C++ Programs

① Addition of 2 numbers

```
#include <iostream>
int main(){
 using namespace std;
 int a, b, sum;
 cout << "enter 2 numbers";
 cin >>a >>b;
 sum = a+b;
 cout << "The Sum of 2 numbers =" <<sum;
 return 0;
}
```

② Find weather the number is even or odd

```
#include <iostream>
int main(){
 using namespace std;
 int n;
 cout << "enter n";
 cin >>n;
 if (n % 2 == 0){
      cout << "the number is even";
 }
 else{
      cout << "the number is odd";
 }
 return 0;
}
```

⑥ Print nos 1 to 10 in loop while and for.

ⓐ
```
#include <iostream>
int main(){
    using namespace std;
for (int i=0; i<11 ; i++){
        cout << i+1<< endl;
}
return 0;
}
```

ⓑ
```
#include <iostream>
int main (){
    using namespace std;
int i=0;
  while(i<11){
      cout<<i+1<<endl;
      i++;
}
return 0;
}
```

④ Using Different Arithematic Operations Using Switch

```
#include <stdio.h>
using namespace std;
int main(){
    int a, b;
    char op;

cout<< "Enter first number";
cin >> a;

cout<< "Enter Second number";
cin>>b;

cout << " Enter an Operator (+, -, *, /):";
cin>>op;

switch (op){
    case '+';
        cout << "Result =" <<a+b <<endl;
    break;
    case '-';
        cout << "Result =" <<a-b<< endl; break;
    case '*';
        cout << "Result =" <<a*b<<endl;
        break;
    case '/';
        cout << "Result =" <<a/b<<endl;
        break;
    default: cout << "Invalid expression";
        break;
}}
```

⑤ Pattern Printing

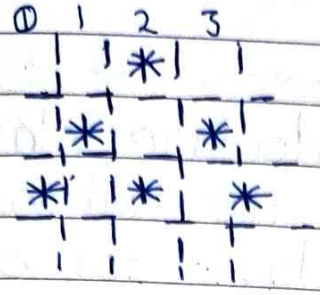
```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

ⓐ

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

ⓑ

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




```
© #include <iostream>
using namespace std;
int main (){
    int rows = 5;

for(int i=1; i<=rows; i++){
    for(int s=1; s<=rows-i; s++){
        cout<<" ";
}
  for (int j=1; j<=2*i-1; j++){
      cout<<"*";
}
  cout<<endl;
}
}
```

9/7/25

Experiement - 1

(1) Write a Program to declare class Stude
have data members as Roll no and nam
Accept and Display data for one objec

(2) Write a Program to declare a class book
having data members as book name,
price, pages. Accept this data for 2
Objects and display name of book
having greater.

(3) Write a Program to declare a class
time. Accept the time is hh:mm:ss
and convert it into total seconds
and display them.

Date
Page

(1)
```
#include <iostream>
#include <string>
using namespace std;

class Student {
    int rollno;
    String name;
public:
    void input(){
        cout << "Enter your name: ";
        cin >> name;
        cout << "Enter your roll no: ";
        cin >> rollno;
    }
    void display(){
        cout << "The name of the student is: " <<
                                    name << endl;
        cout << "The roll no of student is: " << rollno <<
                                    endl;
    }
};
int main(){
    Student dets;
    dets.input();
    dets.display();
    return 0;
}
```

(2).
```
#include <iostream>
using namespace std;
class book{
    public:
    char name[50];
    int pages;
    int price;

    void accept(){
        cout<<"enter the name of the book";
        cin>>name;
        cout<<"enter the price of the book";
        cin>>price;
        cout<<"enter the pages of the book";
        cin>>pages;
}};

int main(){
    book b1, b2;
    b1.accept();
    b2.accept();
    if(b1.price>b2.price){
        cout<<"Book withe greater price is"<<
                                    b1.name;
    }
    else{
        cout<<"Book with greater price is"<<
                                    b2.name;
    }
}
```

(3)
```
#include <iostream>
using namespace std;
Class time{
    public:
    int hh;
    int mm;
    int ss;

void accept(){
    cout<<"enter hours";
    cin>>hh;
    cout<<"enter minutes";
    cin>>mm;
    cout<<"enter Seconds";
    cin>>ss;
}};
    int main(){
        time t1;
        t1.accept();
        int conversion;
        Conversion =3600*t1.hh+60*t1.mm+t1.ss;

cout<<"The conversion of time is"<<conversion;
}
```

23/7/25

Experiement-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 popu

```
#include <iostream>
using namespace std;
class city{
public:
    char name[50];
    int population;
    void accept(){
      cout<<"enter name of city :"<<endl;
      cin>>name;
      cout<<"enter population of city :"<<endl;
      cin>>population;
    }

    void display(){
      cout<<"name of the city :"<<name<<endl;
      cout<<" population of city :"<<population<<endl;
    }
};
int main(){
   city c[5];
   int max=0;
for(int i=0; i<5; i++){
        c[i].accept();
}
```

```
for(int i=0; i<5; i++){
        c[i].display();
}
for (int i=0; i<5; i++){
   if (c[i].population >c[max].population){
                max=i;
      }
}
cout<<"the city having maximum population
     is :"<<c[max].name<<endl;
```

(2). WAP to declare class 'staff' having data members as name and post. Accept this data for 5 staff and display names of staff whor are "HOD"

```
#include <iostream>
#include <cstring>
using namespace std;
class staff{
public:
    char name[50];
    char designation[50];
    void accept(){
        cout<<"enter the name:";
        cin>>name;
        cout<<"enter the designation:";
        cin>>designation;
}
    void display(){
        cout<<"the name is :"<<name;
        cout<<"the designation is :"<<designation;
}};
```

```
int main(){
   Staff s[5];
   Char hod[] = "hod";
 for (int i=0; i<5; i++){
        s[i].accept();
 }
 for (int i=0; i<5; i++){
        s[i].display();
 }
 for (int i=0; i<5; i++){
    If (strcmp (s[i].name, hod) ==0){
 cout<<"the deignation to hod found
    whose name is :- "<<s[i].name;
    }
   }
}
```

(9) Create a Class having Account having data members as accountno and balance Accept this data for 10 accounts and give Intrest of 10% where balance ≥ 5000 and display them

```
#include <iostream>
using namespace std;
Class account{
public:
    int accountno;
    int balance;

void accept(){
cout <<"enter account no";
cin>> accountno;
cout <<" enter account balance";
cin>> balance;
}
void display(){
cout<<" The account no is :"<<acoountno;
cout <<" The account balance :"<<balance;
}
};
int main(){
account a[10];
int t, max = 5000, intrest;
cout<<" enter time (in years)";
cin>>t;
for (int i=0; i<10; i++){
        a[i].accept();
}
```

```
for(int i=0; i<10; i++){
        a[i].display();
}
 for(int i=0; i<10; i++){
        If (a[i].balance > max){
Intrest = a[i].balance *10*t/100;
        a[i].balance = a[i].balance+intrest;
cout<<"account having balance greater
  than 5000 having 10% intrest rate is:
  <<a[i].accountno <<a[i].balance<<endl;
        }
 }
}
```

23/7/25

Experiement No-3

(a) Write a program to declare class 'book' containing data members as book-tittle, author-name and price. Accept and display this info for one object using a pointer to that Object

```
#include <iostream>
#include <cstring>
using namespace std;
Class book{
  String book-tittle;
  String author-name;
  Int price;
Void accept (){
  cout<<"enter book tittle :-"<<endl;
  cin>> this -> book-tittle;
  cout <<"enter author name :-"<<endl;
  cin>> this -> author-name;
  cout<<"enter price :-"<<endl;
  cin>> this -> price;
}
Void display (){
   this -> accept();
cout<<"the book tittle is :-<< this -> book-tittle<<
                                    endl;
cout<<"the author name is :-<< this -> author-name
                                    <<endl;
cout<<"the price is :-<< this -> price <<endl;
}};
```

```
int main(){
    book b;
    book *p;
    p=&b;
    p->display();
}
```

(9) Using this pointer concept create a class student having data members roll-no and percentage and call this for object;

```
#include <iostream>
using namespace std;
class student{
public:
    int roll-no;
    float perc;
void accept(){
    cout<<"enter rollno"<<endl;
    cin>>this->roll-no;
    cout<<"enter percentage:-"<<endl;
    cin>>this->perc;
}
void display(){
    this->accept();
cout<<"the roll no is:-"<<this->roll-no<<endl;
cout<<"the percentage is:-"<<this->perc<<endl;
}};
int main(){
    Student s;
    s.display();
}
```

(8) Use nested Class

```
#include <iostream>
using name space std;

Class demo{
public:
    class demo2{
    public:
        Int rollno;
        Char name [50];
    Void accept(){
    Cout << "enter roll no :";
    Cin >> rollno;
    Cout << "enter the name :";
    Cin >> name;
    }
    Void display(){
    Cout << "the roll no is :" << rollno << endl;
    Cout << "the name is :-" << name << endl;
    }
  };
};
Int main(){
    demo::demo2 Student;
    Student.accept();
    Student.display();
}
```

30/7/25

Experiement No-4

(9)

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

Class result2;

Class result1{
  int m1;
public:
   void accept1(){
      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;
}
```

(9)

```
int main(){
    result1 K;
    result2 f;
    K.accept1();
    f.accept2();
    Calculate (K, f);
}
```

(9). Swapping 2 nos using without frnd function
Obj as func 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 swap (demo &t){
    int temp = t.p;
    t.p = t.q;
    t.q = temp;
}};
int main(){
    demo K;
```

```
K.accept();
K.swap(k);
K.display();
}
```

(Q) 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<<"ent first number"<<endl;
        cin>>num1;
}
friend void greatestnum(A p, B q);
};
Class B{
Public:
    int num2;
    void accept2(){
        cout<<"enter second number"<<endl;
        cin>>num2;
}
friend void greatestnum (A p, B q);
};
```

```
void greatest num (A p, B q){
   If (p.num1 > q.num2){
        cout<<"greatest number : "<<p.num1<<endl;
}
 else{
    cout <<"greatest number :"<<q.num2<<endl;
 }
}
int main(){
    A k;
    B f;
    k.accept1();
    f.accept2();
    greatestnum (k, f);
}
```

(9) # 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();
}
```

(8) 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;
}
 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(){
        A k;
        B f;
    k.accept1();
    f.accept2();
    swap numbers(k, f);
    k.display1();
    f.display2();
}
```

11/8/25

(9) 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 accept1(){
        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 main(){
    A K;
    B f;
    K.accept1();
    f.accept2();
    cout<<" Sum of 2 nos is :- "<<sum(K,f);
}
```

(9) WAP to swap 2 private integers of same class using a friend func.

```
#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();
}
```

(8) Define 2 Classes Box and Cube, each having a private volⁿ. Use a friend funⁿ.

```
#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>>h;
    }
friend void greatervolume(box p, cube q);
};
void greatervolume(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<<"Shape having greater volⁿ is:"<<p.v1
                                        <<endl;
}
 else{
   cout<<"~~box~~ Shape having greater volⁿ is:"
                                    << q.v2<<endl;
 }}
int main(){
    box k;
    cube f;
```

```
K.accept1();
f.accept2();
greatervolume (K, f);
}
```

(8) 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 c2, 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();
}
```

(9) 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 name of student:-"<<endl;
cin>>name;

cout <<" enter marks "<<endl;
cin>> 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();
  calculate avg(K);
}
```

(9) 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, gamma
};
Class beta{
    int b;
public:
   void accept2(){
    cout <<"enter b:-";
    cin>> b;
}
 friend void sum (alpha p, beta q, gamma
};
class gamma{
    int c;
public:
   void accept3(){
    cout <<"enter c:-";
    cin >> c;
}
```

```
friend void sum(alpha p, beta q, gamma x);
};
void sum (alpha p, beta q, gamma x){
    int sum;
    sum = p.a + q.b + x.c;
    cout << " sum is:- " << sum << endl;
}
int main(){
    alpha K;
    gamma z;
    beta q;
 K.accept1();
 q.accept2();
 z.accept3();
 sum (K, q, z);
}
```

(Q) Return distance between 2 points

```
#include <iostream>
#include <cmath>
using namespace std;
class d2;
class d1{
  float x1;
  float y1;
public:
    void accept1(){
    cout <<"enter x1 and y1:";
    cin>>x1>>y1;
}
friend void calcdist (d1p, d2q);
};
class d2{
    float x2;
    float y2;
public:
  void accept2(){
   cout<<"enter value of x2 and y2:"
   cin>>x2>>y2;
}
friend void calcdist (d1 p, d2 q);
};
void calcdist (d1 p, d2 q){
float xf, yf, distf;
xf = sqrt (pow (q.x2 - p.x1, 2));
yf = sqrt (pow (q.y2 - p.y1, 2));
distf = sqrt (pow (xfinal, 2) + pow (yf, 2));
```

```
cout<< " distance is:- "<<distf;
};
int main(){
    d1 k;
    d2 f;
k.accept1();
f. accept2();
calcdist (k,f);
}
```

(9)
```
#include <iostream>
using namespace std;
class audit;
class bankaccount {
    int balanc;
public:
    void accept(){
     cout<<"enter balance;"<<endl;
     cin>>balanc;
}
friend void accessbalance (bankaccount k);
};
class audit{
public:
   friend void accessbalance (bankacoount k);
};
void accessbalance (bankaccount k){
    int pvtbalancc;
    pvtbalance = k.balance;
cout<<"balance for auditing is:-<<pvtbalance;
}
int main(){
    bankaccount x;
    x.accept();
    accessbalance(x);
}
```

13/8

Experiement No-5

① Using friend-function and constructor

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

Class Result1{
public:
   int marks;

 Result1(){
      marks=0
      cout<<"Default Constructor of Resut1\n";
}
Result1 (int m){
    marks =m;
cout<<" Parameterized constructor of Result1";
}
Result1 (Result1 &r){
    marks= r.marks;
cout <<" Copy constructor of Result1\n";
cin>>marks;
}};
class Result2{
public:
   int marks;

 Result2(){
    marks=0;
cout<<" Default constructor of Result2";
```

```
Result2(int m){
 marks = m;
 cout<<"Parameterized constructor of
 Result2 called\n";
}

Result2(const Result2 &r){
   marks = r.marks;
cout<<"Copy constructor of Result2
  called\n";
}
void read(){
    cout<<"Enter marks of Result2:";
    cin>> marks;
}};
void ComputeAverage(Result r1, Result r2){
  float avg = (r1.marks + r2.marks)/2.0;
cout <<"Avg results = "<<avg <<endl;
}
int main(){
  Result r1;
  Result r2;
r1.read();
r2.read();

 Result1 r3(90);
 Result2 r4(80);

Result1 r5(r3);
Resu t2 r6 (r4);
```

```
Compute Average (r1, r2);
Compute Average (r3, r4);
Compute Average (r5, r6);

}
```

Sum of n numbers using
Default, Parameterized,

(a)
```
#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 <<
}
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 st){
    num = t.num;
for(int i=0; i<num; i++){
    summy = summy + num;
}
cout << "Sum of n number is" << summy << endl;
}};
int main(){
    Sum K;
    Sum z(34);
    Sum x(z); }
```

(b)

```
#include <iostream>
#include <cstring>
using namespace std;
Class Student{
private:
    String name;
    float perc;
public:
  Student(){
cout << " name of Student" <<endl;
Cin>>name;
cout<< " percentage of student" <<endl;
Cin>>perc;
}
 void display(){
cout<< " name of student is" <<name <<endl;
cout<< " percentage of Student is << perc<<endl;
}
Student (Stringn, float p){
 name=n;
 perc=p;

cout<< " name of Student is" <<name<<endl;
cout<< " percentage of Student" <<perc<<endl;
}
```

```
Student (student &s){
    name = s.name;
    perc = s.perc;

cout<<"name of student is :"<<name<<endl;
cout<<"percentage of student is :"<<perc<<
                                    endl;
}};
int main(){
    Student k;
    k.display();
    Student s("Siddharth", 67);
    Student z(s);
}
```

(c)

```
#include <iostream> #include <cstring>
using namespace std;
Class college{
    int roll-no;
    String name;
    String course;
public:
 College (int r, String s, String cu="cse"){
      roll-no = r;
      name = s;
      Course = cu;
 }

Cout<<"name of student"<<name <<endl;
Cout<<"roll no of student"<<roll-no <<endl;
Cout<<"course of student is <<course<<endl;
}};
int main(){
 college K(23, "Siddharth");
 college L(26, "Sachin", "Civil");
}
```

(d) Class, Create a program for constructor overloading.

```
#include <iostream>
#include <string>
using namespace std;
class student {
private:
  string name;
  float perc;
public:
   student(){
    name = "Siddharth";
    perc = 98.62;
}
  void display (){
   cout << name << endl;
   cout << perc << endl;
}
Student (string n, float p){
   name = n;
   perc = p;

cout << "name of Student is" << name << endl;
cout << " percentage of Student " << perc << endl;
}
Student (Student &s){
   name = s.name;
   perc = s.perc;
cout << "name is" << name << endl;
cout << "percentage is" << perc << endl; }}
```

```
int main(){
    Student K;
    K.display();
    Student r("Siddharth", 68.28);
    Student s(r);
}
```

Experiement - 6

① Base Class: Person
Derived Class: student

```
#include <iostream>
#include <string>
class person{
public:
   string name;
   int age;
};
class student: protected person{
protected:
  int rollno;
public:
    void accept(){
    cout <<"enter name"<<endl;
    cin>>name
    cout <<"enter age"<<endl;
    cin>>age;
    cout <<"enter rollno"<<endl;
    cin>>rollno;
}
  void display (){
    cout <<"name is"<<name<<endl;
    cout <<"age is"<<age<<endl;
    cout <<"roll no is <<rollno<<endl;
}};
```

```
int main(void){
    Student s;
    s.accept();
    s.display();
}
```

② Multilevel / Multiple Inheritance

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

Class academic{
public:
    int marks;
};
Class sports{
public:
    int sportsscore;
};
Class Result : public academic, public sports{
public:
    int totalScore;

void calculatetotal(){
    total score = marks+sportsscore;
}
void display(){
    Resu
 cout << "marks: " << marks << endl;
 cout << " sports score: " << sportsscore << endl
 cout << " total score: " << totalScore << endl
}};
```

```
Int main(){

Result student;

student.marks =85;
student.sportsscore=15;

student.calculatetotal();
student.display();
}
```

③ Multilevel Inheritance

```
#include <iostream>
#include <string>
using namespace std;
Class vehicle {
public:
   String brand;
   String model;
};
Class car: public vehicle {
protected:
   String cartype;
};
Class electriccar: protected car {
private:
  int batterycapacity;
public:
   Void accept(){
   cout <<"car brand is"<<endl;
   cin>>brand;
   cout <<"model is" <<endl;
   cin>>model;
   cout<<"car type is"<<endl;
   cin>>cartype;
   cout<<"battery capacity is"<<endl;
   cin>>batterycapacity;
}
```

```
void display(){

cout<<"car brand is"<<brand<<endl;
cout<<"car model is <<model<<endl;
cout<<"car type is <<cartype<<endl;
cout<<"battery capacity is << battery
                                capacity<<endl;
}};

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

④ Hirerchial Inheritance

```
#include <iostream>
#include <string>
using namespace std;
class employee {
 public:
    int emp-id;
    string name;
};
class manager: protected employee {
protected:
 string dept-name;
public:
    void accept1(){
cout << "employee name <<endl;
cin>> name;
cout << " employee id is "<<endl;
cin>>emp-id;
cout<< " department name is"<<endl;
cin>>dept-name;
}
void display2(){
cout << " name is "<<name<<endl;
cout << " employee id"<< emp-id<<endl;
cout << " department name is"<<dept-name<<endl;
}};
```

```
class developer: protected employee{
private:
   string codelanguage;
public:
   void accept2(){
    cout<<"coding language"<<endl;
    cin>> code language;
}
   void display2(){
     cout<<"coding language is".
      <<codelanguage<<endl;
}};
int main(){
   manager m;
   developer d;

m.accept1();
m.display1();
d.accept2();
d.display2();
}
```

⑤ Hybrid Inheritance

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

class Person{
public:
 string name;
  int age;
};
class student : protected person{
protected:
   String student_id;
};
class sports{
 protected:
    int sports_score;
};
class academics{

};
class result: protected sports, protected
student{
 private:
  int m1, m2, m3;
 public:
```

```
void accept(){
 cout<<"enter name"<<endl;
 cin>>name;
 cout<<"age is <<endl;
 cin>>age;
 cout<<"student id is <<endl;
 cin>> student_id;
}
void display(){

cout <<"name of student" <<name<<endl;
cout<<"age of student"<<age <<endl;
cout <<" student id"<< student_id<<endl;
}
void calc(){
 int total_marks;
 total_marks = m1+m2+m3;

cout <<"total marks is" <<total_marks
                                <<endl;

cout <<"sports score is"<<sports_score<<
                                endl;
}};
int main(){
   result K;
    K.accept();
    K.display();
    K.calc();
}
```

(9) Virtual Base Class

```
#include <iostream>
#include <string>
using namespace std;
class college_student{
public:
  string student_id;
  int college_code;
};
class test: virtual public college_student{
protected:
  int perc;
};
class sports: virtual public college_student{
protected:
  int grade;
};
class resut: virtual public test, virtual public
sports{
  public:
void accept(){
  cout<<"enter student_id"<<endl;
  cin>>student_id;
  cout<<"enter college code"<<endl;
  cin>>college_code;
  cout<<"enter the percentage"<<endl;
  cin>>perc;
  cout<<"enter grade"<<endl;
  cin>>grade;
}
```

```
void display(){
cout <<" the student id is" <<student-id
                                <<endl;
cout <<" the college code is" <<college-code
                                <<endl;

cout <<" the percentage is" <<perc <<endl;

cout <<" the grade is" <<grade <<endl;
}};
int main(){
    result r;
    r.accept();
    r.display();
}
```

24/9/25

Experiement-7

(a) Write a program using function overloading to calculate area of laboratory and Classroom

```
#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 = s*s;
      Cout << "Class area:" << f << "sq";
}};
Int main (){
    area1 m;
    m.area(20,30);
    Cout << endl;
    m.area(20);
}
```

(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 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(){
  sum1 s1;
  float c[5];
  Int d[10];
cout << "enter 5 float numbers:\n";
  for (int i=0; i<5; i++){
      cin >> c[i]; }
```

```
cout << "enter 10 integer numbers:\n";
 for (int i=0;  i<10; i++){
        cin>>d[i];
}
 S1.sum(c);
 S1.sum(d);
}
```

(c) WAP to implement Unary-operator when used with the Object so that the numeric data member of class is negeted.

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

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

(d) WAP to implement the Unary++ operator when used with the object so that numeric data member of Class is incremented.

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

1/10/25

Experiement-8

(a) WAP to overload the '+' operator so that 2 strings can be concatenated

```
#Include <iostream>
#Include <string>
using namespace std;
class Mystring {
    String str;
public:
    Mystring (string s = "") {
        str = s;
    }
    Mystring operator+(MyString const &obj){
        return Mystring(str + obj.str);
    }
    void display() {
        cout<<str<<endl;
}};
int main() {
    Mystring s1("xyz");
    Mystring s2("pqr");
    Mystring s3 = s1+s2;
  cout<<" Concatenated String:";
    s3.display();
}
```

(b) WAP to create a base class ILogin having data members name and password.

```
#include <iostream>
#include <string>
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{
    String EmailID;
public:
    Void accept() override{
cout << "In -- Email login Details --" <<endl
cout << "Enter name: ";
cin>>name;
```

```
cout << "Enter password: ";
cin>>password;
cout << "Enter Email Id";
cin>> emailID;
}
void display() Override {

cout << "Email login Details: " << endl;
cout << "Enter name: " << name << endl;
cout << "Password: " << password << endl;
cout << "Email ID: " << emailID << endl;
}};

class MembershipLogin:public ILogin{
    String memberID;
public:
    void accept() override{
cout << "\n --Membership Login Details:" << endl;
cout << "Enter name: ";
cin>>name;
cout << "Enter password:";
cin>>password;
cout << "Enter Membership ID: ";
cin>> memberID
}
void display() override{
cout << "\n Membership Login Details:" << endl;
cout << "Name: " << name << endl;
cout << "Password: " << password << endl;
cout << "MembershipID:" << memberID << endl;
}};
```

15/10

Experiement - 9

```
#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";
 return 1;
}
char ch;
while (fin.get(ch)){
      fout.put(ch);
}
fin.close();
fout.close();
cout<<"file opened successfully\n";
fin.open ("source.txt");
int wordcount=0;
string word;
while (fin >> word){
    wordcount++;
}
cout <<" word count: " << wordcount <<"\n";
fin.close();
```

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

fin.open ("source.txt");
int digitcount;
int spacecount;

while (fin.get (ch)){
    if (isdigit (ch)){
        digitcount ++;
    }
    if (isspace (ch)){
        spacecount ++;
    }}
cout <<"Digit: "<< digitcount <<"\n";
cout <<" Space: "<< space cout <<"\n";

return 0;
}
```

16/10/25

```
int main(){

ILogin *login;

EmailLogin e;
MembershipLogin m;

login = &e;
login->accept();
login->display();

login = &m;
login->accept();
login->display();

return 0;
}
```

Experiement-10

#includ

(a). WAP to find Sum of Array Elements using function (eg. Pass integer, Float and Double array of 10 Elements).

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

template <class T>
  T arrayaddition (T arr[], int n){
return accumulate (arr, arr+n, T(0));
}
int main(void){
int Size;
cout <<" Enter Size :"<<endl;
cin>>size;

int intarr[size];
float floatarr [size];

cout<<"Enter the Integer values"<<endl;
for (int i=0; i<size; i++){
      cin>>intarr[i];
}
cout <<"Enter the float values"<<endl;
for (int i=0; i<size; i++){
     cin>>floatarr[i]; }
```

```
cout << "\nThe integer array sum is = " <<
arrayaddition (int arr, size) << endl;

cout << " The float array sum is = " <<
arrayaddition (floatarr, size) << endl;

}
```

(b) WAP of Square function using Template Specialization

→ Calculate the Square of integer no and a String

→ Write a Specialized function for Square of a String.

```
#include <iostream>
#include <numeric>
#include <cstring>
using namespace std;
template <class T>
  T square (T x){
T squared = x*x;
  return squared;
}
template <> string square <strin>
  (string ss) {

return (ss+ss);

}

int main(void){

cout <<" square is :"<< square <int>(45);

cout <<" square is :"<< square <string>
                        ("sid");
```

(c) Calculator Code

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

template <class T>
class A{

public:
  T m1, m2;

void accept(){
 cout <<" Enter 1st no: ";
 cin >>m1;
 cout <<" Enter 2nd no: ";
 cin >> m2;
}
void calc(){
  int Choice;
cout <<"\n--- Simple Calc Menu ---\n";
cout <<" 1. Addition\n";
cout <<" 2. Multiplication \n";
cout <<" 3. Division\n";
cout <<" 4. Subtraction\n";
cout <<" 5. Square root\n";
cout <<" 6. Percentage (m1 % m2)\n";
cout <<" 7. Power (Square of num)\n";
cout <<" 8. Trignometric (Sine values)\n";
cin>>choice;
```

```
Switch (choice){

Case 1:
    cout << "Sum = " << m1+m2 << endl;
    break;

Case 2:
    cout << "Multiplication = " << m1 * m2 << endl;
    break;

Case 3:
    cout << "Division = " << (m1/m2) << endl;
    break;

Case 4:
    cout << "Subtraction = " << m1-m2 << endl;
    break;

Case 5:

    cout << "Square root of " << m1 << " = " << sqrt(m1) << endl;
    cout << "Square root of " << m2 << " = " << sqrt(m2) << endl;
    break;

Case 6:
    cout << m1 << " is " << ((m1 * 100.0)/m2)
                << " % of " << m2 << endl;
    break;
```

```
case 7:
    cout << "Square of " << m1 << " = " << pow(m1,2)
                <<endl;
    cout << " Square of " << m2 << " = " << pow(m2,2)
                <<endl;
    break;

case 8:
    cout << "sin(" << m1 << ") = " << sin(m1)<<endl
    cout << "sin(" << m2 << ") = " << sin(m2)<<endl
    break;

default:
    cout << " Invalid Choice ! " << endl;

};

int main(void) {

    A <double> obj;
    Obj.accept();
    Obj.calc();
    return 0;

}
```

5/11/25

(11).

① ② ③

Experiement -11

(11). Vector (STL) program

① Create a Vector
② modify the value of given Element
③ To display vector in form (10,20,30).

```
#Include <iostream>
#Include <vector>
#Include <numeric>
using namespace std;
template Vector {
    vector <T> V;

Public:
    Void accept(){
     Int size;
     Cout <<"enter the Size:" <<endl;
     Cin>> size;
      T data;
    for (int i= 0; i<size ; i++){
      Cout <<"Enter the data :" <<endl;
      Cin >> data;
      V.push-back (data);
  }}
```

```
void modifyElement(int index, T value){
 If( index >=0 && index< V.size())
     V[index] = value;
  else
     cout << "Invalid index!\n";
}

 void Scalar (){
   T Scalar;
   cout << "Enter Scalar :" <<endl;
   cin>> Scalar;

for( int i=0; i<V.size(); i++){
      V[i] = V[i]* Scalar;

}}

 void display(){

   for( int i=0 ;i<V.size(); i++){
       cout << V[i]<<endl;
  }}
int main(void){

 Vector <int> O;
 O.accept(); O.display();
 O. Scalar();
 O. display();
```

Experiement -12

(8) WAP in c++ to implement stack and queue using STL

```
#Include <iostream>
#Include <stack>
# Include <cstring>
using namespace std;
template <class T > class Stack {

 Stack <T> my-stack;

public;
      void push(){
       int size;
       cout <<" Enter size of Stack:"<<endl;
       cin>> size;
       T data;
       for (int i=0; i< size; i++) {
       cout <<" Enter data :" <<endl;
        cin>>data;
        my-stack.push(data);
  }}
 void pop(){
    T x;
    cout <<" Enter value of x:" <<endl;
    cin >>x;

    my- stack.pop(x);
 }
```

```
void display(){
  cout << " the Stack Structure is:" <<
  while (!my-stack.Empty()){
      cout << my-stack.top();
          my-stack.pop();

}}

int main (void){
    Stack <int> K;
      K.push();
      K.display();
}
```

classmate

Date
Page

(8) Implement Queue using STL

1. Create a Queue
2. Enqueue, Dequeue operations
3. Display the Queue

```
#Include <iostream>
#Include <queue>
using namespace std;
template <class T>
class Queue{
private:
   queue <T>q;
public:

  void Enqueue (T Element){
        q.push(Element);
 cout<<element << "Enqueued Successfully\n"
}

  void dequeue(){
       If (q.Empty()){
 cout << "queue is Empty cannot dequeue\n";
   }else{
       Cout <<q.front()<<" dequeued successfully
\n";

       q.pop();
}}
```

```
void display(){

  cout << " Queue Elements:\n";
  Queue<T> temp = q;
while (!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();

 return 0;

}
```

Qu
5/11