Write a c++ program to declare a class student having data members as Roll-no & name. Accept and display data for single student.

```
class student
{
  int Roll no;
  string name;
  Public :
  void accept ()
{
  cout <<" Enter name and Roll no ";
  cin>>name>> Roll-no;
}
void display ()
{
  cout <<"Name :" <<name;
  cout << "Roll no :" << Roll_no;
};
int main ()
{
  student S1;
  S1. accept ();
  S1. display ();
  return 0;
}
```

Write a c++ code to create a class book having data members as book-name, b-price, b-pages. Accept the data for two books and display the name of book having greater price

```
class book
{
  Public:
  string b-name.
  int b-price;
  int b-pages;
  Public:
  void accept()
{
  cout <<" Enter book name book
        price & no of pages:".
  cin>> b-name>> b-price >> b-pages;
}
void display()
{
  cout <<" Book name :" <<b-name;
  cout << "Book price :" <<b-price;
  cout << "No. of pages :" <<b-pages;
};

int main()
{
  Book B1, B2.
  B1. accept ();
  B2. accept ();
```

```
if(B1.price > B2.price)
{
  B1.display();
}
else
{
B2.display();
}
return 0;
}
```

Write a program to declare class "time", accept time in HH:MM:SS format, convert it into total seconds and display them.

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

class Time {
public:
int hours;
 int minutes;
 int seconds;

public:
 void accept()
 {
  cout << "Enter hours:";
  cin>> hours;

  cout << "Enter minutes:";
  cin>> minutes;

  cout << "Enter seconds:";
  cin>> seconds;
}

int toseconds(){
return (hours * 3600) + (minutes* 60)+
seconds;
}
```

```
void displaySeconds(){
cout << " Total seconds : " << toSeconds()
  << endl;
}
};

int main()
{
  Time t;
  t.accept();
  t.displaySeconds();
  return 0;
}
```

Q2
1/11

## Experiment 2

1. WAP to declare a class 'city' having data-member as name and population. Accept this data for 5 cities and display name of city with highest population.

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

class city {
public:
   string name;
   int population;

void getData() {
  cout << " Enter city name : ";
  cin >> name;
  cout << " Enter population : ";
  cin >> population;
}
}; void display()

int main ()
{
  city cities[5];
  for (int i = 0; i<5; i++)
    {
      cout << " city " << i << ": \n";
    cities[i].getData();
  }
```

```
int max = 0.
for( int i = 1; i<5; i++)
 {
   if (cities[i].population > cities [max]
                                 population
 }
cout << "\n City with highest population
    cities [max].name << endl;
return 0;
```

Q.2 WAP to declare a class 'Account' having data members as Account no and balance. Accept this data for 10 accounts and give intrest of 10% where balance is equal or greater than 5000 and display them.

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

class Account {
Public:
    int accno;
    float balance.

    void getData()
{
  cout << "Enter Account Number:";
  cin>> accno;
  cout << "Enter Balance:";
  cin>>balance;
}

void Interest()
 {
   balance = balance +(balance * 0.10);
 }

void display()
{
  cout << "Account No: " <<accno<<", Balance:"
  << balance <<endl;
```

```
    {
};
int main()
{
  Account acc[10];
  for(int i=0; i<10; i++)
  {
    cout<<" Account "<<i+1<<":\n";
    acc[i].getData();
  }
cout<<" In Account with balance>=5000
      after 10% interest:\n";
for(int i=0; i<10; i++)
  {
    if(acc[i].balance>=5000)
    {
      acc[i].~~appt~~Interest();
    acc[i].display();
    }
  }
return 0;
}
```

Q.3 WAP to declare a class 'staff' having data members as name and post. Accept this data for 5 staff and display names of staff who are Hod

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

class Staff {
public:
  string name;
  string post;

void getData()
{
  cout<<"Enter Staff name:";
  cin>>name;
  cout<<" Enter post:";
  cin>>post;
}
};
int main()
{
 Staff s[5];
 for(int i=0; i<5; i++)
   {
     cout<<"Staff "<<i+1<<":\n";
     s[i].getData();
}
```

```
cout<<"\n Staff who are HOD : \n";
for(int i=0; i<s; i++)
    {
      if(s[i].post == "HOD" || s[i].post
         == "hod")
    {
      cout<<s[i].name<<endl;
    }
}
return 0;
}
```

Q
29/12/25

## Experiment 3

1. Write a program to declare class 'book' containing data members as book-title, author name price. Accept and display information for one object using a pointer to that object.

```
#include <iostream>
#include <string>
using namespace std;
class Boor {
   string Boor-title;
   string author-name;
    float price;
  public:
   void accept(){
  cout<<"Enter Boor title:";
  cin>> boor-title;
  cout<<"Enter author name:";
  cin>> author-name;
  cout<<"Enter Price:";
   cin>> price;
  }
  void display()
  {
   cout<<" Boor Title :"<<boor-title<<endl;
  cout<<"Author name :"<<author-name<<endl;
  cout<<" Price :"<<price<<endl;
  }
};
```

```
int main()
{
  book b;
  Book *ptr = &b;
  ptr → accept();
  ptr → display();
return 0;
}
```

Output:

Book Title: ABC
Author Name: xyz
Price: 400.

Q.2. WAP to declare a class 'student' having data members roll-no and percentage. Using this pointer invoke member functions to accept and display this data for one object of class.

```
#include <iostream>
using namespace std;
class student
{
  int roll_no;
  float percentage;
public:
void accept()
{
cout << "Enter Roll Number:";
cin>> this → roll_no;
cout << " Enter percentage:";
cin>> this → percentage;
}
void display()
{
 cout << "Roll Number:" << this → roll_no << endl;
cout << "Percentage:" << this→percentage << endl;
}
};
int main()
{
  student s;
  s.accept();
  s.display();
  return 0;
}
```

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

```
#include <iostream>
using namespace std;
class student {
public:
  class Marks {
    public:
      int m1, m2, m3;
  void input() {
  cout << "Enter marks for 3 subjects:";
   cin>> m1 >> m2 >> m3;
  }
```

```
float Percentage()
{
  int total = m1+m2+m3;
  return total / 3.0;
  }
};

int main()
{
 student :: Marks marks;
 marks.Input();
 float percentage = marks.Percentage();
 cout << "Percentage:" << percentage <<
   endl;
 return 0;
}
```

8/5/25

## Experiment 4

1. WAP to swap two numbers from same class using object as function argument. Write swap function as member function.

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

class Number {
   int value;
public:
   void input()
  {
   cout << "Enter value";
   cin >> value;
  }
  void display()
{
   cout << value << endl;
}
void swap(Number &n)
  {
    int temp = value;
    value = n.value;
    n.value = temp;
  }
};

int main()
```

```
{
  number n1, n2;
  cout << "Enter first number:";
  n1.input();
  cout << " Enter second number:";
  n2.input();
  n1.swap(n2);
  cout << "After swapping";
  cout << "n1 = ". n1.display();
  cout << "n2 = "; n2.display();
  return 0;
}
```

Output:

```
Enter first number:
Enter value : 99
Enter second number:
Enter value : 100

After swapping:
n1 = 100
n2 = 99.
```

2. WAP to swap two numbers from same class using friend function.

```
#include <iostream>
using namespace std;
class Number {
  int value;
public:
  void input()
  {
    cout << " Enter value :";
    cin >> value;
  }
  void display()
  {
    cout << value << endl;
  }
  friend void swap(Number &x, Number &y);
};

void swap(Number &x, Number &y)
{
  int temp = x.value;
  x.value = y.value;
  y.value = temp;
}
int main()
{
  Number n1, n2;
  cout << "Enter first number:";
  n1.input();
  cout << " Enter second number:";
  n2.input();

  swap(n1, n2);
```

```
cout << " After swapping : ";
cout << "n1 = " ; n1.display();
cout << "n2 = " ; n2.display();
return 0;
}
```

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

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

class B;

class A {
  int numA;
public:
  void input() {
  cout << "Enter number for A:";
  cin >> numA;
  }
  void display()
  {
  cout << "A:" << numA << endl;
  }
  friend void swap (A &x, B &y);
};

class B {
  int numB;
public:
  void input()
  {
  cout << "Enter number for B:";
  cin >> numB;
  }
  void display()
  {
  cout << "B:" << numB << endl;
  }
  friend void swap (A &x, B &y);
};

void swap(A &x, B &y)
{
  int temp = x.numA;
  x.numA = x.numB;
  y.numB = temp;
}
int main()
{
  A obj1;
  B obj2;
obj1.input();
obj2.input();
swap(obj1, obj2);
cout << "After swapping :" << endl;
obj1.display();
obj2.display();
return 0;
}
```

4. WAP to create two classes Result1 and Result2 which stores the marks of the students. Read the value of marks for both the class objects and compute the avg. of two results.

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

class Result1 {
 public:
  float marks;
  void marks()
 {
   cout << " Enter marks for result1 :";
   cin >> marks;
  }
};

class Result2 {
 public:
  float marks;
 void marks()
 {
  cout << " Enter marks for Result2 :";
   cin >> marks;
  }
};

int main()
{
 Result1 r1;
 Result2 r2;

 r1.marks();
 r2.marks();

 float avg = (r1.marks + r2.marks)/2.0;

 cout << "Average marks of two
   results :" << avg << endl;
 return 0;
 }
```

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

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

class number2;

class Number1 { int num1;
void input() cout << "Enter 1st Number :";
   cin >> num1;
  }
  friend void greatest(Number1, Number2);
};

class Number2 {
  int num2;
 public:
```

```
void input()
{
 cout << " Enter 2nd Number : ";
  cin >> num2;
 }
friend void greatest (Number1,
                      Number2);
};

void greatest (Number1 n1, Number2 n2)
{
 if(n1.num1 > n2.num2)
   cout << " Greatest Number is : "
        << n1.num1 << endl;
 else
   cout << " Greatest number is : "
        << n2.num2 << endl;
}
 int main()
{
  Number1 a;
  Number2 b;

a.input();
b.input();

greatest (a,b);
  return 0;
}
```

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

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

class ClassB;

class ClassA {
  int numA;
public:
  ClassA (int a) {numA = a;}
  friend int sum(ClassA, ClassB);
};

class ClassB {
 int numB;
public:
  classB(int b) { numB = b; }
  friend int sum (ClassA, ClassB);
};

int sum (Class A a, ClassB b)
 {
   return a.numA + b.numB;
 }
```

```
int main()
{
 Class A objA(20);
 Class B objB(10);
 cout << "Sum = " << sum(objA, objB);
 return 0;
}
```

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

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

class Number {
   int value;
public:
   Number (int v)
   {
      value = v;
   }
void display()
{
  cout << value << endl;
}
friend void swap (Number &a,
                  Number &b);
};

void swap (Number &a, Number &b)
{
   int temp = a.value;
    a.value = b.value;
     b.value = temp;
}
int main()
{
  Number n1(10), n2(20);
  swap(n1, n2);
cout << "After swap" << endl;
  n1.display();
  n2.display();
  return 0;
}
```

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

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

class Cube;

class Box {
 int volume;
public:
  Box(int v) {volume = v;}
```

```
friend void Greater(Box, Cube);
};

class Cube {
 int volume;
public:
  Cube(int v) {volume = v;}
friend void Greater(Box, Cube);
};

void Greater(Box b, Cube c)
{
 if(b.volume > c.volume)
  cout << "Box has greater volume";
else
  cout << "Cube has greater volume:";
}

int main()
{
  Box b(100);
  Cube c(120);
  Greater(b, c);
  return 0;
}
```

9. Create a class complex with real & imaginary parts as private members. Use a friend function to add two complex numbers and return the result as new complex object

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

class complex {
  int real, imag;
 public:
   void input()
{
    cout << "Enter first complex no:";
    cin >> real >> imag;
 }
 void display
  {
    cout << real << "+" << imag << "i";
  }
friend complex add(complex, complex);
};

complex add(complex c1, complex c2)
 {
  complex temp;
 temp.real = c1.real + c2.real;
 temp.imag = c1.imag + c2.imag;
 return temp;
}
```

```
int main()
{
  Complex c1, c2, sum;
cout<<"Enter first complex number(
            real & imaginary)";
c1.input();
cout <<"Enter second complex
      number(real & imaginary):";
c2.input();
sum = add(c1, c2);
cout <<"sum = "; sum.display();
}
```

10. Create a class student with private data members: name & three subject marks. Write a friend function calculate Average (student) that calculates and displays average marks.

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

class student {
  string name;
int m1, m2, m3;
 public:
void input()
{
 cout <<"Enter name of student: ";
cin>> name;
cout <<"Enter marks of 3 subjects:";
cin>> m1>> m2>>m3;
}
friend void Average (student);
};

void Average (student s)
{
  cout <<" Average = "<<(s.m1+s.m2+
                          s.m3)/3.0;
}
int main()
{
 student s;
 s.input();
  Average(s);
  return 0;
}
```

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

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

class Beta;
class Gramma;

class Alpha {
 int a;
public:
```

```
    void input()
    {
     cout << "Enter value of Alpha :";
      cin>> a;
    }
    friend void sum(Alpha, Beta, Gamma);
};

class Beta {
   int b;
public :
    void input ()
{
  cout << "Enter value of Beta :";
   cin>> b;
  }
  friend void sum(Alpha, Beta, Gamma);
};

class Gamma {
  int c;
public:
  void input()
{
  cout << " Enter the value of Gamma:";
   cin>> c;
}
friend void sum (Alpha, Beta, Gamma);
};
```

```
void sum (Alpha x, Beta y, Gamma z)
{
  cout << "Sum = " <<x.a+y.b+z.c<<endl;
}
int main()
{
  Alpha a;
  Beta b;
  Gamma c;

 a.input();
 b.input();
 c.input();

 sum(a,b,c);
}
```

12. Create a class point with private member x and y. Write a friend function that calculates and returns the distance between two point objects.

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

class Point {
   float x,y
public:
   void input()
{
   cout << " Enter x :";
   cin>> x;
```

```
    cout << " Enter Y: ";
    cin >> y;
  }
  friend float distance (Point, Point);
};

float Distance (Point p1, Point p2)
{
  float dx = p2.x - p1.x;
  float dy = p2.y - p1.y;
  return dx * dx + dy * dy;
}

int main ()
{
  Point p1, p2;
  cout << "first point ";
  p1.input();
  cout << "second point ";
  p2.input();
  cout << "Distance squared = " << distance
                                    (p1, p2);
  return 0;
}
```

Q.13. Create two classes : BankAccount holds private balance information. Write a friend function in Audit that accesses and prints balance information for auditing.

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

class Audit;

class BankAccount {
    float balance;
public:
  void input()
{
  cout << "Enter account balance : ";
  cin >> balance;
 }
 friend void Account (BankAccount, Audit);
};

class Audit {
public:
  friend void Account (BankAccount, Audit);
};

void Account (BankAccount acc, Audit)
 {
   cout << " Auditing account ... :";
   cout << "Account balance = " << acc.balance;
 }
int main () {
BankAccount acc;
 Audit a;
acc.input();
Account (acc, a);
 return 0;
}
```

7/8/25

8/10/25

## Experiment 5

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

1) default constructor:

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

class Sum {
  int n = 10;
  int result = 0;

public:
 Sum ()
 {
   for(int i = 1; i<=n; i++)
       result += i;
 }
 void display ()
 {
   cout << " sum of numbers from 1 to "
        <<n<< "is: " << result <<endl;
 }
};int main()
{
  Sum s;
  s.display (); return 0; }
```

o/p. sum of numbers from 1 to 10 is: 55.

2) Parametirized

```
#include <iostream>
using namespace std;
class Sum {
    int n;
    int sum = 0;
    public:
 Sum(int num)
 {
    n = num;
    for(int i=0; i<=n; i++)
    {
      sum = sum +i;
    }
 }
 void display ()
 {
    cout<<"Sum = "<<sum;
 }
};int main()
 {
    Sum s1(5);
    s1.display();
 return 0;
 }
```

o/p Sum = 15

3 Copy Constructor

```
#include <iostream>
using namespace std;
class sum {
    int n;
    int s = 0;
    public:
    sum()
    {
      n = 4;
    }
sum(sum &obj)
    {
        n = obj.n;
        int i;
    for(i=1; i<=n; i++)
  {
      s = s+i;
  }
    cout<<"sum = "<<s;
    }
};
int main()
  {
    sum s1;
    sum s2(s1);
    return 0;
  }
```

O/P Sum = 10

b. Write a program to declare a class "student" having data members as name and percentage. Write a constructor to initialize these data members. Accept and display data for one student.

1) Default :

```
#include <iostream>
using namespace std;
class student {
 string name;
  float percentage;
  public:
  student()
{
  name = "Anushka";
  percentage = 90.0;
}
 void display()
{
 cout<<"Name and Percentage:"<<
 roll-no<<" "<<percentage;
}
};
int main()
{
 Student s1;
 s1.display();
 return 0;
}
```

O/P Name and Percentage: Anushka 90

2) Parameterized

```
#include <iostream>
using namespace std;
class student {
   float per;
   string name;
   public:
student (float p, string n)
{
   per = p;
   name = n;
}
void display ()
{
  cout << "Percentage :" << per << endl;
  cout << "Name : " << name;
  }
};
int main()
{
  student s1 (90, "anushka");
  s1.display();
  return 0;
}
```

O/P Percentage : 90
Name: anushka

3) copy

```
#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<<"%";
   {
};
int main()
{
   string n;
   float p;
   cout<<"Enter Student Name: ";
   getline (cin, n);
```

```
cout << "Enter Percentage : ";
cin >> p;
student s1(n, p);
student s2(s1);
cout << "In student Data" << endl;
s2.display();
return 0;
}
```

O/P Enter student Name : abc
Enter Percentage : 98

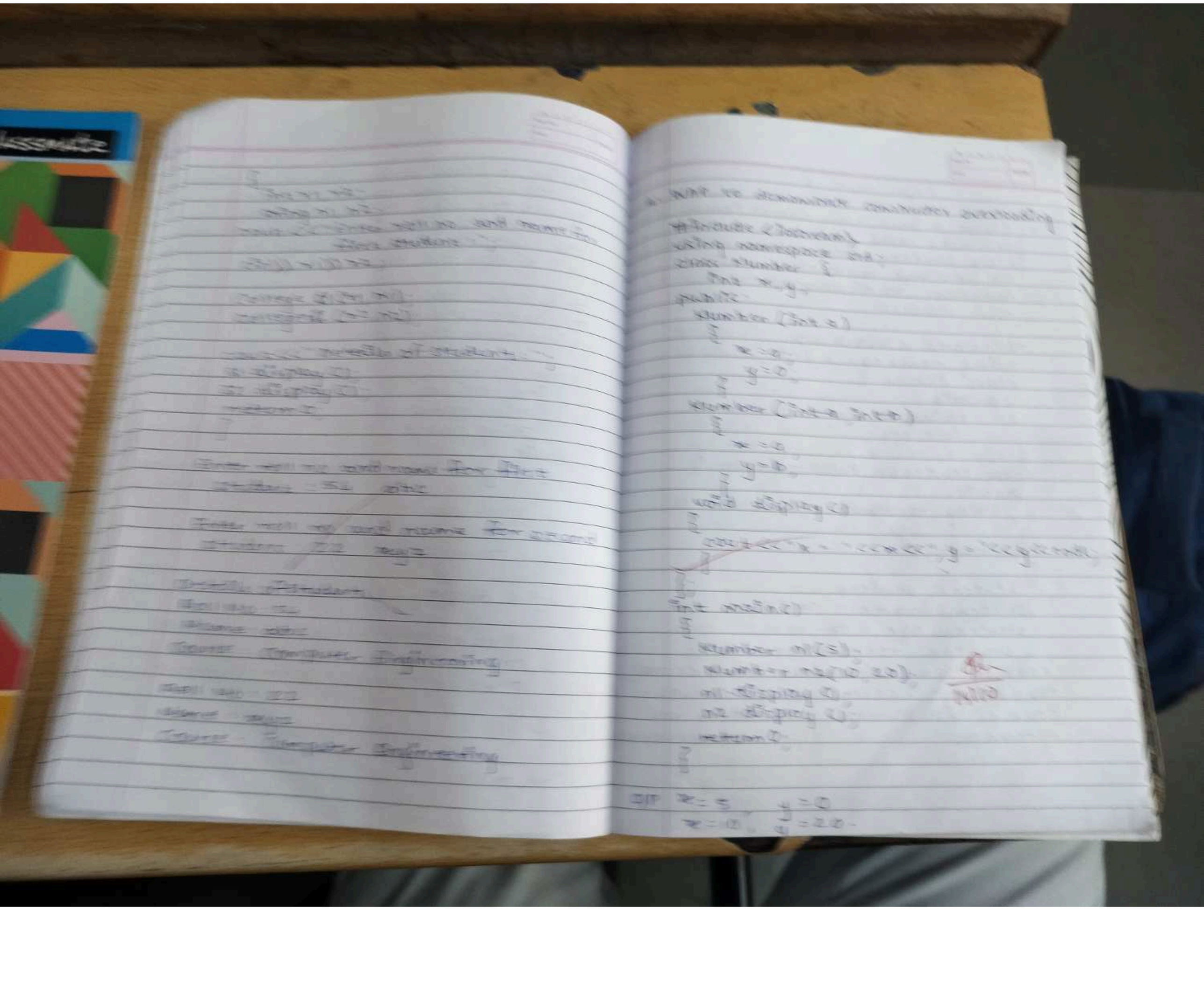
Student Data
Name : abc
Percentage : 98%

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

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

class college {
    int roll-no;
    string name;
    string course;
public:
    college(int r, string n, string c = "Computer Science")
    {
        roll-no = r;
        name = n;
        course = c;
    }
void display()
    {
        cout << "Roll no: " << roll-no << endl;
        cout << "Name: " << name << endl;
        cout << "Course: " << course << endl;
    }
};

int main()
```



Experiment 6

a. Create a base class called Person with attributes name and age. Derive a class student from person that adds an attribute rollNumber. Write functions to display all details of the student

```
#include <iostream>
using namespace std;
class Person {
  public:
    string name;
    int age;
};
class student : Public Person {
 public:
   string rollNumber;
};
int main()
{
  student student;
  student.name = "sahil";
  student.age = 17;
   student.rollNumber = "2";
  cout << "Name : " << student.name << endl;
  cout << "Age : " << student.age << endl;
  cout << "Roll Number : " << student.roll
                              Number << endl;
  return 0;
}
```

output :
Name : Sahil
Age : 17
Roll Number : 2

b. Create two base classes, Academic and sports

Academic class contains marks of a student

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

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

class Result : public Academic, public
   sports {
  public:
    int totalscore;
```

```
void calculate Total()
  {
    totalscore = marks + score;
  }
void display()
{
 cout << "Marks: " << marks << endl;
 cout << "sports score!" << score << endl;
 cout << "Total score: " << totalscore << endl;
  }
};
int main()
Result studentResult;

studentResult.marks = 90;
studentResult.score = 95.
studentResult.calculate Total();
studentResult.display();
return 0;
}
```

O/P Marks: 90
Sports Score: 95.
Total Score: 185

c. Create a class Vehicle with attributes like brand and model. Derive a class Car from vehicle which adds an attribute type. Further derive a class ElectricCar from Car which adds battery capacity. Write functions to display all the details.

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

class Vehicle {
  public:
    string brand;
    string model;
void display()
 {
   cout << "Brand" << brand << endl;
   cout << "Model" << model << endl;
  }
};
class Car : Public vehicle {
public:
    string type;
void display()
  {
    vehicle :: display();
    cout << "Type:" << type << endl;
  }
};
```

```
class ElectricCar : public Car {
public:
    int BatteryCapacity;
    void display()
    Car :: display();
cout << " Battery Capacity : " << batteryCapacity
     << " kWh " << endl;
  }
};
int main()
{
  Vehicle v;
  v.brand = "BMW";
  v.model = "m5cs";
  v.display();

  Car c;
  c.brand = "Suzuki";
  c.model = "Swift";
  c.type = "Hatchback";
  c.display();

  ElectricCar e;
  e.brand = "Tesla";
  e.model = "model s";
  e.type = "sedan";
  e.batteryCapacity = 100;
  e.display();
  return 0;
}
```

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

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

class Employee {
    public:
        int empID;
        string name;
    void display(){
    cout << " Employee ID : " << empId << endl;
    cout << "Name : " << name << endl;
    }
};
class Manager : public Employee {
  public:
    string Department;
    void display() {
     Employee :: display();
     cout << " Department : " << department << endl;
    }
};
class Developer : public Employee {
 public:
```

```
    string ProgrammingLanguage;
    void display() {
      Employee :: display();
cout << "Programming Language": <<
        programmingLanguage << endl;
    }
};
int main()
{
  Manager m;
  m.empID = '101';
  m.name = 'ABC';
  m.department = "Marketing";
  m.display();

  Developer d;
  d.empId = 102;
  d.name = "MNM";
  d.programmingLanguage = "C++";
  d.display();
  return 0;
}
```

c. Combine multilevel and multiple inheritance Create a base class Person with attributes name and age. Derive class student from person. Create two classes Sports and Academics. Derive class Result from student and Sports.

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

class Person {
  public:
    string name;
    int age;
void displayPerson()
{
  cout << "Name: " << name << endl;
  cout << "Age: " << age << endl;
  }
};
class student : public Person {
  public:
    int studentId;
void displayStudent()
  {
    displayPerson();
cout << "Student ID: " << studentId << endl;
  }
};
```

```
class Sports {
 public:
  string sportName;
   int sportScore;

void displaySports()
{
   cout<<"Sport :"<<sportName<<endl;
   cout<<" sport score :"<<sportScore<<endl;
  }
};
class Result : public student, public Sports{
 public:
   float academicScore;
 void displayResult(){
   displaystudent();
    display Sports();
 cout<<" Academic score :"<<academicScore<<en
  }
};
 int main()
 {
   Result r;
  r.name = "Sahil";
  r.age = 17;
  r.studentId = 177;
 r.sportName = "Football";
  r.sportScore = 99;
  r.academicScore = 92.5;
  r.displayResult();
 return 0;
}
```

f. Write a c++ program to implement following in heritance.

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

class college_student {
 protected:
    int student_id;
    int college_code;
 public:
    void InputStudent()
 {
    cout<<"Enter student id :";
    cin>> student_id;
  cout<<" Enter college code:";
    cin>> college_code;
  }
 void displayStudent()
 {
   cout<<"student ID:"<<student_id<<endl;
  cout<<"college code:"<<college_code<<endl;
  }
};
class Test : virtual public college_student {
 protected:
    float percentage;
public:
    void InputTest() {
    cout<<"Enter Test Percentage:";
     cin>> percentage;
    }
```

```
void displayTest()
{
    cout << "Test Percentage : " << percentage
         << "%" << endl;
}
};
class Sports : virtual public college_student
protected:
    char grade;
public:
    void InputSports() {
    cout << "Enter Sports Grade:";
    cin >> grade;
    }
    void displaySports() {
    cout << "Sports Grade : " << grade << endl;
    }
};
class Result : public Test, public Sports {
public:
    void InputResult() {
     inputStudent();
     inputTest();
     inputSports();
    }
void displayResult() {
    displayStudent();
     displayTest();
     displaySports();
    }
};
```

```
int main()
{
    Result r;
    r.inputResult();
    r.displayResult();
    return 0;
}
```

Q1

1+1+0

ASSMATE

Experiment 7

1. WAP using function overloading to calculate the area of laboratory (rectangle) and area of class room (square)

```
#include <iostream>
using namespace std;
class Area {
public:
float calculate (float length, float
breadth)
{
    return length * breadth;
}
float calculate (float side)
{
    return side * side;
}
};
int main()
{
    Area a;
    float length, breadth, side;
cout <<"Enter length and breadth
        for laboratory: ");
cin>> length >> breadth;
cout<<" Area of laboratory (rectangle): "
    <<a.calculate (length, breadth) <<endl;
cout <<" Area of classroom (square): "
    <<a.calculate(side) <<endl;
return 0;
}
```

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

```
#include <iostream>
using namespace std;
class SumCalculate {
public:
float sum ( float a, float b, float c,
            float d, float e)
{
    return a+b+c+d+e;
}

int Sum (int a, int b, int c, int d,
         int e, int f, int g, int h,
         int i, int j)
{
    return a+b+c+d+e+f+g+h+i+j;
}
};
int main() {
SumCalculate s;
float fsum = s.sum(1.1f, 2.2f, 3.3f, 4.4f,
                   5.5f);
int isum = s.sum(1,2,3,4,5,6,7,8,9,10);
cout<<"Sum of 5 float nos: "<<fsum<<endl;
cout <<"Sum of 10 integer nos: "<<isum<<endl;
return 0;
}
```

O/P
Sum of 5 float nos: 16.5
Sum of 10 integer nos: 55

Q 3 WAP to implement unary (-) operator when used within object so that numeric data member of class is negated.

```
#include <iostream>
using nameespace std;
class number {
  int a;
  public:
  void accept()
{
   cout <<" a " <<endl;
    cin>> a;
  }
void display() {
  cout <<" a " << a <<endl;
  }
void operator -()
  {
    a = -a;
  }
};
int main() {
 number n1;
  n1.accept();
  -n1;
  n1.display();
  return 0;
}
```

d. WAP to implement the Unary ++ operator (for Pre increment and post Increment) when used with the object so that the numeric data member of class is incremented.

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

class Number {
    int n;

public :
    Number (int x) {
        n = x;
    }
    void increment()
    {
        n++;
    }
    void increment (int x)
    {
        n += x;
    }
void operator++()
    {
        ++n;
    }
void operator++(int)
    {
        n++;
    }
```

```
void show()
 {
   cout << "Value = " << num << endl;
 }
};

int main()
{
  Number num(5);
 cout << "Initial value:";
  num.show();
cout << "Using function overloading:";
 num.increment();
  num.show();
num.increment(3);
  num.show();

cout << "Using operator overloading:";
  ++num;
 num.show();

 num++;
 num.show();

 return 0;
}
```

```
O/P  Initial value
     Value = 5

     Using function overloading:
     Valde = 6
      Value = 9

     Using operator overloading:
     Value = 10
      Value = 11
```

14/10

## Experiment 8

a. WAP to overload the '+' operator so that two strings can be concentrated. Eg: "xyz" + "pqr" then output will be "xyzpqr".

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

class Addstring {
    string str;
public:
    Addstring (string s = "")
    {
        str = s;
    }
Addstring operator + (Addstring s2)
    {
        Addstring temp;
        temp.str = str + s2.str;
        return temp;
    }
void show()
{
    cout << "Concentrated string : " << str <<
}
};

int main()
{
    Addstring s1("xyz");
    Addstring s2("pqr");
    Addstring s3;

    s3 = s1 + s2;
    s3.show();

    return 0;
}
```

O/P Concentrated string : xyzpqr

b. WAP to create a base class I Login having data members name and password. Declare accept() function virtual Derive Email Login and Membership Login classes from II Login. Display Email login details and membership login details of employee.

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

Class ILogin {
protected:
    string name, password;
public:
    virtual void accept()
    {
        cout << "Enter name :";
        cin >> name;
```

```
    cout << "Enter password : ";
    cin >> password;
    }

virtual void display ()
    {
    cout << "Name : " << name << "Password :"
        << password << endl;
    }
};

class EmailLogin : public ILogin {
    string email;
public:
    void accept() override {
    cout << "Enter email login name:";
    cin >> name;
    cout << " Enter Password :";
    cin >> password;
    cout << " Enter email ID : ";
    cin >> email;
    }
void display() override {

cout << " --- EMAIL LOGIN DETAILS ---";
cout << "Name : " << name << " Password : "
    << password << "\n Email ID : "
        << email << endl;
    }
};
```

```
class membershipLogin : public ILogin {
    Int memberID;
public:
    void accept () override {
    cout << " Enter membership login name : ";
    cin >> name;
    cout << " Enter password : ";
    cin >> password;
    cout << " Enter membership ID : ";
    cin >> memberID;
    }
void display () override {
cout << "\n --- Membership Login
        Details ---";
cout << "Name : " << name << "\nPassword :"
    << password << "\n membership
        ID : " << memberID << endl;
    }
};
int main()
{
    ILogin *ptr;
    EmailLogin e;
    membershipLogin m;

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

    ptr = &m;
    ptr -> accept();
    ptr -> display();
    return 0; {
```

Enter email login name : ABC
Enter password : 7878
Enter email ID : abc@gmail.com

--- EMAIL LOGIN DETAILS ---
Name : ABC
Password : 7878
Email ID : abc@gmail.com
Enter membership login name : NN
Enter password : 7788
Enter membership ID : 123

--- MEMBERSHIP LOGIN DETAILS ---

Name : NN.
Password : 7788
Membership ID : 123.

14/10

# Experiment 9

1.

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
  fstream new-file.
new-file.open("new-file.txt",
                ios::out);
if (!new_file)
 {
  cout<<"File creation failed"<<endl;
  }
  else
  {
   cout<<"New file created"<<endl;
    new-file.close();
  }
 return 0;
}
```

2.

```
#include <iostream>
#include <fstream>
using namespace std;
int main()
{
  ofstream file1("file1.txt", ios::app);
  if(!file1)
  {
    cout<<" Error opening file1.txt"
        <<endl;
    return 1;
  }
file1<<"Welcome to MIT"<<endl;
 file1.close();

ifstream file1_read("file1.txt");
 ofstream file2("file2.txt");

if(!file1_read)
 {
   cout<<" Error opening file1.txt
  for reading"<<endl;
   return 1;
 }
char ch;
while(file1_read.get(ch))
 {
   file2.put(ch);
 }
```

```
cout<<" content copied succesfully
    from file1.txt to file2.txt"
    <<endl;
file1_read.close();
file2.close()
return 0;
}
```

## Experiment 10

1. Write a c++ program to find sum of Array elements using function template

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

template <class T>
T sumArray (T arr[], int size)
  {
     T sum = 0;
   for(int i = 0; i<size ; i++)
     {
       sum = sum + arr[i];
     }
    return sum;
  }
int main()
{
  int intArr[5] = { 10, 20, 30, 40, 50};
  cout << "Sum of int array = " <<
          sumArray (intArr, 5) << endl;
  return 0;
}
```

O/P Sum of int array : 150

2. Write a c++ program of Square Function using template specialization.

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

template <class T>
 T square (T num)
  {
     return num * num;
  }
template <>
string square <string> (string str)
  {
     string result;
      result.resize (str.size()*2);
    strcpy (&result[0], str.c_str());
    strcat (&result[0], str.c_str());
    return result;
  }
int main()
{
  int num = 22;
  string str = "xyz";
cout << "Square of integer: " << square(num);
cout << "Square of string: " << square(str);
  return 0;
}
```

O/P Square of integer : 484
Square of string : xyzxyz

3. Write a c++ program to build simple calculator using a Class Template.

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

template <class T>
class A
{
  public:
   T num1, num2;
   void accept()
   {
    cout << " Enter first value: ";
    cin >> num1;
    cout << " Enter second value: ";
    cin >> num2;
   }
  void add()
   {
    cout << "Addition : " << num1+num2;
   }
  void subtract()
  {
   cout << "Subtraction : " << num1-num2;
  }
  void multiply()
  {
   cout << "Multiplication : " << num1*num2;
  }
  void divide()
  {
   cout << "Division : " << num1/num2;
  }
};
int main()
{
  A <int> obj;
  obj.accept();
  obj.add();
  obj.subtract();
  obj.multiply();
  obj.divide();
  return 0;
}
```

O/P
```
Enter first value : 8
Enter second value : 2
Addition : 10
Subtraction : 6
Multiplication : 16
Division : 4
```

## Experiment 11

Write a c++ program to implement generic vectors

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

int main()
{
  vector <char> v(10);
  cout << "size = " << v.size() << endl;

  for(int i=0; i<10; i++)
    v[i] = i + 'a';

  cout << "Current contents : ";
  for (int i = 0; i<v.size(); i++)
    cout << v[i] << " ";
  cout << "\n\n";

  cout << "\n Expanding vector ... \n";
  for (int i=0; i<10; i++)
    v.push_back(i+'a'+10);

  cout << "size now = " << v.size() << endl;
  cout << "Current contents : \n";
  for (int i = 0; i<v.size(); i++)
    cout << v[i] << " ";
  cout << "\n\n";

  for (int i = 0; i<v.size(); i++)
    v[i] = toupper(v[i]);

  cout << "Modified contents : \n";
  for (int i = 0; i<v.size(); i++)
    cout << v[i] << " ";
    cout << endl;
  return 0;
}
```

O/P

```
size = 10
Current contents:
a b c d e f g h i j

Expanding vector
size now = 20
Current contents:
a b c d e f g h i j k l m n o
p q r s t

Modified contents:
A B C D E F G H I J K L M N
O P Q R S T
```

2.
```
#include <iostream>
#include <vector>
using namespace std;
int main()
{
   vector <int> v = {1, 2, 3, 4, 5}
   int scalar;
   cout << "Enter scalar value to
            multiply :";
   cin>> scalar;

cout << "vector before multiplication:";
for(int i =0; i<v.size(); i++)
  {
    cout << v[i]<<" ";
  }
 cout << endl;
 for( int i =0; i< v.size(); i++)
  {
    v[i] *= scalar;
  }
 cout << "vector after Multiplication:"
 for (int i=0; i<v.size(); i++)
  {
    cout << v[i] << " ";
  }
 cout <<endl;
 return 0;
}
```

3. With using Iterator.

```
#include <iostream>
#include <vector>
using namespace std;
int main()
{
   vector <int> v = {1, 2, 3, 4, 5}
cout << "vector before multiplication:";
 for (vector <int>::iterator it =
          v.begin; it != v.end(); ++it)
  {
    cout << *it << " ";
  }
  cout << endl;
 Int scalar;
cout << "Enter scalar value :";
 cin>> scalar;
 for (vector <int>::iterator it = v.begin();
    it != v.end(); ++it)
    {
      * it = (*it) * scalar;
    }
cout << "vector after Multiplication:";
 for (vector <int>::iterator it = v.begin()
    it != v.end(); ++it)
    {
      cout << *it << " ";
    }
  cout << endl;
 return 0;
}
```

Experiment 12

a WAP using STL to implement stack

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

int main()
{
  stack <int> s;
cout <<"stack";
 s.push(10);
 s.push(20);
 s.push(30);

cout <<" pop() <<endl;
cout <<" Display elements ";
stack <int> temp = s;
while (!temp.empty())
  {
     cout << temp.top() <<" ";
     temp.pop();
  }
cout <<endl;
cout << top element : <<s.top();
return 0;
}
```

b WAP using STL to implement queue

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

int main() {
 queue <int> q;
 q.push(10);
 q.push(20);
 q.push(45);

cout <<" Queue elements \n" <<endl;
 queue <int> temp = {
  while (!temp.empty())
    {
      cout << " temp.front() <<" ";
      temp.pop();
  }
cout << endl;
return 0;
}
```

HackerRank | Prepare Certify Compete Search

# Sahil Lale

@sahillale17

## Personal Information

sahillale17@gmail.com

Add your mobile number

India

Complete your profile

**Add your missing details →**

This data will be helpful to auto-fill your job applications

0%

## My Badges


## My Resume

+ Add Resume

Add your resume here

## My Certifications


## EEO settings