1. Write a C++ program to add 2 numbers

2. Write a C++ program : Arithmetic operation (+, −, *, /) using switch case

3. Check if the number is even or odd using C++

4. Print numbers 1-10 using 'for' loop

5. Print numbers 1-10 using while loop

6. Print the below pattern in C++

a)
```
   *
  * *
 * * *
```

b)
```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

c)
```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

c) Print

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
```

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

int main()
{
rows = 5;
for (i=1; i<=rows; ++i)
{
 for (j=1; j<=rows; ++j)
{
 cout << i << " ";
}
}
```

Q1

3/7

Experiment 1-1

1. Write a program to declare class student having data members as roll number and name. Accept and display data for one object

```
#include <iostream>
using namespace std;
class student
{
int roll no;
string name;

Public:
void accept()
{
cout << "enter value of roll & name";
cin >> roll >> name;
}
void display()
{
cout << "Roll no = " << roll;
cout << "Name = " << name;
}
};
void main()
{
student S1;
```

S1. accept();
S1. display();
}

Experiment 1-2
Write a program to declare a class book having data members as book name, price and number of pages
Accept this data for two objects and display the name of book having greater price.

```
#include <iostream>
using spacename std;
class book
{
~~int~~ char name;
   float price;
int pages;
   Public:
   void accept()
   {
   cout << "enter book name, book price and
           book pages";
   cin >> name >> price >> page;
   void display()
   {
   cout << "book name" = << name;
   cout << "book pgiges=" << paiges;
```

Experiment 1-3

3. Write a program to declare a class time Accept the time in this format HH:MM:SS And convert it into total seconds and display it.

```
#include<iostream>
using namespace std;
class time
{
int H,M,S,t;
Public:
void accept()
{
cout<<"enter H,M&S";
cin>>H>>M>>S;
}
void display()
{ int t;
t=((H*3600)+(M*60)+S);
cout<<"total time in second is">>t;
}
};
int main()
{
time t1;
t1.accept();
t1.display(); }
```

9
3/7

```
}
int get_population()
{
return population;
}
};
int main()
{
city c[5];
for(i;i<5;i++)
{
c[i].accept();
}
int max_population=0;
for(i=1;i<5;i++)
{
if(c[i].population()>c[max_population].popula
{
max_population=i;
}
}

cout<<"\n city with highest population";
c[max_population].display();
return 0;
}
```

o/p

Enter city name & population: Pune 5000
Enter city name & population: Mumbai 7000
Enter city name & population: Delhi 8000
Enter city name & population: Harare 1500
Enter city name & population: Bulawayo 1000

City with highest population
name : Delhi
population: 8000.

Experiment 2-2

Write a Program to declare a class 'Accnt' having data members as Account no and balance. Accept this data for 10 account and give interest of 10% where balance is equal or greater than 500 and display them.

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

class Account
{
 int account_no;
float balance;
Public:
void accept(){
cout << "Enter account number and balan
cin >> account_no >>balance;
}
void applyInterest
if (balance >=500){
 balance += balance * 0.10;
}
}
void display(){
cout << "Account no=" << account no;
cout << "Balance -" << balance; end
} };
```

a class 'Accnt'
Account no and
for 10 accounts
where balance
n 500 and

number and balance
alance;

ount no.

```
int main(){
 Account acc[10];
  for (i=0; i<10; i++)
  {
  cout << "Enter acc[i].accept();
  }
  cout << "Applying 10% interest";
  for (i=0; i<10; i++)
  {
   acc[i]. applyInterest();
   acc[i]. display();
  }
  return 0;
  }
```

Experiment 2-3

Write a program to declare a class "Staff" having data members as name and post. Accept this data for 5 staff and display names of staff who are "HOD"

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

class Staff{
string name;
string post;

Public:
void accept(){
cout<<"Enter name and post";
cin>>name>>post;
}
int HOD(){
return strcmp(post,"HOD")==0;
}
void diplay(){
cout<<name<<endl;
}
};
int main()
{
```

Staff

```
for(i=0;
 staff
} cout<
for(i=
 if(sta
  staff
}
}
return
}
```

O/p

Enter na
Enter na
Enter na
Enter na
Enter na
Staff
Cephas
Ap

31/7

```
Staff staff[5];

for(i=0; i<5; i++){
  staff[i].accept();
} cout<<"Staff members who are HOD";
for (i=0;i<5;i++){
 if (staff[i].HOD()) {
  staff[i].display();
 }
}
 return 0;
}
```

O/p

```
Enter name and post : Paroshe    HR
Enter name and post : Cephas     HOD
Enter name and post : Natasha    Chef
Enter name and post : Sasha      Accountant
Enter name and post : Ap         HOD
Staff member who are HOD;
Cephas
Ap
```

31/7/25

Experiment 3-1

Write a C++ program a class 'book' containing data members as book_title, author_name and price. Accept and display the information for one object using a pointer to that object.

```
#include<iostream>
#include<string>
using namespace std;
class book
{
string b_title, a_name;
float price;
public:
void accept()
{
cout<<"enter book title"<<endl;
cin>>b_title;
cout<<"enter author name"<<endl;
cin>>a_name;
cout>>"enter book price"<<endl;
cin>>price;
}
void display()
{
cout>>"book title="<<b_title<<endl;
cout<<"author name="<<a_name<<endl;
cout<<"price of book="<<price<<endl;
```

```
    }
    };
    int main()
    {
      book b1;
      book* p;
      p = &b1;
      p -> accept();
      p -> display();
      return 0;
    }
```

O/p

Enter book title

The_sky

Enter author name

Natasha_Lopez

Enter book price

200

Book title = The Sky

Author name = Natasha_Lopez

Price of book = 200

```
int main()
 {
 student s1;
 s1.display();
 return 0;
}
```

Output
enter student roll number
70
enter student percentage
98.62

roll number = 70
percentage = 98.62

Q.
1118

Experiment 3-3

Write a program to demostrate the use of nested class

```
#include<iostream>
using namespace std;
class student {
int roll;
string name;
Public:
void accept(){
cout<<"Enter roll number & name"<<endl;
cin>>roll>>name;
 void display {
cout<<"Enter roll number"<<roll;
cout<<"Enter student name"<<name;}
 class Marks{
 int m1, m2 ,avg;
 public:
 void accept1(){
cout<<"Enter marks for 2 subjects"<<endl;
cin>>m1>>m2,}
void calculate avg(){
avg=(m1+m2)\2;
cout<<"The average is"<<avg;}};};
int main(){
student s1;
s1.accept();
student::Marks sm;
```

Experiment 4.1

WAP to swap two number from ~~different~~ same class using concept ~~friend function~~ object as function argument. Write swap function as member function.

```
#include <iostream>
wing namespace std;
class number {
  int n1, n2;
 public:
  void accept(){
cout << "enter first number" << endl;
  cin >> n1;
cout << "enter second number" << endl;
cin >> n2;
 }
void swap(number &n)
 {
  int temp;
  temp = n.n1;
 n.n1 = n.n2;
 n.n2 = temp;
 }
 void display() {
 cout << "num1 = " << n1 << endl;
 cout << "num2 = " << n2 << endl;
    }
    };
```

Experiment 4: 2

WAP to swap two numbers from same class using concept of friend function

```
#include <iostream>
using namespace std;
class number {
int n1=5;
int n2=6;
public:
void display(){
cout << "num1 =" << n1 << endl;
cout << "num2=" << n2 << endl;
}
friend void swap (number &nm);
};
void swap(number &nm)
  {
    int t;
    t= nm.n1;
  nm.n1 =nm.n2;
  nm.n2=t;
}
int main(){
number p;
swap (p);
p.display();
return 0
}
```

O/p

num 1=6
num2=5

Experiment 4-3:

WAP to swap two numbers from different classes using friend function.

```
#include <iostream>
using namespace std;
class classB;
class classA {
  int num1;
 public:
void accept(){
cout<<"enter first number"<<endl;
cin >>num1;
}
void display(){
cout<<"num1="<<num1<<endl;
}
friend void swap(classA &a, classB &b);
};
class classB{
  int num2;
public:
  void accept(){
cout<<"enter second number"<<endl;
cin >>num2;
}
void display(){
cout<<"num2="<<
```

Experiment 4-4

WAP to create two classes Result1 and Result2 which stores the marks of the students. Read the value of a mars for both the class and compute the average of two results

```
#include<iostream>
using namespace std;
class result2;
class result1{
int mark1
 public:
void accept(){
cout<<"enter first mark:";
cin>>mark1;
}
void display(){
cout<<"mark
friend float average(result1 &a, result2
};
class result2{
 int mark2;
 publi:
void accept(){
 cout<<"enter second mark:";
 cin>>mark2;
 }
friend float average(result1 &a, result2 &b)
```

```
};
float average(result1 &a, result2 &b){
  float avg;
  avg=(a.mark1+b.mark2)/2;
  return avg;
}
int main(){
result1 obj1;
result2 obj2;
obj1.accept();
obj2.accept();
average(obj1,obj2);
cout<<"average="<<average(obj1,obj2);
  return 0;
}
```

Output

```
enter first mark : 50
enter second mark: 30
 average = 40
```

5. WAP to find the greatest number among two numbers from two different classes using friend function.

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

class classA{
int num1;
public:
void accept()
{
cout<<"enter first number"<<endl;
cin>>num1;
}
friend void greater(classA &a, classB &b);
};
class classB{
int num2;
public:
void accept()
{
cout<<"Enter second number"<<endl;
cin>>num2;
}
friend void greater(classA &a, classB &b);
};
```

```
void greater (classA &a, classB &b) {
if (a.num1 > b.num2) {
cout << "first number is greater" << a.num1;
}
else if (a.num1 < b.num2) {
cout << "second number is greater" << b.num2;
}
else {
cout << "both number are equal" << endl;
} }
int main()
{
classA obj1;
classB obj2;
obj1.accept();
obj2.accept();
greate (obj1, obj2);

return 0;
}
```

O/p
enter first numbe = 50
enter second numbe = 70

second numbe i greate.

11/8.

6. Create two classes, class A and class B, each with a private integer. Write a friend function sum() that can access private data from both classes and return the sum.

```
#include <iostream>
using namespace std;
class B;
class A{
int x;
public:
void accept(){
cout<<"Enter first number";
cin>>x;
}
friend void sum (classA &a, classB &b);
};
class classB{
int y;
public:
void accept(){
cout<<"Enter second number";
cin>>y;
}
friend void sum (classA a, classB b);
};
void sum (classA a, classB b){
  int t;
  t = a.x + b.y
```

7. Write a program with a class number that contains a private integer. Use a friend function swap Number (Number &, Number &) to swap the private value of two Numbers objects.

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

class NumberB;
class NumberA{
 int n1=5;
  public
 void display();
 cout<<"num1"<<n1<<endl;
 }
 friend void swap(NumberA &a, NumberB &b
 };
 class B{
 void display(){int n2=6;
 cout<<num public:
  void display(){
 cout<<"num2"<<n2<<endl;
 }
 friend void swap(NumberA &a, NumberB &b);
 };
 void swap(NumberA &a, NumberB &b){
   int temp;
   temp= a.n1;
   a.n1 = b.n2;
```

8 Define two classes Box and Cube, each having a private volume. Write a friend function find greater (Box, Cube) that determines which object has a larger volume.

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

class Box {
int vol1;
public:
void accept();
{ cout<<" Enter volume of the box"<<endl;
cin>>vol1;
}
void display(){
friend void greater(Box &a, Cuble &b);
};
class Cube {
int vol2;
public:
void accept(){
cout<<"Enter volume of the Cube"<<endl;
cin>>vol2;
}
friend void greater(Box &a, Cube &b);
};
```

```
}
if (a.vol1>b.vol2){
cout<<"volume of box is greater"<<a.vol1;
}
else if (a.vol1<b.vol2){
cout<<"volume of cube is greater"<<b.vol2<<endl;
}
else{
cout<<"both are equal"<<endl;
}
}
int main(){
Box objA;
Cube objB;
objA.accept();
objB.accept();
greater(objA,objB);
return 0;
}
```

O/P

enter volume of Box = 56

enter volume of Cube = 80

volume of Cube is greater

```
int main() {
complex c1, c2, c3;
 c1.accept();
 c2.accept();
c3 = add(c1, c2);
c3.display();

return 0;
}
```

O/p

Enter real and imaginary part: 3 , 3

Enter real and imaginary part: 4 , 5

Sum = 7.8i

```
S1.accept;
average (S1);
return 0;
}
```

O/P

Enter Student Name : Parashe
Enter three marks : 46,50,50

Average mark = 48.66

11. Create three classes: Alpha, Beta and Gamma each with a private data member. Write a single friend function that can access all three and print their sum.

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

class Beta;
class Gamma;

class Alpha{
 int a;

Public:
void accept(){
cout<<"Enter value";
cin>>a;
}
};
class Beta{
int b;
public: void accept(){
cout<<"Enter second value";
cin>>b;
}
};
class Gamma{
int c;
```

```
public
void accept(){
cout << "Enter third value";
cin >> c;
}
friend void sum(Alpha &1, Beta &2, Gamma &3);
};
void sum(Alpha &1, Beta &2, Gamma &3)
{ int sum;
 sum = 1.a + 2.b + 3.c;
 cout << "sum of Alpha, Beta and Gamma = " << sum;
}
int main(){
 Alpha  obj A;
 Beta   obj B;
 Gamma  obj C;
 obj A.accept();
 obj B.accept();
 obj C.accept();
 sum = (obj A, obj B, obj C);
 sum.display();
 return 0;
 }
```

O/p

Alpha = 5

Beta = 4

Gamma = 3

sum of Alpha, Beta and Gamma = 12.

12. Create a class Point with Private members x & y. Write a friend function that calculates and returns the distance between two point objects.

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

class Point {
int x, y;

Public
void accept(){
cout << " Enter two distance " of two points ";
cin >> x >> y;
}
friend calculate distance (Point &p1, Point &p2);
};
calculate distance (Point &p1, Point & p2){
 int distance
 distance x = p2.x - p1.x;
 distance y = p2.y - p1.y;
 return sqrt (dx*dx + dy*dy);
}
int main () {
Point obj P1 , obj P2;
Point obj P1. accept
    obj P2. accept
distance=(obj P1, obj P2);
 return 0;
}
```

Output
Ente distance of two points = 3, 4 7, 1
Distance = 5

13. Create tw
Account
a friend
prints ba

```
#incl
using

class

class
doul

Pub
Ban
bala
}
frien
};
class
Publ
void
{ A
}
};
void
{ co
}
int
```

```
BankAccount myAccount (1243.57);
Audit auditor;
auditor.audiAccount (MyAccount);
return 0;
}
```

O/p

Account Balance : $1234.57

Q.
21/8

Experiment 5:
Title: Write a C++ program to implement types of constructors

1. Write a program to find the sum of numbers between 1 to n using a constructor where the value of n will be passed to the constructor.

a. Default constructor

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

class Sum {
int n, i;
int result=0;

public:
Sum() {
n=5;
for (i=1; i<=n; i++){
  result += i;
}
}
void display() {
cout << "Sum =" << result << endl;
}
```

```
};
int main() {
Sum S;
S.display();
return 0;
}
```

O/p
Sum = 15

b. Parameterized Constructor

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

class Sum {
int n;
int result=0;

public:
sum(int a) {
 n=a;
for (int i=1; i<=n; i++) {
   result += i;
 }
}
void display() {
cout<<" sum="<< result<<endl;
```

```
    }
};
int main() {
    Sum s(5);
    s.display();
    return 0;
}
```

O/p

Sum = 15

Copy constructor

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

class Sum {
    int n;
    int result=0;
    public:

    Sum(int a) {
        n=a;
        for(int i=1; i<=n; i++){
            result += i;
        }
    }
}
```

```
Sum (Sum &sm)
{
 n=sm.n;
 result=sm.result;
 }
void display() {

cout << "Sum=" <<result<<endl;
 }
 };
int main(){
Sum s1(5)
Sum s2(s1);
S1. display();
S2. display();
 return 0;
}
```

O/p

Sum = 15

Sum = 15

2. Write a
having d
Write a
members
student

a. Default

#inclu
using

cl
st
in

pu
S

O/p

Name : Panashe
Percentage: 90

Parameterized Constructor.

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

class Student {
string name;
int per;

public:
Student (string a, int b){
   name = a;
   per = b;
}
void display (){
 cout << "Name:" << name << endl;
 cout << "Percentage:" << per << endl;
 }
};
int main();
Student s("Panashe", 80);
s.display();
return 0;
}
```

O/p

Name
Percentag

Copy

#inclu
using

das
str
int

publi
Stud

}
Stud
nam
per
}
void
cout
cout
}
};

```
int main() {
 Student s1 ("Panashe", 80);
 Student s2 (31);
  s1.display();
  s2.display();
  return 0;
}
```

O/p

Name : Panashe
Percentage : 80

Name : Panashe
Percentage : 80

3. Defin
roll
const
Engine
for
the

#
wi

vo

3. Define a class "College" members variables as roll-no, name, course. WAP wing constructor with default value as "Computer Engineering" for course. Accept this data for two objects of class and display the data

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

class College {
int roll;
string name;
string course;

public:
College (int a, string b) {
  roll = a
  name = b
  course = "Computer Engineering";
}
void display () {

cout << "roll number :" << roll << endl;
cout << "Name:" << name << endl;
cout << "Curse:" << course << endl;
} ~college () { }
};
```

4. Write a program to demostrate constructor overloading

```
#include <iostream>
using namespace std;
class rectangle{
{
int l, w;
public:
rectangle()
{
  l=1;
 w=2;
}
rectangle(int a)
{
  l=a;
 w=a;
}
rectangle(rectangle &r)
{
  l=r.l;
 w=r.w;
}
void calculate()
{
 int a;
 a = l*w;
 cout<<"Area="<<a<<endl;
```

```
};
int main(){
rectangle r1;
rectangle r2(5);
rectangle r3(r1);
r1.calculate();
r2.calculate();
r3.calculate();
return 0;
}
```

O/p

Area = 2
Area = 25
Area = 2

Experiment

Single Inh
Create a
name and a
Person that
Write fun

#inclu
wing

class
{
pu
str
int
};
class
{
pr

va
{
ca
a
a

```
cout<<"enter roll number:";
cin>>roll;
}
void display()
{
cout<<"Name:"<<name<<endl;
cout<<"Age:"<<age<<endl;
cout<<"Roll number:"<<roll<<endl;
}
};
int main()
{
Student s;
s.accept();
s.display();
return 0;
}
```

O/p

```
enter name : Parashe
enter age : 19
enter roll number: 70

Name : Parashe
Age : 19
Roll number: 70
```

Experiment 6 : 2

Multiple Inheritance

Create two base classes Academic and Sports

- Academic class contains marks of a student
- Sports class contains sports score
  Create a derived class Result that inherits from both Academic and Sports. Write a function to calculate the total score and display details.

```
#include<iostream>
using namespace std;
{
public:
int marks
};
class S
```

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

class Academic
{
public:
int marks;
```

```
};
class Sports
{
protected:
int score;
};
class Result : public Academic, protected
                                    Sports
{
private:
int t;
public:
void accept()
{
cout << "Enter academic marks";
cin >> marks
cout << "Enter sports score";
cin >> score;
}
void calculate()
{
t = marks + score;
cout << "Academic marks" << marks <<
cout << "Sports score :" << score <<
cout << "Total score :" << t <<
}
};
int main()
{
Results r;
```

```
r. accep
r. calcul
return 0;
}

O/p

enter acad
enter spo

Academic
Sports
Total sc
```

Experiment 6 : 3

Multilevel Inheritance

1. Create a class Vehicle with attributes like brand and model.
2. Derived a class Car from Vehicle which adds an attributes type (eg sedan, SUV).
3. Further derive a class ElectricCar from Car which adds battery capacity. Write functi to display all the details.

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

class Vehicle
{
protected:
string brand;
string model;
};
class Car : protected Vehicle
{
protected:
 string type;
};
class ElectricCar : protected Car
{
```

```
private:
string battery_cap;

public:
void accept()
{
cout<<"Enter vehicle brand:";
cin >>brand;
cout<<"Enter vehicle model";
cin >>model;
cout<<"Enter vehicle type";
cin >>type;
cout<<"Enter battery capacity";
cin >>battery_cap;
}
void display()
{
cout<<"Vehicle brand:"<<brand<<endl;
cout<<"Vehicle model:"<<model<<endl;
cout<<"Vehicle type:"<<type<<endl;
cout<<"Battery capacity:"<<battery_cap;
}
};
int main()
{
ElectricCar c;
c.accept();
c.display();
}
```

O/p

enter vehicle brand : Mercedes-Benz
enter vehicle model : ~~EQS 450+~~ GLE
enter vehicle type : SUV
enter battery capacity : 107.8 kWh

Vehicle brand : Mercedes-Benz
Vehicle model : ~~EQS 450+~~ GLE
Vehicle type : SUV
Battery capacity : 107.8 kWh

Experiment

Hierarchica

1. Create a
attributes
2. Derive t
from G
7. Manager
Developer
Write

Experiment 6 : 4

Hierarchical Inheritance

1. Create a base class Employee with attributes empID and name;
2. Derive two classes Manager and Developer from Employee
3. Manager has an attribute department and Developer has an attribute programmingLanguage Write functions to display details for both.

```
#include <iostream>
using namespace std;
class Employee;
{
 protected:
 int empID;
 string name;
 };
 class Manage : protected Employee
 {
  private:
  string name;
  public:
  void accept()
  {
  cout<< "enter employee id: ";
```

```
    cin >>empID;
    cout<<"enter employee name:";
    cin >> name;
    cout <<" enter department name";
    cin >> dname;
    }
    void display()
    {
    cout<<"Employee ID "<<empID<<endl;
    cout<<"Employee name"<<name<<endl;
    cout<<"Department:"<<dname<<endl;
    }
};
class Developer : protected Employee
{
 private:
 string prolang;
 public:
 void accept()
 {
 cout<<"enter programming language:";
 cin >>prolang;
 }
 void display()
 {
 cout<<"Programming language:"<<
    prolang<<endl;
};
```

O/p

enter
enter
enter
enter

Experiment 6: 5

Combine multilevel and multiple inheritance
Create a base class Person with attributes name and age.
Derive class Student from Person.
Create two classes Sports and Academics
Derive class Result from Student and Sports
Write functions to calculate and display total marks and sports score along with student details.

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

class Person
{ protected:
string name;
int age;
};
class Student :protected Person
{

protected:
int roll;
public:
void acceptStudent(){
cout<<"Enter name and age"<<endl;
cin >>name>>age;
```

```
cout <<"enter roll numbe";
cin >>roll;
}
void displayStudent()
{
  cout << "Name: " << name << endl;
  cout << "Age: " << age << endl;
  cout << "Roll no" << roll << endl;
}
};
class Sports
{
protected:
int score;
public:
void acceptSports(){
cout <<"Enter sports score";
cin >> score;
}
};
class Academics {
protected;
int marks
public:
void acceptAcademics(){
  cout << "enter marks";
  cin >>marks;
}
};
```

```
class Result: public Student, public Academic,
              public Sports.
{
 int total;
 public:
void acceptResult(){
   acceptStudent();
   acceptAcademic();
   acceptSports();
}
void displayResult(){
  displayStudent();
 total = marks + score;
cout<<"Total = "<<total<<endl;
}
};
int main(){
Result obj;
obj.acceptResult();
obj.displayResult();
return 0;
}
```

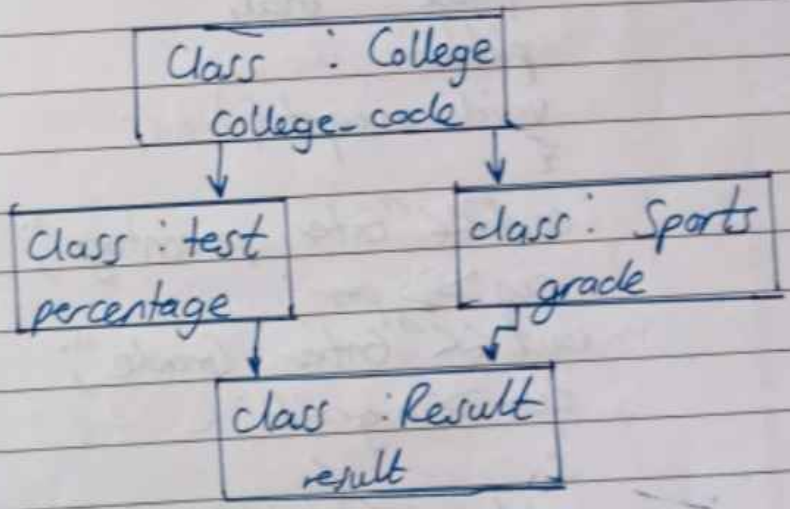
O/p

```
Enter name and age: Panase, 19
Enter roll number: 70
Enter marks: 60
Enter score: 45
```

Name : Paraske
Age : 19
Rollno : 70
Total : 105

Experiment 6:6

Virtual base class




```
#include <iostream>
using namespace std;
class College
{
protected:
int clg_code;
};
class Test : protected virtual College
{
```

```
protected:
float per;
};
class Sports : virtual protected College
{
protected:
float grade;
};
class Result : protected Test, protected Sports
{
float total;
public:
void accept().
{
cout<<"Enter percentage";
cin>>per;
cout<<"Enter Grade";
cin>>grade;
}
void display(){
total=per+grade;
cout<<"Total:"<<total;
}
};
int main()
{ Result r;
r.accept();
r.display();
}
```

O/p

Enter
Enter
Total :

O/p

Enter percentage 45.6
Enter grade 27.5
Total : 73.1

25/9/25

Experiment 7 - 1

Title :- WAP to demostrate the Compile Time Polymorphism (function Overloading and Operator Overloading - Unary

a) Write a program using function overloading to calculate the area of a laboratory (which is rectangular in shape) and area of classroom (which is square in shape)

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

class Area
{
public:
void calculate (int b, int l)
{
cout<< "Rectangle=" <<b*l<<endl;
}
void calculate (int s)
{
cout<<"Square =" << s*s<<endl;
}
};
int main () {
Area obj;
```

obj.cal
return
}

O/p
Recte
Squa

```
obj.calculate(10);
return 0;
}
```

O/p

Rectangle = 300

Square = 100

Experiment 7-2

b) Write a program using function overloading to calculate the sum of 5 float values and sum of 10 integer values.

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

class SumCalculate {
    float fArr[5];
    int iArr[10];
    public:
    float sum(float arr[], int n){
        float total=0;
    for(int i=0; i<n; i++){
     total += arr[i];
    }
    return total;
    }
    int sum(int arr[], int n){
     int total=0;
    for(int i=0; i<n; i++){
     total += arr[i];
    }
    return total;
    }
    void accept(){
    cout<<"Enter 5 float values";
```

```
(fArr, 5) << endl;
for (int i=0; i<5; i++){
  cin >> fArr[i];
 }}
void display (){
cout<<"Sum of 5 float values = "<<sum
(fArr, 5)<< endl; }
 void getdata(){
cout <<"\n Enter 10 integer values: ";
 cin >> iArr [i];
 }}
 void showdata (){
cout<<"Sum of 10 integer value = "<<
sum (iArr, 10)<< endl;
} };
int main (){
SumCalculate calc;
 calc . accept ();
 calc . display ();
 calc . getdata ();
 calc. showdata ();
 return 0;
 }
```

O/p

Enter 5 float values : 1.5, 2.5, 3.5, 4.5, 5.5. Sum of 5 float values = 17.5

Enter 10 inter value : 1, 2, 3, 4, 5, 6, 7, 8 9, 10 Sum of 10 inter values = 55

Experiment 7-3

c) Write a program to implement Unary - operator when used with the object so that the numeric data member of the class is negated

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

class Number
{
int a;
public:
void accept() {
cout << "Enter value of a";
cin >> a;
}
void display()
{
cout << "value of a is" << a;
}
void operator -()
  {
    a= -a;
  }
};
int main(){
Number n1;
n1. accept();
```

```
-n1;
n1. display();
}
```

O/p

enter value of a : 20
value of a is -20

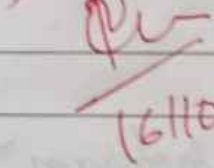
Experiment 7-4

d) Write a programe to implement the Unary ++ operator (for pre increment and post increment) when used with the object so that the numeric data member of the class is incremented.

```
#include <iostream>
using namespace std;
class Number
{
int a=10;
public:
void display()
{
cout << "value of a=" << a;
}
void operator++()
```

```
    {
     a=++a;
    }
};
int main()
{
Number n;
 ++n;
 n.display();
}
```

O/P
value of a=11



Experiment :- 8-1

Title : WAP to demonstrate the Compile Time and Run time Polymorphism (Operator overloading Binary).

a) Write a program to overload the '+' operator so that two strings can be concatenated eg "xyz" + "pqr" then output will "xyzpqr".

```
#include <iostream>
using namespace std;
class Addition {
 string str;
 public:
 void accept() {
  cout<<"enter value of str";
  cin >> str;
 }
Addition operator + (Addition x){
  Addition temp;
 temp.str = str + x.str;
  return temp;
}
  void display() {
     cout << "result:" <<str;
 }
};
```

```
int main(){
Addition s1, s2, s3
s1.accept();
s2.accept();

s3 = s1 + s2;
s3.display();
return 0;
}
```

enter value of str xyz
enter value of str pqr

result = xyzpqr

Experiment

Write a pr
Login having
password. U
Derive Emp
classes from
details and
the employ

```
#inclu
using

class
prote
stri
strin

publ
vir
cou
cin
cou
cin
}
vir
cou
cout
}
```

Experiment 8:2

Write a program to create a base class ILogin having data members name and password. Declare accept() function virtual. Derive EmailLogin and MembershipLogin classes from ILogin. Display EmailLogin details and membership login details of the employee.

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

class ILogin {
protected:
string name;
string password;

public:
virtual void accept () {
cout << "Enter name: ";
cin >> name;
cout << " Enter password: ";
cin >> password;
}
virtual void display () {
cout << "Name: " << name << endl;
cout << " Password: " << password << endl;
} };
```

```
class EmailLogin : public ILogin {
public:
void accept(){
cout << " Email Login" << endl;
cout << " Enter name: ";
cin >> name;
cout << " Enter Password: ";
cin >> password;
}
void display() {
cout << " Email Login details: ";
cout << " Name: " << name << endl;
cout << " Password: " << password << endl;
}
};
class MembershipLogin : public ILogin {
public:
void accept() {
cout << " Membership Login " << endl;
cout << " Enter Name: ";
cin >> name;
cout << " Enter Password";
cin >> password;
}
void display() {
cout << " Membership Login details " << endl;
cout << " Name: " << name << endl;
cout << " Password: " << password << endl;
} };
```

```
int mai
ILogin *
EmailLog
Membership
login =
login →
login →
login =
login →
login →
return
}
```

Output

```
Ema
Enter na
Enter pass

Email
Name :
Password

Me
Ente
Enter pa

N
Name
Passwor
```

```
int main() {
 ILogin * login;
 EmailLogin eLogin;
MembershipLogin mLogin;
 login = &eLogin;
 login -> accept();
 login -> display();
 login = &mLogin;
 login -> accept();
 login -> display();
 return 0;
}
```

Output

Email Login
Enter name : pana@gmail
Enter password : pana2006

Email Login Details
Name : pana@gmail
Password : pana2006

Membership Login
Enter name : abc@o
Enter password : abc123

Membership Login
Name : ~~pana~~ abc@
Password : abc123

10/10

Experiment 9.1

WAP to perform various operations on

a) Write a program to copy the contents of one file into another. Open "first.txt" in read (ios::in) mode and "second.txt" file in write (ios::out) mode. Copy the contents of "first.txt" into "second.txt". Assume "first.txt" is already created.

```
#include <iostream>
#include <fiostream>
using namespace std;
int main()
{
ifstream fin("source.txt", ios::in);
ofstream fout("destination.txt", ios::out);

if(!fin){

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

int main(){
fstream new_file;
new_file.open("new file.txt", ios::
```

```
if(!new_f
cout<<"file
}
else{
cout<<"
new_file
}
ifstream
if(!fin
cout<<
}
ofstream
if(!fou
cout<<
}
char
while
fou
}
cout<
fin
fout
cet
}

O/p
new
file
```

```
if(!new_file){
cout<<"file creation failed";
}
else{
cout<<"new file created"<<endl;
new_file.close();
}
ifstream fin("source.txt",ios::in);
if(!fin){
cout<<"Cannot open file"<<endl;
}
ofstream fout("destination.txt,"ios::app);
if(!fout){
cout<<"Cannot open file",<<endl;
}
char ch;
while(fin.get(ch)){
fout.put(ch);
}
cout<<"file copied successfully";
fin.close();
fout.close();
return 0;
}
```

O/p
new file created
file copied succesfully

Experiment 9:2

b) Write a C++ program to count digits and space spaces using file handling.

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

int main() {
ifstream fin("source.txt", ios::in);
if(!fin) {
cout << "Cannot open file" << endl;
}
char ch;
int digits=0, spaces=0;
while(fin.get(ch)) {
 if(isdigit(ch))
  digits++;
 if(ch==' ')
  spaces++;
 }
 cout << "Total Digits:" << digits << endl;
 cout << "Total Spaces:" << spaces;
  fin.close();
  return 0;
}
```

O/p

Total Digits: 0 Total Spaces: 4

Experiment 9

b) Write a C++
file handling

```
#includ
#include
using na

int ma
ifstream
if(!fi
cout <<
}
string
int
while(

cou
}
cout <<
fin.clo
return
}
```

O/p

Total nu

Experiment 9:3

a) Write a C++ program to count words using file handling

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

int main(){
ifstream fin("source.txt", ios::in);
if (!fin){
cout<<"Cannot open file"<<endl;
}
string word;
int count=0;
while (fin>>word){

  count ++
}
cout<<"Total number of words:"<<count;
fin.close();
return 0;
}
```

O/p
Total number of words: 5

Experiment 9.4

d) Count Occurence of a given word using file handling

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

int main() {
ifstream fin("source.text", ios::in);
if (!fin){
cout<< "cannot open file" << endl;
}
string word, target;
int count = 0;
cout<<" Enter the word to search:";
cin >>target;
while (fin >> word ){
if (word == target)
count ++;
}
cout << "The word " << target << " occured"
<< count << "times " << endl;
fin.close()
return 0; }
```

O/p

enter the word to search : Pune

The word Pune occured 2 times.

9/
16/10

Experiment

Write a p
template

a) Write a
Array elem
Pass Integ
10 eleme

#in
using
templ
T su
T
for (
sum
retu
}
int
int
float
double

cout<

cout <

cout<<

Experiment 10

Write a program to implement a function template and Class template

a) Write a C++ program to find Sum of Array elements using function template (e.g Pass Integer, Float and Double array of 10 elements.

```
#include <iostream>
using namespace std;
template <class T>
T sumArray (T arr[], int n){
  T sum = 0;
for (int i=0; i<n; i++)
sum += arr[i];
return sum;
}
int main(){
int intArr[5] = {1, 2, 3, 4, 5};
float floatArr[5] = {1.2, 2.3, 3.4, 4.5, 5.6};
double doubleArr[5] = {1.22, 2.33, 3.44, 4.55, 5.66};

cout<<"sum of interger numbers "<<sumArray(intArr,
                5<<endl;
cout<<"sum of float numbers"<<sumArray
        (floatArr, 5)<<endl;
cout<<"sum of double numbers"<<sumArray(doubleArr,
                5<<endl;
```

O/p
sum of integer numbers: 15
sum of float number : 17
sum of double numbers : 17.2

6.) Write a C++ Program of Square Function using template specialization
- calculate the square of integer no and a string (Square of string is nothing but concatenation of a string with itself).
- Write a specialized function for the square of a string.

```
#include <iostream>
using namespace std;
template <class T>
T square (T x) {
  T result;
result = x * x;
return result;
}
template < > string square <string>
(string ss) {
  return (ss + ss);
}
int main()
{
```

```
int i=2
string ww=
i = square
cout << "sq
cout << "sq
return 0;
}
```

Output
square if
square of

```
int i=2;
string ww=("Aaa");
i=square<int>(i);
cout<<" square of integer: "<<endl;
cout<<" square of string: "<<square(ww);
return 0;
}
```

Output

square of integer : 2

square of string : AaaAaa

c) Write a C++ Program to build a Simple calculator using a class template

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

template<class T>
class Calculator {
 public:
 T num1, num2;

 Calculator(T n1, T n2) {
 num1 = n1;
 num2 = n2;
 }
 void menu() {
 int choice;
cout<<"1. Addition \n";
cout<<"2. Subtraction \n";
cout<<"3. Multiplication \n";
cout<<"4. Division \n";
cout<<"5. Modulus \n";
cout<<"6. Power \n";
cout<<"7. Square Root \n";
cout<<"8. Maximum \n";
cout<<"9 Minimum\n";
cout<<"10. Average \n";
cout<<"Enter your choice:";
```

```
cin >> choice;

switch (choice) {
case 1:
cout << "Result:" << add() << endl; break;
case 2:
cout << "Result" << subbract() << endl;
  break;
case 3:
cout << "Result: " << multiply() << endl;
break;
case 4:
cout << "Result: " << divide() << endl;
break;
case 5:
cout << "Result: " << modulus() << endl;
break;
case 6:
cout << "Result: " << power() << endl;
break;
case 7:
cout << "Result: " << squareRoot() << endl;
break
case 8:
cout << "Result: " << maximum() << endl;
break;
case 9:
cout << "Result: " << minimum() << endl;
break;
```

```
case 10:
cout << "Result" << average() << endl;
  break;
  default: cout << "Invalid choice!" << endl;
  break;
  }
  }
T add() { return num1 + num2; }
T subtract() { return num1 - num2; }
T multiply() { return num1 * num2; }
T divide() { return (num2 != 0) ? num1 / num2
  : 0; }
T modulus() {
  if constexpr (is_integral<T>::value) {
    return num1 % num2;
  } else {
  cout << "Modulus operation is only valid for
    integers!" << endl;
  return 0; }
  }
T power() { return pow(num1, num2); }
T squareRoot() { return sqrt(num1); }
T maximum() { return (num1 > num2) ?
    num1 : num2; }
T minimum() { return (num1 < num2) ?
    num1 : num2; }
T average() { return (num1 + num2) / 2; }
};
int main()
```

```
o/p
Enter

1. Additio
2. Subtrac
3. Multipl
4. Divisio
5. Modulu
6. Power
7. Square
8. Maximu
9 Minimu
10. Averag
Enter yo

Result:
```

```
    int
~~double~~ a, b;
cout << "Enter two numbers:";
cin >> a >> b;
Calculator <int> calc(a, b);
calc.menu();
return 0;
}
```

O/p

Enter two number: 15 4

1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Modulus
6. Power
7. Square Root
8. Maximum
9. Minimum
10. Average

Enter your choice: 4

Result: 3.75

d) Write a C++ Program to implement push and pop methods from stack using class template

```
#include <iostream>
template<class T>
class Stack {
  private:
T arr[5];
int top;
public
stack() {
top = -1;
}
void push (T value) {
 if (top == 4)
cout << "Stack Overflow " << value << endl;
else {
 top ++;
arr[top] = value;
cout << value << "pushed to stack " << endl;
 }
}
void pop() {
 if (top == -1)
 cout << "Stack Underflow " << endl;
 else {
cout << arr[top] << "popped from stack"
   << endl;
   top --;
```

```
    }
    }
    void display(){
    if (top==-1)
cout<<"Stack is empty"<<endl;
    else{
    cout<<"Stack elements:";
    for(int i=top;i>=0;i--)
      cout<<arr[i]<<" ";
      cout<<endl;
      }
      }
    };
     int main(){
    Stack<int>s;
     s.push(10);
    s.push(20);
    s.push(30);
    s.display();
    s.pop();
    s.display();
    return 0;
    }
```

Output

```
10 pushed to stack
20 pushed to stack
30 pushed to stack
Stack elements: 30 20 10
30 popped from stack
Stack elements: 20 10
```

10/11

80.
Experiment : 11

Write a C++ program to implement generic vectors
Include following members functions
To create the vector.
a) To modify the value of a given element
b) To multiply by a scalar value.
c) To display the vector in the form (10,20
30 ...).

Qa) To modify the value of a given element

```
#include<iostream>
using namespace std;
template<typename>
class vector {
T a[100];
int size;
public:
vector(int s) : size(s) {
void set(int i, T val) {
if (i>=0, && i<size)
a[i]=val;
else
cout << "invalid";
}
T get(int i) {
```

```
if (i>=0 && i<size)
return a[i];
cout << "invalid";
return T();
}
void display(){
for(int i=0; i<size; i++)
cout<<a[i]<<" ";
cout<<endl;
}
};
int main(){
vector<int> v(5);
for(int i=0; i<5; i++){
v.set(i, i*10);
v.display
v.set(2, 99);
cout<<"after modification";
v.display();
return 0;
}
```

output

0 10 20 30 40

after modification : 0 10 99 30 40

Q6 To multiply by scalar value

```
#include<iostream>
#include<vector>
```

```
using namespace std;
int main() {
vector<int> vec = {1, 2, 3, 4, 5};
int scalar = 3;
for (int &val vec)
{ val = val * scalar;
}
for (int val : vec) {
cout<<val <<" ";
}
cout<<endl;
return 0;
}
```

Output

3 6 9 12 15

Q To display the vector in form (10, 20, 30...

```
#include <iostream>
#include <vector>
using namespace std;
int main() {
vector<int> vec = {10, 20, 30, 40, 50};
cout<<"( ";
for (int i=0; i<vec.size(); i++){
cout <<vec[i]
if (i != vec.size()-1)
```

```
    cout << i " ";
}
cout << ")" << endl;
return 0;
```

Output

10 20 30 40 50

10/11

Experiment 12

Q. Write a C++ program using STL

a) Implement stack

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

int main(){
 stack <int> s;
 s.push (10);
 s.push (20);
 s.push (30);
 cout<<" Top element : "<< s.top()<<endl;
 s.pop();
cout<<" Top element after pop "<<s.top()<<
endl;
cout<<"stack size:"<<s.size()<<endl;
 if (s.empty()){
cout <<"stack is empty "<<endl;
 } else{
  cout<<"stack is not empty";
 }
  return 0;
 }
```

output
Top element : 30
Top element after pop : 20
stack size : 2
stack is not empty

Q To implement queue

```
#include <iostream>
#include <queue>
wing namespace std;
int main(){
queue <int> q;
  q.push(10);
  q.push(20);
  q.push(30);
cout<<"front element:"<<q.front()<<endl;
cout<<"Back element:"<<q.back()<<endl;
  q.pop();
cout<<"afte pop operation:"<<endl;
cout<<"queue size:"<<q.size()<<endl;
  if(q.empty){
cout<<"queue is empty"<<endl;
} else
{ cout<<"queue is not empty"<<endl;
}
```

Output
Front element: 10
Back element: 30
after pop operation
front element: 20
queue size: 2
queue is not empty

Q Implement sorting & searching with user defined recorded such as personal record, item record

```
#include <iostream>
#include <vector>
using namespace std;
struct person {
string name;
int age
person(string n, int a) : name(n)
 age(a) {
};
int compareAge(const Person &p1 const
person &p2){
return (p1.age < p2.age) ? 0;
}
int main(){
vector<person> people {
person("Alice", 30);
```

```
person ("Bob", 25);
Person ("Charlie", 35);
person ("Panashe", 28);
sort (people.begin(), people.end(), [](
cout<<p.name<<" - "<<p.age<<"\n";
}
int searchAge=28;
int foundindex = -1;
for (int i=0; i<(int)people.size(); i++){
if (people[i].age == search.Age ? 1:0){
foundindex=i;
break;
}
} if (foundindex = -1)
{ cout<<"person with age "<<searchAge
<<"found "<<people.foundindex.name<<
"\n"
} else {
cout<<"person with age "<<searchAge
<<"not found";
return 0;
```

O/p

sorted records by Aage

Bob - 30

Alice - 25

Charlie - 35

Panashe - 28

~~Peter~~ Person with age 28 found : Panashe

10/11