## INPUT:

#include <iostream> #include <stdio.h> #include <stdlib.h> using namespace std; class complex

{

# ASSIGNMENT NO.: 1

## ROLL NO.: 102

float x; float y;

public:

complex operator+(complex); complex operator-(complex); complex operator\*(complex); complex operator/(complex); complex()

{

x = 0;

y = 0;

}

complex(float a, float b)

{

x = a; y = b;

}

friend void operator<<(ostream &out, complex &d)

{

cout << "\n" << d.x << "+" << d.y << "i";

}

friend istream &operator>>(istream &input, complex &num)

{

input >> num.x >> num.y; return input;

}

};

complex complex::operator+(complex c)

{

complex temp; temp.x = x + c.x; temp.y = y + c.y; return (temp);

}

complex complex::operator\*(complex c)

{

complex temp2;

temp2.x = (x \* c.x) - (y \* c.y);

temp2.y = (y \* c.x) + (x \* c.y); return (temp2);

}

int main()

{

complex c1, c2, c3, c4; int ch;

do

{

cout << "\n=========================";

cout << "\n 1.Create constructor"; cout << "\n 2.Accept data";

cout << "\n 3.Display number"; cout << "\n 4.Add Number"; cout << "\n 5.Multiplication"; cout << "\n 6.Exit";

cout << "\n Enter your choice :"; cin >> ch;

cout << "\n=========================";

switch (ch)

{

case 1:

cout << "\n Complex number after initialization"; cout << c1;

cout << c2; break;

case 2:

cout << "\nEnter first complex number:"; cin >> c1;

cout << "\nEnter second complex number:";

cin >> c2; break;

case 3:

cout << "\n First complex number:"; cout << c1;

cout << "\n Second complex number:"; cout << c2;

break; case 4:

c3 = c1 + c2;

cout << "\n addition is "; cout << c3;

break; case 5:

c4 = c1 \* c2;

cout << "\n multiplication is "; cout << c4;

break; default:

if (ch != 6)

{

cout << "\n Wrong choice ";

break;

}

}

} while (ch != 6);

return 0;

}

## OUPUT:

=========================

1.Create constructor 2.Accept data 3.Display number 4.Add Number 5.Multiplication 6.Exit

Enter your choice :1

=========================

Complex number after initialization 0+0i

0+0i

=========================

1.Create constructor 2.Accept data 3.Display number 4.Add Number 5.Multiplication 6.Exit

Enter your choice :2

=========================

Enter first complex number:12 31

Enter second complex number:10 30

=========================

1.Create constructor 2.Accept data 3.Display number 4.Add Number 5.Multiplication 6.Exit

Enter your choice :3

=========================

First complex number: 12+31i

Second complex number: 10+30i

=========================

1.Create constructor 2.Accept data 3.Display number 4.Add Number 5.Multiplication 6.Exit

Enter your choice :4

=========================

addition is 22+61i

=========================

1.Create constructor 2.Accept data 3.Display number 4.Add Number 5.Multiplication 6.Exit

Enter your choice :5

=========================

multiplication is

-810+670i

=========================

1.Create constructor 2.Accept data 3.Display number 4.Add Number 5.Multiplication 6.Exit

Enter your choice :6

=========================

## INPUT:

#include <iostream> using namespace std; class data

{

private:

string name; char bg;

string address; string dob; string cldiv; string lic; static int c;

int roll;

long int phone;

# ASSIGNMENT NO.: 2

## ROLL NO.: 102

public:

data();

static int getcount(); void getdata();

void show(); data(data \*obj);

data(int roll, long int phone, string name, string address, string dob, string cldiv, string lic);

~data();

};

int data::c = 0; data::data(data \*obj)

{

cout << "Copy constructor implemented" << endl;

}

data::~data()

{

cout << "Destructor called \n";

}

void data::getdata()

{

cout << "Enter roll number \n"; cin >> roll;

cout << "Enter telephone number \n"; cin >> phone;

cout << "Enter Name \n"; ws(cin);

cin >> name;

cout << "Enter address \n"; ws(cin);

cin >> address;

cout << "Enter D.O.B \n"; cin >> dob;

cout << "Enter Class and Division\n"; ws(cin);

cin >> cldiv;

cout << "Enter license number \n"; cin >> lic;

}

void data::show()

{

cout << "Name :" << name << endl; cout << "Roll Number :" << roll << endl;

cout << "Telephone Number :" << phone << endl; cout << "Address :" << address << endl;

cout << "Date of birth :" << dob << endl;

cout << "Class and division: " << cldiv << endl; cout << "License number :" << lic << endl;

}

data::data(int roll, long int phone, string name, string address, string dob, string cldiv, string lic)

{

cout << "Parameterized Constructor" << endl; c++;

this->roll = roll;

this->phone = phone; this->name = name;

this->address = address; this->dob = dob;

this->cldiv = cldiv; this->lic = lic;

}

data::data()

{

roll = 0;

phone = 0; name = "Name";

address = "Address"; dob = "DOB";

cldiv = "Class and Division"; lic = "License no.";

cout << "Default Constructor" << endl; c++;

}

int data::getcount()

{

return c;

}

int main()

{

int num;

data \*d1 = new data(); d1->show();

delete d1;

data \*d2 = new data(23, 9822794182, "Dhruvatara", "Mahatma Nagar", "22.02.97", "SE A", "MSIN92U");

d2->show();

data \*d3 = new data(d2);

d3->show(); delete d2;

cout << "Enter size of database"; cin >> num;

data dx[num];

for (int i = 0; i < num; i++)

{

dx[i].getdata();

}

for (int i = 0; i < num; i++)

{

dx[i].show();

}

cout << "Number of constructor calls and total number of objects:" << data::getcount() << endl;

return 0;

}

## OUTPUT:

Default Constructor Name :Name

Roll Number :0

Telephone Number :0 Address :Address Date of birth :DOB

Class and division: Class and Division License number :License no.

Destructor called Parameterized Constructor Name :Dhruvatara

Roll Number :23

Telephone Number :1232859590 Address :Mahatma Nagar

Date of birth :22.02.97 Class and division: SE A License number :MSIN92U

Copy constructor implemented Name :

Roll Number :0

Telephone Number :0 Address :

Date of birth : Class and division: License number :

Destructor called

Enter size of database2 Default Constructor Default Constructor Enter roll number

234

Enter telephone number 23451769

Enter Name Sara

Enter address Baramati Enter D.O.B 17/09/2003

Enter Class and Division

S.E A

Enter license number Enter roll number 432

Enter telephone number 86376255

Enter Name Sakshi

Enter address malegaon Enter D.O.B

12/04/2003

Enter Class and Division

S.E A

Enter license number Name :Sara

Roll Number :234

Telephone Number :23451769 Address :Baramati

Date of birth :17/09/2003 Class and division: S.E License number :A

Name :Sakshi Roll Number :432

Telephone Number :86376255 Address :malegaon

Date of birth :12/04/2003 Class and division: S.E License number :A

Number of constructor calls and total number of objects:4 Destructor called

Destructor called

## INPUT:

#include <iostream> #include <string> #include <conio.h> using namespace std; class publication

{

private: string title; float price;

public:

void getdata()

{

# ASSIGNMENT NO.: 3

## ROLL NO.: 102

cout << "Enter title of publication: "; cin >> title;

cout << "Enter price of publication: "; cin >> price;

}

void putdata()

{

cout << "Publication title: " << title << endl; cout << "Publication price: " << price << endl;

}

};

class book : public publication

{

private:

int pagecount;

public:

void getdata()

{

publication::getdata();

cout << "Enter Book Page Count: "; cin >> pagecount;

}

void putdata()

{

publication::putdata();

cout << "Book page count: " << pagecount << endl;

// Show book data

}

};

class tape : public publication

{

private:

float ptime;

public:

void getdata()

{

publication::getdata();

cout << "Enter tap's playing time: "; cin >> ptime;

}

void putdata()

{

publication::putdata();

cout << "Tap's playing time: " << ptime << endl;

}

};

int main()

{

book b; tape t; b.getdata();

t.getdata();

b.putdata();

t.putdata(); return 0;

}

## OUPUT:

Enter title of publication: Summer Enter price of publication: 230 Enter Book Page Count: 1500 Enter title of publication: Dream Enter price of publication: 340 Enter tap's playing time: 2 Publication title: Summer Publication price: 230

Book page count: 1500 Publication title: Dream Publication price: 340 Tap's playing time: 2

## INPUT:

#include <iostream> #include <fstream> using namespace std; class file

{

# ASSIGNMENT NO.: 4

## ROLL NO.:102

char name[40]; int emp\_id; float salary;

public:

void accept()

{

cin >> name; cin >> emp\_id; cin >> salary;

}

void display()

{

cout << "\n"

<< name << "\t" << emp\_id << "\t" << salary;

}

};

int main()

{

file obj[5]; fstream f; int i, n;

f.open("input.txt", ios::out);

cout << "\nHow many employee information want to store:"; cin >> n;

cout << "\nEnter information of employee(name employee\_id salary)"; for (i = 0; i < n; i++)

{

cout << "\nEnter information of " << i + 1 << " employee:"; obj[i].accept();

f.write((char \*)&obj[i], sizeof(obj[i]));

}

f.close(); f.open("input", ios::in);

cout << "\nEnter information of employee is "; for (i = 0; i < n; i++)

{

f.read((char \*)&obj[i], sizeof(obj[i])); obj[i].display();

}

f.close(); return 0;

}

## OUPUT:

How many employee information want to store:2

Enter information of employee(name employee\_id salary) Enter information of 1 employee:Sanskruti

123

200000

Enter information of 2 employee:Sanchita 134

300000

Enter information of employee is Sanskruti 123 200000

Sanchita 134 300000

## INPUT:

#include <iostream> using namespace std;

template <class T> void sort()

{

# ASSIGNMENT NO.: 5

## ROLL NO.:102

int i, j, min, n; T temp;

T a[10];

cout << "\n Enter number of element you want to sort: "; cin >> n;

for (i = 0; i < n; i++)

{

cin >> a[i];

}

for (i = 0; i < n; i++)

{

min = i;

for (j = i + 1; j < n; j++)

{

if (a[j] < a[min])

{

min = j;

}

}

if (min != i)

{

temp = a[i]; a[i] = a[min]; a[min] = temp;

}

}

cout << "\n The array in the sorted order is : " << endl; for (i = 0; i < n; i++)

{

cout << "\t" << a[i];

}

}

int main()

{

int choice; char ans; do

{

cout << "\n 1. Integer sort. \n 2. Float sort."; cout << "\n Enter the input you want to sort : "; cin >> choice;

switch (choice)

{

case 1:

sort<int>();

break; case 2:

sort<float>(); break;

case 3:

cout << "\n Invalid choice."; break;

default:

cout << "\nEnter valid choice.."; break;

}

cout << "\n Do u wish to continue (Y/N)?"; cin >> ans;

} while (ans == 'Y' || ans == 'y'); return 0;

}

## OUPUT:

1. Integer sort.
2. Float sort.

Enter the input you want to sort : 1

Enter number of element you want to sort: 4 9

1

3

7

The array in the sorted order is : 1 3 7 9

Do u wish to continue (Y/N)? y

1. Integer sort.
2. Float sort.

Enter the input you want to sort : 2

Enter number of element you want to sort: 5 11.3

66.5

3.4

72.9

32.8

The array in the sorted order is : 3.4 11.3 32.8 66.5 72.9

Do u wish to continue (Y/N)? n

## INPUT:

#include <iostream> #include <list>

using namespace std; class record

{

# ASSIGNMENT NO.: 6

## ROLL NO.: 102

list<string> name, dob, phone, ni; list<string>::iterator it1, it2, it3, j, k, l, c, n; list<string> code;

list<int> number; list<float> cost; list<int>::iterator no, j1; list<float>::iterator f, i;

public:

void getp(); void display();

void searchp(string); void sortp();

void checkempty(); void getlist();

void displayit(); void searchlist(); void sortitem();

};

void record::getp()

{

int count; string n, d, p;

cout << "Enter the number of members in record:" << endl; cin >> count;

for (int i = 1; i <= count; i++)

{

cout << "Enter name:" << endl; cin >> n;

name.push\_back(n);

cout << "Enter date of birth:" << endl; cin >> d;

dob.push\_back(d);

cout << "Enter phone number:" << endl; cin >> p;

phone.push\_back(p);

}

}

void record::searchp(string data)

{

int flag = 0;

it1 = name.begin(); it2 = dob.begin();

it3 = phone.begin();

while (it1 != name.end() && it2 != dob.end() && it3 != phone.end())

{

if (\*it1 == data)

{

cout << "Record found!" << endl;

cout << "Corresponding D.O.B: " << \*it2 << endl;

cout << "Corresponding phone number: " << \*it3 << endl; flag = 1;

break;

}

if (\*it2 == data)

{

cout << "Record found!" << endl;

cout << "Corresponding name " << \*it1 << endl;

cout << "Corresponding phone number: " << \*it3 << endl; flag = 1;

break;

}

if (\*it3 == data)

{

cout << "Record found!" << endl;

cout << "Corresponding name: " << \*it1 << endl; cout << "Corresponding D.O.B: " << \*it2 << endl; flag = 1;

break;

}

it1++; it2++;

it3++;

}

if (flag == 0)

cout << "Record not found." << endl;

}

void record::display()

{

it1 = name.begin(); it2 = dob.begin(); it3 = phone.begin();

while (it1 != name.end())

{

cout << \*it1 << "\t" << \*it2 << "\t" << \*it3 << endl; it1++;

it2++; it3++;

}

}

void record::sortp()

{

string temp;

it1 = name.begin(); it2 = dob.begin(); it3 = phone.begin(); j = it1;

k = it2;

l = it3; j++; k++; l++;

while (it1 != name.end())

{

while (j != name.end())

{

if (\*it1 > \*j)

{

temp = \*it1;

\*it1 = \*j;

\*j = temp; temp = \*it2;

\*it2 = \*k;

\*k = temp; temp = \*it3;

\*it3 = \*l;

\*l = temp;

}

j++; k++; l++;

}

it1++; it2++; it3++;

}

}

void record::getlist()

{

cout << "Enter the number of items:" << endl; int c, no;

string n; float f; cin >> c;

for (int i = 1; i <= c; i++)

{

cout << "Enter item name:" << endl; cin >> n;

ni.push\_back(n);

cout << "Enter item code:" << endl; cin >> n;

code.push\_back(n);

cout << "Enter cost:" << endl; cin >> f;

cost.push\_back(f);

cout << "Enter the quantity:" << endl; cin >> no;

number.push\_back(no);

}

}

void record::displayit()

{

c = code.begin(); n = ni.begin();

no = number.begin(); f = cost.begin();

while (c != code.end())

{

cout << \*c << "\t" << \*n << "\t" << \*no << "\t" << \*f << endl; c++;

n++;

no++; f++;

}

}

void record::sortitem()

{

string temp; int tempno;

float tempf;

c = code.begin(); n = ni.begin();

no = number.begin(); f = cost.begin();

i = f;

j1 = no;

k = c; l = n; i++; j1++; k++; l++;

while (f != cost.end())

{

while (i != cost.end())

{

if (\*f > \*i)

{

tempf = \*f;

\*f = \*i;

\*i = tempf;

temp = \*n;

\*n = \*l;

\*l = temp;

temp = \*c;

\*c = \*k;

\*k = temp;

tempno = \*no;

\*no = \*j1;

\*j1 = tempno;

}

i++; j1++; k++; l++;

}

f++; n++;

no++; c++;

}

}

void record::searchlist()

{

string key;

cout << "Enter the item code:" << endl;

cin >> key;

c = code.begin(); n = ni.begin();

no = number.begin(); f = cost.begin();

while (c != code.end())

{

if (key == \*c)

{

cout << "Item available!" << endl; cout << "Item name: " << \*n << endl;

cout << "Item quantity: " << \*no << endl;

cout << "Item cost: " << \*f << endl;

}

c++; n++;

no++; f++;

}

}

int main()

{

record obj; string key; int ch, chr;

char x = 'y'; do

{

cout << "1. Personal record\n2. Item record\nEnter choice:\n"; cin >> ch;

do

{

if (ch == 1)

{

cout << "1. Enter details\n2. Display\n3. Search entry\n4. Sort records\nEnter choice\n";

cin >> chr; switch (chr)

find\n";

{

case 1:

obj.getp(); obj.display(); break;

case 2:

obj.display(); break;

case 3:

cout << "Enter either name, d.o.b or phone number you want to

cin >> key; obj.searchp(key);

break; case 4:

obj.sortp(); obj.display(); break;

default:

cout << "Wrong choice" << endl;

}

}

else if (ch == 2)

{

cout << "1. Enter details\n2. Display\n3. Search entry\n4. Sort records\nEnter choice\n";

cin >> chr; switch (chr)

{

case 1:

obj.getlist(); obj.displayit(); break;

case 2:

obj.displayit(); break;

case 3:

obj.searchlist(); break;

case 4:

obj.sortitem(); obj.displayit(); break;

default:

cout << "Wrong choice" << endl;

}

}

else

{

cout << "Wrong choice" << endl; break;

}

cout << "Do you wish to continue? Y or N\n"; cin >> x;

} while (x == 'y' || x == 'Y');

cout << "Do you wish to select another type of record? Y or N\n"; cin >> x;

} while (x == 'y' || x == 'Y'); return 0;

}

## OUPUT:

1. Personal record
2. Item record Enter choice:

1

1. Enter details
2. Display
3. Search entry
4. Sort records Enter choice

1

Enter the number of members in record: 2

Enter name:

SUMIT

Enter date of birth: 30/10/2004

Enter phone number: 97354264

Enter name:

SUMAN

Enter date of birth: 23/04/2004

Enter phone number: 886623453

SUMIT 30/10/2004 97354264

SUMAN 23/04/2004 886623453

Do you wish to continue? Y or N Y

1. Enter details
2. Display
3. Search entry
4. Sort records Enter choice

2

SUMIT 30/10/2004 97354264

SUMAN 23/04/2004 886623453

Do you wish to continue? Y or N Y

1. Enter details
2. Display
3. Search entry
4. Sort records Enter choice

3

Enter either name, d.o.b or phone number you want to find SUMAN

Record found!

Corresponding D.O.B: 23/04/2004 Corresponding phone number: 886623453 Do you wish to continue? Y or N

Y

1. Enter details
2. Display
3. Search entry
4. Sort records Enter choice

4

SUMAN 23/04/2004 886623453

SUMIT 30/10/2004 97354264

Do you wish to continue? Y or N N

Do you wish to select another type of record? Y or N Y

1. Personal record
2. Item record Enter choice:

2

1. Enter details
2. Display
3. Search entry
4. Sort records Enter choice

1

Enter the number of items: 2

Enter item name: PAPERS

Enter item code:

2340

Enter cost:

23

Enter the quantity:

50

Enter item name:

PAINT

Enter item code:

456

Enter cost:

450

Enter the quantity:

20

2340 PAPERS 50 23

456 PAINT 20 450

Do you wish to continue? Y or N Y

1. Enter details
2. Display
3. Search entry
4. Sort records Enter choice

2

2340 PAPERS 50 23

456 PAINT 20 450

Do you wish to continue? Y or N Y

1. Enter details
2. Display
3. Search entry
4. Sort records Enter choice

3

Enter the item code:

456

Item available! Item name: PAINT Item quantity: 20

Item cost: 450

Do you wish to continue? Y or N Y

1. Enter details
2. Display
3. Search entry
4. Sort records Enter choice

4

2340 PAPERS 50 23

456 PAINT 20 450

Do you wish to continue? Y or N N

Do you wish to select another type of record? Y or N N

**Assignment No : 7**

**Roll no : 102**

**Input :**

**Program : 1**

**Input :**

#include <map>

#include <iostream>

#include<string>

using namespace std;

int main()

{

map<int,string>mymap;

mymap[1] = "Hi";

mymap[2] = "This";

mymap[3] = "is";

mymap[4] = "DYP";

cout<<mymap[4];

return 0;

}

**OUTPUT :**

DYP

**Program : 2**

**Input:**

#include<iostream>

#include<map>

#include<string>

using namespace std;

int main()

{

typedef map<string,int> mapType;

mapType populationMap;

populationMap.insert(pair<string, int>("Maharashtra", 7026357));

populationMap.insert(pair<string, int>("Rajasthan", 6578936));

populationMap.insert(pair<string, int>("Karanataka", 6678993));

populationMap.insert(pair<string, int>("Punjab", 5789032));

populationMap.insert(pair<string, int>("West Bengal", 6676291));

mapType::iterator iter;

cout<<"========Population of states in India==========\n";

cout<<"\n Size of populationMap"<<populationMap.size()<<"\n";

string state\_name;

cout<<"\n Enter name of the state :";

cin>>state\_name;

iter = populationMap.find(state\_name);

if( iter!= populationMap.end() )

cout<<state\_name<<" 's population is "<<iter->second ;

else

cout<<"Key is not populationMap"<<"\n";

populationMap.clear();

}

**Output :**

========Population of states in India==========

Size of populationMap5

Enter name of the state :Maharashtra

Maharashtra 's population is 7026357