data with printdata().

1. Define a **student** class (with necessary constructors and member functions) in Object Oriented Programming (abstract necessary attributes and their types). (Write a complete code in C++  programming language).

Derive a **computer Science and Mathematics** class from **student** class adding necessary attributes (at least three subjects).

Use these classes in a main function and display the average marks of computer science and mathematics students.

1. Write a program according to the specification given below:

 Create a class Teacher with data members tid & subject and ember functions for reading and displaying data members. Create another class Staff with data members sid & position, and member function for reading and displaying data members. Derive a class Coordinator from Teacher and Staff and the class must have its own data member department and member functions for reading and displaying data members. Create two object of Coordinator class and read and display their details.

1. Create class called Time (hr , min, sec). Write a C++ program that create three Time objects t1(1,40,50) ,t2(2,40,50) and t3(0,0,0) and then add then add t1 and t2 and store result in t3 . Finally display object t3

**Unit 5**

1. Write a program in C++ that overloads unary operator ++
2. Write a program in C++ that overloads unary operator - -
3. Write a program in C++ that overloads unary operator -
4. Write a program in C++ that add two complex numbers by overloading binary operator +
5. Write a program in C++ that finds difference of two complex numbers by overloading binary operator -
6. Write a program in C++ that finds product two complex numbers by overloading binary operator \*
7. Write a program in C++ that adds two distances (feet, inch) by overloading binary operator +
8. Write a program in C++ that finds difference of two distances (feet, inch) by overloading binary operator -
9. Write a program in C++ to add two times (hr, min, sec) objects by overloading + operator
10. Write a program in C++ to concatenate two string by overloading binary operator ‘+’ using friend function
11. Write a program in C++ that finds difference of two complex numbers by overloading binary operator – using friend function
12. Write a program in C++ that add two complex numbers by overloading binary operator + using friend function
13. Write a program in C++ that finds product two complex numbers by overloading binary operator \* using friend function
14. Write a program in C++ that adds two distances (feet, inch) by overloading binary operator + using friend function
15. Write a program in C++ that finds difference of two distances (feet, inch) by overloading binary operator - using friend function
16. Write a program in C++ to add two times (hr, min, sec) objects by overloading + operator using friend function
17. Write a program in C++ to convert an object of one class into an object of another class using conversion routine in source class
18. Write a program in C++ to convert an object of one class into an object of another class using conversion routine in destination class

**Unit 6:**

1. Write a C++ program to demonstrate function overriding
2. Write a C++ program to demonstrate virtual function
3. Write C ++ program to demonstrate the use of “this” pointer

**Unit 7**

1. Write a C++ program to handle divide by zero exception
2. Write a C++ that handles all types of exceptions
3. Write a C++ program to demonstrate multiple catch statements
4. Write a C++ program to sort 10 numbers using template function
5. Write a C++ program create template class called stack. Implement two functions push and pop

**Unit 8**

1. Write a C++ program write lines of text to a file
2. Write a C++ program that reads lines of text from file
3. Write a C++ program that copies content of one file to another file
4. Write a Program in C++ that write an object to file and reads the objects from file
5. Write a program in C++ for reading and writing multiple objects to file and from file