Experiment No : 01

Q.1] // Simple addition using fn.

→

```
#include <iostream>
using namespace std;
class num {
    private:
    public:
    int a=4, b=5, c;
    void add () {
        c=a+b;
    cout <<" Addition = " <<c;
    }
};

int main () {
    num n1;
    n1.add ();
    return 0;
}
```

Output :

Addition = 9

Q.2] // Arthamatic operation.

→

```
#include <iostream>
using namespace std;
int main () {
    int a,b,sum;
    cout <<" Enter a no = ";
    cin >> a;
    cout <<" Enter another no = ";
    cin <<b;
    sum = a+b;
    cout <<"The sum of 2 No = "<<sum;
    return 0;
}
```

output:

```
Enter a no = 4
Enter another no = 5
The sum of 2 No = 9
```

Q.3] // Check odd or even.

→

```
#include<iostream>
using namespace std;
int main () {
    int a;
    cout <<" Enter a no = ";
    cin >> a;
    if (a%2 == 0) {
    Cout << " The no is even ";
}
```

```
else {
  cout << "The no is odd ";
  }
  return 0;
}
```

output:

Enter a no = 5
The no is odd

Q.4] // Print no 1-10 using for loop.

→

```
#include <iostream>
  using namespace std;
  int main () {
     int i;
    for (i=0; i<=10; i++) {
       cout <<i << "\n";
    }
   return 0;
}
```

Output:

1
2
3
4
5
6

Q.5] // Using switch case statement.

→

```
#include <iostream>
using namespace std;
int main () {
  int choice, a, b;
  cout.<<"Enter two no.= ";
  cin >>a>> b;
  cout << " 1. Add \n 2. Sub \n 3. Mult \n
          4. divid ";
  cout <<" Enter choice = ";
  cin >> choice;
  switch (choice) {
    case 1:
    cout <<a+b;
    break;

  case 2:
  cout <<a-b;
  break;

  Case 3:
  cout <<a* b;
  break;
```

```
        case 4:
        cout << a/b;
         break;

        default:
         cout << " wrong input ... ";
      }
     return 0;
}
```

Output:

```
Enter two no = 4 5
1] Add
2] Sub
3] Mult
4] Divide
```

```
Enter choice = 3
 20
```

Q.6] // Print 1-10 no using while loop.

→

```
#include <iostream>
 using namespace std;
 int main () {
  int i=1;
 while (i<=10) {
  cout <<i<< "\n";
   i++;
   }
return 0;
}
```

Output :

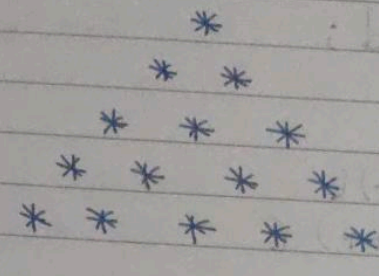
1
2
3
4
5
6
7
8
9
10

Q.7] Create the following patterns.

a]
→

```
# include <iostream>
using namespace std;
int main () {
  int i,j,k;
for (i=0;i<5;i++){
   for (j=0;j<=5-i;i++){
printf (" ");
}
for (k=0; k<i+1; k++){
    cout <<" * ";
    }
   cout <<" \n ";
   }
}
```

Output :



b)

→
```
#include <iostream>
using namespace std;
int main () {
    int i,j;
for (i=1; i<=5; i++) {
    for (j=1;j<=i;j++) {
    cout <<j;
    }
    cout <<"\n";
    }
    return 0;
}
```

Output :

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

c]

→
```
#include <iostream>
using namespace std;
int main(){
  int i,j;
 for(i=1;i<=5;i++){
    for(j=1;j<=i;j++){
      cout<<i;
    }
    cout<<"\n";
  }
  return 0;
}
```

Output:

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

e.8] // Using Class fn:

a] Write a program to declare a class student... data member - Roll no., name - Accept/display data from ~~one~~ one object.

→

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

```
Private :
Public:
int roll, prn;
string name [10];

void accept ()
{
cout <<" \n ";
cout << " Enter name, roll & prn = ";
cin >> name >> roll >> prn;
}
};
int main ()
{
  student s[20];
   int i,n;
cout <<" Enter no. of student = ";
cin >> n;
for (i = 0; i<n; i++)
{
   s[i].accept ();
}
printf (" \n %d ", s[2].prn);
  return 0;
}
```

output:

Enter no. of student = 2

Enter name, roll & prn = abc 12 126
Enter name, roll & prn = xyz 13 126
126

b] Write a program to declare - class → book, member - no-page, price, name:

→

```
#include <iostream>
using namespace std;
 class book {
   private:
  public;
  int n, i, great;
  int no-page [5], price [5];
  string name [5];

   void accept () {
cout <<"Enter no.of books =";
cin >> n;
 for (i=0; i<n; i++) {
cout <<"Enter book name, price, no.of pages:";
cin >> name [i] >> price [i] >>
   no-pages [i];
 cout <<"\n";
   }
};
 void display () {
  for (i=0; i<n; i++) {
cout <<"The book details =" <<name << " " <<
       price <<" " << no-pages ",\n";
 }
greate = price [0];
  for (i=0; i<n; i++) {
  if (price [i] > greate) {
  great = price [i];
```

```
cout << " The book of greater price = << name[i];
 break;
  }
  }
  }
};
 int main() {
     book b1;
     b1.accept();
     b1.display();
  }
```

Output:

Enter the no.of books = 2

Enter book name, price, no.of pages = xyz 150 120

Enter book name, price, no.of pages = YYz 120 154

The book details = xyz 150 120

~~The book details~~ = YYz 120 154

The book of greater price = xyz

c] To convert hrs, min & sec to total sec

→

```
#include <iostream>
using namespace std;
class time {
  private:
  public:
int hrs, min, sec, total;
   void A&D() {
cout << "Enter time in hrs, min & sec = ";
 cin <<hrs << min <<sec;
  total = ((hrs * 3600) + (min * 60) + sec);
  cout =<< " The time in total sec = "
        << total;
    }
};
int main () {
    time t1;
   t1. A&D();
```

output:

Enter time in hrs, min & sec = 12 34 55
The time in total seconds = 45295

11/8

a.1]

→

Experiment No:02

Q.1] // To accept & display name & population of a city.

→

```
#include <iostream>
using namespace std;
class city
{
 Private:
 string name[5];
 int pop[5], n, i;
 public:
  int great;
  void accept ()
 cout <<" Enter the no.of city = ";
 cin >>n;
 for (i=0; i<n; i++){
 cout <<" Enter name & Pop = ";
 cin >> name >> Pop;
 }
}
 void display ()
  for (i=0; i<n; i++){
 cout <<" \n City = " <<name
 <<" Population = " <<
   pop [i];
 }
}
```

```
void comp ()
{
 great = pop[0];
for(i=0;i<n;i++)
{
 if(pop[i]>great)
{
 great = pop[i];
cout<<"\n The city with greater pop = "
    <<name[i];
 break; }
    }
  }
};

int main()
{
  city c1;
  c1.accept();
  c1.display();
  c1.comp();
}
```

Output:

```
Enter the no. of city = 5
 Enter name & pop = a 10000
 Enter name & pop = b 5001
 Enter name & pop = c 4001
 Enter name & pop = d 12000
 Enter name & pop = e 11000
```

city = a  pop = 10000
city = b  pop = 5001
city = c  pop = 4001
city = d  pop = 12000
city = e  pop = 11000

The city with greater pop = d

Q.2] // Write a program to data member - Acc no. & Bal with class Account.

→

```
#include <iostream>
using namespace std;
class account {
   private:
   public:
int acc-no, bal;
 string name;
  void accept () {
cout <<" Enter the name. acc-no & bal = ";
cin >> name >> acc-no >> bal;
  }
};
 int main () {
 int i, n;

printf (" Enter the no. of person = ");
 scanf ("%d", &n);
account a[n];
for (i=0; i<n; i++) {
a[i].accept ();
```

```
}
for (i=0; i<n ; i++){
 if (a[i].bal >= 5000){
  a[i].bal = a[i].bal + a[i].bal*0.1;
 }
}
for (i=0; i<n; i++){
 cout << "\n costomer name = " << a[i].number;
 cout << "\n balance Acc.no. = " << a[i].acc_no;
cout << "\n balance after intrest = " <<
  a[i].bal;
 cout << endl;
 }
 return 0;
}
```

Q.3] →

Output :

Enter the no. of person = 3
Enter the name, acc-no & bal = a
123 500.1 6001
Enter the name, acc-no & bal = b 234
Enter the name, acc-no & bal = c 345 3001

Customer name = a
acc-no = 123
balance after interest = 5001

Customer name = b
acc-no = 234
balance after interest = 6001

Customer name = c
acc-no = 345
balance after interest = 3001

Q.3] Write a program to data members - name & post with class staff

→

```
#include <iostream>
using namespace std;
 class staff {
   private:
   public:
string name, post;
   void accept (){
 cout <<" Enter the name & post = ";
 cin >> name >> post;
   }
};
int main (){
int i, n;
cout <<" Enter the no. of person = ";
cin >> n;
staff s[n];
for (i=0; i<n; i++){
  s[i].accept ();
  }
for (i=0; i<n; i++){
  if (s[i].post == "HOD") {
cout <<" " <<s[i].name " is an HOD \n";
}
}
```

```
return 0;
}
```

Output:

```
Enter the no. of person=3
 Enter the name & post = Prathamesh HOD
 Enter the name & post = Ansh HOD
Enter the name & post = Vithal staff
 Prathamesh is an HOD
 Ansh is an HOD
```

11/8

Experiment No: 03

a.] Write a program to class book - data members as book - title, author - name & price. Accept & display into for 1 object using this pointer.

→

```
#include <ioshream>
using namespace std;
class Book:{
  Private:
string b-title, a-name;
  int price;
Public;
  void accept ():{
cout <<" Enter book title = ";
getline (cin, this → b_title);
cout <<" Enter author name = ";
getline (cin, this → a_name);
  cout <<" Enter price = ";
  cin >> Price;
};
Void display ():{
cout <<" The book " <<b- title <<" author
name = " <<a-name <<" & price = "
<< Price << endl;
}
}; b1;
```

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

Output:

```
Enter the book title = Apple
Enter the author name = Einsten
Enter the price = 151
The book Apple. Author name Einsten &
 price = 151
```

Q. 2] // Write a program to class student, data members roll-no, percent. Accept & display using this pointer for 1 object.

→

```
#include <iostream>
using namespace std;
class student {
  public:
  int roll-no;
  float percent;
void accept (){
cout<<"Enter roll no. & percentage =";
cin>> this ->roll - no >> this->percent;
}
```

Q. 3]

→

```
void display ()
{
  this -> Accept ();
  cout << "Roll no. =" << this -> roll_no;
  cout << " Percentage = " << this -> percent;
  cout << " Percentage =
  }
};
int main ()
{
  student s1;
  s1.display ();
  return 0;
}
```

Output:

Enter roll no. & percentage = 12 93.6
Roll No. = 12
Percentage = 93.6

Q.3] Write a program to demonstrate the use of nested class:

→

```
#include <iostream>
using namespace std;
class demo
{
  public:
  int roll_no;
  string name;
```

```
void accept () {
cout <<"Enter the name of student &
    roll_no = ";
cin >>name >>roll_no;
void display () {
cout <<" \n Name = " <<name <<"\t
roll_no = " <<roll_no << endl;
  }
 };
};
int main () {
   demo ::demo2 s1;
    s1.accept ();
    s1.display ();
    return 0;
 }
```

Output:

```
Enter the name of student & roll_no =
  Apple 12
Name = Apple
roll_no = 12
```

11/8.

# Experiment No: 04

Q.1

```
#include <iostream>
using namespace std;
class result2;
class result1 {
  int m1;
public:
  void accept(){
cout<<"Enter marks of first student:";
  cin >>m1;
}
 friend void calculate (result1 p, result2 q);
};
class result2 {
   int m2;
public:
 void accept2(){
 cout <<"Enter marks of second student:";
 cin >>m2;
}
  friend void calculate (result1 p, result2 q)
};
void calculate (result1 p, result2 q) {
    int total = (p.m1+q.m2)/2;
cout<<"total marks of both students:"<<total
                                    <<endl;
}
```

```
int main (){
 result 1 k;
 result 2 f;
 k.accept1();
 f.accept2();
 calculate (k, f);
}
```

Q.2] Swapping 2 nos using without friend function obj as function argument.

```
#include <iostream>
using namespace std;
class demo {
  public:
int p, q;
void accept (){
cout<<"enter 2 nos: "<<endl;
 cin >>p>>q;
}
 void display (){
cout<<"after swapping: "<<" value of p= "<<p
                        <<" value of q= "<<q;
}
void ~~display()~~ swap (demo &t) {
~~cout<<"after swapping~~
   int temp = t.p;
   t.p = t.q;
   t.q = temp;
 }
};
```

```
int main (){
  demo k;
  k.accept();
  k.swap (k);
  k. display();
}
```

Q.3] WAP to find the greatest number among 2 numbers from 2 different classes using friend function

```
#include <iostream>
using namespace std;
 class B;
 class A{
  public:
 int num1;
  void accept1(){
 cout<<" enter first number "<<endl;
  cin >>num1;
 }
  friend void greatest num (Ap, Bq);
};
 class B{
 public:
   int num 2;
  void accept2(){
 cout <<"enter second number "<<endl;
  cin>>num 2;
 }
 friend void greatest num(Ap, Bq);
};
```

```
void greatest num (Ap, Bq) {
 if (p.num1 > q.num2) {
 cout << "greatest number : " << p.num1 << endl;
 }
 else {
 cout << " greatest number : " << q.num2 << end
 }
}
int main () {
 Ak;
 Bf;
 k.accept1();
 f.accept2();
greatest num (k, f);
 }
```

Q.4] # swapping 2 numbers from same class by using concept of friend function

```
#include <iostream>
using namespace std;
class demo {
 int a, b;
 public:
 void accept () {
cout << "enter 2 numbers :" << endl;
 cin >> a >> b;
}
 void display () {
cout << "value of a :" << a;
cout << "value of b :" << b;
```

```
};
 friend void swapnums (demo &t);
};
 void swapnums (demo &t) {
    int temp=t.a;
     t.a=t.b;
     t.b=temp;
 }
 int main (){
  demo k;
  k.accept ();
  swapnums(k);
  k.display ();
 }
```

Q.5] WAP to swap two numbers of different class using concept of friend function.

```
#include <iostream>
using namespace std;
 class B;
 class A {
  int a;
  public:
 void accept1 (){
  cout <<"enter a :" <<endl;
   cin >>a;
 }
  void display1 (){
  cout<<"value of a :" <<a;
}
```

Q.6]

```
friend void swapnumbers (A&p, B&q);
};
class B{
  int b;
  public:
  void accept2(){
  cout<<"enter b:" <<endl;
   cin>>b;
  }
   void display2(){
   cout<<" value of b:"<< b;
  }
  friend void swapnumbers (A&p, B&q);
};
void swapnumbers (A&p, B&q){
   int temp = p.a;
   p.a = q.b;
   q.b =temp;
}
int main (){
    Ak;
    Bf;
  k.accept1();
   f.accept2();
 swap numbers (k, f);
 k.display1();
 f.display2();
}
```

Q.6] Create two classes, class A and class B, each with a private integer write a frnd func sum() that can access private data from both classes and return the sum.

```
#include <iostream>
using namespace std;
class B;
class A {
    int a;
public:
    void accept() {
        cout << "enter first no :-" << endl;
        cin >> a;
    }
    friend int sum (A p, B q);
};
class B {
    int b;
public:
    void accept2() {
        cout << "enter 2nd no :" << endl;
        cin >> b;
    }
    friend int sum (A P, B q);
};
int sum (A P, B q) {
    int sum;
    sum = P.a + q.b;
    return sum;
}
```

```
int sum main (){
    A K;
    B f;
    K. accept1();
    f. accept2();
    cout<<" sum of 2 no is :-"<<sum(K,f);
}
```

q.7] WAP to swap 2 private integers of ~~of~~ of same class using a friend fun$^c$.

```
#include <iostream>
using namespace std;
class A {
    int a,b;
public:
    void accept(){
    cout<<"enter 2 numbers :-"<<endl;
    cin>>a>>b;
}
friend void swapnumbers (A &t);

 void display(){
  cout<<"enter a :-"<<endl;
  cout<<"enter b :-"<<b<<endl;
}
friend void swapnumbers (A &t);
};
void swapnumbers (A &t){
    int temp = t.a;
    t.a = t.b;
    t.b = temp;
}
int main(){
    A k;
    k.accept();
    swapnumbers (k);
    k.display();
}
```

q.8] Define 2 classes Box and Cube, each having a private voln. use a friend func:

```
#include <iostream>
using namespace std;
class box;
class cube{
    int l,b,h,v1;
public:
    void accept1(){
    cout<<"enter dimension of box"<<endl;
     cin>>l>>b>>b;
}
friend void greatevolume (box p, cube q);
};
void gredtervolume (box p, cube q){
 p.v1 = p.l*p.h*p.b;
 q.v2 = q.side*q.side*q.side;

if (p.v1 > q.v2){
   cout<<"box having greater voln is :"<<p.v1
                                    <<endl;
}
else{
cout<<" shape having greater voln is :"
                                    <<q.v2<<endl;
}}
int main(){
  box k;
  cube f;
```

```
K.accept1();
f.accept();
greater volume(k,f);
}
```

q. 9] Addition of 2 numbers using class cmplx

```
#include <iostream>
using namespace std;
 int re, im;
public
     void accept(){
     cout << "enter real and imaginary" <<endl;
      cin>>re>>im;
 }
 friend cmplx addcmplx (cmplx c1, cmplx c1,
                        cmplx result);
 void display(){

 cout<<re<<"+"<<im<<"i"<<endl;
 }
 friend cmplx addcmplx (cmplx c1, cmplx c2,
                        cmplx result);
};

Cmplx addcmplx (cmplx c1, cmplx c2, cmplx result)
{ result.re = c1.re + c2.re;
result.im = c1.im + c2.im;
 return result;
};
```

```
int main(){

 cmplx K, f, r;
  K.accept();
  f.accept();
  r = addcmplx(K, f, r);
  r.display();
}
```

q.10] Create a class student with private data members ; name and 3 subjects. Calculate Average.

```
#include<iostream>
using namespace std;
class student{
  char name[50];
  int m1, m2, m3
public:
  void accept {

cout << "enter marks" << endl;
(in)>>m1>>m2>>m3;
}
friend void Calculateavg (student p);
};
void Calculateavg (student p) {
 int avg;
 avg = (p.m1 + p.m2 + p.m3)/3;
Cout << "avg is :- " << avg << endl;
}
```

```
int main () {
   student K;
   K.accept ();
   Calculateavg (K);
}
```

q.10] Create 3 classes: Alpha, beta and Gamma, with each private member.

```
#include<iostream>
using namespace std;
class beta;
class gamma;
class Alpha {
   int a;
public
      void accept1 () {
      cout << "enter a :-";
      cin >> a;
}
friend void sum (alpha p, beta q, gumma x);
};
class beta {
   int b;
public:
   void accept2 (){
   cout << "enter b :-";
   cin >> b;
}
friend void sum (alpha p, beta q, gamma x);
};
```

```
class gamma {
    int c;
public:
    void accept3() {
    cout << "enter c:-";
    cin >> c;
3.
friend void sum(alpha p, beta q, gamma z);
};
void sum(alpha p, beta q, gamma z) {
  int sum;
  cout << "sum is:-" << sum << endl;
}
int main() {
    alpha k;
    gamma z;
    beta q;
  k.accept1();
  q.accept2();
  z.accept3();
  sum(k,q,z);
}.
```

q. [2]

```
# includ <iostream>
using namespace std;
class audit;
class bankaccount {
  int balance;
public:
  void accpt() {
  cout << "enter balance;" << endl;
```

```
    cin>>balance;
}
friend void accessbalance(bankaccount k);
};
class audit{
public:
friend void accessbalance(bankaccount k);
};
 void accessbalance(bankaccount k){
  int pvtbalance;
  pvtbalance = k.balance;
cout<<"balance for auditing is:-"<<
                                  pvtbalance;
}
int main(){
   bankba
   bankaccount x;
   x.accept();
   accessbalance(x);
}.
```

Qn

26/8

# Experiment No: 05

Q.1]

```
#include <iostream>
using namespace std;
class sum
{
  private:
 int num, summy = 0;
  public:
  sum ()
{
  num = 5;
for(int i=0;i<num;i++)
{
  summy = summy + num;
  }
cout<<"sum of n number is "<<summy<<endl;
}
sum(int n) {
   num=n;
for(int i=0;i<num;i++){
  summy = summy + num;
  }
cout<<"sum of n number is "<<summy<<endl.
}
sum(sum&t){
   num = t.num;
for(int i=0; i<num ;i++){
      summy = summy+n ym;
}
```

Q.2]

→

```
cout <<"sum of n number is"<<summy<<endl;
    }
};
int main()
 {
  sum k;
  sum z(34);
  sum k(z);
}
```

Q.2] Code for class 'student' having data members as name & percentage.

→
```
#include <iostream>
using namespace std;
   class student
{
  float per;
  string name;
   public:
  student()
{
cout <<" Enter Name:";
 cin >>name;
cout<<"Enter the percentage:";
 cin >>per;
}
 void display ()
{
 cout<<" Name:"<<name<<endl;
 cout<<"Percentage:"<<per<<endl;
```

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

Output:

```
Enter Name: Prathamesh
Enter the Percentage = 80
  Name = Prathamesh
Percentage = 80
```

e.3) Define class "college" data members as roll no, name, course. Accept & display data for 2 students.

→

```
#include<iostream>
using namespace std;
    class college
    {
     int roll;
     string name, course;
     public:
    college (int r, string n)
    {
       roll r;
      name n;
    course = "computer science";
}
```

```
void display ()
 {
  cout<<" The name of student = "<<name<<endl;
  cout<<" The roll number of student = "<<roll<<endl;
  cout<<" The course of student = "<<course<<endl;
      }
  };
int main ()
{
  college S1 (11, "Aayush "), S2(10, " Aditya ");
    S1. display ();
    S2. display ();
}
```

Output:

The name of student = Aayush
The roll number of student = 11
The course of student = Computer science
The name of student = Aditya
The roll number of student = 10
The course of student = Computer science

Q.4] Constructor over loading

→
```
#including <iostream>
using namespace std;
    class Rectangle;
{
int l, w;
 public:
rectangle (){
 l = 1;
 w = 2;
}

rectangle (int a)
{
 l = a;
 w - b;
}
rectangle (int a, int b)
 {
  l = a;
  w = b
 }

Void calculate ()
{
 int a;
 a = l x b;
 cout << " Area = " << a;
 }
};
```

```
int main()
{
rectangle r1;
rectangle r2(5);
rectangle r3(4,5);
r1.calculate();
r2.calculate();
r3.calculate();
}
```

Q2

20 19 25

# Experiment No: 06

Q.1] Single inheritance

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

class person {
    protected;
    string name;
    int age;
};
class derived : protected person {
    private:
    int roll;
    public:
    void accept() {
    cout<<"enter name, roll no. & age = ";
    cin>>name>>roll>>age;
    }
    void display() {
    cout<<endl<<" Name = "<<name<<endl<<
    "Roll no. = "<<roll<<endl<<" Age = "<<
    age<<endl;}
};
int main() {
    derived d1;
    d1.accept();
    d1.display();
    return 0;
}
```

Q. 2

Q. 2] Multiple Inheritance

→
```
#include <iostream>
using namespace std;
class academic {
    public:
  int marks;
 };
class sports {
    protected:
  int score;
};
class result: protected academic, protected
  sports {
 int total;
   public:
void accept () {
cout<<"enter clg marks &sports score =";
cin >>marks >>score;
 }
void calq () { total= marks+score;
}
void disp () { calq ();
cout<<" The total marks =" <<total;
}
int main () {
 result.rl;
 rl.accept ();
 rl.display ();
 return 0;
}
```

Q. 3] Multi-level inheritance

→
```
#include <iostream>
using namespace std;
class vehicle {
    public:
  string brand, model;
  };
class car:public vehicle {
    public:
  string attribute;
};
class ecar:protected car {
    private:
  string battery="capacity";
  public:
  void accept() {
  cout<<"enter brand, car type, model, battery-
      capacity="<< model<<
  cin<<brand<<attribute<<battery_capacity;
  }
  void display() {
  cout<<endl<<"car brand="<<brand<<endl<<
  "car type="<<attribute<<endl<<"car model="
  <<model<<endl<<"car_battery="<<
    battery_capacity<<endl;
  }
};
int main() {
  ecar e1;
  e1.accept();
  e1.display();
  return 0;}
```

e.g] Hierarchical inheritance

→
```
#include <iostream>
using namespace std;
class employee {
    public:
    string id_name;
};
class manage : protected employee {
    private:
    string dept;
    public:
    void acc() {
    cout<<"enter id-name, dept = ";
    cin >>id >>name >>dept;
    }
    void disp() {
    cout<<"emp_id = "<<id<<"\n name = "
    <<name<<"\n emp dept = "<<dept<<endl;
    }
};
class developer : protected employee {
    private:
    string programming_language = "Rust";
    public:
    void disp() {
    cout<<"programming language = "<<
        programming language<<endl;
    }
};
```

```
int main () {
   manager m1;
    developer d1;
   m1.acc();
   m1.disp();
   d1.disp();
   return 0;
  }
```

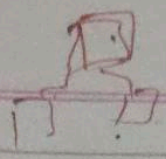
Q.5] Hybrid inheritance

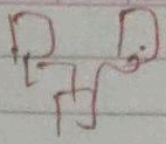
→
```
#include <iostream>
using namespace std;
 class person {
    protected;
  string name;
   int age;
 };
class sports {
  public:
 int score;
 void acc() {
cout<<"enter sport score=";
 cin>>score;
 }
};
class academic {
    public:
 int marks;
 void acc() {
cout<<"enter marks of student = ";
cin>>marks;
```

```
    }
};
class student : protected person {
    public:
    void accept(){
    cout<<"enter age &name of std =";
    cin>>age>>name;
    }
};
class result : protected sports protected
    student, protected acadamic {
        public:
        int total;
    void disp(){
    student::accept();
    acadamic::accept();
    sports::acc();
    }
    void cal(){
    total = marks + score;
    cout<<endl<<"total marks of student="<<
        total<<endl;
    }
};
int main(){
    result r1;
    r1.disp();
    r1.cal();
    return 0;
}
```

Q.6] Virtual base class

→
```
#include <iostream>
using namespace std;
class college {
  protected:
  int id;
  string name;
};
class test : virtual protected college {
  public:
  int per;
  void acc() {
  cout << endl << "enter student id & name = ";
  cin >> id >> name;
  }
  void acc() {
  cout << endl << "enter student marks = ";
  cin >> per;
  }
};
class sports : virtual protected college {
  public:
  char grade;
  void acc() {
  cout << endl << "enter grade = ";
  cin >> grade;
  }
};
class result : protected test, protected sports {
  public:
  void disp() {
```





```
test :: acc();
test :: acc();
sports :: acc();
  }
};
int main(){
  result r;
  r.disp();
  return 0;
}
```

26/9/25

# Experiment No: 07

Q.1]

```
#include <iostream>
using namespace std;
 class area1{
    public:
    int l, b;
  void area (int a, int b){
     int c=a*b;
  cout<<" laboratory Area: "<<c<<"sq";}
     void area (int s){
        int f=5*s;
     cout<<" class Area: "<<f<<"sq";
  }};
  int main (){
   area1 m;
   m.area (20, 30);
   cout<<endl;
    m.area(20);
  }
```

Q.2]

```
#include<iostream>
using namespace std;
    class sum1{
    public :
      int i;
    void sum (float a[5]){
          float s=0;
    for(i=0;i<5;i++){
       s+= a[i];
  }
  cout<<"sum of 5 float numbers:"<<s
   <<endl;
  }
  void sum (int b[10]){
       int s=0;
   for(i=0;i<10;i++){
       s+= b[i];
  }
  cout<<"sum of 10 integer numbers:"<<s
   <<endl;
    }
};
int main(){
  sum s1;
  float c[5];
   int d[10];
cout<<"Enter 5 integer float nos.:\n";
  for(int i=0;i<5;i++){
       cin>>c[i];}
cout<<"Enter 5 integer nos.:\n";
 for(int i=0;i<10;i++){
```

```
    cin >> d[i]; }
s1.sum(c);
s1.sum(d);
}
```

Q.3]

```
#include <iostream>
using namespace std;
    class num {
      int a;
    public:
    void accept(){
cout<<"Enter value of a: ";
    cin>>a;
  }
  void disp(){
cout<<"Value of a:"<<a;
}
  void operator -(){
      a= -a;
    }
};
int main(){
    hum n1;
  n1.accept();
  -n1;
  n1.disp();
}
```

Q.4]

```
#include <iostream>
using namespace std;
    class num {
    int a,b,c;
     public:
    void accept(){
    cout<<"Enter value of a:";
      cin >>a>>b;
   }
    void disp(){
     cout<<"Value of a:"<<a;
   }
    void operator. ++(){
        a=++a;
        }
   };
int main(){
     num n1;
    n1.accept();
     ++n1;
    n1.disp();
}
```

12/11

# Experiment No: 08

Q.1]

```
#include <iostream>
using namespace std;
class mstring {
    string str;
    public:
    mstring (string s){
      str = s;
    }
    mstring (){
      str = "";
    }
    void operator + (mstring obj) {
        str = str + obj.str;
    }
    void disp (){
      cout << str;
    }
};
int main () {
  mstring s1 ("xyz"), s2 ("Pqr"), s3;
      s1+s2;
  cout << "Concatenated string: ";
      s3 = s1;
      s3.disp ();
}
```

Q2]

```
#include <iostream>
using namespace std;
class ilogin {
    protected:
    string name;
    string pass;
  public:
  virtual void accept (){
    cout <<"Enter Name and password :";
    cin >>name >>pass;
  }
   virtual void disp (){
    cout<<" Name " <<name<<" Password "<<pass;
     }
  };
class elogin : public :ogin {
        string email;
        string pass;
       public:
          void accept (){
          Count >> email and Password :";
        cin >> email >> pass;
    }
    void disp (){
    cout << " Email "<<email<<"password "<<pass;
    }
}
class mlogin : public ilogin{
  string mid;
  string pass;
```

```
public:
   void accept(){
    cout<<"Enter Mid and Password";
   cin disp(){
   cout<<"Mid:id "<<mid<<"Password"<<pass;
   };
int main(){
   ilogin *ptr
   ilogin i
   elogin e;
   mlogin m;
   iptr=&i;
   iptr->accept();
   iptr->disp();
   cout->>end l;
   iptr->accept();
   iptr->disp();
   cout<<end l;

   iptr=&m;
   iptr->accept();
   iptr->disp();
   cout<<end l;
   cout
   return 0;
}
```

12/11

Experiment no:09

q.1]

```
#include <iostream>
#include <fstream>
using namespace std;
int main (){
ifstream fin;
ofstream fout;

Fout.open ("destination.txt");
Fin.open ("source.txt");
if (!fin){
Cout << "error opening source file:\n";
retur 1;
3

char ch;
while (fin opened successfully \n";
fin.open ("source.txt");
int word cout = 0
string word;
while (find >> word){
word cout";
3
3

cout << "word count;" << word cout:" << word
                                count << "n";
fin.close();
```

```
fin.open("source.txt");
string target;
int count = 0;
while (fin >> word) {
if (word == target) {
cout << " ";
}
}
cout << "Enter target " << endl;
cin >> target;
cout << " word accurance " << count << endl;
fin.close();

fin.open("source.txt");
int digit count;
int space count;

while (fin.get(ch)) {
  if (isdigit(ch)){
  digit count ++;
}
if (isspace(ch)) {
  space count ++;
}
}
cout << "Digit : " << digit count << "\n";
cout << "space : " << space out << "\n";
return 0;
}
int main () {
login * login ;
} mail login e;
```

```
Membership login m;
login = de;
login → accept ();
login → display ();

login = &m;
login → accept ();
login → display ();
return 0;
}
```

Q

12/11

Experiment no : 10

Q. 1

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

template <class T>
class A {
Public:
T. m1, m2;

Void accept (){
cout << " Enter the first number :";
cin >> m1;
cout << " Enter the second number: ";
cin >> m2;
}
void calc (){
int choice;
cout << "\n-- simple Calculator Menu --\n";
cout << "1. addition \n";
cout << "2. multiplication \n";
cout << "3. division \n";
cout << "4. substraction \n";
cout << "5. square root \n";
cout << "6. percentage (m1 is what % m2) \n";
cout << "7. Power (square of both numbers) \n";
cout << "8. Trignometric (in values) \n";
cout << " Enter your choice:";
cin >> choice;
switch (choice) {
```

```
case:1
cout << "sum = " << m1+m2 << endl;
break;
Case 2 :
cout << "multiplication = " << m1 * m2 << endl;
break;
cout << "Division = " << m1 / m2) << endl;
bredk;
Case 4:
cout << "subtraction = " << m1 - m2 << endl;
break
Case 5 :
cout << "square root of " << m1 << = " << sqrt(m1) << end
Case 6:
Cout << m1 << "P@ " << ((m1 * 100.0) / m2) << " % of " << m2 << end
break
Case 7 :
cout << "square of " << m1 << " = " << pow(m1.2) << endl;
Cout << "square of " << " = " << pow(m2, 2) << endl;
Break;
Case 8 :
~~cout << "square of~~
cout << "sin " << m1 << ") = " << sin(m1) << endl;
cout << sin (" << m2 << ") = " << sin(m2) << end;
break:

defaut:
cout << "invalid choice;" << endl;
}
}
int main(){
A < double > obj;
```

```
obj.calc();
return 0;
}
```

2. queue

```
#include<iostream>
#include<queue>
using namespace std;
template<class T>
class queue{
private:
queue<T>;
public;
void Enque(T elements){
q.push(element);
cout<<"element"<<"enqued Successfully\n"}
void dequeue(){
if(q.empty()){
Cout<<"queue is empty cannot deque\n";
}.
else{
cout<<q.Front()<<"dequeed succesfully\n"q.pop()
}
void display(){
cout<<"queue elements:\n";
queue<T> temp=q;
while(!temp.Empty()){
cout<<temp.Empty()){
cout<<temp.Front()<<" ";
temp.pop();
}
cout<<endl;
}
};
```

```
int main () {
queue <int> mq ;
mq. enqueue (30);
mq. enqueue (20);
mq. Enqueue (10);

mq. display ();
mq. dequeue ();
mq. display ();

retur 0;
}
```

Q.

12/4

Experiment no: 11

q.1]

```
#include <iostream>
#include <vector>
#include <cctype>
using namespace std;
int main(){
vector <int> vec (5);
int i;
cout << "Enter vector 5 elements:";
for (i=0; i<5; i++){
cin >> vec[i];
}
Cout << endl;
cout << "vector elements are:" << endl;
for (i=0; i<5; i++) {
cout << vec[i] << endl;
}
cout << "modified elements are;";
for (i=0; i<5; i++){
cout << endl
for (i=0; i<5; i++){
cout << vec[i] << " ";
}

cout << endl;
int scalar;
cout << "enter a scalar value to multiply:";
cin >> scalar;
cout << "After multiplying by scalar:",
```

```
for (i=0 ; i<5 ; i++)
vec [i] = vec [i] * scalar;

for (i=0; i<5 ; i++)
cout << vec [i] << " ";
cout << endl;
```

Experiment No: 11

Q.

→
```
#include <iostream>
#include <vector>
using namespace std;
 int main () {
  int s;
cout<<"Enter the size of vector ";
 cin>>s;
 vector <int >vec (s);
 cout <<"Enter the siz vector elements:" <<endl;
  for (int i=0;i<s ;i++) {
   cout << " Element " <<i+1<<":";
   cin>> vec [i];
  }
cout<<" \n Vector elements are (with iteration):"
   <<endl;
  cout<<" (";
  for (int i=0; i<s; i++) {
   cout<<vec[i];
   if (i!=s-1)
   cout<<",";
}

 cout " )" <<endl;
  int scalar;
cout<<"\n Enter a scalar value to multiply:";
 cin >> scalar;
 for (int i=0; i<s;i++) {
  cout<<" iteration " <<i+1 <<":"
    <<vec [i] <<" * " <<scalar;
```

```
    vec[i] *= scalar;
    cout<<" = "<<vec[i]<<endl;
}
cout<<"\n After multiplying by scalar:";
cout<<"(";
for(int i=0; i<s; i++){
    cout<<vec[i];
    if(i!=s-1)
        cout<<",";
}
cout<<")"<<endl;
return 0;
}
```

12/11/

Experiment No: 12

```
}
#include <iostream>
#include <string>
#include <stack>
using namespace std;
template <class T>
class stack {
stack <T> my_stack;
Public:
void accept (){
int size,
cout<<"Enter the size of the stack"<<endl
cin>>size;
T data;
for (int i=0;i<size;i++){
cout <<"Enter the data for the stack"endl;
cin >> data;
my-stack. push (data);
}
}
void pop (){
T x;
cout<<"Enter the value which you want
to delete \n"<<endl;
cin >> x;
if (my_stack.empty ()){
cout<<" the stack is empty"<<endl;
}
if (My stack.empty ()) {
```

```
cout << "the stack is empty" << endl;
}
if (!my_stack.empty ()) {
   my_stack.pop (x);
   }
}
void display (){
 if (my_stack.empty ()){
   cout << "The stack is empty" << endl;
}
while (! My_stack.empty ()) {
    cout << my_stack.top ();
my_stack.pop ();
    }
  }
}
int main (void) {
  stack <int> k;
  k.accept ();
  k.display;
}
```

2 Queue

```
#include <iostream>
# include < queue>
using namespace std;
template <class T>
 class queue{
Private:
Queue <T>;
Public:
void enqueue (T elements {
 e. push (Element);
cout<<" element<<" enqueued succesfully\n"}
 void dequeue (){
 if (q.empty ()){
cout <<q.front ()<<" dequeued succes Fully \n"
        q.pop ();
   }
}
void display (){
cout<<" Queue elements: \n";
 Queue <T> temp = q;
 while (! temp. empty ()) {
cout << temp.front ()<<" ";
     temp.pop ();
   }
  cout <<endl;
   }
};
```

12/11

```
int main () {
  Queue <int> mq;
mq. enqueue (30);
mq. enqueue (20);
mq. enqueue (10);

mq. display ();

mq. dqueue ();
 mq. display ();

return 0;
}
```