

**LAB MANUAL**

**(OBJECT ORIENTED PROGRAMMING)**

**COMP2111**

## Submitted By

Name:Ammara Sajid

Reg#: 21-UE-02160

**Submitted To**

Ma’am Uzma Umar

## CONTENTS

Introduction --------------------------------------------------------------------------------3

Lab1: Recall ------------------------------------------------------------------------------------4

Lab2: Class(Basic) ---------------------------------------------------------------6

Lab3: Class ------------------------------------------------------------------------------------9

Lab4: Array of objects------------------------------------------------------------------------12

Lab6: Composition/Aggregation--------------------------------------------------------------18

Lab9: Inheritance -------------------------------------------------------------------------------21

Lab10: Multiple Inheritance -----------------------------------------------------------------25

Lab11:Polymorphism---------------------------------------------------------------------------29

Lab12:Abstract class polymorphism---------------------------------------------------------------------------33

# INTRODUCTION

**T**he objective of this lab manual is to give students step-by-step examples to become familiar with programming concepts, design, and coding.

### F E AT U R E S

To ensure a successful experience for instructors and students alike, this lab manual includes the following features:

* **Lab Objectives**—Every lab has a brief description and list of learning objectives.

* **Materials required—**Every lab includes information on software you will need to complete the lab.

* **Completion Times**—Every lab has an estimated completion time so that you can plan your activities more accurately.

* **Activity Sections**—Labs are presented in manageable sections; where appropriate, additional Activity Background information is provided to illustrate the importance of a particular activity.

* **Step-by-Step Instructions**—Every lab provides steps to enhance technical proficiency; some labs include Critical Thinking exercises to challenge students.

* **Review Questions**—Some labs include review questions to help reinforce concepts presented in the lab.

* **SOFTWARE REQUIREMENTS** —Computer running Windows XP, Recommended compiler is DEV C++ 5.11 version and web-based IDE Code board.

# Lab1: RECALL

**Activity No.1** Date: Sept.29,2022

/\*write a C++ program that defines two functions: input and processing. The input function

is meant to take input of 20 integers in array and the processing function displays the number of even and odd numbers in that array\*/

#include<iostream>

using namespace std;

void input(int a[]);

void EvenOdd(int a[]);

int main()

{

int a[20];

int size=20;

input(a);

EvenOdd(a);

return 0;

}

void input(int a[20])

{

cout<<"Enter 20 integers:";

for(int i=0;i<20;i++)

{

cin>>a[i];

}

}

void EvenOdd(int a[])

{

int even=0;

int odd=0;

for(int i=0;i<20;i++)

{

if(a[i]%2!=0)

odd++;

else

even++;

}

cout<<odd<<" integers are odd"<<endl;

cout<<even<<"integers are even"<<endl;

}

**Output of the Program:**

Enter 20 integers:987654321123456789

5 integers are odd

15integers are even

**Activity No.2**

|  |
| --- |
|  |

Date: Sept.29,2022

/\*

C++ program to check prime number. Identify and correct error.

Prime numbers are those numbers which are divisible by itself only. Here, we will read an integer number and check whether it is Prime or Not,

to check prime number we implemented a function isPrime() that will take integer number as argument and return 1 if it is prime else it return 0.\*/

#include <iostream>

using namespace std;

int isPrime(long n);

int main()

{

int num;

cout<<"Enter an integer number: ";

cin>>num;

if(isPrime(num))

cout<<num<<" is a prime number"<<endl;

else

cout<<num<<" is not a prime number"<<endl;

return 0;

}

int isPrime(long n)

{

int prime=1;

for(int i=2;i<(n-1);i++)

{

if(n%i==0)

{

prime=0;

break;

}

}

return prime;

}

**Output of the Program:**

Enter an integer number: 13

13 is a prime number

# Lab2: CLASS(BASIC)

# Activity No. 1

Date: Sept.29,2022

/\*Write a C++ program that defines a class for employee. The class should have four data members for employee's name, Id, age in year 2020, and retirement year.

Define three member functions in the class one of which should take input of first three data members,

the second method should calculate the retirement year for the employee provided the retirement age is 60 years.

The third method should display the name, Id, age and retirement year of the employee.\*/

#include<iostream>

using namespace std;

class employee{

char name[50];

int id;

int age;

int Retire;

public:

void TakeData()

{ cout<<"enter the name of employee"<<endl;

cin>>name;

cout<<"\nenter id"<<endl;

cin>>id;

cout<<"\nenter age"<<endl;

cin>>age;

}

void calculate()

{

int year=60-age;

Retire=2020+year;

}

void output()

{

cout<<"\n\nname of the employee "<<name<<endl;

cout<<"\nid of employee "<<id<<endl;

cout<<"\nage of employee in 2020 "<<age<<endl;

cout<<"\nemployee retirement year "<<Retire;

}

};

int main()

{

employee e1;

e1.TakeData();

e1.calculate();

e1.output();

return 0;

}

**Output of the Program:**

Enter the name of employee

Maro

Enter id

13

Enter age

19

Name of the employee Maro

Id of employee 13

Age of employee in 2020 19

Employee retirement year 2061

**Activity No. 2**

## Activity No. 2

Date: Sept.29,2022

//class, constructor

#include <iostream>

using namespace std;

class Car {

public:

int speed(int maxSpeed);

};

int Car::speed(int maxSpeed) {

return maxSpeed;

};

int main() {

Car myObj;

cout << myObj.speed(200);

return 0;

}

**Output of the Program:**

200

# Lab3: CLASS

## Activity No. 1

Date: Oct. 20,2022

/\*Write a program that creates a class to display Utility\_Bill. The class should take customer name, bill number, units consumed, and customer address as inputs and displays the input values and the bill amount which is calculated after multiplying the units consumed with per unit price.

If units consumed are less than 300, the per unit price is Rs. 8; if greater than or equal to 300 but less than 1000, the per unit price is Rs. 5. If units consumed are greater than 999, the per unit price is Rs. 14.\*/

#include <iostream>

using namespace std;

class U\_bill

{

private:

char c\_name[20];

int units;

double bill;

char c\_address[40];

public:

void get()

{

cout<<"Enter Details of Customer Below : \n" <<endl;

cout<<"Enter Customer Name : ";

cin>>c\_name;

cout<<"\nEnter No. of Units used : ";

cin>>units;

**Output of the Program:**

Enter customer’s name : Maro

Enter bill number : 13

Enter consumed units : 2003

Enter address : Muslim town

Total amount to be paid : 28042

## La Activity No. 2

Date: Oct. 20,2022

# b4: Selection structure

/\* This program implements a class of Book and displays the data of three books entered by user.

identify and correct errors\*/

#include <iostream>

using namespace std;

class book

{

string title;

float price ;

public:

void getdata ();

void putdata ();

} ;

void book:: getdata ()

{

cout<<"Title: "<<endl;

cin>>title;

cout<<"Price: "<<endl;

cin>>price;

**Output of the Program:**

Enter details o£ book 1

Title:Turestory

Price:500

Enter details o£ book 2

Title:Honesty

Price:300

Enter details o£ book 3

Title:Fake story

Price:

Book 1

Title:Turestory

Price:500

Book 2

Title:Honesty

Price:300

# Lab4: ARRAY OF OBJECTS

## Activity No. 1

Date: Oct. 20,2022

/\* Write a C++ program that creates a class to take and display details of student. It takes input for student's name, id, marks, and CGPA.

Create an array of objects of 8 students and display the details of student who took highest marks among the 8 students.\*/

#include<iostream>

using namespace std;

class student

{

public:

string name;

int id;

int marks;

float CGPA;

void getdata(){

cout<<"enter student name: "<<endl;

cin>>name;

cout<<"enter student id: "<<endl;

cin>>id;

cout<<"enter student marks: "<<endl;

cin>>marks;

cout<<"enter student CGPA: "<<endl;

cin>>CGPA;

}

void putdata(){

cout<<endl<<name<<endl<<id<<endl<<marks<<endl<<CGPA<<endl;

}

};

int main()

{

int i;

student s[8];

for(i=0;i<8;i++)

{

cout<<"enter details of "<<i+1<< "student"<<endl;

s[i].getdata();

}

cout<<"details of students";

for(i=0;i<8;i++)

s[i].putdata();

for(i = 1;i <8; ++i)

{

if(s[0].marks < s[i].marks)

s[0] = s[i];

}

cout << "Student with the highest marks " ;

s[0].putdata();

return 0;

}

**Output of the Program:**

Student 1

Enter name

Maro

Enter id

13

Enter marks

996

Enter CGPA

3.1

Student 2

Enter name

Hafsa

Enter id

15

Enter marks

1026

Enter CGPA

3.7

Student 3

Enter name

Hira

Enter id

16

Enter marks

906

Enter CGPA

3.3

Student 4

Enter name

Noor

Enter id

13

Enter marks

1000

Enter CGPA

3.5

Student 5

Enter name

Ammara

Enter id

12

Enter marks

922

Enter CGPA

3.3

Student 6

Enter name

Nasira

Enter id

17

Enter marks

800

Enter CGPA

3.4

Student 7

Enter name

Zara

Enter id

19

Enter marks

1067

Enter CGPA

3.8

Student 8

Enter name

Zanib

Enter id

17

Enter marks

1056

Enter CGPA

3.2

Heighest marks scored : 1067

## Activity No. 2

Date: Oct. 20,2022

// This program creates a user defined type Employee. It takes and displays data of 5 employees. (correct errors)

#include<iostream>

using namespace std;

class Employee

{

int Id;

char name[25];

int Age;

long Salary;

public:

void GetData()

{

cout<<"\n\tEnter Employee Id : ";

cin>>Id;

cout<<"\n\tEnter Employee Name : ";

cin>>name;

cout<<"\n\tEnter Employee Age : ";

cin>>Age;

cout<<"\n\tEnter Employee Salary : ";

cin>>Salary;

}

void PutData()

{

cout<<"\n"<<Id<<"\t"<<name<<"\t"<<Age<<"\t"<<Salary;

}

};

int main()

{

Employee E[5];

int i;

for(i=0;i<=5;i++)

{

cout<<"\nEnter details of "<<i+1<<" Employee";

Employee i;

i.GetData();

}

cout<<"\nDetails of Employees";

for(i=0;i<=5;i++)

E[i].PutData();

}

**Output of the Program:**

Enter details of 1 Employee

Enter Employee Id : 13

Enter Employee Name : noor

Enter Employee Age : 19

Enter Employee Salary : 2

Enter details of 2 Employee

Enter Employee Id : 220000

Enter Employee Name : zara

Enter Employee Age : 18

Enter Employee Salary : 300000

Enter details of 3 Employee

Enter Employee Id : maro

# Lab6: COMPOSITION/AGGREGATION

### Activity No.1

Date: Nov. 2,2022

/\*Write a C++ program that creates a class named Car. It has an object of user defined type called Engine.

The Engine has engine\_number as a data member which is set in the constructor of class Engine.

The class Car has an aggregation relationship with another user defined class called Passenger which has passenger\_name as its data member.

Create an object of class Car that displays the engine number and a passenger name on screen.\*/

#include <iostream>

using namespace std;

class Engine{

int en;

public:

Engine (){

}

Engine (int i){

en=i;

}

int showEN(){

return en;

}

};

class Passenger{

string pass\_name;

public:

void setPassname(string pname){

pass\_name=pname;

}

string showPassName(){

return pass\_name;

}

};

class Car{

Passenger \*p;

Engine e;

public:

Car(Passenger P, Engine E){

p=&P;

e=E;

}

void showCar(){

cout << "ZSB" << p->showPassName() << endl;

cout << e.showEN();

}

};

int main ()

{

Passenger pa;

Engine eng=459;

Car c(pa,eng);

c.showCar();

return 0;

}

**Output of the Program:**

ZSB

459

### Activity No.2

Date: Nov. 2,2022

//This program is meant to post-increment a time's object and display the results

#include <iostream>

using namespace std;

class Time {

int hr, min;

public:

Time(int h, int m)

{

hr = h;

min = m;

}

void operator++(int)

{

hr++;

min++;

cout << "\n Hours and Minutes " <<hr<< min; } };

int main()

{

Time t1(4, 45);

t1++;

return 0;

}

**Output of the Program:**

Hours and Minutes 546

# Lab9: INHERITANCE

### Activity No.1

Date: Nov. 17,2022

/\*Write a C++ program that implements a class hierarchy having a base class named Employee and a derived class named Adhoc\_employee.

The base class has data members of Name, Id, Number of working hours and Per hour salary. The derived class implements two methods

named salary and display\_info. The salary method calculates the salary of employee by multiplying “Number of working hours” by “Per hour salary”. The display\_info method displays

the complete information of employee. Create an object of Adhoc\_employee and display the information and salary against that instance.\*/

#include <iostream>

using namespace std;

class Employee{

protected:

string Name;

string ID;

int working\_hours;

float per\_hour\_salary;

float Total\_salary;

public:

void set(){

cout<<"Enter the name id working hours and per hour salary "<<endl;

cin>>Name;

cin>>ID;

cin>>working\_hours;

cin>>per\_hour\_salary;

}

};

class Adhoc\_employee:public Employee{

public:

int salary(){

set();

Total\_salary=working\_hours\*per\_hour\_salary;

return Total\_salary;

}

void display(){

cout<<" Name : "<<Name<<endl;

cout<<" ID : "<<ID<<endl;

cout<<" Working Hours : "<<working\_hours<<endl;

cout<<" Total Salary : "<<Total\_salary<<endl;

}

};

int main(){

Adhoc\_employee AE;

AE.salary();

AE.display();

return 0;

}

**Output of the Program:**

Maro 13 8 2500

Name: Maro

ID: 13

Working Hours: 8

Total Salary: 20000

### Activity No.2

Date: Nov. 17,2022

/\* identify and correct errors. Also comment the identified error\*/

#include <iostream>

using namespace std;

class Person

{

protected:

string subject;

public:

void setsubject(string subjectname1)

{

subject= subjectname1;

}

};

class MathsTeacher : public Person

{

public:

void display()

{

cout << "Subject" << subject << endl;

}

};

int main()

{

MathsTeacher teacher;

string subjectname;

cout<<"Enter subject’s name";

cin>>subjectname;

teacher.setsubject(subjectname);

teacher.display();

return 0;

}

**Output of the Program:**

Enter subject’s nameMathematics

SubjectMathematics

# Lab10: MULTIPLE INHERITANCE

### Activity No.1

Date :Nov. 24 ,2022

/\*Write a C++ program that implements multiple inheritance through class hierarchy having four classes:

Course, IT, CS, SOS\_course where IT, CS are inherited from Course and SOS\_Course is inherited from IT and CS.

Course class has an attribute course Id and an input() method that takes input of course Id from user. Each child class in the hierarchy implements

a display() method that displays the course Id.

Declare an object of SOS\_Course and display the course Id set by user.\*/

#include<iostream>

using namespace std;

class Course{

protected:

int id;

public:

void input(){

cout<<"Enter the course id:"<<endl;

cin>>id;

}

};

class IT: virtual public Course {

public:

void display(){

cout<<"course id:"<<id<<endl;

}

};

class CS : virtual public Course{

public:

void display(){

cout<<"course id:"<<id<<endl;

}

};

class SOS\_Course: public IT, public CS{

public:

void display(){

cout<<"course id:"<<id<<endl;

}

};

int main(){

SOS\_Course co;

co.input();

co.display();

return 0;

}

**Output of the Program:**

Enter the course id:

1312003

Course id:1312003

### Activity No.2

Date :Nov. 24 ,2022

// Identify and correct errors ( also comment the errors).

#include <iostream>

using namespace std;

class P

{

protected:

int z;

public:

void setz(int q)

{

z=q;

}

};

class A

{

protected:

int x;

public:

void setx(int q)

{

x=q;

}

};

class B

{

protected:

int y;

public:

void sety(int q)

{

y=q;

}

};

class C : public P,public A,public B

{

public:

void sum()

{

cout << "Sum = " << x + y+z;

}

};

int main()

{

C obj1;

obj1.setz(3);

obj1.setx(4);

obj1.sety(5);

obj1.sum();

return 0;

}

**Output of the Program:**

Sum = 12

# Lab11: POLYMORPHISM

### Activity No.1

Date: Dec. 8, 2022

/\*Write a C++ program that defines four classes named: Shape, Triangle, Rectangle and Circle where Shape is parent class of rest of the three classes.

Class Shape has a method named area which is overridden in child classes.

This method calculates and displays the area of its respective type.

The program displays areas of all the shapes through polymorphism.

(area of triangle= (base\* height) /2: area of circle = 3.1415 \* (square of radius); area of rectangle = length \* width)\*/

#include <iostream>

using namespace std;

class Shape

{

protected:

double width, height ,radius,base;

public:

void set\_data (double a, double b,double c,double d)

{

width = a;

height = b;

radius = c ;

base = d ;

}

virtual double area()

{return 0;}

};

class Rectangle: public Shape

{

public:

double area ()

{

return (width \* height);

}

};

class circle: public Shape

{

public:

double area ()

{

return (3.1415 \* radius\*radius);

}

};

class triangle: public Shape

{

public:

double area ()

{

return (base \* height);

}

};

int main ()

{

Shape \*sPtr;

Rectangle Rect;

sPtr = &Rect;

sPtr -> set\_data (5,3,2,2);

cout <<"area of rectangle= "<< sPtr -> area() << endl;

circle cir ;

sPtr = &cir;

sPtr -> set\_data (5,3,2,2);

cout <<"area of circle= "<< sPtr -> area() << endl;

triangle tir ;

sPtr = &tir;

sPtr -> set\_data (5,3,2,2);

cout <<"area of triangle= "<< sPtr -> area() << endl;

return 0;

}

**Output of the Program:**

area of rectangle= 15

area of circle= 12.566

area of triangle= 6

**Activity No.2**

Date: Dec. 8, 2022

/\*This code is meant to polymorphically call two methods print and show.

Identify and correct any logical or compilation errors\*/

#include <iostream>

using namespace std;

class base

{

public:

void print ()

{ cout<< "print base class" <<endl; }

};

class derived:public base

{

public:

virtual void print ()

{ cout<< "print derived class" <<endl; }

virtual void show ()

{ cout<< "show derived class" <<endl; }

};

int main()

{

base \*bptr;

derived d;

bptr = &d;

d.print();

d.show();

return 0;

}

**Output of the Program:**

print derived class

show derived class

# Lab11: ABSTRACT CLASS POLYMORPHISM

### Activity No.1

Date: Dec.15, 2022

/\*Write a C++ program having three classes named: Vehicle, Four\_Wheeler, Two\_Wheeler where Vehicle is parent class of the latter two classes.

The Vehicle class has two data members named mileage and manufacturer. It also has a pure virtual function named

first\_maintenance() that displays appropriate message (in it's overriden versions in derived classes)

if the maintenance of vehicle is due or not according (maintenance is due if the mileage of vehicle reaches 5000 km or above for four wheeler

and 2000 km or above for two wheeler). Vehicle class also has a display function that displays the mileage and manufacturer of the vehicle.

Create one Four\_Wheeler object, named car, and one Two\_Wheeler object, named bike. Display the mileage and manufacturer of both vehicles

(car and bike) polymorphically. Also display appropriate messages in case the vehicles need maintenance or not.\*/

#include <iostream>

#include <string>

using namespace std;

class vehicle{

public:

float mileage;

string manufacturer;

virtual void first\_maintenance()=0;

virtual void display(){

cout << mileage << manufacturer;

}

};

class four\_wheeler : public vehicle{

public:

void first\_maintenance(){

cout << "enter mileage and manufacturer of car : ";

cin >> mileage >> manufacturer;

if (mileage >= 5000)

cout << "Maintenance is due!" << endl;

else

cout << "Your car is in a good condition!" << endl;

}

void display(){

cout << "Mileage of car : " << mileage << endl;

cout << "Manufacturer of car : " << manufacturer << endl;

}

};

class two\_wheeler : public vehicle{

public:

void first\_maintenance(){

cout << "enter mileage and manufacturer of bike : ";

cin >> mileage >> manufacturer;

if (mileage >= 2000)

cout << "Maintenance is due!" << endl;

else

cout << "Your bike is in a good condition!" << endl;

}

void display(){

cout << "Mileage of bike : " << mileage << endl;

cout << "Manufacturer of bike : " << manufacturer << endl;

}

};

void intermediate (vehicle \*v)

{

v->first\_maintenance();

v->display();

}

int main ()

{

four\_wheeler \*car=new four\_wheeler();

two\_wheeler \*bike=new two\_wheeler();

intermediate(car);

intermediate(bike);

delete car;

delete bike;

return 0;

}

**Output of the Program:**

Enter mileage and manufacturer of car : 13pardo

Your car is in a good condition!

Mileage of car : 13

Manufacturer of car : pardo

Enter mileage and manufacturer of bike : 1honda

Your bike is in a good condition!

Mileage of bike : 1

Manufacturer of bike : honda

**Activity No.2**

Date: Dec.15, 2022

/\*This program is intended to have an abstract class called Base which has a pure virtual function named show.

The function is overriden in Derived class and polymorphically called in main(). Identify and correct errors.\*/

#include<iostream>

using namespace std;

class Base

{

public:

virtual void show()=0;

};

class Derived: public Base

{

public:

void show() { cout << "In Derived \n"; }

};

int main()

{

Base \*bp;

bp = new Derived();

bp->show();

delete bp;

return 0;

}

**Output of the Program:**

In Derived

Lab work

:

#include<iostream>

Using namespace std;

Class Laboratory{

String type\_of\_reports;

Public:

Void setter(){

Cout<<”Enter type of report”<<endl;

Cin>>type\_of\_reports;

}

Void getter(){

Cout<<”Type of report: “<<type\_of\_reports<<endl;

}

};

Class Patient{

String name;

Public:

Void pname(){

Cout<<”Enter Patient name: “<<endl;

Cin>>name;

Cout<<”Patient name: “<<name<<endl;

}

};

Class Medical\_Complex{

Protected:

String address;

String administrator\_name;

Int number\_of\_doctors;

Int starting\_time;

Int closing\_timings;

};

Class Clinic : public Medical\_Complex{

Patient \*P;

Int PatientTime;

Public:

Void timeallot(){

P->pname();

Cout<<”Enter patient time: “;

Cin>>PatientTime;

If(PatientTime<closing\_timings&&PatientTime>starting\_time)

{

Cout<<”Time of 30 minutes alloted”<<endl;

}

Else{

Cout<<”Time can not be alloted”<<endl;

}

}

Void setter(){

Cout<<”Enter address: “<<endl;

Cin>>address;

Cout<<”Enter Administrator name: “<<endl;

Cin>>administrator\_name;

Cout<<”Enter no. of doctors: “<<endl;

Cin>>number\_of\_doctors;

Cout<<”Enter starting time: “<<endl;

Cin>>starting\_time;

Cout<<”Enter closing time: “<<endl;

Cin>>closing\_timings;

}

Void getter(){

Cout<<”Address :”<<address<<endl;

Cout<<”Administrator name :”<<administrator\_name<<endl;

Cout<<”no. of doctors :”<<number\_of\_doctors<<endl;

Cout<<”starting time :”<<starting\_time<<endl;

Cout<<”closing time :”<<closing\_timings<<endl;

}

};

Class Diagnostic\_Center :public Medical\_Complex{

Int num\_of\_laboratories;

Public:

Void set(){

Cout<<”Enter number of laboratories: “<<endl;

Cin>>num\_of\_laboratories;

}

Void get(){

Cout<<”Number of laboratories: “<<num\_of\_laboratories<<endl;

}

};

Int main(){

Clinic C;

Diagnostic\_Center D;

Laboratory L;

D.set();

D.get();

L.setter();

L.getter();

C.setter();

C.timeallot();

Output of program :

Enter number of laboratories:

3

Number of laboratories: 3

Enter type of report

2

Type of report: 2

Enter address:

Hghj

Enter Administrator name:

Jg

Enter no. of doctors:

6

Enter starting time:

3

Enter closing time:

8

Enter Patient name:

hhj C.getter();

Return 0;

}

Lab work 2:

#include<iostream>

#include<fstream>

Using namespace std;

Int main()

{

String name;

Int id;

String program;

Ofstream ofile (“file.txt”);

Cout<<”Enter name”;

Cin>>name;

Cout<<”Enter id”;

Cin>>id;

Cout<<”Enter program”;

Cin>>program;

Ofile<<”Enter name”<<name<<”Enter id”<<id<<”Enter program”<<program;

Ofile.close();

String text;

Ifstream ifile(“file.txt”);

While(getline (ifile, text)){

Cout<<text;

}

Ifile.close();

}

Output of program

Enter namemaro

Enter id13

Enter programoop

Enter namemaroEnterid13Enterprogramoop