Experiment - 1

Q1) wap to declare a class student having data member as name, Rollno accept & display data for one student.

→

```
#include <iostream>
using namespace std;
class student {
int roll no;
Public:
void accept () {
cout << " enter student's name : n);
getline (cin, name);
cout << " enter roll number :";
cin >> roll_no;
}
void display c) {
cout << " In student Details : In";
cout << " Name : " << name << endl;
cout << Rollnumber : " << rollno << end l;
}
int main() {
student s;
s accept();
s display ();
return 0;
```

Output :-

Enter student's name :- Atharv
Enter Roll no :- 70

Student details :-
Name :- Atharv
Rollno :- 70.

Q2) WAP to declare a class book having data member asidehave, price Accept data for 2 books & display

→ #include <iostream>
using namespace std;

class Book {
private:
int id;
string name;
float price;

public:
void accept() {
cout << "enter book ID : ";
cin >> id;
cin.ignore();
cout << "enter book name : ";
getline(cin, name);
cout << "enter book price :";
cin >> price;
}

```
void display {

 cout<< "\n Book ID : " << id << endl;
cout << "Book name : " << name << endl;
cout << "Book price : " << price << endl;
}
 Float get price ( ) {
  return price;
 }
};

int main ( ) {
   Book b1, b2;

cout << "enter detail For Book 1: \n")
 b1. accept ( );

cout << "\n enter details For Book 2 : \n";
b2. accept ( );

cout << "\n Book with greater price : \n";
If (b1 . get price ( ) > b2 . get price ( )) {
if (b1 . get price ( ) > b2 . get price ( )) {
   b1 . display ( );
} else {
cout << "Both Books have the same price: \n";
   b 1. display ( );
  b 2 . display ( );
   }
   return 0;
```

Output:-
Enter details For Book1:
Enter book ID :77
Enter book Name:
Enter book price:

Enter details for Book2:
Enter book ID : 23
Enter book name:
Enter book price:
Book with greater price
Book ID : 23 , Book Name : jj, Book price 77

(Q5) WAP to declare a class Time having data members as H, M, S Accept data

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

class Time {
private:
  int H, M, S;

public:
  void gettime() {
  cout << "enter Hours:";
  cin >> H;
cout << "enter mintues:";
cin >> M;
cout << "Enter seconds:";
cin >> S;
}
```

```
int toseconds (){
return (H * 3600) + (M * 60) + S;
}

void display totalseconds () {
cout << " Total time in seconds: " << toseconds
        () << endl;
}
};
int main () {
  Time t;
  t.gettime ();
  t.display totalseconds ();
  return 0;
  }
```

Output :-

```
Enter Hours : 07
enter Minutes : 55
Enter seconds : 55
Total time in seconds : 28555
```

15/7

Experiment - 2

1) WAp to declare a class 'city' having data members as name and population Accept this data for 5 cities and display name of city having highest population.

→

```
#include<iostream>
using namespace std;
class city
{ public:
 string name;
 int population;
 void accept()
 { cout<<"enter city name:";
  cin>>name;
 cout<<"enter city popylation:";
 cin>>population; }
 void disp()
{ count<<" city having highest population:"<<name;}
int main()
{ city C[5];
  int i, max;
for(i=0; i<5; i++)
{C[i].accept();}
max=C[0].population;
For(i=0; i<5; i++)
{ if(( [i] population> max)
 { max=i}
C[max] disp();
 return 0; }
```

2) Wap to declare a class 'Account' having data members as account and ~~balance~~ 'balance' Accept this data For 10 accounts and give interest of 10% where blance is equal or greater than 5000 and display them.

→

```
#include<iostream>
 class account.
{ public:
 int acc_no;
 float balance;
 void accept()
{ count<<" enter Account Number :";
 cin>>acc-no;
 count<<" enter Account Balance:";
 cin >> balance; }
 void disp()
{ count<<"\n Account Number :"<< acc-no;
 count<<"\n Account Balance:"<< balance;}};
int main()
{ account A[10];
 int i;
for(i=0;i<10;i++)
{A[i] accept() }
For(i=0; i<10 ;i++)
{ if (A[i] balance>=5000)
{A [i] balance =A[i] balance +(0.1 * A(i));
A[i]balance);
A[i]disp(); }}
return0; }
```

9) Wap 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>
using namespace std;
class staff
{ string name;
 public:
string post;
void accept()
{ cout<<"enter the staff name:";
 getline(cin,name);
cout<<"enter the staff post:";
getline(cin,post);}
void disp()
{ cout<<"\n staff name:"<<name;
cout<<"\n staff post:"<<post;}};
int main()
{ staff s[5];
 int i;
for(i=0;i<5;i++)
{ s[i].accept();}
for(i=0;i<5;i++)
{if(s[i].post=="HOD")
{ s[i].disp();}}
return 0;}
```

21/9/25

experiment-3

1) write a program to declare a class 'book' containing data members as book_tittle, author name and price. Accept and display the information For one object using a pointer to that object

→

```
# include <iostream>
using namespace std;
class book
{ string booktittle;
  string authorname;
  int price;
public:
void accept()
{ count << " enter Book tittle :";
getline (cin, booktitle);
cout << " enter Author name :";
getline (cin, authorname);
cout << " enter BOOK price:";
cin >> price; }
void disp ()
{ count << "Book Title : " << booktitle;
count << "\n Author Name :" << authorname;
cout << "\n Bookprice :" << price; }};
int main()
{ book B1;
  book *p;
  p = &B1;
  p -> accept ();
  p -> disp ();
  return 0; }
```

2) wAp to declar a class 'student' having

→
```
# include <iostream>
using namespace std;
class student.
{ int roll_no;
float percentage;
public:
void accept()
{ count << " enter the student Roll no :";
cin >> roll_no;
count << " enter student percentage : ";
cin >> percentage; }
void disp()
{ this → accept();
cout << " \n Roll no of the student : " << roll_no;
cout << " \n percentage of the student : " << percentage;
} } ;
int main()
{ student s1;
s1 disp();
return 0;
```

3) write a program to demonstrate the use of nested class?

→
```
# include <iostream>
using namespace std;
class student
{
```

```
private:
  string name;
  int roll_no;
public
  void input Student Details()
{
cout<<"enter Student's name:";
getline(cin,name);
cout<<"enter roll number:";
cin>> roll no;
}
void display Student Details()
{
 cout<<"\n Student Name:"<<name<<endl
cout<<"ROLL NO:"<<roll_no<<endl;
 }
class Marks
{
  private;
  int science;
  int english;
  public:
  void input Marks()
{
cout<<"enter marks in science:";
 cin>> science;
cout<<"enter marks in english:";
cin>> english;
}
void display Markes()
{
```

```
cout << "Marks in science : " << science << endl;
cout << " Marks in english :" << english << endl;
    }
  }
}
int main()
{
 Student S1;
Student :: Marks M1;
S1 . inputStudent Details();
M1 . input Marks();
S1 . display Student Details();
M1 . display Marks ();
return 0;
}
```

2/9/25

* practice Question

1) 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 classB;
class classA{
   int A;
public:
void accept(){
count << "enter value For A:";
 cin >> A;
}
Friend int sum (class A &A, class &B);
 }na;
class classB{
     int B;
public:
 void accept(){
count << " enter value for B:";
cin >> B;
}
Friend int sum (classA &A, class B &B);
}nb;
int sum (classA &A, class &B) {
 return &A.A + &B.B;
}
int main () {
```

```
na accept ();
nb.accept ();
cout << "sum: " << sum (na, nb) << endl;
return 0;
}
```

2) write a program with a class Number that contains a private integer. use a friend function swapnumbers (Number & Number &) to swap the private values of two Number objects.

→

```
# include <iostream>
using namespace, std;
class Number {
    int num;
public:
    void accept () {
    cout << "enter value:";
    cin >> num;
}
void display () {
count << "value : " << num << endl;
}
friend void swap Number (Number &num1,
                         Number is num2);
};n1, n2;
void swap number (Number &, Number &) {
int temp = n1.num;
n1.num = n2.num;
n2.num = temp;
}
```

```
int main() {
  n1.accept();
n2.accept();
cout<<"Before Swape:"<<endl;
 n1.display();
n2.display();
Swap Number(n1, n2);
cout<<"After Swap:"<<endl;
 n1.display();
 n2.display();
 return 0;
}
```

3) 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 cube;
class Box {
  double v;
public:
  void accept() {
  double l, w, h;
  cout<<" enter length, with, and height for
     the Box:";
  cin>>l>>w>>h;
   v=l*w*h;
} Friend void Find greater (Box b, cube c);
```

```
} b;
class cube {
 double v;
public:
 void accept() {
 double s;
cout << "enter Side length for the cube:";
cin >> s;
v = s * s * s;
} friend void Find greater (Box b, cube c);
} c;
void Find greater (Box cube) {
String res = (b.v > c.v) ? "Box" : ((c.v > b.v) ?
"cube": "Both equal");
cout << "Greater volume:" << res << endl;
}
int main () {
 b accept();
 c accept();
Find greater (b, c);
 return 0;
 }
```

4) Create a class complex with real and imaginary parts a private member use a Friend function to add two complex number and return the result as a never complex object.

→

```
# include < iostream >
using namespace std;
class complex {
```

```
double r, i, j
public:
void accept () {
cout << "enter real and imaginary parts :";
cin >> r >> i;
}
void display () {
cout << r << "+" << i << "i" << endl;
} friend complex add (complex c1, complex c2);
} c1, c2;
complex add (complex, complex) {
  complex sum;
  sum r = c1 r + c2 r;
  sum i = c1 i + c2 i;
  return sum;
}
int main () {
  c1 accept ();
  c2 accept ();
complex sum = add (c1, c2);
cout << "the sum is :";
 sum display ();
 return 0;
}
```

5) Create a class student with private data members: name and three subject marks write a friend function calculate-Average (student) that calculate & display the average marks.

→)

```
# include <iostream>
using namespace std;
class student {
  string n;
  int m[3];
public:
  void accept() {
  cout<<" enterstudent name :";
  cin>>n;
  cout<<"enter marks For three subject:";
  cin>> m[0] >>m[1] >> m[2];
  } friend void calculate Average (student s);
} s;
void calculate Average (student) {
double avg = (s.m[0] + sm[1] + sm[2])/3.0;
cout<< " student :" << s.n <<endl;
cout<<" Average marks :" <<avg <<endl;
}
int main() {
  s.accept();
  calculate average(s);
  return 0;
}
```

6) Create a class point with private members x and y. write a Friend Function that calculate & returns the distance between two point object.

```
→ #include<iostream>
  #include<cmath>
  using namespace std;
   class point {
      double x, y;
  public:
   void accept() {
   cout << "enter x and y coordinates:";
   cin >> x >> y;
  } friend double dist(point p1, point p2);
  } p1, p2;
  double dist(point, point) {
   double dx = p1.x - p2.x;
   double dy = p1.y - p2.y;
  return sqrt((dx * dx) + (dy * dy));
  }
  int main() {
    p1 accept();
    p2 accept();
   cout << "Distance:" << dist(p1, p2) << endl;
   return 0;
  }
```

classmate
Date
Page

Experiment 4

Q write a C++ program to resolve the following problem statement and show output.

1) Swap two number from same class using object as function argument write swap function as member function.

→

```
#include <iostream>
using namespace std;
 class num{
public:
    int n1, n2;
 void accept(){
 cout << "enter First number:";
 cin >> n1;
cout << "enter Second number:";
 cin >> n2;
 }
 void display(){
 cout << "Num1: " << n1 << endl;
 cout << "Num2: " << n2 << endl;
 }
 void swap (Num & obj){
 int temp = obj n1;
 obj n1 = obj n2;
 obj n2 = temp;
 cout << " swapped :-\n";
cout << "First number : " << obj n1 << endl;
cout << "Second number :" << obj n2 << endl;
 }
};
```

```
int main(){
 num nu
 nu accept();
 nu display();
 nu.swap(nu);
 return 0;
}
```

Output:-

```
enter First number :5
enter second number :9
Num 1 :5
Num 2 :9
Swapped :-
First number :9
second number :5
```

e) Swap two number From same class using concept of Friend Function.

→

```
# include<iostream>
using namespace std;
class num{
public:
  int n1,n2;
 void accept(){
cout<<"enter first number:";
cin>>n1;
cout<<" enter second number:";
cin>>n2;
}
```

```
void display () {
  cout << "Num1 : " << n1 << endl;
  cout << " Num2 : " << n2 << endl;
  }
Friend void swap (num &nu);
}
void swap (num & nu) {
  int temp = nu.n1;
  nu.n1 = nu.n2;
  nu.n2 = temp;
  cout << " Swapped : \n";
  cout << " First : " << num.n1 << endl;
  cout << " second : " << nu.n2 << endl;
  }
int main () {
   num nu;
   nu.accept();
   nu.display ();
   swap (nu);
   return 0;
  }
```

- Output :-

```
Enter first number : 5
Enter second number : 9
 Num 1 : 5
Num 2 : 9
Swapped :
First : 9
second : 5
```

3) Swap two number from different class using Friend function.

```
#include <iostream>
using namespace std;
class num1;
class num2;
class num1 {
public:
  int n1;
void accept () {
 cout<<" enter First number :";
 cin>>n1;
 }
 void display () {
 cout<<" Num1 :"<<n1<<endl;
 }
 Friend void swap (num1 &nu1, num2 &nu2);
};
class num2 {
public:
int n2;
void accept () {
 cout<<" enter second number :";
 cin>>n2;
}
void display () {
 cout<<"Num2 :" <<n2<<endl;
}
Friend void swap (num1 &nu1, num2 &nu2);
};
```

```
void swap (num1 &nu1, num2 &nu2) {
    int temp = nu1, n1;
    nu1, n1 = nu2, n2;
    nu2, n2 = temp;
    cout << "swapped : \n";
    cout << "First : " << nu1, n1 << endl;
    cout << "Second : " << nu2 n2 << endl;
}
int main () {
    num1 nu1;
    num2, nu2;
    n1 accept ();
    n2 accept ();
    nu1 display ();
    nu2 display ();
    swap (nu1, nu2);
    return 0;
}
```

output :-

```
enter First Number : 5
enter second Number : 9
Num 1 : 5
Num 2 : 9
swapped :
First : 9
second : 5
```

4) Create two classes Result 1 and Result 2 which stores the marks of the student Read the value of marks for both the class object & compute average.

→

```
#include<iostream>
using namespace std;
class Result 1 {
public
  int r1;
 void accept () {
 cout << " enter First result : ";
 cin >> r1;
}
} res 1;
class result 2 {
public:
  int r2;
void accept () {
 cout << " enter second result : ";
  cin >> r2;
  }
} res2;
Float avg (const Result 1 & res1, const Result
2 & res2) {
return (res1 r1 + res2 r2) /2 of;
}
int main () {
res 1 accept ();
res 2 accept ();
Float av = avg (res 1, res2);
```

```
cout << " Average : " << av;
return 0;
}
```

- Output :-

```
Enter First result: 95
Enter second result: 99
Average : 97
```

5) Find the greatest number among two numbers From two different classes using Friend.

->

```
#include <iostream>
using namespace std;
class num1;
class num2;

class num1 {
public:
 int n1;
void accept() {
 cout << " enter First number :";
 cin >> n1;
}
 friend int gtnum(const num1 ,const num2 );
};nu1;
class num2 {
public:
 int n2;
 void accept(){
cout << " enter second number :";
 cin >> n2;
}
```

```
friend int gnum (const num1 &, const num2 &);
};
int gnum (const num1 &a, const num2 &b) {
return (a.n1 > b.n2) ? a.n1 : b.n2;
}
int main () {
num1.accept ();
num2.accept ();
int gr = gnum (n1, n2);
cout << "Greatest:" << gr;
return 0;
}
```

Output:-

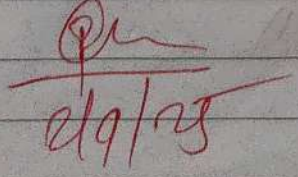
```
Enter First number : 5
Enter second number : 9
Greatest : 9.
```

6) Create three classes: Alpha, Beta & Gramma, 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 Gramma;
class Alpha {
int n; public:
void accept () {
```

```
cout << "enter value for Alpha : ";
cin >> n;
} friend int sum (Alpha a, Beta b, Grammac);
};
class Beta {
int n; public:
void accept () {
cout << " enter value for Beta :";
cin >> n;
}
Friend int sum (Alpha a, Beta b, Grammac);
} b;
class Gramma {
int n; Public :
void accept () {
cout << "enter value for Gramma :";
cin >> n;
} friend int sum (Alpha a, Beta b, Grammac);
} c;
int sum (Alpha a, Beta b, Grammac) {
return an + bn + cn; }
int main () {
a accept ();
b accept ();
c accept ();
cout << "sum :" << sum (a, b, c) << endl;
return 0;
}
```



7) Create two classes : Bank account and Audit. Bank Account holds private balance information and a friend function in Audit that accesses and prints balance information for auditing

→

```
#include <iostream>
using namespace std;
class BankAccount;
class Audit {
public:
    void pBal (BankAccount);
};
class BankAccount {
double b; public:
    void accept(){
cout<< "enter bank account balance:";
cin >> b;
}ba;
void Audit::pBal (BankAccount) {
 cout<< "Auditing .... Bank Account balance:"
    << ba.b << endl;
}
 int main () {
    ba.accept();
    and.pBal (b, a);
    return 0;
}
```

Experiment No 5

Q1)

→

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

class number printer {
  public:

  Number printer (int n) {
cout << "Numbers from 1 to " << n << ":" << endl;
  for(int i=1; i<=n ; i++) {
    cout << i << " ";
  }
  cout << endl;
  }
};

int main() {
    int n;

cout << " enter the value of n : ";
    cin >> n;

Number printer np(n);

    return 0;
}
```

output :-

Enter the value of n : 2

number From 1 to 2 :

1 2

Q2)

→

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

class student {
private:
 string name;
 float percentage;

public:

student (string n, float p) {
    name = n;
    percentage = p;
}

 void display () {
cout << "Student Name :" << name << endl;
cout << "percentage :" << percentage << "%." << endl;
 }
};
int main () {
string name;
float percentage;

cout << "enter studet name:";
 getline (cin, name);

cout << "Enter percentage:";
cin >> percentage.
```

```
student S(name, percentage);
 cout << "\n student Details : \n";
    S.display();

  return 0;
}
```

Output :-

Enter student name : Atharv
Enter percentage : 87

Student Details :
student Name : Atharv
percentage : 87%.

Q3)

→

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

class collage {
private:
   int roll_no;
    string name;
    string course;
public;
  college( int r, string n, string c : "computer Enginning"){
  roll_no = r;
  name = n;
  course = c;
  }
```

```
void display () {
cout << "Roll no :" << roll_no << endl;
cout << " Name :" << name << endl;
cout << " course " << course << endl << endl;
    }
};
    int main () {
    int roll;
  string name, course;

cout << " enter details for student 1:" << endl;
cout << " Roll No:";
  cin >> roll;
  cin ignore ();
cout << " Name :";
get line ( cin, name );
college student ( roll, name);

cout << " In Enter details for student 2:" << end
   cout << " Rollno:";
    cin >> roll;
    Cin . ignor()
    cout << " Name:";
  get line ( cin, name);
  coub << " course:";
   get line ( cin, course);
college Student 2 (roll, name, course);

cout << " In student details In ";
   student display ();
  Student 2 . display ();
```

```
return 0;
}
```

Output:-

```
Enter details for student1:
    Roll No : 9
    Name : shripad.

Enter details For student 2:
    Roll No : 69
    Name : Atharv

Student Details:
Roll No : 9
Name : Atharv
course : computer Engineering

Roll No : 69
Name : Atharv
course : CSE
```

d)

→

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

class Box {
private:
  double length, width, height;

public:
  Box() {
  length = width = height = 0;
cout << "Default constructor called" << endl;
 }
Box ( double side ) {
  length = width = height = side;
cout << cube constructor called" << endl;
}
Box(double l, double w, double h) {
    length = l;
  width = w;
    height = h;
  cout << "Box constructor with length,
width, height called." << endl;
}
  double volume() {
  return length * width * height;
}
};
  int main() {
   Box box1;
   Box box2 (5.0);
```

```
Box box3 (2.0, 3.0, 4.0);

cout << "volume of box1: " << box1.volume() << endl;
cout << "volume of box2: " << box2.volume() << endl;
cout << "volume of box3: " << box3.volume() << endl;

return 0;
}
```

Output:-

```
Default constructor called,
cube constructor called,
Box constructor with length, width, height called
volume of box1: 0
volume of box2: 125
volume of box3: 24
```

12/11

experiment NO 6

1)

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

class person {
protected:
string name;
int age;

public
person(string n, int a): name(n), age(a) {}
};

class Student : public person {
private:
string roll_no;

public:
student (string n, int a, string r) : person(n, a),
roll_no (r) {}

void display_details () {
cout << "Name: " << name << endl;
cout << " Age: " << age << endl;
cout << "RollNO: " << roll_no << endl;
}

int main () {
Student student ("Alie", 20, "A123");
```

```
student display details();

return 0;
}
```

output:-

Name: Alice
Age: 20
Roll NO: A123

2)

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

class Academic {
protected:
    int accademic_score;

public:
Academic (int score): accademic_score(score){}
};
classsports{
 protected:
    int sports_score;

public:
sports (int score): sports_score (score){}
};
class Result: public, Academic, public Sports {
public:
Result (int a score, int s score): Academia score, sports(s.score)
                                                    {}
```

3) WAP to implement hierarchical inheretance Assume suitable data.

→

```
#include<iostream>
using namespace std;
class vehical.
{
 protected:
   string brand;
   int model,
};

class car: protected vehical
{
 protected:
   string types;
};

class electricCar : proteced car.
 {
 int batterycapacity.
public:
void accept()
 {
cout<<"enter the brand of car:";
cin >> brand;
cout << enter the model of the car:";
cin >> model;
cout<<"enter the type of car:";
cin >> type;
```

```
cout<<"enter the battery capacity of the car:";
cin>>battery capacity;
}
void display ()
{
cout<<"\n the brand of the car:"<< brand<<endl;
cout<<" the model of the car:"<<model<<endl;
cout<<" the type of the car:"<<type<<endl;
cout<<" the battery capacity of the car:"<<battery
capacity<<endl;
}
};

int main()
{
electricar e;
e.accept();
e.display();
return 0;
}
```

output.

enter the brand of the car: BMW
enter the model of the car: M4
enter the type of the car: Sedan sport.
enter the battery capacity: 115.

4) WAP to implement hybrid inheritance

→
```
#include <string>
#include <iostream>
using namespace std;
class person
{
public:
string name;
int age;

void getPersonDetails()
{
cout << "enter name:";
cin >> name;
cout << "enter age:";
cin >> age;
}
void showDetails()
{
cout << "Name" << name << ", Age:" << age << endl;
}
};

class student: public person.
{
public:
string course;
void getDetails()
{cout << "enterCourse:";
}cin >> course;
```

```
void showDetails()
{
cout<<" Course : "<<course<<endl;
}
};

class employee : public person
{
public:
 string company;
 void getemployeeDetails()
{
 cout<<"enter company:";
 cin>>company;
}
void Showemplayee Details()
{
cout<<"company:"<<company<<endl;
}
};

class Intern: public student, public
 employee {
public:
 void showInternDetails()
 {
cout<<"n--- Intern Details ---\n";
student :: showpersonDetails();
 show employe Details();
 }
};
```

```
int main ()
{
 Intern it;
it student:: get personDetails ();
it getDetails ();
it get employeeDetails ();
it Show InteanDetails ();

return 0;
}
```

Output
enter name: Aditya
enter age: 21
Enter course: Computer Science
enter company: Microsoft.

—— Intern Details ——
Name: Aditya, Age: 21
course: Computer Science.
company: Microsoft.

12/11

Experiment NO.7

a)

→
```
#include<iostream>
using namespace std;
class Area;
{ private:
 float l,b;
public
 void area (float l)
{ float area;
   area = l *b
  cout<<"Area of square :"<< area << endl;
}
void area ( float l, float b)
{
 float area;
 area = l*b;
cout << Area of rectangle :" << area;
 }
};
int main()
{
Area a1;
a1.area (8)
a1.area (4,12);
 return 0;
}
```

output:-

Area of square : 64

Area of rectangle : 48

b)

→
```
#include <iostream>
using namespace std;
class sum
{ private :
int a,b,c,d,e,f,g,h,i,j;
float k,l,m,n,o;
public:
void sum(float k, float l, float m, float n, float o);
float sum;
sum = k+l+m+n+o;
cout<< sum of 5 floating number : "<<sum<<endl;}
void sum (int a, int b, int c, int d, int e, int f, int g,
          int h, int i, int j ) {
int sum;
sum = a+b+c+d+e+f+g+h+i+j;
cout<< sum of 10 intergers : "<<sum;
}};

int main ()
{
sum s1;
s1.sum (1.2, 3.5, 5.6, 8.9, 1.5);
s1.sum (1,2,3,4,5,6,7,8,9,10);

return 0; }
```

Output :

Sum of 5 floating number : 20,2

Sum of 10 integers : 55.

c)

```
#include <iostream>
using namespace std;
class Number
{
private: int x;
public void accept()
{
cout<<"enter a number:";
cin>>x;

void operator -()
{
  x=-x;
}
  void display()
{
cout<<"Negated number:"<<x<<endl;
}
};
  intmain()
{
Number n1;
n1.accept();
-n1;
n1 display();
  return0;
}
```

output

Enter a number : 5

Negated number : -5

d)

→
```
#include<iostream>
using namespace std;
class Number
{
private : int x; int accept;
public : { void accept ()
        cout<<"enter a number";
    cin>>x;
temp = x; }

  void operator ++ ()
  {
    x = ++x;
  }
  void reset ()
  { x = temp;
  }
  void operator ++ (int)
{
  x = x++;
}
  void display ()
{ cout<<(pre) The number is :"<< x<<endl;
}
void display 2()
{
cout<<" (post) The numb eris : "<<x;
}
};
```

```
int main()
{
 Number n1;
n1.accept ();
++n1;
n1.display ();
n1.reset ();
n1++;
n1 display 2 ();
return 0;
}
```

Output:

enter a number : 5
(pre) The number is 5
(post) Th number is : 5

Experiment No. 8

a)

→
```
#include<iostream>
#include<string.h>
using namespace std;
class mystring
{ char str[20];
 public:
 mystring(char a[20])
{ strcpy(str,a);
}
 void display()
{ cout<<"string is: "<<str<<"\n";
}
 my string operator +(mystring);
};
mystring mystring::operator +(mystring s2)
{ mystring s3;
  strcpy(s3.str, str);
  strcat(s3.str, s2.str);
  return s3;
}
int main()
{ mystring s1("abc"); mystring s2("xyz");
 s3=s1+s2;
 cout<<"\n concatenated";
 s3.display();
 return 0;
}
```

b)

→

b)

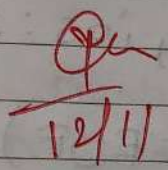
```
#include <iostream>
using name space ; std;
class ilogin
{ protected:
    char n[20], p[20];
public
   virtual void accept ()
{ cout << "enter name:";
   cin >> n;
   cout << "enter password:";
   cin >> p;
}
 virtual void public()
 { cout << "\n name is: " << n;
   cout << "\n password is :" << p;
 }
};
class email_login: public ;login
{ public:
    void accept ()
  { cout << " enter email id: ";
cin << "enter mail-id password:";
cin >> l;
}
Void display ()
{ cout << "\n email id is : " << n;
  cout << "\n email id password is :" << p;
}
};
```

```
class membership-login : public ilogin
{ public:
    void accept ()
  { cout<<"enter membership name:";
    cin>>n;
    cout<<"enter membership password:";
    cin>>p;
  }
  void display ()
  { cout<<"\n membership name is :"<<n;
  cout<<"\n membership password is :"<<p;
  }.
};
int main()
{ ilogin* ill, ill;
  email_login el;
membership-login ml;

int type;
cout<<"\n1:- For email login \n:- For membership
        login \n3:- normal login:";
cin>>type;

if(type==1)
{ ill = &el
  ill->accept();
  ill->display();
}
elseif (type==2)
{ ill = &ml;
```

```
i1 -> accept();
i1 -> display();
}
else if(type == 3)
{ i1 = &i11;
  i1 -> accept();
  i1 -> display();
}
return 0;
}
```



experiment no 9

1)

→
```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{ ifstream f1;
  ofstream f2;
  char ch;

f1.open("myfile.txt", ios::in);
f2.open("abc.text", ios::out);

while((ch=f1.get())!=EOF)
{
  f2.put(ch);
}
f1.close();
f2.close();
cout<<"contents copied:";
return 0;
}
```

b)

→
```
#include <iostream>
#include <fistream> #include <ctype.h>
using namespace std;
int main()
{
  ifstream f1("myfile.txt");
  char ch;
  int d=0; s=0;
```

c)

→

```
while (ch = fl.get ()) != EOF)
{
  if (isdigit (ch))
  {
    d++;
  }
  else if (isspace (ch))
  {
    s++;
  }
}
cout << "\n number of digits in file are:" << d;
cout << "\n number of spaces in file are:" << s;

return 0;
}.
```

c)

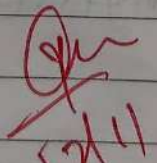
→

```
#include <iostream>
#include <fstream>
#include <ctype.h>
using namespace std;
int main()
{
ifstream fl("myfile.txt");
char ch;
int w=0;
while((ch=fl.get()) != EOF)
{ if (isspace (ch))
    { w++;
}
} cout << " total words are:" << w;
```

d)

→

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
 {
ifstream fl("myfile.txt");
 string ch;
 int w=0;
cout<<"enter a word:";
cin>>ch;
 string word;
 while (fl>>word)
 {
 cout<<"\n"<<word;
  if(ch=word)
   { w++;
  }
}
 fl.close();
cout<<"\n occuriance of a word in file is
          :"<<w;
return 0;
}
```



Experiment NO-10

1)

→
```
#include<iostream>
using namespace std;
template<class t>
t sum (t arr[10])
{ int i;
 t add=0;
For (i=0; i<10; i++)
{
 add = add + a[i];
}
 return add;
}
int main()
{ Float arr[] = {12,1,3,5,6,3,4,9,10,14,15};
cout<<"\nsum of array elements is :"<<sum(arr);
return 0;
}
```

2)

→
```
#include<iostream>
using namespace std;
template<class t> t square (t num)
{
return num*num;
}
template<> string square (string str)
{
template return str * str
}
int main()
```

```
{ int n=5;
  string t="hello";

cout<<"Square of no is :"<<square(n);
cout<<"square of sting is :"<<square(t);

return 0;

}
```

3)

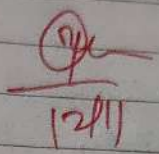
→

```
#include<iostream>
#include<math.h>
using namespace std;
template<class t> class cal
{  t a,b;
public:
 cal(tc,td)
 { a=c;
   b=d;
 }
 t add()
 { return a+b;
 }
 t sub t()
 { return a-b;
 }
 t div()
 { return a/b;
 }
 t mul t()
```

```
{ return a-b;
}
tsqrt()
{ return sqrt(a);
}
tcbrt()
{ return cbrt(a);
}
tpow()
{ return pow(a);
}
tsin()
{ return sin(a);
}
tcos()
{return cos(a);
}
tTan()
{ return tan(a)
}
};
int main()
{ int a,b,ch;
cout<<"enter two numbers";
cin>>a>>b;
cal c Float>c1(a,b);
do {
printf("\n1: to add \n2: Subtract\n3: divide\n4:
multiply \n5: square\n6: cube root\n7: power
\n8: Sin \n9: cos\n10: Tan \nall : exit \n"
enter your choice :";
```

```
scanf("%d", &ch);
switch (ch)
{ case1:
  cout<<"addition is:"<<cl.addu;
break;
case2:
cout<<" subtraction is:"<<cl.subt C);
break;
case3:
cout<<" division is:"<<cl.div C);
break;
case4:
cout<<" multiplication is:"<<cl.mult();
break;
case5:
 cout<<" Square root is:"<<cl.sqrt();
break;
case6:
cout<<" Cube root is:"<<cl.(brt();
break;
 case7:
cout<<" sin is:" <<cl.sin();
break;
case 8:
cout<<"cos is:"<<cl.cos();
break;
case9:
cout<<" tan is:"<<cl tan();
break;
case10;
cout<<" power is:"<<cl.pow();
```

```
}
} while (ch!='n');
return 0;
}
```



experiment no. 11

a)

→
```
#include<iostream>
using namespace std;
template<typename T> class vec
{
  T *v;
  int size;
public:
vec(int n):size(n)
{
v=new T[size];
}
void set(int idx, T val)
 {
  v[idx]=val;
}
void num()
{
 for(int i=0; i<size; i++)
{ v[i]*=2;
}
}
void display()
{
cout<<"c"
for(int i=0; i<size; ++i)
{ cout<<v[i];
 if(i!=size-1)
  { cout<<",";
  }
}
```

```
};
int main()
{ int n;
cout << "enter no of elements: ";
cin >> n;
vec <int> v(n);
for (int i=0; i<n; i++)
{ cout << "enter value for elements " << i+1 << ":";
int val;
cin >> val;
v set (i, val);
}
v.display();
int idx, new val;
cout << "enter index to modify: " << n-1 << ") ":";
cin >> idx;
cout << "enter new value : " << idx;
cin >> new val;
v.set (idx, new val);
cout << "vector after multiply:";
v.display;
return 0;
}.
```

b) with iterator:-

→

```
#include <iostream>
#include <vector>
using namespace std;
int main() {
    vector <char> v(10);
    vector <char> :: iterator p;
   int i;
  p = v.begin();
   i=0;
while (p1 = v.end()) {
    p = it 'a';
    p++;
    it++;
}
cout << "original content : \n";
   p = v.begin();
 while (p1 = v.end()) {
     cout << *p << " ";
    p++;
}
```

12/4

Experiment NO:-12

a) Implement Stack

```
#include<bits/stdc++.h>
using namespace std;
int main(){
    Stack<char> car;
    cars.push("BMW").cars.push("Audi").
    cars.push("Mercedes").car.push("Ferrai");
    cout<<"Top element is:"<<cars.top()<<endl;
    cout<<"size of stach is:"<<cars.size()<<endl;
    cars.pop(); car.pop();
while(!cars.empty()){
 cout<<"element in stack are:"<<cars.top()<<endl;
car.pop();
}
 return 0;
}
```

Output:-

Top elements: Ferrari
Size of stack is: 4
element in stack are: Audi
element in stack are: BMW

b) Implement Queue:-

```
#include <lists/stdc++h>
using namespace std;
int main() {
    queue<int> age;
age.push(21); age.push(22); age.push(23);
age.push(24);
cout<<" Front element is :"<<age.front()<<endl;
cout<<" Back element is :"<<age.back()<<endl;
age.pop(); age.pop();
while(!age.empty())
{
cout<<"element in queue are :"<<age.front()<<endl;
age.pop(); }
return 0;
}
```

op:-

Front element is: 21
back element is: 24
element in queue are : 23
element in queue are : 24

12/11