- NAME: PREM CHUNIYAN
- ROLL N0.:75

ASSIGNMENT NO.:01

TITLE : Implement a class Complex which represents the Complex Number data type. Implement the following

1. Constructor (including a default constructor which creates the complex number 0+0i).
2. Overloaded operator+ to add two complex numbers.
3. Overloaded operator* to multiply two complex numbers.
4. Overloaded << and >> to print and read Complex Numbers.

CODE :

```
#include<iostream>

#include<iomanip> using

namespace std; class

complex

{

        public:

                float real_num,img_num;

                complex()

                {

                real_num=0;

img_num=0;

                }

                complex operator +(complex);

complex operator *(complex);

                friend ostream &operator <<(ostream &,complex &);

friend istream &operator >>(istream &,complex &);

};

istream &operator >>(istream &is,complex &obj)

{

        is>>obj.real_num;

is>>obj.img_num;        return is;

}

ostream &operator <<(ostream &out,complex &obj)
```

```
{
        out<<" "<<obj.real_num;
out<<"+"<<obj.img_num<<"i";             return
out;
}
complex complex::operator+(complex obj)
{
        complex temp;
        temp.real_num=real_num+obj.real_num;
temp.img_num=img_num+obj.img_num;          return
temp;
}
complex complex::operator*(complex obj)
{
        complex temp;
        temp.real_num=real_num*obj.real_num-img_num*obj.img_num;
temp.img_num=img_num*obj.real_num+real_num*obj.img_num;
        return temp;
}
int main()
{
        complex a,b,c,d;
        int ch;
        cout<<"\n The first complex number is:";
cout<<"\nEnter real and img:";
        cin>>a;
        cout<<"\n The second complex number is:";
cout<<"\nEnter real and img:";
        cin>>b;
        do
        {
```

```
        cout<<"Enter Your Choice\n1.Adition\n2.Multiplication\n3.Exit\n";  cin>>ch;
        switch(ch)
        {
        case 1:
c=a+b;
        cout<<"\n Addition=";
cout<<c<<endl;
        break;
case 2:
d=a*b;
        cout<<"\n Multiplication=";
        cout<<d<<endl;
        break;
        }
        }
        while(ch!=3);
return 0;
}
```

OUTPUT:

```
The first complex number is:
Enter real and img:25 6
The second complex number is:
Enter real and img:35 8
Enter Your Choice
1.Adition
2.Multiplication
3.Exit 1
Addition=  60+14i
Enter Your Choice
1.Adition
```

2. Multiplication

3.Exit

2

Multiplication= 827+410i

Enter Your Choice

1.Adition

2.Multiplication

3.Exit

3(EXIT THE CODE)

- NAME: PREM CHUNIYAN
- ROLL N0.:75

ASSIGNMENT NO.:02

TITLE : Develop an object oriented program in C++ to create a database of student information system containing the following information: Name, Roll number, Class, division, Date of Birth, Blood group, Contact address, telephone number, driving license no. and other. Construct the database with suitable member functions for initializing and destroying the data viz constructor, default constructor, Copy constructor, destructor, static member functions, friend class, this pointer, inline code and dynamic memory allocation operators-new and delete.

CODE:

```
#include <iostream>
#include <string>
#include <vector>
using namespace std;
class Student { private:
   string name;     int rollNumber;
string studentClass;     char
division;     string dob;        //
Date of Birth
   string bloodGroup;
string address;     long
long telephone;     string
licenseNo;     static int
studentCount; public:
   // Default constructor
   Student() : rollNumber(0), division('A'), telephone(0) {
name = "Unknown";        studentClass = "Unknown";
dob = "00/00/0000";         bloodGroup = "Unknown";
      address = "Unknown";
licenseNo = "Unknown";
studentCount++;
   }
   // Parameterized constructor
```

```
   Student(string name, int rollNumber, string studentClass, char division,
string dob, string bloodGroup, string address, long long telephone,          string
licenseNo) : name(name), rollNumber(rollNumber),
studentClass(studentClass), division(division), dob(dob),
bloodGroup(bloodGroup), address(address), telephone(telephone),
        licenseNo(licenseNo) {
studentCount++;
   }
   // Destructor
~Student() {
studentCount--;
   }
   // Static function to get student count
static int getStudentCount() {
return studentCount;
   }
   // Friend class declaration
friend class StudentHelper;    //
Inline function to display details
inline void display() const {       cout
<< "\nStudent Details:"
          << "\nName: " << name
          << "\nRoll Number: " << rollNumber
          << "\nClass: " << studentClass
          << "\nDivision: " << division
          << "\nDate of Birth: " << dob
          << "\nBlood Group: " << bloodGroup
          << "\nAddress: " << address
          << "\nTelephone: " << telephone
          << "\nDriving License No.: " << licenseNo << endl;
   }
```

```
  // Function to set data using 'this' pointer     void setData(string name, int
rollNumber, string studentClass, char division,                string dob, string
bloodGroup, string address, long long telephone,
              string licenseNo) {
      this->name = name;        this-
>rollNumber = rollNumber;        this-
>studentClass = studentClass;
this->division = division;        this-
>dob = dob;        this->bloodGroup =
bloodGroup;        this->address =
address;        this->telephone =
telephone;        this->licenseNo =
licenseNo;
  }
  // Dynamic memory allocation and data entry  static Student*
createStudent() {   string name, studentClass, dob, bloodGroup,
address, licenseNo;
      int rollNumber;        char division;
long long telephone;        // Taking input
from user        cout << "Enter Student
Name: ";        getline(cin >> ws, name);
cout << "Enter Roll Number: ";        cin >>
rollNumber;        cout << "Enter Class: ";
getline(cin >> ws, studentClass);        cout
<< "Enter Division (A/B/C/...): ";        cin
>> division;

      cout << "Enter Date of Birth (DD/MM/YYYY): ";
      getline(cin >> ws, dob);      cout <<
"Enter Blood Group: ";   getline(cin >> ws,
bloodGroup);   cout << "Enter Address: ";
getline(cin >> ws, address);      cout <<
```

```
"Enter Telephone Number: ";        cin >>
telephone;        cout << "Enter Driving License
Number: ";        getline(cin >> ws, licenseNo);
      // Using new to dynamically create a student object
      return new Student(name, rollNumber, studentClass, division, dob, bloodGroup, address,
telephone, licenseNo);
   }
   // Function to delete dynamically allocated student object
static void deleteStudent(Student *student) {
      delete student;
   }
};
// Initialize static member int
Student::studentCount = 0; //
Friend class definition class
StudentHelper { public:
   static void showStudentInfo(const Student &student) {        cout << "Accessing private data
from friend class:\n";        cout << "Name: " << student.name << "\nRoll Number: " <<
student.rollNumber << endl;
   }
};
int main() {     int
numStudents;
cout << "Enter the
number of
students you want
to add: ";     cin >>
numStudents;
   // Creating a vector to store pointers to Student objects
vector<Student*> students;
   // Taking inputs for each student and storing in vector
for (int i = 0; i < numStudents; ++i) {        cout << "\nEnter
```

```
details for student " << (i + 1) << ":\n";
students.push_back(Student::createStudent());
  }
  // Display all students' details     cout
<< "\n--- Student Information ---";    for
(int i = 0; i < numStudents; ++i) {
students[i]->display();
  }
  // Displaying total student count     cout << "\nTotal Students: " <<
Student::getStudentCount() << endl;
  // Deleting all dynamically allocated student objects
for (int i = 0; i < numStudents; ++i) {
     Student::deleteStudent(students[i]);
  }
  return 0;
}
```

OUTPUT:

```
Enter the number of students you want to add: 2
Enter details for student 1:
Enter Student Name: Arpita
Enter Roll Number: 40
Enter Class: SE
Enter Division (A/B/C/...): A
Enter Date of Birth (DD/MM/YYYY): 13/08/2005
Enter Blood Group: O+
Enter Address: Nashik
Enter Telephone Number: 1234567890
Enter Driving License Number: 987

Enter details for student 2:
Enter Student Name: Tanvi
```

Enter Roll Number: 40

Enter Class: SE

Enter Division (A/B/C/...): A

Enter Date of Birth (DD/MM/YYYY): 10/11/2005

Enter Blood Group: A+

Enter Address: Jalna

Enter Telephone Number: 9876543210

Enter Driving License Number: 123

--- Student Information ---

Student Details:

Name: Arpita

Roll Number: 40

Class: SE

Division: A

Date of Birth: 13/08/2005

Blood Group: O+

Address: Nashik

Telephone: 1234567890

Driving License No.: 987

Student Details:

Name: Tanvi

Roll Number: 40

Class: SE

Division: A

Date of Birth: 10/11/2005

Blood Group: A+

Address: Jalna

Telephone: 9876543210

Driving License No.: 123

Total Students: 2

- NAME: PREM CHUNIYAN
- ROLL N0.:75

ASSIGNMENT NO.:03

TITLE : Imagine a publishing company which does marketing for book and audio cassette versions. Create a class publication that stores the title (a string) and price (type float) of a publication. From this class derive two classes: book, which adds a page count (type int), and tape, which adds a playing time in minutes (type float). Write a program that instantiates the book and tape classes, allows user to enter data and displays the data members. If an exception is caught, replace all the data member values with zero values.

CODE :

```
#include<iostream> using
namespace std;
class publication
{
        public:
char title[20];         float
price;
                virtual void getdata()
                {
                        cout<<"\nEnter the title:";
                        cin>>title;
                        cout<<"\nEnter the price:";
                        cin>>price;
                }
                virtual void display()
                {
                        cout<<"Title:"<<title<<endl;
                        cout<<"Price:"<<price<<endl;
                }

};
class book:public publication
{
        public:
                int page_count;
        void getdata()
                {
                        cout<<"\nEnter the no of pages:";
cin>>page_count;
                }
                void display()
                {
                        cout<<"Page Count of book:"<<page_count<<endl;
                }
};

class tape:public publication
{
        public:
                float playtime;
```

```
		void getdata()
		{
			cout<<"\nEnter the time in minutes:";
			cin>>playtime;
		}

		void display()
		{
			cout<<"Playing time in tape:"<<playtime<<endl;
		}
};

int main()
{
	publication *ptr;
publication obj;	book
b;
	tape t;

	obj.getdata();
obj.display();

	ptr=&b;
	ptr->getdata();		ptr-
>display();

	ptr=&t;
	ptr->getdata();		ptr-
>display();

	return 0;
}
```

OUTPUT:

```
Enter the title:In_search_of_lost_time

Enter the price:1995
Title:In_search_of_lost_time
Price:1500

Enter the no of pages:4000
Page Count of book:4000

Enter the time in minutes:4560
Playing time in tape:4560
```

- NAME: PREM CHUNIYAN
- ROLL N0.:75

ASSIGNMENT NO.:04

Write a C++ program that creates an output file, writes information to it, closes the file andopen it again as an input file and read the information from the file.

CODE :

```
#include<iostream>
#include<fstream>
#include<cstdlib> using
namespace std; class
sample
{
        public:
int roll_num;           char
name[20];               void
getvalue()
                {
                        cout<<"\nEnter the roll_num no:";
                        cin>>roll_num;
                        cout<<"\nEnter the name:";
                        cin>>name;
                }

                void show()
                {
                        cout<<roll_num<<"\n";
cout<<name<<"\n";
                }
};
int main()
{
        sample obj;
        ofstream os;
        os.open("Home:\\test.txt",ios::in|ios::out);
        if(!os)
        {
        cerr<<"Could not open output file\n";
exit(1);
        }

        cout<<"Writing the contents to the file...\n";
        obj.getvalue();
        os.write((char *)&obj,sizeof(obj));
        if(!os)
        {
        cerr<<"Could not write to file\n";
exit(1);
        }
```

```
        os.close();

        ifstream is;
        is.open("Home:\\test.txt",ios::in|ios::out);
        if(!is)
        {
        cerr<<"Could not open input file\n";
exit(1);
        }
        cout<<"Reading the contents from the file...\n";
        obj.show();
        is.read((char *)&obj,sizeof(obj));
if(!is)
        {
        cerr<<"Could not read from file\n";
exit(1);
        }
        return 0;
}
```

OUTPUT:

Could not open output file

- NAME: PREM CHUNIYAN
- ROLL N0.:75

ASSIGNMENT NO.:05

TITLE :Write a function template selection Sort. Write a program that inputs, sorts and outputs an integer array and a float array.

CODE ;

```
#include<iostream> using
namespace std; //#define
Size 10
int n;
template <class T>
class sort
{ int
i,j;
T a[50]; public:
void insert()
{
        cout<<"\nHow many elements are there?";
        cin>>n;
        cout<<"\nEnter the Numbers"<<endl;
for(i=0;i<n;i++)          cin>>a[i];
selection(a);
}
void selection(T a[])
{
        T temp,exchange=0,cmp=0;      for(i=0;i<n-
1;i++)
        {
                cout<<"\n\tAfter Pass"<<i+1<<" "<<endl;
                for(j=i+1;j<n;j++)
                {
                        cmp++;
```

```
                        if(a[j]<a[i])
                        {
                        exchange++;
        temp=a[i];
a[i]=a[j];                      a[j]=temp;
                        }
                        for(int k=0;k<n;k++)
                        {
                                cout<<"\t"<<a[k];
                        }
                        cout<<"\n";
                        //cout<<"\nno of exchanges"<<exchange<<endl;
                        //cout<<"\nno of comparision"<<cmp<<endl;
                }
        }
        cout<<"\n\tNo of Exchange/no of Comparision\n Total no of exchanges =>
"<<exchange<<endl;    cout<<"\n\t Total no of comparision =>
"<<cmp<<endl<<"Dispaly of sorted list";
        cout<<endl;
}
void display()
{
        cout<<"\n The sorted List Is...\n";
        for(i=0;i<n;i++)
cout<<"\t"<<a[i]<<endl;
} };
int main()
{
        cout<<" All values should be in Integer form "<<endl;
```

```
        sort<int> obj1;
        obj1.insert();
obj1.display();

        cout<<"Float value"<<endl;
sort<float> obj2;
obj2.insert();  obj2.display();
        return 0;
}
```

OUTPUT:

All values should be in Integer form

How many elements are there?4

Enter the Numbers

30

10

60

90

After Pass1

10 30 60 90

10 30 60 90

10 30 60 90

After Pass2

10 30 60 90

10 30 60 90

After Pass3

10 30 60 90

No of Exchange/no of Comparision

Total no of exchanges => 1

Total no of comparision => 6

Dispaly of sorted list

The sorted List Is...

10

30

60

90

Float value

How many elements are there?2

Enter the Numbers

11.34

78.98

After Pass1

11.34 78.98

No of Exchange/no of Comparision

Total no of exchanges => 0

Total no of comparision => 1

Dispaly of sorted list

The sorted List Is...

11.34

78.98

- NAME: PREM CHUNIYAN
- ROLL N0.:75

ASSIGNMENT NO.:06

TITLE : Crete User defined exception to check the following conditions and throw the exception if the criterion does not meet.

a) User has age between 18 and 55
b) User stays have income between Rs. 50,000 – Rs. 1,00,000 per month
c) User stays in Pune/ Mumbai/ Bangalore / Chennai
d) User has 4-wheeler Accept age, Income, City, Vehicle from the user and check for the conditions mentioned above. If any of the condition not met then throw the exception

CODE :

```
#include<iostream>

#include<cstring> using

namespace std; int

main()

{    int

age;

  int vehicleType;

double income;    char

city[20];


  cout << "\nEnter the age of the person: ";

cin >> age;

  try

  {

    if (age < 18 || age > 55)

    {

      throw age;

    }

    cout << "\nValid age: " << age;

  }

  catch (int)

  {

    cout << "\nError: Age must be between 18 and 55.";
```

```
    }
    cout << "\nEnter the income of the person: ";
cin >> income;
    try
    {
        if (income < 50000 || income > 100000)
        {
            throw income;
        }
        cout << "\nValid income: " << income;
    }
    catch (double)
    {
        cout << "\nError: Income should be in the range of 50000 to 100000.";
    }
    cout << "\nEnter the city of residence: ";
    cin >> city;
    try
    {
        if (strcmp(city, "Pune") && strcmp(city, "Mumbai") && strcmp(city, "Bangalore") &&
strcmp(city, "Chennai"))
        {
            throw city;
        }
        cout << "\nValid city: " << city;
    }
    catch (char[])
    {
        cout << "\nError: City must be Pune, Mumbai, Bangalore, or Chennai.";
    }
    cout << "\nEnter type of vehicle (2 for two-wheeler, 4 for four-wheeler): ";
cin >> vehicleType;
```

```
    try
    {
        if (vehicleType != 2 && vehicleType != 4)
        {
            throw vehicleType;
        }
        cout << "\nValid vehicle type: " << vehicleType << " wheeler";
    }
    catch (int)
    {
        cout << "\nError: Vehicle type must be either 2 or 4 wheeler.";
    }

    return 0;
}
```

OUTPUT:

```
Enter the age of the person: 19
Valid age: 19
Enter the income of the person: 100000
Valid income: 100000
Enter the city of residence: nashik
Error: City must be Pune, Mumbai, Bangalore, or Chennai.
Enter type of vehicle (2 for two-wheeler, 4 for four-wheeler): 2
Valid vehicle type: 2 wheeler

Enter the age of the person: 13
Error: Age must be between 18 and 55.
Enter the income of the person: 1000
Error: Income should be in the range of 50000 to 100000.
Enter the city of residence: NASHIK
Error: City must be Pune, Mumbai, Bangalore, or Chennai.
```

Enter type of vehicle (2 for two-wheeler, 4 for four-wheeler): 4

Valid vehicle type: 4 wheeler

- NAME: PREM CHUNIYAN
- ROLL N0.:75

ASSIGNMENT NO.:07

TITLE ; Write C++ program using STL for sorting and searching with user defined records such as person record(Name, DOB, Telephone number), Item record (Item code, name, cost,quantity) using vector container.

CODE:

```
#include <iostream>

#include <algorithm>

#include <vector> using

namespace std; class

Product

{

public:

    char productName[10];

    int quantity;

int price;      int

id;

    bool operator==(const Product& p)

    {

      return id == p.id;

    }

    bool operator<(const Product& p)

    {

      return id < p.id;

    }

};

vector<Product> inventory;

void print(Product &p);

void display(); void

insert(); void search(); void

remove();
```

```
bool compareByPrice(const Product &p1, const Product &p2)
{
   return p1.price < p2.price;
}
int main()
{
   int choice;
do
   {
      cout << "\n***** Menu *****";
cout << "\n1. Insert";       cout <<
"\n2. Display";       cout << "\n3.
Search";       cout << "\n4. Sort";
cout << "\n5. Delete";       cout <<
"\n6. Exit";       cout << "\nEnter
your choice: ";       cin >> choice;
switch(choice)
      {
         case 1:
insert();
break;           case
2:
display();
break;           case
3:
search();
break;           case
4:
            sort(inventory.begin(), inventory.end(), compareByPrice);              cout << "\n\nSorted
by Price";
            display();
 break;           case
```

```
5:
remove();
break;          case
6:              exit(0);
      }
   } while(choice != 7);
return 0;
}
void insert()
{
   Product p;    cout << "\nEnter
Product Name: ";    cin >>
p.productName;    cout << "\nEnter
Quantity: ";    cin >> p.quantity;
cout << "\nEnter Price: ";    cin >>
p.price;    cout << "\nEnter Product
ID: ";
   cin >> p.id;
   inventory.push_back(p);
}
void display()
{
   for_each(inventory.begin(), inventory.end(), print);
}
void print(Product &p)
{
   cout << "\n";
   cout << "\nProduct Name: " << p.productName;
cout << "\nQuantity: " << p.quantity;    cout <<
"\nPrice: " << p.price;    cout << "\nProduct ID: "
<< p.id;
}
```

```
void search()
{
   vector<Product>::iterator it;    Product
p;    cout << "\nEnter Product ID to
search: ";
   cin >> p.id;    it = find(inventory.begin(),
inventory.end(), p);    if(it == inventory.end())
   {
      cout << "\nNot found.";
   }
   else
   {
      cout << "\nFound.";
   }
}
void remove()
{
   vector<Product>::iterator it;    Product
p;    cout << "\nEnter Product ID to
delete: ";
   cin >> p.id;    it = find(inventory.begin(),
inventory.end(), p);    if(it == inventory.end())
   {
      cout << "\nNot found.";
   }
   else
   {
      inventory.erase(it);
cout << "\nDeleted.";
   }
}
```

OUTPUT:

***** Menu *****

1. Insert

2. Display

3. Search

4. Sort

5. Delete

6. Exit

Enter your choice: 1

Enter Product Name: pen

Enter Quantity: 20

Enter Price: 15

Enter Product ID: 101

***** Menu *****

1. Insert

2. Display

3. Search

4. Sort

5. Delete 6. Exit

Enter your choice: 2

Product Name: pen

Quantity: 20

Price: 15

Product ID: 101

***** Menu *****

1. Insert

2. Display

3. Search

4. Sort

5. Delete

6. Exit

Enter your choice: 1

Enter Product Name: notebook

Enter Quantity: 10

Enter Price: 50

Enter Product ID: 102

***** Menu *****

1. Insert

2. Display

3. Search

4. Sort

5. Delete 6. Exit

Enter your choice: 3 Enter

Product ID to search: 102

Found.

***** Menu *****

1. Insert

2. Display

3. Search

4. Sort

5. Delete

6. Exit

Enter your choice: 4

Sorted by Price

Product Name: pen

Quantity: 20

Price: 15

Product ID: 101

Product Name: notebook

Quantity: 10

Price: 50

Product ID: 102

***** Menu *****

1. Insert

2. Display

3. Search

4. Sort

5. Delete 6. Exit

Enter your choice: 5 Enter

Product ID to delete: 102

Deleted.

***** Menu *****

1. Insert

2. Display

3. Search

4. Sort

5. Delete 6. Exit

Enter your choice: 2

Product Name: pen

Quantity: 20

Price: 15

Product ID: 101

***** Menu *****

1. Insert

2. Display

3. Search

4. Sort

5. Delete

6. Exit

Enter your choice: 6(EXIT)

- NAME: PREM CHUNIYAN
- ROLL N0.:75

ASSIGNMENT NO.:08

TITLE : Write a program in C++ to use map associative container. The keys will be the names of states and the values will be the populations of the states. When the program runs, the user is prompted to type the name of a state. The program then looks in the map, using the state name as an index and returns the population of the state.

CODE :

```
#include<iostream>
#include<map>
#include<string>
using namespace std;
int main()
{
   typedef map<string, int> mType;

   mType popMap;

   popMap.insert(pair<string, int>("Maharashtra", 7026357));
popMap.insert(pair<string, int>("Rajasthan", 6578936));
popMap.insert(pair<string, int>("Karnataka", 6678993));
popMap.insert(pair<string, int>("Punjab", 5789032));
popMap.insert(pair<string, int>("West Bengal", 6676291));

   mType::iterator it;

   cout << "\tPopulation of states in India\n";
   cout << "\nSize of popMap: " << popMap.size() << "\n";

   string state;     cout << "\nEnter
name of the state: ";
   cin >> state;     it =
popMap.find(state);     if
(it != popMap.end())
      cout << state << "'s population is " << it->second;
else
      cout << "State not found in popMap\n";
popMap.clear();
}
```

OUTPUT:

```
Population of states in IndiaS
Size of popMap: 5
Enter name of the state: Punjab

Punjab's population is 5789032
```

Population of states in India

Size of popMap: 5

Enter name of the state: MAHARASHTRA
State not found in popMap