```
/*Implement a class Complex which represents the Complex Number data type.
Implement the following operations:
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.*/

#include<iostream>
using namespace std;

class Complex
{
	private:
	int real,img;

	public:
	Complex()
	{
		real = 0;
		img = 0;
	}

	Complex operator + (Complex b)
	{
		class Complex a;
		a.real =  real + b.real;
		a.img = img + b.img;

		return(a);
	}

	Complex operator * (Complex b)
	{
		class Complex multi;
		multi.real = real * b.real;
		multi.img = img * b.img;
		return (multi);
	}

	friend istream &operator >> (istream &input,Complex &com)
	{
		cout<<"Enter real part : ";
		input>>com.real;
		cout<<"Enter img part : ";
		input>>com.img;
	}
	friend ostream &operator << (ostream &output,Complex &com)
	{
		output<<com.real<<" + "<<com.img<<"i\n";
	}

};
```

```
int main()
{
	Complex i1,i2,i3,i4;
	cin>>i1;
	cin>>i2;

	i3 = i1+i2;
	cout<<i3;

	i4 = i1*i2;
	cout<<i4;

}
```

**Output**

```
Enter real part : 25
Enter img part : 12
Enter real part : 30
Enter img part : 40
55 + 52i
750 + 480i
```

*/*Develop a program in C++ to create a database of student's 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. Make use of constructor, default constructor, copy constructor,destructor, static member functions, friend class, this pointer, inline code and dynamic memory allocation operators-new and delete as well as exception handling.*

*/

```
#include <iostream>
#include <string>
using namespace std;
class Data
{
   string name;
   int roll;
   int year;
   char *div = new char[1];
   string dob;
   long int phone;
   string college;
   char *bloodgrp = new char[3];
   string address;

public:
   static int count;
   Data()
   {
      name = "";
      roll = 0;
      year = 0;
      dob = "";
      phone = 0;
   }
   ~Data()
   {
      delete[] div;
       delete[] bloodgrp;
   }
   Data(Data &obj)
   {
      this->college = obj.college;
   }
   inline static int getCount()
   {
      return count;
   }
   void inputData()
   {
      cout << "Enter Student Name : ";
      cin.ignore();
      getline(cin, name);
      cout << "Enter Roll No : ";
      cin >> roll;
```

```
        cout << "Enter Year : ";
        cin >> year;
        cout << "Enter Division : ";
        cin >> div;
        cout << "Enter Date of Birth (dd/mm/yy) : ";
        cin.ignore();
        getline(cin, dob);
        cout << "Enter Blood Group : ";
        cin >> bloodgrp;
        cout << "Enter Address : ";
        cin.ignore();
        getline(cin, address);
        cout << "Enter Phone No : ";
        cin >> phone;
        count++;
        cout << "\n\n";
    }
    void setCollege(string college)
    {
        this->college = college;
    }
    friend class Display;
};
int Data::count = 0;
class Display
{
public:
    void display(Data &obj)
    {
        cout << "\nCollege Name : " << obj.college;
        cout << "\nStudent Name : " << obj.name;
        cout << "\nRoll No : " << obj.roll;
        cout << "\nYear : " << obj.year;
        cout << "\nDivision : " << obj.div;
        cout << "\nDate of Birth : " << obj.dob;
        cout << "\nBlood Group : " << obj.bloodgrp;
        cout << "\nAddress : " << obj.address;
        try
        {
            if (obj.phone >= 1000000000 && obj.phone <= 9999999999)
                cout << "Phone No : " << obj.phone;
            else
                throw obj.phone;
        }
        catch (long int phone)
        {
            cout << "Invald Phone Number";
        }
    }
};
int main()
{
```

```
    string college;
    cout << "Enter College Name : ";
    getline(cin, college);
    Data a;
    a.inputData();
    a.setCollege(college);
    Display ad;
    Data b(a);
    b.inputData();
    ad.display(a);
    ad.display(b);
    cout << "\nTotal Student Entries : " << a.getCount() << "\n";
}
```

## Output

```
Enter College Name : I2IT

Enter Student Name : John
Enter Roll No : 12
Enter Year : 2022
Enter Division : A
Enter Date of Birth (dd/mm/yy) : 16/02/2003
Enter Blood Group : AB+
Enter Address : Warje
Enter Phone No : 9876321234

Enter Student Name : James
Enter Roll No : 19
Enter Year : 2022
Enter Division : B
Enter Date of Birth (dd/mm/yy) : 19/06/2003
Enter Blood Group : A+
Enter Address : Pimpri
Enter Phone No : 7348321234
College Name : I2IT

Student Name : John
Roll No : 12
Year : 2022
Division : A
Date of Birth : 16/02/2003
Blood Group : AB+
Address : Warje
Phone No : 9876321234

College Name : I2IT
Student Name : James
Roll No : 19
Year : 2022
Division : B
Date of Birth : 19/06/2003
```

Blood Group : A+
Address : Pimpri
Phone No : 7348321234

Total Student Entries : 2

/*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 publications. 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 class, allows user to enter data and displays the data members. If an exception is caught, replace all the data member values with zero values.*/

```
#include<iostream>
#include<string>

using namespace std;

class Publication
{
   public:
        string title;
        float price;

        Publication()
        {
                title = "";
                price = 0.0;
        }

        void get_data()
        {

                cout<<"Enter title : ";
                cin.ignore();
                getline(cin,title);
                cout<<"Enter Price : ";
                cin>>price;
        }

        void display_data()
        {
                cout<<"-------------------------------------------------------------------------";
                cout<<"\nTitle : "<<title<<"\n";

                try{
                        if(price<=0)
                        {
                                throw price;
                        }
                        else
                        {
                                cout<<"Price : "<<price<<"\n";
                        }
                }catch(float f)
                {
                        cout<<"Invalid Price : "<<f<<"\n";
                        price = 0;
                        title = "";
```

```
		}
	}
};

class Book : public Publication
{
	int pc;
	public:
	Book()
	{
		pc=0;
	}
	void get_data()
	{
		Publication::get_data();
		cout<<"Enter pages : ";
		cin>>pc;
	}

	void display_data()
	{
		Publication::display_data();
	try{
		if(pc<=0)
		{
			throw pc;
		}
		else
		{
			cout<<"Pages : "<<pc<<"\n";
		}
	}
	catch(int f)
	{
		cout<<"Pages are invalid : "<<f<<"\n";
		pc = 0;
		title = "";
		price = 0.0;
	}

	cout<<"-----------------------------------------------------------------\n";
	}


};

class Tapes : public Publication
{
	int tapes;
	public:
	Tapes()
	{
```

```
		tapes = 0;
	}

	void get_data()
	{
		Publication::get_data();
		cout<<"Enter Tapes Size : ";
		cin>>tapes;
	}

	void display_data()
	{
		Publication::display_data();
		try{
			if(tapes<=0){
				throw tapes;
			}
			else
			{
				cout<<"Tapes Size : "<<tapes<<"\n";
			}
		}catch(int f)
		{
			cout<<"Invalid Tape Size : "<<f<<"\n";
			tapes = 0;
			title = "";
			price = 0.0;
		}
		cout<<"-------------------------------------------------------------------------------\n";
	}
};

int main()
{
	Book b[100];
	Tapes t[100];

	int n=-1,ib=0,it=0,c=0;
	while(n!=0)
	{
		cout<<"\n1.Enter Pages";
		cout<<"\n2.Enter Tapes";
		cout<<"\n3.Display Pages Data";
		cout<<"\n4.Display Tapes Data";
		cout<<"\n0.Exit";
		cout<<"\n\nEnter your choice : ";
		cin>>n;

		switch(n)
		{
			case 1:
				b[ib].get_data();
```

```
                        ib++;
                        break;
                case 3:
                        if(ib==0)
                        {
                                cout<<"\nNo Data to Display\n";
                        }
                        else
                        {
                                for(int j=0;j<ib;j++)
                                        b[j].display_data();
                        }
                        break;
                case 2:
                        t[it].get_data();
                        it++;
                        break;
                case 4:
                        if(it==0)
                        {
                                cout<<"\nNo Data to Display\n";
                        }
                        else
                        {
                                for(int j=0;j<it;j++)
                                        t[j].display_data();
                        }
                        break;
                case 0:
                        cout<<"\nThank You!!";
                        break;
                }
        }

}
```

## Output

```
1.Enter Pages
2.Enter Tapes
3.Display Pages Data
4.Display Tapes Data
0.Exit

Enter your choice : 1
Enter title : Hobbit
Enter Price : 200
Enter pages : 500

1.Enter Pages
2.Enter Tapes
3.Display Pages Data
4.Display Tapes Data
```

```
0.Exit

Enter your choice : 2
Enter title : Pop
Enter Price : 200
Enter Tapes Size : 2

1.Enter Pages
2.Enter Tapes
3.Display Pages Data
4.Display Tapes Data
0.Exit

Enter your choice : 1
Enter title : Pirates
Enter Price : 500
Enter pages : 200

1.Enter Pages
2.Enter Tapes
3.Display Pages Data
4.Display Tapes Data
0.Exit

Enter your choice : 3
------------------------------------------------------------------------
Title : Hobbit
Price : 200
Pages : 500
------------------------------------------------------------------
------------------------------------------------------------------------
Title : Pirates
Price : 500
Pages : 200
------------------------------------------------------------------

1.Enter Pages
2.Enter Tapes
3.Display Pages Data
4.Display Tapes Data
0.Exit

Enter your choice : 4
------------------------------------------------------------------------
Title : Pop
Price : 200
Tapes Size : 2
------------------------------------------------------------------------

1.Enter Pages
2.Enter Tapes
3.Display Pages Data
```

4.Display Tapes Data
0.Exit

Enter your choice : 0

Thank You!!

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

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

int main()
{
   int n, roll_no[10];
   string name[50];
   cout << "Enter the number of students whose data is to be saved \n";
   cin >> n;
   cout << "  " << endl;

   ofstream outf;
   outf.open("stud_data.txt", ios::trunc);

   for (int i = 0; i < n; i++)
   {
      cout << "Enter name : ";
      cin >> name[i];
      outf << name[i] << "\n";

      cout << "Enter Roll number : ";
      cin >> roll_no[i];
      outf << roll_no[i] << "\n";
      cout << "     " << endl;
   }

   outf.close();

   ifstream inf;
   inf.open("stud_data.txt");

   for (int i = 0; i < n; i++)
   {
      while (inf)
      {
         inf >> name[i];
         inf >> roll_no[i];
         if (inf.eof() != 0)
         {
            cout << "End of file\n";
            exit(1);
         }
         cout << "Name is        : " << name[i] << endl;
         cout << "roll number is : " << roll_no[i] << endl;
      }
   }
```

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

## Output

```
Enter the number of students whose data is to be saved
3

Enter name : Sahil
Enter Roll number : 12

Enter name : John
Enter Roll number : 13

Enter name : Parth
Enter Roll number : 25

Name is       : Sahil
roll number is : 12
Name is       : John
roll number is : 13
Name is       : Parth
roll number is : 25
End of file
```

/*Implement a function template selection Sort. Write a program that inputs,
sorts and outputs an integer array and a float array.*/

```
#include <iostream>
using namespace std;
int n;
#define size 10
template <class T>
void sel(T A[size])
{
   int i, j, min;
   T temp;
   for (i = 0; i < n - 1; i++)
   {
      min = i;
      for (j = i + 1; j < n; j++)
      {
         if (A[j] < A[min])
            min = j;
      }
      temp = A[i];
      A[i] = A[min];
      A[min] = temp;
   }
   cout << "\nSorted array: ";
   for (i = 0; i < n; i++)
   {
      cout << " " << A[i];
   }
   cout << "\n";
}
int main()
{
   int choice;
   char C[size];
   int A[size];
   float B[size];
   int i;
   cout << "-----------------------";
   do
   {
      cout << "\n 1.Enter Integer Array : ";
      cout << "\n 2.Enter Float Array : ";
      cout << "\n 3.Exit : " << endl;
      cout << "\n Enter Choice : ";
      cin >> choice;
      switch (choice)
      {
      case 1:
         cout << "\nEnter Total Number Of Integer Elements:";
         cin >> n;
         cout << "\nEnter Integer Elements:";
```

```
            for (i = 0; i < n; i++)
            {
               cin >> A[i];
            }
            sel(A);
            break;
         case 2:
            cout << "\nEnter Total Number Of Float Elements:";
            cin >> n;
            cout << "\nEnter Float Elememts:";
            for (i = 0; i < n; i++)
            {
               cin >> B[i];
            }
            sel(B);
            break;
         case 3:
            cout << "Program End" << endl;
            exit(0);
         default:
            cout << "\n Invalid";
         }
      } while (choice != 4);
      return 0;
}
```

## Output

```
----------------------
1.Enter Integer Array :
2.Enter Float Array :
3.Exit :
Enter Choice : 1
Enter Total Number Of Integer Elements:5
Enter Integer Elements:8 3 5 12 9
Sorted array: 3 5 8 9 12
1.Enter Integer Array :
2.Enter Float Array :
3.Exit :
Enter Choice : 2
Enter Total Number Of Float Elements:5
Enter Float Elememts:8.5 8.2 9.0 12.3 12.9
Sorted array: 8.2 8.5 9 12.3 12.9
1.Enter Integer Array :
2.Enter Float Array :
3.Exit :
Enter Choice : 3
Program End
```

```
/*
Write C++ program using STL for Sorting and searching with user-defined records such as
Person Record (Name, birth date, telephone no), item record (item code, item name, quantity
and cost)
*/
#include <iostream>
#include <algorithm>
#include <vector>
   using namespace std;
class Item
{
public:
   char name[10];
   int quantity;
   int cost;
   int code;
   bool operator==(const Item &i1)
   {
      if (code == i1.code)
         return 1;
      return 0;
   }
   bool operator<(const Item &i1)
   {
      if (code < i1.code)
         return 1;
      return 0;
   }
};
vector<Item> o1;
void print(Item &i1);
void display();
void insert();
void search();
void dlt();
bool compare(const Item &i1, const Item &i2)
{
   return i1.cost < i2.cost;
}
int main()
{
   int ch;
   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 >> ch;
        switch (ch)
        {
        case 1:
            insert();
            break;
        case 2:
            display();
            break;
        case 3:
            search();
            break;
        case 4:
            sort(o1.begin(), o1.end(), compare);
            cout << "\n\n Sorted on Cost";
            display();
            break;
        case 5:
            dlt();
            break;
        case 6:
            exit(0);
        }
    } while (ch != 7);
    return 0;
}
void insert()
{
    Item i1;
    cout << "\nEnter Item Name:";
    cin >> i1.name;
    cout << "\nEnter Item Quantity:";
    cin >> i1.quantity;
    cout << "\nEnter Item Cost:";
    cin >> i1.cost;
    cout << "\nEnter Item Code:";
    cin >> i1.code;
    o1.push_back(i1);
}
void display()
{
    for_each(o1.begin(), o1.end(), print);
}
void print(Item &i1)
{
    cout << "\n";
    cout << "\nItem Name:" << i1.name;
    cout << "\nItem Quantity:" << i1.quantity;
    cout << "\nItem Cost:" << i1.cost;
    cout << "\nItem Code:" << i1.code;
}
void search()
```

```
{
   vector<Item>::iterator p;
   Item i1;
   cout << "\nEnter Item Code to search:";
   cin >> i1.code;
   p = find(o1.begin(), o1.end(), i1);
   if (p == o1.end())
   {
      cout << "\nNot found.";
   }
   else
   {
      cout << "\nFound.";
   }
}
void dlt()
{
   vector<Item>::iterator p;
   Item i1;
   cout << "\nEnter Item Code to delete:";
   cin >> i1.code;
   p = find(o1.begin(), o1.end(), i1);
   if (p == o1.end())
   {
      cout << "\nNot found.";
   }
   else
   {
      o1.erase(p);
      cout << "\nDeleted.";
   }
}
```

Output

```
***** Menu *****
1.Insert
2.Display
3.Search
4.Sort
5.Delete
6.Exit
Enter your choice:1
Enter Item Name:Book
Enter Item Quantity:10
Enter Item Cost:500
Enter Item Code:4
***** Menu *****
1.Insert
2.Display
3.Search
4.Sort
5.Delete
```

6.Exit
Enter your choice:1
Enter Item Name:Compass
Enter Item Quantity:25
Enter Item Cost:120
Enter Item Code:13
***** Menu *****
1.Insert
2.Display
3.Search
4.Sort
5.Delete
6.Exit
Enter your choice:2
Item Name:Book
Item Quantity:10
Item Cost:500
Item Code:4
Item Name:Compass
Item Quantity:25
Item Cost:120
Item Code:13
***** Menu *****
1.Insert
2.Display3.Search
4.Sort
5.Delete
6.Exit
Enter your choice:3
Enter Item Code to search:13
Found.
***** Menu *****
1.Insert
2.Display
3.Search
4.Sort
5.Delete
6.Exit
Enter your choice:4
Sorted on Cost
Item Name:Compass
Item Quantity:25
Item Cost:120
Item Code:13
Item Name:Book
Item Quantity:10
Item Cost:500
Item Code:4
***** Menu *****
1.Insert
2.Display
3.Search

4.Sort
5.Delete
6.Exit
Enter your choice:5
Enter Item Code to delete:13
Deleted.
***** Menu *****
1.Insert
2.Display
3.Search
4.Sort
5.Delete
6.Exit
Enter your choice:2
Item Name:Book
Item Quantity:10
Item Cost:500
Item Code:4
***** Menu *****
1.Insert
2.Display
3.Search
4.Sort
5.Delete
6.Exit
Enter your choice:6

/*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
*/

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

   populationMap.insert({"Maharashtra", 11});
   populationMap.insert({"Punjab", 2});
   populationMap.insert({"Tamil Nadu        ", 7});
   populationMap.insert({"Goa", 1});
   populationMap.insert({"Rajasthan", 6});

   map<string, int>::iterator iter;
   string state;
   int flag = 0;
   cout<<"Enter State Population You Want : ";
   cin>>state;

   cout << "Size of population map: " << populationMap.size() << '\n';
   for (iter = populationMap.begin(); iter != populationMap.end(); ++iter)
   {
      if(iter->first == state){
      cout<<"\n" << iter->first << " : "<< iter->second << " Crore\n";
      flag = 1;
      }
   }

   if(flag == 0)
      cout<<"\nState Not Found!!";



   populationMap.clear();
}
```

# Output

```
Enter State Population You Want : Maharashtra
Size of population map: 5

Maharashtra : 11Crore
```