Experiment - 1

Q.1] Write a C program to declare Class Student having data members as Roll no and name accept and display data for one object.

```
#include <iostream>
Using namespace std;
Class Student
{
  int roll;
  String name;
  public:
   Void accept()
  {
  Cout<<"Enter Value of roll number:";
  Cin >> roll;
  Cout<<"Enter the student name:";
  Cin >> name;
}
   Void disp()
{
  Cout<<"\nRoll number=" << roll;
  Cout<<"\nName =" << name;
}
};
  int main()
{
    Student S1;
    S1. accept();
```

```
S1.disp();
return 0;
}
```

Output:

enter the roll number : 50
enter the student name : Sanket

roll number = 50
name = Sanket

Write a program to declare class 'book' having data member as book name, price and no. of pages. Accept the data for 2 object and display the name of book having greater price.

```
#include<iostream>
Using namespace std;
Class book
{
 String name;
  int price;
  int pages;

 Public:
   Void accept()
  {
 Cout<<" enter the book name:";
 Cin >> name;
 Cout<<" enter the price of book:";
  Cin >> price;
```

```
    cout<<"enter the number of pages:";
    cin >> pages;
}
  void disp()
{
   cout<<"\n Name of book="<< name;
   cout<<"\n Price of book="<< Price;
   cout<<"\n Pages="<< Pages;
}
};
   int main()
   {
      book b1, b2;
       b1. accept();
       b2. accept();
     cout<<"\n Book with greater price:\n";
      if(b1.Price() > b2.Price())
      {
          b1. disp()
      }else if(b2. Price() > b1.Price()){
          b2.disp();
      }else{
         cout <<"Both books have same price:\n";
          b1. disp();
          b2. disp();
      }
       return 0;
      }
```

output:

```
Enter the book name: maths
Enter the price of book: 500
Enter the no. of pages: 170
Enter the book name: OOP
Enter the price of book: 600
Enter the no. of pages: 180
```

```
Book with greater price:
Name of Book: OOP
Price of Book: 600
No. of pages: 180
```

Q.3] Write a C program to declare Class time Accept ettime HH:MM:SS format, Convert it in seconds and display them.

→

```
#include<iostream>
Using namespace std;
Class time
{
  int H;
  int M;
  int S;

 Public:
    Void accept();
 {
   Cout" Enter the time:";
   Cin >> H >>M>>S;
 }
  Void disp()
 {
   int total;
  total=(H*3600)+(M*60)+S);
  Cout<<" Time in seconds="<< total;
 }
 };
   int main()
  {
     time t1;
     t1.accept();
     t1.disp();
  }return 0;
  }
```

Output:

```
Enter the time(H M S): 1 30 15
Time in seconds: 5415
```

31/7/25

Experiment no: 2

Write a C++ program to demostrate the
Write WAP to declare a class 'city' ha
data member as name and popula
Accept this data for 5 cities and
name of city having highest pop

```
# include <iostream>
Using namespace std;
Class city
{
 int population;
 String name;

 Public:
    Void accept ()
 {
  Cout <<" enter city name :";
  Cin >> name;
  Cout<<" enter the population :";
  Cin >> population;
 }
   Void disp()
 {
   Cout<<"\n name="<<name;
   Cout <<"\n Population = "<< Population;
 }
};
  int main ()
 {
    int i;
 City C[5];
 for(i=0;i<5;i++){
```

```
C[i].accept();
}
for(i=1; i<5; i++){
 if(C[i].Population() > C[max].Population()){
     max=i;
 }
}
 Cout<<" city Details\n";
  for(i=0; i<5; i++){
   C[i].disp();
 }
 Cout<<"\n city with highest population\n";
  C[max].disp();

  return 0;
}
```

output:

```
enter Cityname: khamgaon
 enter population: 5000
enter city name: Akola
enter Population: 6000
enter City name: Shegaon
 enter Population: 4000
 enter city name: nagpur
 enter population: 7000
 enter city name: ambavati
 enter Population: 3000
```

```
City details:
name = khamgaon
Population = 5000
name = Akola
Population = 6000
name = Shegaon
Population = 4000
name = nagpur
Population = 7000
name = ambavati
 Population = 3000
City with highest
Population:
name: nagpur
```

WAP to declare class 'Account' having data member as Account no. and balance. Accept this data for 10 Account and give interest 10% where balance is equal or greater than 5000 and display them.

```
#include<iostream>
Using namespace Std;
Class Account
{
  int acc-no;
  float balance;
  String name;

Public:
  Void accept()
{
  Cout<<"Acc No:";
  Cin >> acc-no;
  Cout<<"Balance:";
   Cin >> Balance;
   Cout<<"Name:";
   Cin >> Name;
}
  Void update(){
   if(balance >5000) balance *=0.1;
}
  Void disp()
  {
   Cout<<"Acc No:" << acc-no;
```

```
    cout<<"Name:"<<name;
    cout<<" Balance"<<Balance;
  }
};
  int main()
{
  Account a[10];
  for(int i=0; i<10; i++){
     a[i].accept();
  }
   for(int i=0; i<10; i++){
      a[i].update();
   }
    cout<<"\nUpdated Accounts:\n";
    for(int i=0; i<10; i++){
        a[i].disp();
    }
     return 0;
}
```

Output:

```
Acc No: 9988
Balance: 5000
Name: Ram
Acc No: 8898
Balance: 6000
Name: Sham
Acc No: 8523
Balance: [redacted]7000
Name: Yash
Acc No: 4654
Balance: 8000
Name: Om
```

DOHS | Page No.
Date / /

Acc No: 4641
Balance: 10000
Name: Sanket
Acc No: 4646
Balance: 9000
Name: Aunj
Acc No: 8464
Balance: 6000
Name: Vijay
Acc No: 4648
Balance: 6000
Name: amit
Acc No: 4555
Balance: 9000
Name: Shiv
Acc No: 8465
Balance: 9000
Name: Ramesh

Updated Accounts.

Acc No: 9988, Name: Ram, Balance: 5000
Acc No: 8898, Name: Sham, Balance: 600
Acc No: 8523, Name: yash, Balance: 700
Acc No: 4654, Name: om, Balance: 800
Acc No: 4641, Name: Sanket, Balance: 1000
Acc No: 4646, Name: anuj, Balance: 900
Acc No: 8464, Name: vijay, Balance: 600
Acc No: 4648, Name: amit, Balance: 5000
Acc No: 4555, Name: Shiv, Balance: 900
Acc No: 8465, Name: Ramg, Balance: 800

Q3) WAP to declare a class 'staff' having data member as name and post. Accept this data for 5 staff and display name of staff who are HOD.

→
```
#include <iostream>
Using namespace Std;
Class Staff
{
 Public:
  String name;
  String Post;

 Void accept()
 {
 Cout<<" Enter name:";
 Cin >> name;
 Cout<<" Enter post:";
  Cin >> Post;
 }
  Void disp()
 {
   if( Post =="HOD" || Post == "hod"){
  Cout << name;
 }
 }
};
  int main(){
  Staff Stafflist[5];
  int i;
  for(int i=0; i<5; i++){
```

```
Cout<<"\n Enter details for staff"<<i+1
        <<":\n";
    Staff list[i].accept();
}
  Cout<<"\n staff who are HOD:\n";
  for(int i=0; i<5; i++){
     Staff List[i].disp();
  }
   return 0;
}
```

Output:

```
Enter details for staff 1:
 Enter name: Ram
 Enter Post: Prof.

Enter details for staff 2:
 Enter name: Sham
 Enter Post: Prof.

Enter details for staff 3:
 Enter name: Sanket
 Enter Post: HOD

Enter details for staff 4:
 Enter name: Disha
 Enter Post: Assi.Prof.

Enter details for staff 5:
 Enter name: Radhika
 Enter Post: HOD
```

DOMS | Page No.
Date / /

Staff Who are HOD:
Sanket
Radhika

31/7/25

Experiment no. 3

Pointer to object, The 'this' Pointer nes
Class

a) Write a program to declare a class book' containning data member as book-title, author_name and price. Accept and display the information for one object using pointer to that object.

→
```
#include<iostream>
Using namespace std;
Class book
{
  String book_title;
  String author_name;
   float Price;

Public:
   Void accept()
{
Cout <<"Enter the book title:"<<endl;
Cin >> book_title;
Cout<<" Enter the author name:"<<en
Cin>> author_name;
Cout <<"Enter the book Price:"<<endl;
Cin >> Price;
}
  Void disp()
  {
   Cout<<" Book title:"<<book_title<<end
   Cout<<" author name:"<<author_name<<en
   Cout<<" Price:" << Price<< endl;
```

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

    return 0;
}
```

output:

Enter the book title:
The Blue Umbrella

Enter the author name:
RUSKIN BOND

Enter the book price: 1000

Book title: The Blue Umbrella
author name: RuskinBond
Book price: 1000

6] WAP to declare a class 'Student' having data member roll_no and percentage. Using 'this' pointer invoke member function to accept and display this data for one of the class.

→
```
#include <iostream>
Using namespace Std;
Class Student
{
  int roll_no;
  float Percentage;

Public:
  Void accept()
{
  Cout<<" Enter the Student roll no: "<<e
  Cin >> roll_no;
  Cout<<" Enter the student Percentage:
                                <<en
  Cin >> Percentage;
}
  void dispc)
{
  Cout<<" Roll no = "<<roll_no<< endl;
  Cout << "Percentage = "<< Percentage
                          << % << endl;
}
};
```

```
int main (){
  Student S1;
   Student *P;
    P= &S1;
    P-> accept();
     P-> disP();

       return 0;
}
```

Output:

```
Enter the Student Roll no:
  50
Enter the Student Percentage:
  80
  Roll no = 50
  Percentage = 80%
```

Write a program to demonstrate the use of nested class.

```
#include <iostream>
Using namespace std;
Class demo{
  Public:
  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<<
     roll-no.=" <<roll-no<<endl;
  }
  };
};
   int main(){
    demo::demo2 S1;
    S1.accept();
    S1.disp();
    return 0;
    }
```

Write a program to demonstrate the use of nested class

```
#include <iostream>
Using namespace std;
Class demo{
  Public:
  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<<
     roll -no. =" <<roll-no <<endl;
  }
  };
};
   int main(){
    demo:: demo2 S1;
     S1.accept();
     S1. disp();
     return 0;
     }
```

output:

Enter the name of Student & roll-no =
Apple 12

Name = Apple
roll-no = 12

Experiment.no.4

WAP to swap two nos from same Class using object as function argument. Write Swap function as member function.

```
#include <iostream>
Using namespace std;
Class Number {
  int [redacted]num;
  Public:
    Void accept(){
  Cout<<" Enter number:";
    Cin >> num;
  }
   Void disp(){
   Cout <<" Number:" <<num<<endl;
  }
   Void Swap (Number & obj)
   {
     int temp = num;
         num = obj. num;
         obj.num = temp;
  }
  };
   int main (){
    number n1, n2;
    Cout<<" enter first number:" <<endl
      n1. accept();
    Cout<<" enter second number:" <<en
      n2. accept();
```

```
n1.swap(n2);
cout<<"\n After swap:" <<endl;
 cout<<" first";
 n1.disp();
 cout<<" second";
 n2.disp();
 return 0;
 }
```

WAP to Swap two nos from same Class Using cocept of friend function.

```
#include<iostream>
Using namespace Std;
Class AA{
 int a, b;
 Public:
  Void input(){
  Cout<<"enter two value: ";
  Cin >> a >>b;
 }
 friend void Swap(AA a1);
};
  void Swap(AA a1),
  {
    int temp;
    temp =a1.a;
     a1. a = a1. b;
      a1.b = temp
     Cout<<"\n value after swapping:
                    <<a1. a <<a1;
  }
 int main(){
  AA a1;
   a1. input();
    Swap (a1);
    }
```

3] wap to swap two nos from different class using concept of friend function.

→
```
#include<iostream>
Using namespace std;
Class A{
 input numA;
 Public:
  Void accept(){
  Cout<<"Enter number A:";
  Cin >> numA;
  }
  Void disp(){
   Cout<<"Number A = "<<numA << endl;
  }
   friend  Void swap numbers(A &, B &);
};
  Class B{
   Int numB;
   Public:
    Void accept(){
    Cout <<"Enter number B:";
    Cin >>numB;
    }
    Void disp(){
    Cout<<"Number A = "<<numB<< endl;
    }
    friend void Swapnumber(A&, B&);
};
```

```
void SwapNumbers(A &a, B &b)
{
  int temp = a.numA;
      a.numA = b.numB;
      b.numB = temp;
}
int main()
{
  A c1;
  B d1;
  c1.accept();
  d1.accept();
  SwapNumbers(c1, d1);
  cout<<"\n After Swapping:"<<e
    c1.disp();
    d1.disp();
    return 0;
}
```

4] WAP to create two classes Result 1 & Result 2 which shows marks of the student. Read the value of marks for both the Class objects and Computer the avg of two result.

→

```
#include<iostream>
Using namespace std;
Class Result 1{
    int a;
    Public:
    Void accept(){
    Cout<<" Enter marks out of 50";
    Cin >>a;}
    friend Void cal(Result R1,
     Result R2);
    };
     Class Result 2 {
      int b;
      Public:
      Void accept(){
      Cout<<" Enter marks out of 50";
       Cin >> b;}
     friend Void cal(Result 1 R1, Result2 R2);
     };
     Void cal(Result1 R1, Result2 R2){
     float avg = (float)(R1.a+R1.b)/2;
     Cout<<"\n average :" <<avg;}
      int main(){
       Result1 x;
       Result2 y;
```

```
x.accept();
y.accept();
cal(x,y);
}
return 0;
}
```

5] WAP to Find the greatest number among 2 number from two digit different classes using friend function.

→
```
#include <iostream>
Using namespace Std;
Class A{
  int a;
  Public:
  Void accept()
   {
   Cout<<" Enter a value ";
    Cin >> a;
   }
    friend Void greater(A a1, B b1);
    };
     Void greater(A a1, B b1)
     {
      if (a1.a > b1.b){
      Cout<<"first value is greater:";
      }
       else{
       Cout<<"second value is greater";
       }
        int main(){
         A x;
         B y;
         x.accept();
         y.accept();
         greater(x,y);
         }
```

~~Assignment~~

Experiment No. 6

1. Single Inheritance

Problem:

Create a base class person with attributes name and age. Derive a class student from person that adds an attribute, pollNumber. Write function to display all details of the student.

→

```
#include <iostream>
Using namespace std;
Class Person {
Protected:
    String name;
    int age;
Public:
    Void accept () {
    Cout << "Enter name : ";
     Cin >> name;
     Cout << "Enter age:";
      Cin >> age;
}
 Void display () {
   Cout << "Name:" << name << endl;
   Cout << "Age:" << age << endl;
}
};
Class Student : Public Person {
Private:
 int pollNumber;
```

```
Public:
    void acceptStudent (){
     accept ();
     Cout << " Enter Roll number: ";
      cin >> rollnumber;
    }
      void displayStudent (){
        display ();
      Cout << "Roll Number: " << rollNumber
                                  endl;
    }
    };
      int main (){
       Student S;
         S.acceptStudent ();
         Cout << "\n student Details: \n
         S.displaystudent ();
          return 0;
      }
```

output:

```
Enter name : Rahul
Enter age : 20
Enter roll number : 101

Student Details:
Name: Rahul
Age: 20
Roll number: 101
```

2. Multiple Inheritance
Problem:
Create two classes Academic Sports.
- Academic Class contais marks of a student.
- Sports class contains sports score
Create a derived Class Result that inherits from both Academic and sports. Write a function to calculate the total score and display details.

→
```
#include<iostream>
Using namespace Std;
Class Academic{
  Public:
  int makks;
  Academic(int m){
    makks = m;
  }
  };
    Class sports{
    Public:
      int sportsScore;
      Sports(int s){
        Sports Score = s;
      }
    };
      Class Result: Public Academic, Public
          sports
      {
```

```
Public:
    Result(int m, int s): Academic(m),
    Sports(s) {}
     Void display(){
    Cout<<"Academic Marks:" <<marks<<
                                endl;
    Cout<<"Sports Score:" << score <<
                                endl;
    Cout<<"Total score:" <<(marks +
                            sportsScore)
                          << endl;
}
};
    int main(){
     int m,s;
     Cout<<"Enter academic marks!";
      Cin >>m;
      Cout<<"Enter Sports Score:";
       Cin >>s;
       Result result(m, s);
        result.display();

          return 0;
    }
```

output:

```
Enter academic marks: 85
 Enter Sports Score: 15
  Academic Marks: 85
   Sports Score: 15
  Total Score: 100
```

3. Multilevel Inheritance

Problem:

Create a class Vehicle with attributes like brand and model.

Derive a class car from Vechicle which adds an attributes type (e.g. sedan, SUV). further derive a class Electric - Car from Car which adds battery capacity. Write function to display all the details.

→

```
#include<iostream>
Using namespace std;
Class Vehicle {
  Public:
    String brand;
    String model;
};
Class Car: Public Vehicle {
    Public:
      int batteryCapacity;
    Void display() {
    cout<<"Brand:" <<brand <<endl;
    cout <<" Model:" << model <<endl;
    cout <<" Type:" << type <<endl;
    cout<<"Battery capacity :" << battery
      capacity <<"kwh" <<endl;
  }
};
  int main(){
   ElectricCar car;
```

```
Car.brand = "Tesla";
Car.model = "Model y";
Car.type = "SUV";
Car.batteryCapacity = 75;
Car.display();
return 0;
}
```

Output:

Brand : Tesla
Model : Model y
Type : SUV
Battery capacity : 75 kWh

4. Hierarchical Inheritance

Problem:

Create a base class Employee with attributes empID and name. Derive two classes Manager and Developer from Employee. Manager has an attributes department and Developer has an attribute programming Language. Write function to display details for both.

→

```
#include <iostream>
Using namespace std;
Class Employee
{
  Public:
   int empID;
   String name;

   Employee(int id, String nm) {
      empID = id;
      name = nm;
   }
   ~~void display()~~
   Void displayEmployee() {
     Cout << "Employee ID:" << empID <<
                                endl;
     Cout << "Name:" << name << endl;
   }
};
```

```
Class Manager : Public Employee {
 Public:
  String department;

  Manager(int id, String nm, String
  Employee (id, nm){
      department = dept;
  }
   Void displayManager (){
      display Employee();
   Cout<<"Department:" << departme
                    <<endl;
 }
};
Class Developer: Public Employee{
 Public:
   String ProgrammingLanguage;

   Developer(int id, String nm, String
   Employee(id, nm){
     ProgrammingLanguage = lang;
   }
   Void displayDeveloper(){
      display Employee ();
   Cout<<" Programming Language :" <
    ProgrammingLanguage <<endl;
   }
};
```

```
int main (){
Manager mgr(101, "Alex", "HR");
Developer dev (102, "Stev", "C++");
Cout << " -- Manager Details --" <<endl;
 mgr. displayManager();
 Cout <<"\n -- Developer Details--" <<
  endl;
  dev. displayDeveloper();

  return 0;
}
```

output:

```
-- Manager Details --
   Employee ID: 101
     Name: Alex
   Department: HR

-- Developer Details --
     Employee ID: 102
       Name: Stev
       Programming Language : C++
```

5. Hybrid Inheritance

Problem:
Combine multilevel and multiple inh
Create a base class Person with
attributes name and age. Deriv
Class Student from Person.
Create two classes Sports and
Academics. Derive class Result fr
Student and sports. Write functio
to calculate and display total
marks and Score along wi
Student details.

```
#include <iostream>
Using namespace std;
Class Student {
Public:
int rollno;
void Set_rollno(int r){
rollno = r;
}
Void display_rollno(){
Cout<<"Roll number:"<<rollno<
}
};
Class Test : Public
```

```
#include <iostream>
Using namespace std;
Class Person {
 Public:
  String name;
  int age;
};
 Class student : Public Person{
  Public:
   int id;
 };
  Class sports {
   Public:
    int sportsScore;
};
 Class Result: Public student, Public sports
   {
   Public:
   Void display () {
     Cout<<"Name:"<<name<<endl;
     Cout<<"Age:"<<age<<endl;
     Cout<<"ID:"<<id<<endl;
      Cout<<"Sports Score:"<<SportScore
                          <<endl;

   }
  };
    int main (){
     Result p;
     p.name = "Sanket";
      p. age= 20;
```

```
p.id = 101;
p.sportsscore = 85;
p.display();
return 0;
}
```

output:

Name: Sanket
Age: 20
ID: 101
SportScore: 85

Q→

26/9/25

Experiment 5

* Write a C++ program to implement types of constructors.

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

→ default constructor

```
#include<iostream>
Using namespace std;
Class Sum{
    int Total;
    Public:
     Sum(){ // Default Constructor
     int n;
     Cout<<"Enter a number:";
      Cin>>n;
      Total=n*(n+1)/2;
    }
    Void display(){
     Cout<<"Sum="<<Total<<endl;
    }
};
 int main(){
    Sum S;
     S.display();
     return 0;
 }
```

Copy Constructor

```
#include <iostream>
Using namespace std;
Class Sum{
  int Total;
 Public:
  Sum(int n){
   Total = n*(n+1)/2;
  }
   Sum(const Sum &s){
    Total = s.Total;
   }
    void display(){
    Cout <<"Sum =" Total << endl;
  }
 };
   int main(){
   int n;
   Cout <<"Enter a number:";
   Cin >>n;
    Sum S1(n);
    Cout <<" object S1 :";
     S1. display();
     Sum S2 (S1);
      Cout<<"object S2 :";
       S2. display();
        return 0;
      }
```

* Parameterized Constructor

```
#include<iostream>
Using namespace std;
Class Sum{
 int Total;
 Public:
   Sum(int n){
   Total=n*(n+1)/2;
 }
 Void display(){
   Cout<<" Sum =" << Total <<endl;
 }
};
  int main(){
   int n;
   Cout<<" enter a number:";
   Cin >>n;
    Sum S(n);
     S.display();
     return 0;
    }
```

output:

enter a number: 5

sum: 15

6] Write a program to declare a class "Student" having data members as no. and percentage. Write a constructor to initialize these data members. Accept and display data for one Stu.

→ Default Constructor

```
#include<iostream>
Using namespace Std;
Class student{
  String name;
  float Percentage;
  Public:
    Student ()
    {
      name = "Ram";
      Percentage = 80;
    }
    Void display (){
  Cout<<"name of Student :" << name<<
  Cout<<"percentage of Student:"<< percent
                    <<"%."<<endl;
  }
};
    int main (){
      Student S1;
      S1. display ();
      Return 0;
    }
```

Copy Constructor

```
#include<iostream>
Using namespace Std;
Class Student{
 String name;
 float Percentage;
 Public:
  Student(string n, float p){
    name = n;
    Percentage = P;
  }
  Student(const student &S){
    name = S.name;
    Percentage = S.percentage;
  }
    Void display (){
  Cout<<"name =" <<name<<endl;
  Cout<<"Percentage=" <<percentage<<"%"
                    <<endl;

  }
};
  int main (){
   student S1 ("Pam", 80);
    Student S2(S1);
    S1.display ();
    S2. display ();
  }
```

Paramaterized Constructor

```
#include <iostream>
Using namespace std;
Class Student{
  String name;
  flodt percentage;
  Public:
  Student(string n, float p){
    name=n;
    Percentage=P;
  }
  Void display(){
  Cout<<"name="<<name<<endl;
  Cout<<"percentage="<< Percentage<<
                          <<endl;
  }
};
    int main(){
    Student S1("Pam", 80);
     S1. display();
     Return 0;
  }
```

Output:
name: Pam
Percentage: 80%

Define a "College" members variables as Roll_no, name, course WAP using constructor with default value as "Computer Engineering" for course. Accept the data for two object of class and display the data.
Default constructor

```
#include <iostream>
Using namespace std;
Class College {
 int roll;
 String name;
 String course;
 Public:
  College(){
   roll=12;
    name="Ram";
    course="Computer science";
  }
   void display(){
   cout<<"ROLL="<<roll<<endl;
   cout<<"NAME="<<name<<endl;
   cout<<"COURSE="<<course<<endl;
   }
};
   int main(){
    college c1;
    college c2;
    c1.display();
    c2.display();
```

Copy constructor

```
#include<iostream>
Using namespace Std;
Class College{
  int roll;
  String name;
  String Course;
  Public:
   College (int r, String n, String C){
    roll = r;
     name = n;
     Course = C;
  }
    College(const College &C){
     roll = C.roll;
     name = C.name;
     Course = C.Course;
  }
    Void display(){
    Cout<<"ROLL="<< roll<<endl;
    Cout<<"NAME ="<< name<<endl;
    Cout<<"COURSE="<< Course<<endl;
  }
};
    int main(){
    College C1(12,"Ram","Computer sci
     College * C2 = C1;
     C1.display();
      C2.display();
  }
```

Parameterized constructor

```
#include<iostream>
Using namespace std;
Class College{
    int roll;
    String name;
    String course;
    Public:
     College (int r, String n, string c){
      roll = r;
      name = n;
      course = c;
  }
    Void display (){
    Cout<<"ROLL=" << roll << endl;
    Cout <<"NAME=" << name<< endl;
    Cout<<"COURSE=" << course <<endl;
  }
};
  int main (){
    College c1(12, "Ram", "computer science"
                );
    College c2(15, "Shyam", "Mechanical
     Engineering");
      c1. display();
      c2. display();

      return 0;
  }
```

output:

ROLL = 12
NAME = Ram
COURSE = Computer Science
ROLL = 15
NAME = Shyam
COURSE = Mechanical Engineering

Write a program to demostrate constructor Overloading

```
#include<iostream>
Using namespace std;
Class Student{
  int roll;
  String name;
  Public:
   Student(){
    roll=0;
   name="Ram";
  }
   Student(int r, string n){
    roll=r;
    name=n;
  }
   Student(const Student &s){
    roll=s.roll;
   name=s.name;
  }
   Void display(){
  Cout<<"Roll="<<roll<<endl;
  Cout<<"NAME="<<name<<endl;
  }
};
   int main(){
    Student S1;
    Student S2(1,"Shyam");
     Student S3(S2);
```

```
S1.display();
S2.display();
S3.display();

return 0;
}
```

output:

```
ROLL = 0
NAME = Ram
ROLL = 1
NAME = Shyam
ROLL = 1
NAME = Shyam
```

26/9/25

Experiment no 7

Write a program to demonstrate the compile Time Polymorphism (function overloading and Operator Overloading - Unary).

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

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

// Function overloading
int area(int side){
 return side * side;
}
int area(int l, int b){
 return l * b;
}
// operator overloading
Class Number{
 int x;
Public:
 Number(int v) { x = v; }
 Void display (){
 Cout << x << endl;
 }
  Void operator-() {
   x = -x;
```

```
int main(){
cout << "Square Area:" << area(5) <<
                                  endl;
cout << "Rectangle Area:" << area(10,6)
                                  << endl;
 Number n(20);
  -n;
  cout << "After Unary-:";
    n.display();

    return 0;
}
```

output:

Square Area: 25
Rectangle Area: 60
After Unary-: -20

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;
float sum(int n){
  float S=0, x;
  Cout<<"Enter"<<n<<"float value:
  \n;
  for (int i=0; i<n; i++){
    cin>>x;
    S+=x;
  }
  return S;
}
int sum(long n){
  int s=0, x;
  Cout<<"Enter"<<n<<"integer values::
  \n";
  for (int i=0; i<n; i++){
    Cin>>x;
    St=x;
  }
  return S;
}
  int main(){
  Cout<<"sum of 5 floats="<<sum(5)
    <<endl;
  Cout<<"sum of 10 int="<<sum(10)
    <<endl;
```

output:

Enter 5 float values:
1.1 2.2 3.3 4.4 5.5
Sum of 5 floats = 16.5

Enter 10 integers values:
1 2 3 4 5 6 7 8 9 10
Sum of 10 ints = 55

Write a program to implement Unary operator when used with the object so that the numeric data member of the class in negated.

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

Class Num{
  int x;
  Public:
   Void get (){
   cin >> x;
    }
   Void show (){
   cout << x << endl;
  }
   Void operator - (){
     x = -x;
   }
  };
  int main(){
    Num n;
    n.get();
    -n;
    n.show();
  }
```

Output:

Enter number: 5
After negation: -5

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

→

```
#include <iostream>
Using namespace Std;
Class Num{
 int x;
Public:
 Void get(){
 Cin >> x;
 }
  Void Show(){
  Cout << x << endl;
  // Pre - increment
   Void operator ++(){
       ++x;
   }
   // Post - increment
   Void operator ++(int){
     x++;
   };
     int main(){
      Num n;
      n.get();
      ++n;
       n.show();
       n++;
       n.show();
      }
```

output:

Input: 5
After pre-increment: 6
After post-increment: 7

Experiment no 8

* Write a program to demonstrate the Compile Time and Run Time Polymorphism (operator overloading)

A] Write a program to overload the + operator so that two strings can be concatenated.
e.g. "xyz" + "pqr" then output will be "xyzpqr".

→
```
#include <iostream>
#include <string>
Using Namespace Std;
Class AddString{
  String S;
 Public:
  Void get(){
   Cin >> S;
  }
  Add String temp;
   temp.S = S + obj.S;
   Return temp;
 }
  Void Show(){
    Cout << S << endl;
  }
   Void Show (){
    Cout << S << endl;
  }
 };
```

```
int main()
{
 Addstring a,b,c;
   a.get();
   b.get();
   c= a+b;
    c.show();
}
```

output:

Enter first string : xyz

Enter second string: pqr

Concatenated string: xyzpqr

DOMS | Page No.
Date / /

Write a program to [redacted] Clea
a base class ILogin having data mem
name and password. Declare accept() fu
Virtual Derive EmailLogin and Member
Login Classes from ILogin. Display Email l
details and membership login details of the

```
#include<iostream>
Using namespace std;
Class Login {
Protected:
  String name, pass;
Public:
  virtual void accept(){
    cin >> name >> pass;
  }
    virtual void Show(){
    cout << name << " " << pass <<
}
};
  Class EmailLogin : Public Login {
  Public:
    Void Show(){
    Cout << "Member:" << name << " "
      <<endl;
    }
  };
      int main(){
        EmailLogin e;
        Member Login m;
        e.accept();
```

```
e.show();
m.accept();
m.show();
}
```

O/P

Sanket : 1234
Email : Sanket 1234
Rahul : 4321
Member: Rahul 4321

12/11

DOMS | Page No. | Date / /

Experiment no. 9

* 1 Write a program to perform various operations on file.

a] Write a program to copy the content of one file into another open "first.te in read (ios:in) mode and "second.tex file in write (ios:out) mode. Copy the contents of "First.text" into "second.te Assume "First.text is already created.

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

int main(){
ofstream Create("First.text");
Create<<"This is a sample file.\n"
Create<<"It will be copied to second.
Create.close();
ifstream fin("First.text");
ofstream fout("Second.text");
char ch;
while(fin.get(ch)){
fout.put(ch);
}
cout<<File copied successfully!"<<
fin.close();
fout.close();
return 0;
}
```

O/P :

File copied successfully

Write a C++ program to count digit and spaces using File Handling.

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

int main(){
ofstream fout("First.text");
fout<<"Hello 123 welcome 43 to CSE";
fout.close();

ifstream fin("First.txt");
char ch;
int digits=0, spaces=0;

while(fin.get(ch)){
if(isdigit(ch))
digits++;
else if(ch==' ')
spaces++;
}
fin.close();
cout<<"Total
cout<<"To
return 0;
}
```

Output:

Total digits=6

Total Spaces=5

Write a C++ Program to count words using File Handling.

```
#include<iostream>
#include<sstream>
Using namespace std;

int main(){
ifstream inputfile("input.txt");
if(!inputFile){
    return 0;
}
String line;
  int words=0;
While (getline(inputFile,line)){
  Stringstream ss(line);
    String word;
  While (ss >> word){
    words++;
}
}
  inputFile.Close();
Cout << "Total words=" << words <<
  return 0;
  }
```

Output:

Total words=7

Write a C++ Program to count occurrence of a given word using File Handling.

```
#include <iostream>
#include<fstream>
#include<string>
Using namespace std;

int main(){
 ifstream file("input.txt");
 string target, word;
  int count = 0;
  cout << "Enter word to search",
  cin >> target;
  while(file >>word){
    if(word == target)
     count ++;
  }
 Count <<"word\n"<< target<<"\n" occurs"
   << count <<" times" <<endl,
   return 0;
}
```

Output:

Enter word to search: hello
word "hello" occurs 0 times.

10/11

Experiment no. 10

* Write a program to implement a function template and class template.

a) Write a C++ Program to find sum of Array elements using function template (e.g. Pass Integer, float and double array of 10 elements.

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

template<class T>
T SumArray(T arr[], int n){
  T sum=0;
  for(int i=0; i<n; i++){
    sum+=arr[i];
  return sum;
}
  int main(){
  int a[5]={1,2,3,4,5};
  float b[5]={1.1,2.2,3.3,4.4,5.5};
  double c[5]={1.11,2.22,3.33,4.44,5.5
 cout<<"Int sum:"<<SumArray(a,5)
cout<<"Float Sum:"<<SumArray(b,5)<<
cout<<"double Sum:"<<SumArray(c,5)<<
}
```

Output:

Int sum: 15
Float sum: 16.5
Double Sum: 16.65

DOTS | Page No.
Date / /

Write a C++ Program of squar Function using template specialization
Calculate the squake of integer no. and a string (squake of string is noting but concatenation of a string with itself).
- Write a specialized function for the squake of a string.

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

template<class T>
T squake(T x){
  return x*x;
}
  template<>
  String squake(string s){
    return s+s;
  }
    int main(){
      int n=5;
      String str="Hello"
    Cout<<"Squake Of int:"<<squake(n)<<
      endl;
      Cout<<" Squake of string:<< squake(str)
      <<endl;
      return 0;
      }
```

Output:
Squake of int: 25
```
```

DOMS | Page No. | Date / /

3. Write a c++ Program to build simple calculator using a class temp.

→

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

template <class T>
Class Calculator{
  T a,b;
 Public:
  Calculator(T x, T y){
     a=x;
     b=y;
  }
   Void operations(){
cout<<"Addition:"<< a+b <<endl;
Cout<<"Substraction:"<<a-b<<endl;
cout<<" Multiplication:<< a*b <<endl;
Cout<<" Division:"<<a/b <<endl;
 }
};
     int main(){
      Calculator<int> C1(10,5);
       Calculator<float> C2(5.5,2.2);

    cout<<" Integer Calculator\n";
      C1. Operation();
    Cout<<"\nFloat Calculator \n";
      C2. operation();
     }
```

output:

Integer calculator
Addition : 15
Substraction : 5
Multiplication : 50
Division : 2

Float calculator
Addition : 7.7
Substraction : 3.3
Multiplication : 12.1
Division : 2.5

d] Write a C++ Program to implement Push and Pop methods from Stack using class template.

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

template<class T>
 Class stack{
  T s[5];
  int top = -1;
  Public:
 Void Push(T x){
   S[++top] = x;}
 Void Pop(){
  Cout<<"Popped:"<<S[top--]<< endl;}
  Void Show(){
  for(int i=top; i>=0; i--)
  Cout<<s[i]<<" "; Cout<<endl;}
  };
    int main(){
      Stack<int> St;
      St.Push(10);
      St.Push(20);
      St.Push(30);
      St.Show();
       St.Pop();
        St.Show();
    }
```

12/11

O/P:

30 20 10
Popped: 30

Experiment no 11

* Write a C++ Program to implement generic Vectors Include following member functions:-
  - To create the vector
  - To modify the value of a given element.
  - To multiply by a scalar value
  - To display the vector in the form (10, 20, 30 ....)
  - combined all three (a, b, c) in single program.

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

  template <class T>
    Class Vector{
      T V[10];
      int size;
    Public:
      Void Create(int n){
        size = n;
        for(int i=0; i<size; i++)
          Cin >> V[i];
      }
      void modify(int index, T value){
        V[index] = value;
      }
```

```
void multiply(T scalar){
 for(int i=0; i<size; i++)
  V[i] *= scalar;
 }
 void display(){
  cout<<"(";
  for(int i=0; i<size; i++){
    cout<<V[i];
   if(i<size-1)
    cout<<", ";
   }
    cout<<")"<<endl;
 }
};
    int main(){
     Vector<int> vec;
      int n;
      cout<<"Enter size:";
       cin>>n;
      cout<<"Enter elements:";
       vec.create(n);
       vec.modify(1,50);
       vec.multiply(2);
        vec.display();
     }
```

output:
Enter size: 3
Enter elements: 10 20 30
(20, 100, 60)

Experiment No. 12

* Write C++ Program using STL.

Implement stack
Implement queue
Implement Sorting and Searching Records Such as Person Record (Name, birth date, telephone no) item Record (item code, item name, quantity and cost)
Combined all three (a, b, c) in one single Program.

```
#include<iostream>
#include<stack>
#include<queue>
#include<vector>
#include<algorithm>
using namespace std;

class Person{
public:
  string name, dob, Phone;
  void get(){
  cin >> name >> dob >> Phone;
  }
  void Show(){
  cout<<name<<" "<<dob<<" "<<Phone
   <<endl;
  }
};
```

```
int main(){
  //Stack
  stack<int> s;
    s.push(10);
    s.push(20);
  cout<<"stack top"<<s.top()<<endl;
    s.pop();

  //Queue
  queue<int> q;
    q.push(1);
    q.push(2);
  cout<<"Queue front"<<q.front()
  <<endl;
    q.pop();

  //Sort & Search
    vector<Person> V(2);
  for(int i=0; i<2; i++)
    V[i].get();
  Sort(V.begin(),V.end(),[]
  (Person a, Person b)
    {return a.name<b.name;
    }
  );
  for(auto &P:V)
    P.Show();

  String key;
  cin>>key;
```

```
auto it = find_if(v.begin(), v.end(),
  [&] (Person P)
  {return P.name == key;
   }
  );
 if(it != v.end()) it->Show();
  else cout<<"Not found\n";
  }
```

output:

```
Stack top: 20
Queue front: 1
Ravi  12-05-2000   9999
Sham  03-09-1999   8888
Ram   04-08-2001   7777
```

12/11