Assignment No 1

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.

Code:

```
#include<iostream> //including header files

using namespace std; //declaring the scope of program

class complex //class name "complex" {

public: float real,img; //declared variable of type float

complex() //default constructor {

} complex operator+ (complex);

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

friend istream &operator<<(istream &,complex&); }; complex complex:: operator + (complex obj)

{ complex temp;

temp.real=real+obj.real; temp.img=img+obj.img;

return (temp); }

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

is>>obj.real; is>>obj.img;

return is; }
```

```
ostream &operator<<(ostream &outt,complex &obj) {
outt<<""<<obj.real; outt<<"+"<<obj.img<<"i";
return outt; }
complex complex :: operator * (complex obj) {
complex temp; temp.real=real*obj.real-img*obj.img;
temp.img=real*obj.img+img*obj.real; return (temp);
} int main()
{ complex a,b,c,d;
cout<<"\nEnter first complex number\n"; cout<<"\nEnter real and imaginary:\t";
cin>>a;
```

Output:

```
Enter first complex number
Enter real and imaginary: 2 3
Enter second complex number
Enter real and imaginary: 4 6
Arithmetic operations
Addition =6+9i
Multiplication=-10+24i cout<<"Enter second complex number \n";
cout<<"\nEnter real and imaginary:\t"; cin>>b;
cout<<"\n\tArithmetic operations"; c=a+b;
cout<<"\n\tAddition ="; cout<<c;
d=a*b; cout<<"\n\tMultiplication=";
cout<<d; cout<<endl;
return 0;
}
```

ASSINGMENT NO 2

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

Code:

```
#include<iostream>
#include<string.h>
#include<iomanip>
using namespace std;
class db
{
 int roll;
 char Name[20];
 char Class[10];
 char Div[10];
 char dob[10];
 char contact[10],bg[3];
 char phone[10],license[12];
  public:
 static int stdno;
 static void count()
```

```
{
   cout<<"\nNo. of objects created: "<<stdno;
 }
 void fin()
{
cout<<"\nInline Function !";
}
  db()
  {
     roll=0;
     strcpy(Name,"Sachin");
     strcpy(Class, "I");
     strcpy(Div,"A");
    strcpy(dob,"11/12/2003");
    strcpy(bg,"A");
    strcpy(contact,"city");
    strcpy(phone,"8787505011");
    strcpy(license,"A404040");
    ++stdno;
   }
  db(db *ob)
 {
       strcpy(Name,ob->Name);
   strcpy(dob,ob->Class);
   strcpy(Class,ob->Div);
   strcpy(bg,ob->bg);
   strcpy(contact,ob->contact);
   strcpy(license,ob->license);
   ++stdno;
 }
```

```
void getdata()
 {
   cout<<"\n\nEnter:
Name,roll,Class,Div,dob,bg,contact,phone,lincense \n\n\n";

cin>>Name>>roll>>Class>>Div>>dob>>bg>>contact>>phone>>license
;
 }
friend void display(db d);
~db()
   {
  cout<<"\n\n"<<this->Name<<"(Object) is destroyed!";
   }
};
 void display(db d)
{

cout<<"\n\n"<<setw(12)<<d.Name<<setw(5)<<d.roll<<setw(4)<<d.Cl
ass<<setw(3)<<d.Div<<setw(12)<<d.dob<<setw(4)<<d.contact<<"
"<<setw(12)<<d.phone<<" "<<setw(12)<<"  "<<d.license;
}
 int db::stdno;
int main()
{
   int n,i;
   db d1,*ptr[5];
   cout<<"\n\nUse Values:";
   display(d1);
   d1.getdata();
   display(d1);
   db d2(&d1);
```

```
  cout<<"\n\nUse of copy constructor :\n";
   display (d2);
  cout<<"\nHow many objects you want to create? :" ;
  cin>>n;
  for(i=0;i<n;i++)
  {
   ptr[i]=new db();
    ptr[i]->getdata();
   }
cout<<"\n"<<setw(12)<<"Name"<<setw(5)<<"roll"<<setw(4)<<"Div"
<<setw(12)<<"dob"<<setw(4)<<"bg"<<setw(12)<<"contact"<<setw(1
2)<<"phone"<<setw
 (12)<<"license";
for(i=0;i<n;i++)
display(*ptr[i]);
 db::count();
 for(i=0;i<n;i++);
 {
  delete(ptr[i]);
 }
 cout<<"\nObjects deleted!" ;
 return 0;
}
```

Output:

```
Use Values:
    Sachin  0  I  A11/12/2003citycity 8787505011A404040
A404040
Sachin(Object) is destroyed!
Enter: Name,roll,Class,Div,dob,bg,contact,phone,lincense
xyz
```

20

1

c

10/10/2003

adas

jnisd

8945612377

ae7845346

xyz 20 1 c10/10/2003jnisdjnisd 8945612377ae7845346 ae7845346

xyz(Object) is destroyed!

Use of copy constructor :

xyz 4096 c 1jnisd 8945612377ae7845346 ae7845346

xyz(Object) is destroyed!

How many objects you want to create? :

1

Enter: Name,roll,Class,Div,dob,bg,contact,phone,lincense

abc

4

2

c

10/10/2003

gyud

yt

1452639870

ac784613

** Process exited - Return Cod

oop

Assigment No 3

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.

Code:

```
#include<iostream>
#include<string>
using namespace std;
//base class publication
class publication
{
private:
string title;
float prices;
public:
publication()
{
title="";
prices=0.0;
}
void get_data()
```

```
{
cout<<"\nEnter Title :";
cin.ignore();//clear input buffer
getline(cin,title);
cout<<"\nEnter Price : ";
cin>>prices;
}
void put_data()
{
cout<<"\n _____ \n";
cout<<"\n Information : " <<endl;
cout<<"\n Title :"<<title;
cout<<"\n Price :"<<prices;
}
};
class book: public publication
{
private:
int pages;
public:
book(){
pages=0;
}
void get_data()
{
publication::get_data();
cout<<endl;
cout<<"Enter Page Count : \n";
cin>>pages;
}
```

```
void put_data()
{
try{
if(pages==0)
throw pages;}
catch(int f)
{
cout<<"\n error: pages not valid :"<<f;
pages=0;
}
cout<<"\n Pages Are :"<<pages;
publication::put_data();
}
};
class tape: public publication
{
private:
float playtime;
public:
tape()
{
playtime=0.0;
}
void get_data()
{
publication::get_data();
cout<<"Enter Play Time Of Cassette \n";
cin>>playtime;
}
void put_data()
```

```
{
try
{
if(playtime==0.0)
throw playtime;
}
catch(float r)
{
cout<<"\n Error: Invalid Playtime : "<<playtime;
playtime=0.0;
}
cout<<"\n Playtime is : "<<playtime;
publication::put_data();
}
};
int main()//main program
{
book b[10];// arrray of objects
tape t[10];
int choice=0,bookCount=0,tapeCount=0;
cout<<"----------------------";
do
{
cout<<"\n 1. Add book ";
cout<<"\n 2. Add tape: ";
cout<<"\n 3. Display book ";
cout<<"\n 4. Display tape";
cout<<"\n 5. Exit:"<<endl;
cout<<"\n Enter Choice : ";
cin>>choice;
```

```
switch(choice)
{
case 1:
{
cout<<"\n--------------\n";
cout<<"Add Book: \n";
b[bookCount].get_data();
bookCount++;
break;
}
case 2:
{
cout<<"\n--------------\n";
cout<<"Add Tape: \n";
t[tapeCount].get_data();
tapeCount++;
break;
}
case 3:
{
cout<<"\n (books)";
for(int j=0;j<bookCount;j++)
{
b[j].put_data();
}
break;
}
case 4:
{
cout<<"\n (tape)";
```

```
for(int j=0;j<tapeCount;j++)
{
t[j].put_data();
}
break;
}
case 5:
{
cout<<"**********Program Exited Successfully**********"<<endl;
exit(0);
}
default:
{
cout<<"\n Invalid";
}
}
}
while(choice!=5);
return 0;
}
```

Output:

```
----------------------
1. Add book
2. Add tape:
3. Display book
4. Display tape
5. Exit:
Enter Choice : 1
--------------
```

Add Book:

Enter Title : dream

Enter Price : 20

Enter Page Count :

50

1. Add book

2. Add tape:

3. Display book

4. Display tape

5. Exit:

Enter Choice : 2

Add Tape:

Enter Title :OOP

Enter Price : 100

Enter Play Time Of Cassette

40

1. Add book

2. Add tape:

3. Display book

4. Display tape

5. Exit:

Enter Choice : 3

(books)

Information :

Title :dream

Price :20

Pages Are :50

1. Add book

2. Add tape:

3. Display book

4. Display tape

5. Exit:

Enter Choice : 4

----

1. Add book

2. Add tape:

3. Display book

4. Display tape

5. Exit:

Enter Choice : 2

Add Tape:

Enter Title :gd

Enter Price : 88

Enter Play Time Of Cassette

0.0

1. Add book

2. Add tape:

3. Display book

4. Display tape

5. Exit:

Enter Choice : 4

(tape)

Error: Invalid Playtime : 0

Playtime is : 0

Information :

Title :gd

Price :88

1. Add book

2. Add tape:

3. Display book

4. Display tape

5. Exit:

Enter Choice :5

oop

Assignment No 4

Write a c++ program that creates an output file , writes information to file ,closes the file and

open it again as an input file, and read info from file.

CODE:-

```
#include<iostream>
#include<fstream>
#include<stdlib.h>
using namespace std;
class student
{
private:
int roll;
float marks;
char name[20];
public:
void accept()
{
cout<<"\nEnter roll no:";
cin>>roll;
cout<<"\nEnter marks of student:";
cin>>marks;
cout<<"\nEnter name of student:";
cin>>name;
}
void display()
{
```

```
cout<<"\nName : "<<name;
cout<<"\nRoll no: "<<roll;
cout<<"\nMarks are: "<<marks;
}
};
int main()
{
int ch,n,i;
fstream f;
student s[10];
while(1)
{
cout<<"\nMenu :";
cout<<"\n1.Create :";
cout<<"\n2.write :";
cout<<"\n3.read :";
cout<<"\n4.append :";
cout<<"\n5.exit :";
cout<<"\nEnter ur choice:";
cin>>ch;
switch(ch)
{
case 1:
f.open("file.txt",ios::out);
cout<<"\nFile is successfully created :";
f.close();
break;
case 2:
f.open("file.txt",ios::out | ios::app);
cout<<"\nhow many students do u want to enter:";
```

```
cin>>n;
for(i=0;i<n;i++)
{
s[i].accept();
f.write((char*)& s,sizeof (s));
}
f.close();
break;
case 3:
f.open("file.txt",ios::in );
for(i=0;i<n;i++)
{
f.read((char*)& s,sizeof (s));
s[i].display();
}
f.close();
break;
case 4:
{
f.open("file.txt",ios::app);
s[i].accept();
f.write((char*)& s,sizeof (s));
n++;
f.close();
}
break;
case 5:
exit(0);
}
}
```

}

OUTPUT:-

Menu :

1.Create :

2.write :

3.read :

4.append :

5.exit :

Enter ur choice:1

File is successfully created :

Menu :

1.Create :

2.write :

3.read :

4.append :

5.exit :

Enter ur choice:2

how many students do u want to enter:2

Enter roll no:1

Enter marks of student:89

Enter name of student:xyz

Enter roll no:4

Enter marks of student:90

Enter name of student:pqr

Menu :

1.Create :

2.write :

3.read :

4.append :

5.exit :

Enter ur choice:3

Name : xyz

Roll no: 1

Marks are: 89

Name : pqr

Roll no: 4

Marks are: 90

Menu :

1.Create :

2.write :

3.read :

4.append :

5.exit :

Enter ur choice:4

Enter roll no:3

Enter marks of student:98

Enter name of student:xyz

Menu :

1.Create :

2.write :

3.read :

4.append :

5.exit :

Enter ur choice:5

oop

Assignment No 5

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

CODE:-

```
#include <iostream>
#include<stdlib.h>
#include<string.h>
using namespace std;
template<typename T>
void sort(T a[],int n)
{
int pos;
T temp;
for(int i=0;i<n-1;i++)
{
pos=i;
for(int j=i+1;j<n;j++)
{
if (a[j]<a[pos])
pos=j;
}
if (pos!=i)
{
temp=a[i];
a[i]=a[pos];
a[pos]=temp;
}
}
//display ele
cout<<"sorted elements are ";
for(int k=0;k<n;k++)
{
cout<<"\n"<<a[k];
}
}
int main()
{
int ch;
```

```
int no;
int ele;
int fno;
while(1)
{
cout<<"\nmenu \n1.int sorting\n2.char sorting\n3.float
sorting\n4.exit";
cout<<"\nEnter choice";
cin>>ch;
switch(ch)
{
case 1:
cout<<"\ninteger sorting\n";
cout<<"\nenter no of element to be sorted\n";
cin>>no;
int sor1[100];
cout<<"\nenter element\n";
for(int i=0;i<no;i++)
{
cin>>sor1[i];
}
sort<int>(sor1,no);
break;
case 2:
cout<<"\ncharacter sorting\n";
cout<<"\nenter no of element to be sorted\n";
cin>>ele;
char sor2[100];
cout<<"\nenter element\n";
for(int i=0;i<ele;i++)
{
cin>>sor2[i];
}
sort<char>(sor2,ele);
break;
case 3:
cout<<"\nfloating type sorting\n";
cout<<"\nenter no of element to be sorted\n";
cin>>fno;
float sor3[100];
cout<<"\nenter element\n";
for(int i=0;i<fno;i++)
{
cin>>sor3[i];
}
sort<float>(sor3,fno);
```

```
break;
case 4:
exit(0);
}
}
return 0;
}
```

OUTPUT:-

```
menu
1.int sorting
2.char sorting
3.float sorting
4.exit
Enter choice3
floating type sorting

enter no of element to be sorted
3
enter element
7.7
3.4
1.2
sorted elements are
1.2
3.4
7.7
menu
1.int sorting
2.char sorting
3.float sorting
4.exit
Enter choice:4
```

oop

Assignment No 6

Write C++ program using STL for Sorting and searching with user-defined records

such as item record (item code, item name, quantity and cost)using vector

container

Code:

```
#include <iostream> //standard input output stream header file

#include <algorithm> //The STL algorithms are generic because they can operate on a

variety of data structures

#include <vector> //The header file for the STL vector library is vector.

using namespace std;

class Item

{

public:

char name[10];

int quantity;

int cost;

int code;

bool operator==(const Item& i1) //Boolean operators allow you to

create more complex conditional statements

{

if(code==i1.code) //operator will return 1 if the comparison is true, or 0 if

the comparison is false

return 1;
```

```
return 0;
}
bool operator<(const Item& i1)
{
if(code<i1.code) //operator will return 1 if the comparison is true, or 0 if
the comparison is false
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)
{
//if (i1.name != i2.name) return i1.cost < i2.cost;
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:abc

Enter Item Quantity:2

Enter Item Cost:56

Enter Item Code:8465

******* Menu *******

1.Insert

2.Display

3.Search

4.Sort

5.Delete

6.Exit

Enter your choice:2

Item Name:abc

Item Quantity:2

Item Cost:56

Item Code:8465

******* Menu *******

1.Insert

2.Display

3.Search

4.Sort

5.Delete

6.Exit

Enter your choice:3

Enter Item Code to search:8465

Found.

******* Menu *******

1.Insert

2.Display

3.Search

4.Sort

5.Delete

6.Exit

Enter your choice:3

Enter Item Code to search:8946

Not found.

******* Menu *******

1.Insert

2.Display

3.Search

4.Sort

5.Delete

6.Exit

Enter your choice:4

Sorted on Cost

Item Name:abc

Item Quantity:2

Item Cost:56

Item Code:8465

******* Menu *******

1.Insert

2.Display

3.Search

4.Sort

5.Delete

6.Exit

Enter your choice:5

Enter Item Code to delete:8465

Deleted.

******* Menu *******

1.Insert

2.Display

3.Search

4.Sort

5.Delete

6.Exit

Enter your choice:6}

Option B:

Use STL for Sorting and searching with user-defined records such as Person Record (Name,

birth date, telephone no).

Code:

```
#include <algorithm>
#include <iostream>
#include <vector>
#include <string>
using namespace std;
class student
{
```

```
public:
int rollno;
string name;
char dob[15];
bool operator==(const student &student1)
{
return(rollno==student1.rollno);
}
bool operator<(const student &student1)
{
return(rollno<student1.rollno);
}
friend ostream& operator << (ostream &out, const student &);
friend istream& operator >> (istream &in, student &k);
};
ostream & operator << (ostream &out, const student &k)
{
out<<"\n\t\t"<<k.rollno<<"\t\t"<<k.name<<"\t\t"<<k.dob;
return out;
}
istream & operator >> (istream &in , student &k)
{
cout<<"\nEnter Roll No : ";
in>>k.rollno;
cout<<"\nEnter Name : ";
in>>k.name;
cout<<"\nEnter DOB : ";
in>>k.dob;
return in;
}
```

```
bool myfunc(const student &k, const student &i2)
{
return(k.rollno<i2.rollno);
}
vector<student> read()
{
int n;
student k;
vector<student> j;
cout<< "\nEnter total no. of students : ";
cin>>n;
for(int i=0;i<n;i++)
{
cin>>k;
j.push_back(k);
}
return j;
}
void printfunction(const student &k)
{
cout<<k;
}
void print( const vector<student> &j)
{
cout<<"\n\t\tROLL NO\t\tNAME\t\tDATE OF BIRTH";
for_each(j.begin(),j.end(),printfunction);
}
void insert(vector<student> &j)
{
student k;
```

```
cin>>k;
j.push_back(k);
}
void delet(vector<student>&j)
{
student k;
cout<<"\nEnter Student Roll No To Delete : ";
cin>>k.rollno;
vector<student>::iterator p;
p=find(j.begin(),j.end(),k);
if(p!=j.end())
{
j.erase(p);
cout<<"Record deleted successfully";
}
else
cout<<"\nNot found ";
}
void search( vector<student>&j )
{
student k;
cout<<"\nEnter Student Roll No To Search : ";
cin>>k.rollno;
cout<<"\n\n\t\tROLL NO\t\tNAME\t\tDATE OF BIRTH";
vector<student>::iterator p;
p=find(j.begin(),j.end(),k);
if(p!=j.end())
cout<<*p;
else
cout<<"\nNot found ";
```

```
}
void sort( vector<student> &j)
{
sort(j.begin(),j.end(),myfunc);
}
int main()
{
vector<student> j;
int op;
do
{
cout<<"\n\n\t\t-------<< MENU >>---------";
cout<<"\n\t\t1.Create ";
cout<<"\n\t\t2.Display ";
cout<<"\n\t\t3.Insert ";
cout<<"\n\t\t4.Delete ";
cout<<"\n\t\t5.Search ";
cout<<"\n\t\t6.Sort";
cout<<"\n\t\t7.Quit";
cout<<"\n\t\t-------------------------";
cout<<"\n\t\tEnter your choice : ";
cin >> op;
switch(op)
{
case 1:
j=read();
break;
case 2:
print(j);
break;
```

```
case 3:
insert(j);
break;
case 4:
delet(j);
break;
case 5:
search(j);
break;
case 6:
sort(j);
print(j);
break;
}
}while(op!=7);
}
```

Output:

```
-------<< MENU >>---------
1.Create
2.Display
3.Insert
4.Delete
5.Search
6.Sort
7.Quit
-------------------------
Enter your choice : 1
Enter total no. of students : 2
Enter Roll No : 85
```

Enter Name : shixs atul

Enter DOB : 15/8/2003

Enter Roll No : 89

Enter Name : swara

Enter DOB : 85 /8/2003

-------<< MENU >>---------

1.Create

2.Display

3.Insert

4.Delete

5.Search

6.Sort

7.Quit

Enter your choice : 2

ROLL NO	NAME	DATE OF BIRTH
85	atul	15/8/2003
8	swara	8/8/2003

-------<< MENU >>---------

1.Create

2.Display

3.Insert

4.Delete

5.Search

6.Sort

7.Quit

Enter your choice : 3

Enter Roll No : 7

Enter Name : rucha

Enter DOB : 10/5/2002

-------<< MENU >>---------

1.Create

2.Display

3.Insert

4.Delete

5.Search

6.Sort

7.Quit

Enter your choice : 5

Enter Student Roll No To Search : 7

ROLL NO	NAME	DATE OF BIRTH
7	rucha	10/5/2002

-------<< MENU >>---------

1.Create

2.Display

3.Insert

4.Delete

5.Search

6.Sort

7.Quit

Enter your choice : 6

ROLL NO	NAME	DATE OF BIRTH
7	rucha	10/5/2002
8	swara	8/8/2003

85 atul 15/8/2003

-------<< MENU >>---------

1.Create

2.Display

3.Insert

4.Delete

5.Search

6.Sort

7.Quit

Enter your choice : 7

ASSINGMENT NO 7

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> 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
```

Code:

```
#include<iostream>
#include <map>
#include<string>
using namespace std;
int main() {
string ch;
map < string, long int > state;
state = {
{
"Andhra Pradesh",
49577103
},
{
```

"Assam",
31205576
},
{
"Arunachal Pradesh",
1383727
},
{
"Bihar",
104099452
},
{
"Chhattisgarh",
25545198
},
{
"Goa",
1458545
},
{
"Gujarat",
60439692
},
{
"Haryana",
25351462
},
{
"Himachal Pradesh",
6864602

```
},
{
"Jharkhand",
32988134
},
{
"Karnataka",
61095297
},
{
"Kerala",
33406061
},
{
"Madhya Pradesh",
72626809
},
{
"Maharashtra",
112374333
},
{
"Manipur",
2570390
},
{
"Meghalaya",
2966889
},
{
```

"Mizoram",
1097206
},
{
"Nagaland",
1978502
},
{
"Odisha",
41974219
},
{
"Punjab",
27743338
},
{
"Rajasthan",
68548437
},
{
"Sikkim",
610577
},
{
"Tamil Nadu",
72147030
},
{
"Telangana",
35003674

```
},
{
"Tripura",
3673917
},
{
"Uttarakhand",
10086292
},
{
"Uttar Pradesh",
199812341
},
{
"West Bengal",
91276115
},
};
cout << "Enter State (in capital camel case e.g-'Tamil Nadu')" << endl;
cin >> ch;
cout << "Population of " << ch << " is " << state[ch] << endl;
return 0;
}
```

Output:

```
Enter State (in capital camel case e.g-'Tamil Nadu')
Kerala
Population of Kerala is 33406061
```