PAGE NO.
DATE :

expiriment - 1 :

Q) write a C++ program to declare a class student having data member as roll. no & student ~~having data members as~~ name. Accept and display data for single da

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

class student {
    int roll-no;
    String name;
public:
    Void accept () {
        cout <<"enter roll no. of student and name: ";
        cin >>roll_no >> name;
    }
    void display () {
        cout <<"Roll no: " << roll_no<< ";Name: "
            << name <<endl;
    }
};
int main () {
    student S1;
    S1. accept ();
    S1 . display ();
    return 0;
}
```

Out-put :

enter roll no. of student and name: 17 swani
roll no.: 17 , Name : Swani

Q) write a C++ program to creat a class book, having data members as bname, bprice, bpages. Accept two data for two books and display the names of book having greater price.

```
#include <iostream.h>
  using namespace std;
class book {
 public : int bprice, bpapages;
 public : string bname;
 public : void accept (){
          cout << "enter name, price and pages: ";
          cin>>bname>>bprice>>bpages;
        }
};
int main (){
    book b1, b2;
    b1 .accept ();
    b2 .accept ();
    cout <<" Book with greatest price:" <<(b1.bprice>b2.bprice)?
                                    b1.bname:b2.bname;
   return 0;
}
```

out put :

```
enter name, price, and number of pages:
  book1   800  800
enter name, price, and number of pages:
  book 2   399   999
Book with greatest price : book 2
```

Q) write a program to declare class "time", accept time in HH:MM:SS format, consit it into total second and display them.

→

```
#include <iostream>
 using namespace std;
 class time {
  int h,m,s;
public :
  void accept (){
       cout << "enter time in HH:MM:SS format
       cout << "Hours (0-23): ";
       cin >> h;
       cout << "Minutes (0-59): ";
       cin >> m;
       cout << "seconds (0-59): ";
       cin >> s;
  }
  void display (){
       cout << "\n entered time : "<<h<<" . "<<m<<" : "<<s;
       cout<< "\n total seconds :"<< ((long int)h*3600+(long int
                                   *60 +(long int) s)<< endl;
};
  int main (){
    time t1;
    t1. accept ();
    t1. display ();
    return 0;
}
```

```
out put : enter time in HH:MM:SS format
Hours (0-23) : 21
Minutes (0-59): 12
Seconds (0-59): 33

enter time : 21 : 12 : 33
Total seconds : 76353
```

PAGE NO.
DATE :

Experiment No. 2 :

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

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

 class city {
      string city_name;
      int population;
public:
  Void accept () {
      cout << "enter city name:  ";
      cin . ignore();
      getline (cin, city_name);
      cout << "Enter population:  ";
       cin >> population;
     }
     void display () {
          cout<<" city name:  "<< city_name<<
                                  endl;
     }
     int getpopulation () {
          return population;
     }
};
```

```
int main () {
    City c[5];

for (int i=0; i<5; i++) {
 cout << "\nCity" << i+1 << ":\n";
  c[i].accept();
}
int max = c[0].getPopulation();
int index=0;
for (int i=1; i<5; i++) {
   if (c[i].getPopulation()>max) {
        max = c[i].getPopulation();
        index=i;
       }
 }
 cout << "\nCity with highest population:\n";
 c[index].display();
  return 0;
}
```

Out put:

```
enter city name: Mumbai
enter population: 21000
enter city name: Delhi
enter population: 19000
enter city name: Pune
enter population: 9000
enter city name: Kolkata
enter population: 15000
enter city name: Chennai
enter population: 11000
City with highest population:
 city name: Mumbai
 Population: 21000
```

obtain post of workers:

(1)

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

 class Staff {
      string name;
      string post;
 public:
     void accept() {
          cout << "enter name: ";
          cin.ignore();
          getline(cin, name);
          cout << "enter post: ";
          cin >> post;
     }
     void display() {
          cout << "Name: " << name << endl;
          cout << "post: " << post << endl;
     }
     string getPost() {
          return post;
     }
     void displayHOD() {
      if (post == "HOD") {
          cout << "\nHOD Details " << endl;
       display();
      }
   }
};
```

```
int main() {
    Staff s[5];
    for (int i=0; i<5; i++) {
    cout << "\nEnter details for staff "<<i+1<<":\n";
    s[i].accept();
    }
    cout<<"\n List of HODs:\n";
    for (int i=0; i<5; i++) {
     s[i].displayHOD();
    }
    return 0;
}
```

output:
list of HODs:
HOD details
Name: Mr. Sharma
Post: HOD

HOD details
Name: Mr. Khan
Post: HOD

HOD details
Name: Ms. Meena
Post: HOD

2818

Experiment NO. 3:-

Q) #include <iostream>
#include <string>
using namespace std;
class book {
    string book_title;
    string author_name;
    int price;
public:
    void accept() {
    cout << "enter book title: ";
    cin.ignore();
    getline(cin, book_title);
    cout << enter author name: ";
    getline(cin, author_name);
    cout<<"enter price: ";
    cin >> price;
    cout << endl;
    }
    void display() {
    cout << "Book title: " << book_title << endl;
    cout << "Author name: "<< author_name<<endl;
    cout<< "price: "<<author_name<<endl;
    cout<<endl;
    }
};
int main() {
    book b;
    b.accept();
    Book *ptr = &b;
    ptr -> display();
    return 0;
}

output:
enter book title: notebook
enter author name: Swani
enter price: 888
Book title: notebook
Author name: Swani
enter price: 888

Student name and roll no.

① #include <iostream>
using namespace std;

class Student {
int roll_no;
int percentage;
public:
void accept () {
cout << "enter roll.no: ";
cin >> this -> roll_no;
cout << "enter percentage: ";
cin >> this -> persentage;
}
void display ()
cout << "Roll no: " << Roll no.: " << this -> rollno << endl;
cout << "percentage: " << this -> percentage << "%" << endl;
cout << endl;
}
};
int main () {
Student s;
S.accept ();
S.display ();
return 0;
}

Out Put:
enter roll no: 44
enter percentage: 67
Roll no: 44
Percentage: 67%.

② WAP to declare a class 'Student' having data members roll-no and percentage. Using 'this' pointer invoke members functi to accept and display this data for one object of the class.

→ #include <iostream.h>
using namespace std;
class student {
class name;
int roll_no;
Public:
void accept () {
cout << wirte it please";
cin >> this → name → this → roll;;
}
void display () {
cout << write it please";
cin >> this → name → this → roll no << ;
}
};

int main () {
student S1
S1 . display ();
return 0;
}

Q) WAP to demonstrate the use of nested class.

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

  class student {
  private:
    string name;
Public:
    class Address {
private:
    string city;
    string country;
public:
  Address (const string& city:" ", const string&
          country = " ")
        : city (city, country (country) {}}

void accept-address () {
    cout << " enter City:  ";
    getline (cin, city);
    cout <<"enter country: ";
    getline ( country; )
  }
  void display-address () const {
    cout << "Address:" << city << ", " << country << endl;
}
Address student_ address;
Student (): name (" "), student_address (" "," ")}}
```

```
void accept-data () {
cout << " enter student data" << endl;
cout << " enter student name: ";
getline (cin, name);
student_address.accept_address ();
}
 void display_data () const {
cout << "\n Display student data " << endl;
cout << " Student Name: " << this → Name << endl;
this → student_address.display-address ();
 }
};
  int main (){
  Student student;
  student.accept-data ();
Student. display-data ();
cout << "\n creating a standalone Address object" << end

student: address new-address ("london", "UK");
new.address.display-address ();

  return 0;
}
```

@
28/8

PAGE NO.
DATE :

Exp. 4:

Q1) Write a C++ program to resolve the following problems statement
1) Swap 2 numbers from same class using object as functions arguments. Write swap function as member function.

```
#include <iostream>
```

→

```
Using namespace std;
Class num {
public:
int n1, n2;
void accept (){
    cout<<"enter first no.:"<<endl;
    cin >> n1;
    cout << "enter second no:" <<endl;
    cin >> n2;
}
void display (){
cout <<"Num1: " << n1 << endl;
cout<<"Num2: " << n2 << endl;
}
Void swap (num ){
int temp = n1 = n2; n2 = temp;
cout <<"Swapped: In first no.: "<< n1 << /n
        second no."<< n2;
    }
};
int main (){
nu.accept ();
nu. display ();
nu. swap (nu);
return 0;
}
```

O/P:

```
enter first no:
5
enter second no:
9
num1 : 5
num2 : 9
Swapped: num1 : 9
         num2 : 5
```

3) Swap two number from same class using concept of friend function.

→
```
#include <iostream>
using namespace std;
class num {
public:
    int n1, n2;
    void accept (){
    cout <<"enter first number:" << endl;
    cin >> n1;
    cout <<"enter second no.:" <<endl;
    cin >> n2;
    }
    void display (){
    cout <<"Num1:" <<n1 <<endl <<"Num2:"
        << n2 << endl;
        }
    }nu;
Void swap (num) {
int temp = nu.n1;
nu.n1 = nu.n2;
cout <<"swapped \n first :" << nu.n1 <<"\nSec
    -nd:" nu.n2 << endl;
} int main (){
    nu.accept ();
    nu.display ();
    swap. (nu)
    return 0;
}
```

```
O/P:
enter first no: 5
enter second no: 9
num 1 : 5
num 2 : 9
 First : 9
 Second : 5
```

3) Swap two numbers from different class using friend function.

→
```
#include <iostream>
using namespace std;
  class num1 {
Public : int n1;
        void accept () {
         cout <<"enter first no: " <<endl;
         cin >> n1; }
         void display () { cout <<"Num 1:" <<n1 << endl;
} nu1;
  class num2 {
  public : int n2;
   void accept () {
   cout <<"enter second no: " <<endl;
   cin >> n2; }
   void display () { cout <<"Num 2: " << n2 << endl;
  }nu2;
    void swap (num1, num2) {
        int temp = nu1.n1;
        nu1.n1 = nu2.n2;
        nu2.n2 = temp;
   Cout <<"Swapped: \n first :" << nu1.n1 << \n Second:"
        << nu2.n2 << endl;
}
 int main () {
  nu1. accept ();
  nu2. accept ();
  nu1. dsp ();
  nu2. dsp ();
   Swap (nu1, nu2);
  return 0;
}
```

```
O/P:
enter first no: 5
enter Second no: 9
 Num 1: 5
 Num 2: 9
Swapped : First : 9
          second : 5
```

4) creat 2 classes result and Result 2 which stores the marks of the students. Read the value of make fcn both the class objects & compute avg.

```
#include <iostream>
Using namespace std;
class Result {
public:
int r1;
void accept (){
cout<<"enter first result: "<<endl;
cin >>n1; }
}res1;
Class Result 2{
public:
int r2;
Void accept (){
cout <<"enter second result: "<<endl;
cin >> n2; }
} res 2;
float avg ( result 1, Result 2) {
    return ( res1.re + res2.r2)/2;
 }
 int main () {
   res 1 . accept ();
   res 2. accept ();
int main (){
   res 1 . accept ();
   res 2. accept ();
 float avg = avg
```

```
int main () {
 res 1 . accept ();
 res 2 . accept ();
 float avg = avg (res 1, res 2);
 cout << " Average: " <<avg;
    return 0;
 }
```

O/P:
enter first result: 95
enter second result: 99
Avg 97.

5) Find the greatest number among 2 numbers from 2 different classes using friend function.

→

```
#include <iostream>
 using namespace std;
 class num 1 {
public : int n1;
void accept (){
 cout<<" enter first number:"<<endl;
} nu1;
 class num 2{
 public: int n2;
 void accept (){
 cout<<" enter second number:"<<endl;
  cin >> n2; }
} nu2;
int glum(int n1, int n2) { return (n1> n2) (n1:n2); }
```

```
int main () {
nu1. accept ();
nu2 . accept ();
int gr = gnum(nu1 . n1; nu2 . n2);
cout << "Greatest :" << gr << endl;
return 0;
}
```

O/P :-

enter first no : 5
enter second no : 9
Greatest = 9

6) Create two classes, A & B with a private integer, write a pr friend function sum() that can access private data from both the classes and return the sum

```
#include <iostream>
using namespace std;
class ~~out~~ class B;
class class A;
 int A;
Public;
 void accept () {
 cout << "enter value from A: ";
 cin >> A;
}
 Friend int sum (class A OA, class B =B);
} na;
 class class B {
     int B;
Public;
 void accept () {
    cout << "enter value for B: ";
     cin >> B;
  }
 Friend int sum (class A OA, class B OB);
} nb;
 int sum (class A OA, class B OB) {
  return OA . A + OB . B;
 }
 int main () {
 na . accept ();
 nb . accept ();
cout << "sum:" << sum (na, nb) << endl;
 return 0;
}
```

O/P:-

enter value for A: 3
enter value for B: 6
Sum: 9

7) WAP with a class number that contains a private integer. Use a friend function swap numbers (number &, number &) to swap two prive values of two number objects.

```
#include <iostream>
 Using namespace std;
 class Number {
    int num;
 Public;
    Void accept () {
    cout << "enter value: ";
    cin >> num;
    }
    void display () {
    cout << " value " : < num << endl;
}
friend void swap numbers (Number & num1,
                          Number & num 2);
}n1, n2;
void swapnumbers ( Number &, number &){
  int temp = n1. num;
  n1. num = n2. num;
  n2. num = temp;
}
int main () {
  n1. accept ();
  n2. accept ();
cout << "Before swap : " << endl;
n1. display ();
n2. display ();
Swap numbers (n1, n2);
cout << " After swap : " << endl;
  n1. dsp ();
  n2. dsp ();
  return 0;
}
```

O/P :-

```
enter value : 3
enter value : 6
Before Swap:
        value : 6
        value : 3
After swap:
        value: 6
        value: 3
```

8) Define two classes Box & cube, each having pvt volume, write a friend function findgreater (Box, cube) that determines which obj. has a larger volume.

```
#include <iostream>
 using namespace std;
 class cube;
 class Box; {
  double v;
public:
  void accept () {
  double w, l, h;
  cout << "enter length, width & height of Box: ";
  cin >> l >> w >> h;
```

```
    v = l * w * h;
} friend void findgreater (Box b, Cube C);
} b;
  class Cube {
    double v;
Public:
   void accept (){
      double s;
cout << "enter side length for the cube: ";
   cin >> s;
   v = s * s * s;
} friend void findgreater (Box b, Cube C);
} c;
void findGreater (Box, Cube) {
 string res = (b.v > c.v)? "Box": ((c.v > b.v)?
 cout << "greater volume"; << res << endl;
}
int main (){
  b.accept ();
  c.accept ();
  find greater (b, c);
  return 0;
}
```

o/p:

```
enter l, w, h for box: 3, 4, 3
enter side length for cube: 3
  greater volume : Box.
```

9) Create a class complex with real and imaginary parts as a pvt member. use a friend function to add two complex no.'s & return the result as a new complex obj.

```
#include <iostream>
using namespace std;
   class complex {
    double r, i;
  Public:
  void accept (){
  cout << "enter real & imaginary parts: ";
  cin >> r >> i;
  }
  void display (){
  cout << r << " + " << i << "i" << endl;
} friend complex add (complex c1, complex c2);
} c1, c2;
 complex add (complex, complex) {
    complex sum;
    sum.r = c1.r + c2.r;
    sum.i = c1.i + c2.i;
    return sum;
 }
 int main (){
  c1.accept ();
  c2.accept ();
complex.sum = add (c1, c2);
cout << "The sum is : ";
  sum.display ();
  return 0;
 }
```

(6) Create a class student with pvt data members: name & three subject marks. friend function calculate (Avg) (std) Avg. marks

→
```
#include <iostream>
 using namespce std;
 class student {
 String n;
 int n[3];
public:
 void accept (){
  cout<<"enter student name: ";
  cin>>n;
  cout<<"enter marks for 3 sub : ";
  cin>>m[0]>>m[1]>>m[2];
} friend void calculateAvg (student s);
{ s;
  Void calculate Avg (student){
  double avg = (s.m[0]+s.m[1]+s.m[2])/3.0;

  cout<<"student: "<<s.n<<endl;
  cout<<"Avg marks: "<<avg<<endl;
}
 int main (){
   s.accept ();
   calculate Average ();
   return 0;
}
```

Exp: 5:

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

```
#include <stdio.h> <iostream>
using namespace std;
class calcu {
 int sum;
 Public:
  calcu () {
  int n=10;
   sum = 0;
   for (i=0; i<=n; i++){
   sum+=i;
  }
}
 calcu (calcu & x) {
  sum = x.sum
  cout << "sum" << sum";
  }
};
int main (){
  calcu c1;
   calcu c2 (c1);
return 0;
}
```

O/P:

sum of numbers from 1 to 10 is 55.

2) WAP to declare a class "student" having data members as name & percentage. Write a constructor to initialize these data members. Accept & display data for one student.

```
#include<iostream>
#include<string>
using namespace std;
class student{
private:
string name;
float percentage;
public:
   student(){name="N/A"; percentage=0.0;}
   void setStudentData(string sName, float sPercentage)
                {name=sName; percentage=sPercentage;}
   void displayData(){cout<<"Student Name: "<<name<<endl;
                cout<<"Percentage: "<<percentage<<"%"<<endl;}
};
int main(){
string studentName; float studentPercentage;
student student1;
cout<<"Enter student's name: "; getline(cin,studentName);
cout<<"Enter student's percentage: "; cin>>studentPercentage;
student1.setStudentData(studentName, studentPercentage);
cout<<"\nStudent Data\n";
student1.displayData();
return 0;
}
```

(Default)

(Parametrized constructor)

```
#include<iostream>
#include<string>
using namespace std;
class Student{
private:
   string name;
   float percentage;
public:
   Student(string sName, float sPercentage){name=sName; percentage=sPercentage;}

   void displayData(){cout<<"Student Name: "<<name<<endl;
                cout<<"Percentage: "<<percentage<<"%"<<endl;}
};
int main(){
string studentName;
float studentPer;

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

cout<<"enter student percentage: ";
cin>>studentPer;

Student S1(studentName, studentPer);
S1.displayData();

return 0;
}
```

3) copy constructor:

→
```
#include <iostream>
using namespace std;
class calculator {
    private:
    int sum;
    public:
    calculator ()
    {
    int n=10;
    sum=0;
    for (int=0; i<=n, i++){
        sum+=i;
    }
}
calculator (calculator &x)
    {
    sum=+sum;
    cout<<"sum="<<sum;
    }
};
int main(){
 calculator c1;
 calculator c2 (c1);
return 0;
}
```

PAGE NO.
DATE :

* default constructor:

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

class student {
  string name;
   float percentage;
    public:
       student (){
        name = "Swani";
        percentage=100;
  cout<<"Name: "<<name<<"\npercentage
                    <<Percent<<"%"<<en
    }
};
int main (){
 student s1;
 return 0;
}
```

4) WAP to demonstrate constructor over loading;

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

class student {
  int roll;
  string name;
 public:

  student () {
   name = "Unknown";
   roll = 0;
  }
  student (string n){
   name = n;
   roll = 0;

  student (string n, int r){
   name = n;
   roll = r;

  void display (){
   cout<<"Name: "<<name<<endl;
   cout<<"Roll number: "<<roll<<endl;
   }
};
```

```
int main (){
  student s1;
  student s2 ("Swani");
  student s3 ("Tanisha", 18);

  s1.display();
  s2.display();
  s3.display();
  return 0;
}
```

1/1/1

PAGE NO.
DATE :

experiment-6:

Q1) WAP to export multilevel inheritance,

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

class department {
protected:
    string dname;
};
class student : protected department {
protected:
    string sname;
    int roll;
};
class marks : protected student {
    int m1, m2, percentage;
public:
  void accept() {
  cout << "enter department: " << endl;
  cin >> dname;
  cout << "enter name: " << endl;
  cin >> name;
  cout << "Entermarks: " << endl;
  cin >> m1;
  cout << enter marks: " << endl;
  cin >> m2;
```

```
Void calculate (){
  int per = (m1+m2)/2
  cout << "Department : " << dname << endl;
  cout << "Name: " << sname << endl;
  cout << "Percentage: " << percentage << endl;
 }
};

int main (){
 marks m;
 m. accept ();
 m. calculate ();

 return 0;
{
```

Q2) multiple inheritance,

```
#include <iostream>
using. namespace std;
class department{
 protected:
      string dname;
};
 class student {
 protected:
     string sname;
     int roll;
};
class marks: protected department, protected student{
 int m1, m2, percentage;
Public:
  void accept (){
  cout<<" enter department: " << endl;
  cin >> dname;
  cout<<" enter name: " << endl;
  cin>> sname;
  cout<<" enter marks 1: " << endl;
  cin >> m1;
  cout<<" Enter marks 2: " << endl;
   cin >> m2;
} void calculate (){
int per= (m1+m2)/2;
cout<<" Department: " << dname << endl;
cout<<" Name: " << sname << endl;
cout<<" percentage: " << per << endl;
} int main (){
   marks m;
   m. accept ();
 } m. calculate ();
```

03) hierarchical inheritance:

```
#
class Person {
 protected:
 string name;
 int age;
 public:
 void acceptPer(){
cout<<"enter name: "<<endl;
cin>>name;
cout<<"enter age: "<<endl;
cin>>age;
};
class student : public Person{
int roll;
float per;
Public:
void acceptstud(){
cout<<"enter roll number: "<<endl;
cin>>roll;
cout<<"enter Percentage: "<< endl;
cin>>per;
}
void displaystud(){
cout<<"Name: "<<name<<endl;
cout<<"Age: "<<age<<endl;
cout<<"Roll no. "<<roll<<endl;
cout<<"Percentage"<<per<<endl;
 }
};
```

```
void displaystaff(){
cout<<"Name: "<<"name"<<endl;
cout<<"emp Id "<<emp-id<<endl;
cout<<"Age"<<age<<endl;
cout<<"Subject"<<sub<<endl;
}
};

int main(){
student s;
staff t;

s.acceptPer();
s.acceptstud();
s.acceptPer();
s.acceptstaff();

s.displaystud();
t.displaystaff();

return 0;
}
```

(4) hybrid inheritance:

```
#
class college {
protected:
 String cname;
};
class employee: protected college{
    protected
    string emp_name;
    int id;
};
class staff: public employee{
    string ~~name~~ emp_name;
    int dept Id;
    Public:
void accept Emp(){
cout << " enter clg name" << endl;
cin >> cname;
cout << enter emp.name << endl;
cin >> emp_name;
cout << enter ID << endl;
cin >> id;
cout << enter staff name << endl;
cin >> name;
cout << dept ID :" << endl;
cin >> dept Id;
}
```

```
class student: protected college {
  string stu-name;
  int roll;
  public
  void accept Std(){
cout << " enter clg name: " << cname << endl;
cin >> cname;
cout << " enter name: " << name << endl;
cin >> stu-name;
cout << " enter roll number: " << endl;
cin >> roll;

 void display
cout << " college: " << cname << endl;
cout << " student name: " << stu-name << endl;
cout << " roll Number: " << roll << end;

int main (){
staff staff 1;
staff 1 . accept Emp();
Staff 1 . display Emp();

Student stud1;
 Stud 1 . accept Stud ();
 Stud 1 . display Stud ();

 return 0;
}
```

PAGE NO.
DATE :

exp : 7 :

Q1) function overloading; area of laboratory & area of classroom. (rect & sq).

```
#
class Area {
 public:
 float calculate (float length, float breadth){
  return length * breadth;
  }
 float calculate (float side){
  return side * side;
 }
};

int main (){
Area a;
cout<<" Area of laboratory "<<a.calculate (10, 5)<<endl;
cout<<" Area of square : "<<a.calculate (5)<<endl;
}
```

Q2) WAP using function over loading, sum of 5 float values & 10 int values.

```
#
 class sum {
 public:
```

→

```
int total (int a [], int n){
int s=0;
for (int i=0; i<n; i++)
  s += a[i];
 return s;
}
float total (float a[0]; int n){
 float s=0
  for (int i=0; i<n; i++)
    s += a [i];
   return s;
}
};
int main (){
Sum s;
int marks [10] = { 45, 56, 67, 78, 8990, 76, 88, 92 85}
float grades [5] = { 9.2, 8.7, 9.5, 8.9, 9.0}

cout << "sum of 10 std marks : " << total(marks, 10);
cout << "sum of 5 std grade : " << stotal(grades, 5);

return 0;
}
```

Q3) WAP, compile time & implement Unary operation used with obj so that the numeric data member of the class is negated.

```
#
class teacher{
    int experience;
public:
   Teacher (int e){
      experience = e;
   }
   void display (){
cout << "experience : " experience "year" << endl;
}
 void operator - (){
    experience = - experience;
  }
};

int main (){
  Teacher t1 (10);
  t1.display ();
  -t1;
cout << "After negation = ";
t1.display ();

return 0;
}
```

Q4) Unary ++ operator, (incremented)

```
#
class student {
  int count;
public:
 student (int c=0){
   count=c;
 }
 void operator++(){
      ++count;
   }
  void operator (int k) {
      count++;
  void display {
 cout<<"student count : "<<count<<endl;
};
};
 int main (){
student s1 (50);
cout<<"before increment<<endl;
s1 display (),
++s1;
cout<<"After pre increment<<endl;
s1 - dsplay ()
s1 ++;
cout<<"after Post incremen<<endl;
s1 . display ();
return 0;
}
```

Q5)

exp : 8 :

```
# include <iostream>
# include <string>
using namespace std;
class combine {
   string str;
public:
    combine (string s=" ")
      str =s;
    }
combine operator+ (combine &obj ){
return combine (str + obj . str),
}
 void display (){
  cout << str <<endle,
 }
};
int main (){
 combine s1 ("xyz"), s2("pqr"), s3;
 s3 = s1 + s2;
 cout<< "concatenated string : ";
 s3. display ();

}
```

Q2) #

```
class I login {
protected;
string name, password;
Public:
virtual void accept (){
cout << "Name << name << endl;
cout << "Password << password << endl;
}
};
class email login : public I login {
    string email;
Public:
  void accept () override {
cout << center email ID: ;
cin >> email;
I login :: accept ();
}
void disp () override {
cout << email login details" << endl;
cout << email ID << email id << endl;
I login :: display ();
}
};

class member ship login : public I login {
string member ID;

public:
  void accept () override {
      disp
```

```
cout << "In membership login Details" << endl;
cout << "membership ID, " number ID. << endl;
  I login :: display ();
}
};

int main(){
  I login login;
  email login ;
  membership login m;

login = & e;
login → accept ();
login → display ();

login = & m;
login → accept ();
login → display ();

return 0;
}
```

PAGE NO.
DATE :

Exp :- 9:

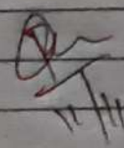
Q1)
```
# include <iostream>
# include <fstream>
using name space std;

int main (){
if stream infile ("first.text");
of stream outfile ("second.txt");

if (!inFile){
cout<<" error opening First-txt "<<endl;
 return 1;
}
char ch;
while (infile.get (ch)){
 outfile.put (ch);
}
cout<<"file copied successfully"<<endl;

infile.close();
outfile.close();
return 0;
}
```

02)
```
#
int main (){
if stream file (first.txt");
if (!file){
cout << "error opening file" << endl;
return 1;
}
char ch;
int digits = 0; spaces = 0;

while (file.get (ch)) {
if (is digit (ch))
digits ++
else if (isspace (ch))
space ++;
}
cout << "Digits: " << digits << endl;
cout "spaces: " << space << endl;
file.close();
return 0;
}
```

03)
```
# include <iostream>
# include <fstream>
# include <string>
using name space std;

int main (){
if stream file ("hxs.txt");
if (!file){
cout << Error << endl;
return 1;
}

String word;
int count = 0;

while (file >> word)
count ++;
cout << "Total words: " << count << endl;

file.close()
return 0;
}
```

Q4) #

```
int main () {
if stream file (" first.txt );
if (!file) {
cout << " error " << endl;
 return 1;
}
string word, target;
int count = 0;
(out << " enter word to count: " << endl;
cin >> target;

while (file >> word ){
if ( word = target )
   count ++;
}
cout " Occurrence of " << target << " : "
      << count << endl;

file close ();

}
```



exp: 10:

a) #

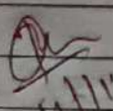
```
template <class T>
T sum (T a[], int n) {
 T s=0;
for (int i= 0; i<n; i++)
   s+ = a[i];
 return s;
}
int main(){
 int a [s];
for (int i=0; i<s; i++)
 cin >> a[i];
 cout << sum (a, s);
}
```

b) #

```
# include <iostream>
# include < string>
using namespace std;
template < class T>
T square (T x ) {
   return x*x;
}
template <>
string square (string s ){
 return s+s;
}
int main (){
cout << Square (4) << endl;
cout << square (string (" Hi ") );
}
```

c) #
```
template<class T>
class calc{
T a, b;
public:
void get() {cin >>a>>b;}
 void add {cout<<a+b;}
};
int main (){
calc<int>c;
c.get ();
c.add ();
}
```

d) #
```
template < class T>
class stack {
T s[10];
int top;
Public:
Stack() { top=-1;}
void push (T x) {s[++top] = x;}
void pop () {if (top>=0) top --;}
void displayy() {for(int i=0; i<=top; i++)
cout<<s[i]<<" ";}
};
int main(){
Stack<int>s;
S.push(1);
S.push(2);
S.push(3);
 S.pop()
S.displayy();
```

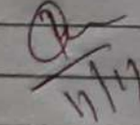


exp 11:

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

int main(){
 vector<int>v;
 for(int i=10; i<=30; i+=10)
  v.push-back(i);
int x;

cin>>x;
 cout<<v[x]<<endl;
 int n;
  cin>>n;
for(int i=0; i<v.size(); i++)
  v[i]*=n;
 for (int i=0; i v.size(); i++)
  cout<<v[i]<<" ";
};
```

without iterators:

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

int main()
{
  int n;
  cout<<'enter scalar value:"<<endl;
  cin>>n;
 vector<int> v={1,2,3,4,5};
 cout<<'initial vector:"<<endl;
 for(int i=0; i<5; i++)
{
 cout<<v[i]<<" "<<endl;
}
 }
 for int i=0; i<5; i++)
{
  v[i]=v[i]*n;
}
 (out<<'new vector:"<<endl;
 for (int i=0; i<5; i++)
  {
   cout<<v[i]<<" "<<endl;
  }
 return 0;
}
```



exp 12:

a)
```
#include<iostream>
#include<stack>
Using namespace std;

int main(){
 stack<int>S;
 S.push(10);
 S.push(30);
 S.push(30);
 while(!S.empty()){
 Cout<<S.top()<<" ";
  S.pop();
  }
}
```

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

int main(){
 queue<int>q;
 q.push(10);
 q.push(20);
 q.push(30);

while(!q.empty()){
  cout<<q.front()<<" ";
    q.pop();
  }
}
```

PAGE NO.
DATE :

★ Calculator:

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

template<<class T>
class Calc{
  T a,b;
 public:
  void get(){ cin >>a>>b;}
T add(){return a+b};
T sub(){
T mul(){
T div(){
T minv(){
[maxv(){
};
int main(){
 int ch;
cin >> ch;
cal <double> c;
 c.get();
switch(ch){
case1: cout << c.add(); break;
case2: cout << c.sub(); break;
case3: cout << c.mul(); break;
case4: cout<< c.divv(); break;
case 5: cout<< c.minv(); break;
case 6: cout<<c.c.maxv(); break;
default: cout<<"invalid"
  }
}
```