Experiment : 4

A) 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;
  str 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;
}
```

3. Write C++ code to create a class book having data members as book_name, book price, book pages. Accept the data from two books and display the name of book having greater price

⇒
```
class book
{
 Pubic:
 string b_name
 int b_price;
 int b_pages
 Pubic
 void accept ()
 {
 cout << "enter book name book price
        & no. of pages:",
 cin >> b_name >> b_price >> b_page
 }
 voide 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 in to total seconds and display

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

Void display seconds () {
```

```
cout << "Total seconds : " << to_seconds()
     << endl;
}
};

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

Q 15/7

Experiment - 2

1) WAP to declare a class 'city' having data members as name and population. Accept this data for 5 cities having population.

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

class city
{ public:
    string name;
    int population;
     public:
    void accept ()
  {
Cout << "enter the name of city:" << en
Cin >> name;
Cout << "enter population:" << endl;
Cin >> population;
   }
   void display ()
{
Cout << "Name:" << name << endl;
Cout << "Population:" << population << endl;
   }
 }
  City c[5]
  int main ()
  {
```

```
for (i=0; i<5 ; i++)

};
int max = 0;
for(i=0; i<5; i++)
{
 if ( [i].population c[max].population)
    {
      max = ii
    }
     }

Cout << "The city with maximum population."
      << endl;
 c[max] . display ();

 return 0;

};
```

Output :-

enter name of city :
city 1: Pune
Population : 23457

city 2: Katraj
population : 23574

city 3: Amritsar
population : 34780

City 4: Mahabaleswar
population : 23008

City 6: Vai
population: 21007

The city with maximum population:
34180

Output :-

enter name of city :
city 1 : Pune
Population : 23457

city 2 : Katraj
population : 23574

city 3 : Amritsar
population : 34780

City 4 : Mahabaleswar
population : 23008

City 6 : Vai
population : 21007

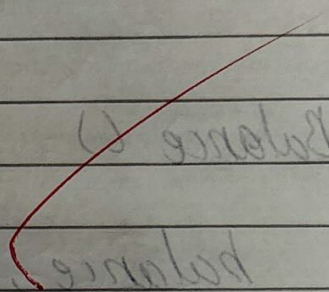
The city with maximum population :
34180

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

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

Class account
  {
     int accno.;
     int balance;
  public:
     void accept ()
  {
Cout << "Enter account no.:" << endl;
Cin >> accno.;
Cout << "Enter balance :" << endl;
Cin >> balance;
  }
   void display ()
  {
Cout << "Account number:" << accno << "Balance:"
                          << balance << endl;
  }

  void addinterest [ ]
  {
   if (balance >= 5000)

   {
```

```
int interest = balance * 0.10;
    balance += interest;
    cout << "Interest of 10% added". << balance
                                    << endl;

  }
  else
{
 cout << "Balance is less than 5000, no
          interest is added" << endl;
  }
}
int getBalance ()
 {
  return balance;
 }
 };
 int main ()
 {
  account accounts [10];
   in numACCounts;

cout << "Enter no. of accounts (max 10)! ";
 cin >> num Accounts;
 for (int i = 0; i < numAccounts; i++)
 {
  cout ("\n Account " << (i+1) << ":" << endl;
  accounts [i] . accept ();
  }

cout << "\n Account Display with 10% interest" <<
                                    endl;
```

```
for (int i=0; i<numAccounts; i++)
  {

cout<<"\n Account"<<(i+1)<<":"<<endl;
accounts[i].display();
accounts[i].addinterest();
  }
    return 0;
  }
```



~~cout <<~~ "

3] WAP to declare a class 'staff' having data members as book - title, author, name & price. Accept & display the info. for 1 object using pointer of that object.

```
#include <iostream>
using namespace std;
class staff
{
string name;
string post;
public:
void accept()
{
cout << "Enter name:";
cin >> name;
cout << "Enter post:";
cin >> post;
}
void display()
{
cout << "Name:" << name << "Post:" << post <<
endl; }
}
string get Post()
{
return post;
}
string get Name()
{
return name;
}
```

```
int main ()
{
 staff s[5];
bool found HOD = false;

for (int i=0; i<5; i++)
{
cout << "Staff member " << (i+1) << ": " << endl;
s[i].accept();
}
for (int i=0; i<5; i++)
{
if (s[i].getPost() == "HOD")
 {
cout << "HOD is :" << s[i] accept () << end l;
  found HOD = True;
   break;
   }
   }
if (!found HOD)
 cout << "HOD not found " << end l;
return 0;

}
```

Output

Name :- Krishana
Post :- HOD

Name :- Sara
Post : staff

Name :- Aishni
Post :- staff

Name :- ~~Administrator~~ ~~HOD~~ Neel
Post :- Administrator

Name : Swati
Post : Head

HOD is :- Krishna

5/8/25

Experiment - 3

1] WAP to declare a class 'book' having data members as book-title, author-name and price. Accept and display the information for one object using a pointer to that object.

```
#include <iostream>
#include <string>
using namespace std;
{
 string name;
 string author;
  in price;
 public:
 void accept() {
 cout << "Book name:" << endl;
 Cin >> name;
 Cout << "Name of Author:" << endl;
 Cin >> author;
 Cout << "Book price:" << endl;
 Cin >> price
 }
   void display () {
Cout << "Book name:" << "name << ". Name
 of author:" << author << "Book price:" << price
   }
   };
```

```
int main ()
{
books B;
books *p;
p = &B
p -> accept ();
p -> display ();

return 0;
}
```

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

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

Class student
{
   int roll_no;
   string name;
   flodt marks[5];
   float total-marks;
   float percentage;
   public:
   void accept ()
Cout << "enter Roll no.:" << endl;
Cin >> this -> roll no;
Cout << "Name :" << endl;
Cin >> this -> name;
Cout << "enter marks for 5 subjects:"
        << endl;
total_marks =0;
for (int i=0; i<5 ; i++)
cout << "Subject " << (i+1) << ":";
Cin >> marks [i]
total_ marks += marks [i];
 }
```

```
calculate percentage ();

}
void calculatepercentage ()
 {
 void display () {
cout << "Roll no.:" << roll_no. endl;
cout << "Name " << name << endl;
cout << "Total marks :" << total_marks <<
                                  / 500" << endl
cout << "percentage:" << "%!" << percentage << "%."
  endl;

}
 }
int main () {
 student s
 s1 . accept ();
 s1 . display ();

  return 0;
```

3) Write a program to demonstrate the use of nested class

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

class Student {
  int rollNo;
  string name;

public :
    void getstudentdata () {
    cout << "Enter Roll number : ";
    cin >> rollNo;
   cin.ignore ();
   cout << " Enter Name : ";
   getline (cin, name);
   }
Void displayStudentData () {
    cout << "\n Student Details :";
    cout << "\n Roll Number :" << rollNo;
    getline (cin, name);
  }
  void displayStudentData ()
  cout << "\n Student Details : ";
  cout << "\n Roll Number :" << rollNo;
cout << "\n Name: " << name << endl;

  }
```

```
Class marks {
    int m1, m2, m3;

public:
    void getMarks () {
    cout << "Enter marks for 3 subject:";
    cin >> m1 >> m2 >> m3;

}
Void displaymarks (){
 Cout << "\n Marks Obtained:";
 cout << "\n Subject 1:" << m1
Cout << "\n Subject 2:" << m2
Cout << "\n Subject 3:" << m3
 int total = m1 + m2 + m3;
 float percentage = total/3.0
cout << "\nTotal Marks:" << total;
cout << "\nPercentage:" << percentage << "%"
     << endl;
  }
 };
};

Student : Marks m;
m.getMarks();      m.getMark
m.displayMarks();

int main () {
```

```
Student s;
s.getStudentData();
s.displayStudentDat

Student::Marks m;
m.getMarks();
m.displayMarks();

return 0;

}
```

5/8/25

Experiment - 4

Q) Write a C++ program to resolve the following problem statement and show output.

1. Swap 2 numbers from same class using object as function argument. Write swap function as member function

Code:-

→
```
#include <iostream>
using namespace std;
class num {
public:
    int n1, n2
void accept() {
cout << "Enter first number :" << endl;
cin >> n1;
cout << "Enter second number :" << endl;
cin >> n2;
}
void display() {
  cout << "Num 1:" << n1 << endl;
  cout << "Num 2:" << n2 << endl
}
  Void Swap (num) {
    int temp = n1; n1 = n2; n2 = temp;
  cout << "Swapped : \n First number : " << n1 << "\n Second number" << n2
  }
} nu;
```

```
int main () {
   nu. accept ();
   nu. display ();
   nu. swap (nu);
   return 0;
}
```

Output

Enter first number : 5
Enter second number : 9

Num 1 : 5
Num 2 : 9

Swapped

First number : 9
Second number : 5

2] Swap two numbers from same class using concept of friend function.

Code:

```
#include <iostream>
using namespace std;
class num {
public:
    int n1, n2;
    void accept () {
    cout << "Enter first number :" << endl;
    cin >> n1;
    cout << "Enter second number:" << endl;
    cin >> n2;
    }
    void display () {
     cout << "Num 1:" << n1 << endl;
     cout << "Num 2:" << n2 << end2;
    }
    } nu;
void swap (num) {
 int temp = nu.n1;
 nu.n1 = nu.n2;
 nu.n2 = temp;
 cout << "Swapped: \n First : " << nu.n1 << " \n
                                  second : " nu.w << endl
}
int main () {
 nu.accept ();
 nu.display ();
 swap (nu);
 return 0;
}
```

Output :-

Ente first number : 5
Ente second number : 9

Num 1 : 5
Num 2 : 9
Swapped

First : 9
Second : 5

3. Swap two number from different class using friend function

Code:

```
#include <iostream>
using namespace std;
class num1{
public:
        int n1;
  void accept (){
    cout<<"Enter first number: "<<endl
    cin>>n1;}
  void display (){cout<<"Num 1: "<<n1<<endl;}
  }nu1;
Class num2{
public:
        int n2;
    Void accept (){
      cout<<"Enter second number: "<<endl
      Cin >> n2;}
void display (){cout<<"Num 2: "<<n2<<endl;}
  }nu2;
Void swap (num1, num2)
      int tem = nu1.n1;
      nu1.n1 = nu2.nu2
     nu2.n2 = temp;
Cout<<"Swaped: \n First"<<nu1.n1<<\n Second:
                    <<nu2.n2<<endl;
}
int main (){
 nu1.accept ();
nu2. accept ();
nu1. display ();
```

```
nu2.display();
Swap(nu1, nu2);

return 0;

}
```

Output:-

```
enter first number : 5
enter second number : 9

Num : 5
Num : 9

Swapped

first : 9
second : 5
```

4. Create 2 class Result1 and Result2 which stores the marks of the students Read the value of marks both the class objects & compute avg.

Code:

```
#include <iostream>
using namespace std;
class Result1{
public:
    int r1;
    void accept(){
cout<<"Enter first result:"<<endl;
cin >> r1;}
}res1;
class Result2{
public:
    int r2;
    void accept(){
cout<<"Enter second result<<endl;
cin >>r2;}
}res2;
float avg(Result1, Result2){
    return (res1.r1+res2.r2)/2;
}
int main(){
    res1.accept();
    res2.accept();
    float aveg = avg(res1, res2);
cout<<"Average:"<<avg;

    return 0;
}
```

Output

Enter first result:
95

Enter second result;
99

Avege :- 97

5. Find the greatest of two numbers using 2 classes [illegible]

Code:

```
#include <iostream>
using namespace std;
class num1 {
public:
    int n1;
    void accept() {
cout << "Enter first number : "<<endl;
cin >> n1;
};
class num2 {
public:
    int n2;
    void accept() {
cout << "Enter second number : "<<endl;
cin >> n2;
};
int main() {
num1 a1;
num2 a2;
a1.accept();
a2.accept();
int g = greater(a1.n1, a2.n2);
cout << "Greater : "<< g << endl;
return 0;
}
```

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

Code:

```
#include <iostream>
using namespace std;
class num1 {
public:
    int n1;
    void accept() {
cout << "Enter first number: " << endl;
cin >> n2; }
} nu1;
class num2 {
public:
    int n2;
    void accept() {
cout << "Enter second number: " << endl;
cin >> n2; }
} nu2;
int main() {
  nu1.accept();
  nu2.accept();
  int gr = gnum(nu1.n1, nu2.n2);
  cout >> "Greatest: " << gr << end;
    return 0;
}
```

Output :-

Enter first number : 5

Enter second number : 9

Greatest : 9

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

```
#include <iostream>
using namespace std;
class ClassB;
class ClassA {
    int A;
public:
void accept() {
  cout << "Enter value for A: ";
  cin >> A;
}
friend int sum(ClassA oA, ClassB oB);
};
class ClassB {
    int B;
public:
void accept() {
  cout << "Enter value for B: ";
  cin >> B;
}
friend int sum(ClassA oA, ClassB oB);
};
int sum(ClassA oA, ClassB oB) {
  return oA.A + oB.B;
}
```

```
int main () {
  na.accept ();
  nb.accept ();

Cout << "Sum (na, nb) << endl;

   return 0;
}
```

Output :-

Enter value for A : 6
Enter value for B : 6

Sum : 9

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 object.

→
```
#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 swapNumbers (Number &num1,
Number &num2 }; 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:" << end l;
 n1. display ();
 n2. display ();
 SwapNumbers (n1, n2);
```

```
cout << "After swap : " << endl;
n1.display ();
n2.display ();

  return 0;

}
```

Output

```
Enter value : 3
Enter value : 6
Value swap
 Value : 3
 Value : 6
After swap :
 Value : 6
 Value : 3
```

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

→
```
#include <iostream>
using namespace std;
class Cube;
class Box {
    double v;
public:
    void accept() {
    double l, w, h;
cout << "Enter length, width, and height for
                the Box: ";
cin >> l >> w >> h;
} friend void findGreater(Box b, Cube c);
};
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);
};
Void findGreater(Box, Cube) {
string res = (b.v > c.v) ? "Box" : (c.v > b.v) ? "Cube" : "Both
                                    equal.");
```

```
cout << "Greater Volume:" << res << endl;
}
int main (){
    b. accept ();
    c. accept ();
    findGreater (b, c);

    return 0;

    }
```

Output :-

```
Enter length, width, and height for
  the Box 3 4 3
Enter side length for the cube : 3
Greater Volume : Box
```

Q. Create a Class Complex with real imaginary parts as private members. Use a friend function to add two complex number and return the result as a new complex object.

→
```
#include <iostream>
using namespace std;
class Complex {
    double r, i;
public:
    void accept () {
cout << "Enter real and 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;
 }
```

Output

Enter real and imaginary parts : 3 4
Enter real and imaginary parts : 1 2

The sum is : 4 + 6i

10] Create a class Student with private data members: name and three subject marks. Write a friend function calculateAverage (Student) that calculates and displays the average marks.

```
#include <iostream>
using namespace std;
    class Student {
        string n;
        int m[3];
public:
    void accept () {
cout << "Enter student name: ";
cin >> n;
cout << "Enter marks for three subjects: ";
cin >> m[0] >> m[1] >> m[2];
} friend void calculateAverage (student s);
} s;
void calculateAverage (Student) {
double avg = (s.m[0] + s.m[1] + s.m[2]) / 3.0;
cout << "Student: " << s.n << endl;
cout << "Average Marks: " << avg << endl;
}
int main () {
   s. accept ();
 calculateAverage (s);
   return 0;

}
```

Output

Enter student name : Arsalan
Enter marks for three subjects : 94 97 93
Student : Arsalan
Average Marks : 97

11. Createte three Classes : Alpha, Beta and Gamma, 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 Gamma;
Class Alpha {
int n;
public:
    void accept () {
Cout << "Enter value for Alpha:";
Cin >> n;
} friend in sum(Alpha a, Beta b, Gamma c);
} a;
 Class Beta{
int n; public:
  void accept (){
Cout << "Enter value for Beta:";
Cin >> n;
  } friend int sum ( Alpha a, Beta b, Gamma c)
};
Class Gamma {
int n; public:
  void accept (){
Cout << Enter value for Gamma:";
Cin >> n;
  } friend int sum( Alpha a, Beta b, Gamma c);
} c;
```

```
int sum(Alpha a, Beta b, Gamma c){
return a.n + b.n + c.n;
int main(){
a.accept();
b.accept();
c.accept();

Cout << "sum:" << sum(a,b,c) << endl;

return 0;

}
```

Output :-

Enter value for Alpha
Enter value for Beta
Enter value for Gamma
Sum : 12

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

→
```
#include <iostream>
#include <cmath>
using namespace std;
class Point {
    double x, y;
public:
    void accept () {
cout << "Enter x & y coordinates : ";
cin >> x >> y;
    } friend double dist (Point p1, Point p2);
} p1, p2;
double dist (Point, Point) {
    double dx = p1.x - p2.x;
    double dy = p1.x - p2.x;
    double
    return sqrt ((dx = dx) + (dy = dy));
}
int main () {
p1. accept ();
p2. accept ();
cout << "Distance:" << dist(p1, p2) << end
return 0;
}
```

Enter x & y coordinates : 2 5
Enter x & y coordinates : 5 2
Distance : 4.24264

13] Create two classes. BankAccount and Audit. 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 BankAccount;
class Audit {
public:
    void pBal (BankAccount);
};
class BankAccount {
double b; public:
    void accept() {
cout << "Enter bank account balance: ";
cin >> b;
} friend void Audit::pBal(BankAccount);
} ba;
void Audit::pBal (BankAccount) {
cout << "Auditing ... Bank Account balance.
    = " << ba.b << endl;
}
int main () {
    ba.accept();
    aud.pBal(ba);

return 0;

}
```