Grp A -1 : 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. Overload operator+ to add two complex numbers.

3. Overload operator\* to multiply two complex numbers.

4. Overload operators << and >> to print and read Complex Numbers

# include<iostream>

using namespace std;

class Complex //decaring Class Complex

{

double real;

double img;

public:

Complex(); // Default Constructor

friend istream & operator >> (istream &, Complex &); // Input

friend ostream & operator << (ostream &, const Complex &); // Output

Complex operator + (Complex); // Addition

Complex operator \* (Complex); // Multiplication

};

Complex::Complex() // Default Constructor

{

real = 0;

img = 0;

}

istream & operator >> (istream &, Complex & i)

{

cin >> i.real >> i.img;

return cin;

}

ostream & operator << (ostream &, const Complex & d)

{

cout << d.real << " + " << d.img << "i" << endl;

return cout;

}

Complex Complex::operator + (Complex c1) // Overloading + operator

{

Complex temp;

temp.real = real + c1.real;

temp.img = img + c1.img;

return temp;

}

Complex Complex::operator \* (Complex c2) // Overloading \* Operator

{

Complex tmp;

tmp.real = real \* c2.real - img \* c2.img;

tmp.img = real \* c2.img + img \* c2.real;

return tmp;

}

int main()

{

Complex C1, C2, C3, C4;

cout<<"Default Complex Number Values: " <<endl;

cout<<C1<<endl;

cout<<C2<<endl;

cout << "Enter Real and Imaginary part of the Complex Number 1 :"<<endl;

cin >> C1;

cout << "Enter Real and Imaginary part of the Complex Number 2 :"<<endl;

cin >> C2;

cout << "Complex Number 1 : " << C1 << endl;

cout << "Complex Number 2 : " << C2 << endl;

C3 = C1+C2;

cout << "Addition : " << C3 << endl;

C4 = C1 \* C2;

cout << "Multiplication : " << C4 << endl;

return 0;

}

Grp A-2 Experiment Number 2 : 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, Contactaddress, 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.h>

using namespace std;

class StudData;

class Student{

string name;

int roll\_no;

string cls;

char\* division;

string dob;

char\* bloodgroup;

static int count;

public:

Student() // Default Constructor

{

name="";

roll\_no=0;

cls="";

division=new char;

dob="dd/mm/yyyy";

bloodgroup=new char[4];

}

~Student()

{

delete division;

delete[] bloodgroup;

}

static int getCount()

{

return count;

}

void getData(StudData\*);

void dispData(StudData\*);

};

class StudData{

string caddress;

long int\* telno;

long int\* dlno;

friend class Student;

public:

StudData()

{

caddress="";

telno=new long;

dlno=new long;

}

~StudData()

{

delete telno;

delete dlno;

}

void getStudData()

{

cout<<"Enter Contact Address : ";

cin.get();

getline(cin,caddress);

cout<<"Enter Telephone Number : ";

cin>>\*telno;

cout<<"Enter Driving License Number : ";

cin>>\*dlno;

}

void dispStudData()

{

cout<<"Contact Address : "<<caddress<<endl;

cout<<"Telephone Number : "<<\*telno<<endl;

cout<<"Driving License Number : "<<\*dlno<<endl;

}

};

inline void Student::getData(StudData\* st)

{

cout<<"Enter Student Name : ";

getline(cin,name);

cout<<"Enter Roll Number : ";

cin>>roll\_no;

cout<<"Enter Class : ";

cin.get();

getline(cin,cls);

cout<<"Enter Division : ";

cin>>division;

cout<<"Enter Date of Birth : ";

cin.get();

getline(cin,dob);

cout<<"Enter Blood Group : ";

cin>>bloodgroup;

st->getStudData();

count++;

}

inline void Student::dispData(StudData\* st1)

{

cout<<"Student Name : "<<name<<endl;

cout<<"Roll Number : "<<roll\_no<<endl;

cout<<"Class : "<<cls<<endl;

cout<<"Division : "<<division<<endl;

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

cout<<"Blood Group : "<<bloodgroup<<endl;

st1->dispStudData();

}

int Student::count;

int main()

{

Student\* stud1[100];

StudData\* stud2[100];

int n=0;

char ch;

do

{

stud1[n]=new Student;

stud2[n]=new StudData;

stud1[n]->getData(stud2[n]);

n++;

cout<<"Do you want to add another student (y/n) : ";

cin>>ch;

cin.get();

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

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

{

cout<<"---------------------------------------------------------------"<<endl;

stud1[i]->dispData(stud2[i]);

}

cout<<"---------------------------------------------------------------"<<endl;

cout<<"Total Students : "<<Student::getCount();

cout<<endl<<"---------------------------------------------------------------"<<endl;

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

{

delete stud1[i];

delete stud2[i];

}

return 0;

}

Grp A – 3 Imagine a publishing company which does marketing for book and audiocassette 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.

# include<iostream>

# include<stdio.h>

using namespace std;

class publication // declaring class Publication

{

private:

string title;

float price;

public:

void add()

{

cout << "\*\*Enter the Publication information\*\* " << endl;

cout << "Enter Title of the Publication : ";

cin>>title;

cout << "Enter Price of Publication : ";

cin >> price;

}

void display()

{

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

cout << "\nTitle of Publication : " << title;

cout << "\nPublication Price : " << price;

}

};

class book : public publication // declaring class book which inherits class publication in public mode.

{

private:

int page\_count;

public:

void add\_book()

{

try

{

add();

cout << "Enter Page Count of Book : ";

cin >> page\_count;

if (page\_count <= 0)

{

throw page\_count;

}

}

catch(...)

{

cout << "\nInvalid Page Count!!!";

page\_count = 0;

}

}

void display\_book()

{

display();

cout << "\nPage Count : " <<page\_count;

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

}

};

class tape : public publication // declaring class tape which inherits class publication in public mode

{

private:

float play\_time;

public:

void add\_tape()

{

try

{

add();

cout << "Enter Play Duration of the Tape : ";

cin >> play\_time;

if (play\_time <= 0)

throw play\_time;

}

catch(...)

{

cout << "\nInvalid Play Time!!!";

play\_time = 0;

}

}

void display\_tape()

{

display();

cout << "\nPlay Time : " <<

play\_time << " min";

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

}

};

int main()

{

book b1[10]; // object of class book

tape t1[10]; // object of class tape

int b\_count = 0, t\_count = 0;

cout<<"Enter number of books:";

cin>>b\_count;

cout<<"\n Add Information to Books"<<endl;

for(int i=0; i<b\_count; i++)

{

b1[i].add\_book();

}

cout<<"Enter number of Tapes:";

cin>>t\_count;

cout << "Add Information to Tapes"<<endl;

for(int i=0; i<t\_count; i++)

{

t1[i].add\_tape();

}

cout <<"\*\*Display Books Information\*\*"<<endl;

for(int i=0; i<b\_count; i++)

{

b1[i].display\_book();

}

cout << "\*\*Display Tapes Information\*\*";

for(int i=0; i<b\_count; i++)

{

t1[i].display\_tape();

}

return 0;

}

Grp B – 4 Write a C++ program that creates an output file, writes information to it, closes the file, open it again as an input file and read the information from the file.

#include<iostream>

#include<fstream>

#include<string>

using namespace std;

class File

{

private:

string name;

int salary;

public:

void writefile();

void readfile();

};

void File :: writefile()

{

ofstream out("Data.txt", ios::out);

out<<"Employee Name"<<"\t"<< "Employee Salary"<<endl;

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

{

cout<<"Enter name of employee: -";

cin>>name;

cout<<"Enter salary of employee: -";

cin>>salary;

out<<name<<"\t\t"<<salary<<endl;

}

out.close();

}

void File :: readfile()

{

ifstream in("Data.txt", ios::in);

cout<<"-------- Data From the file ------ "<<endl;

for(string line; getline(in, line); )

{

cout << line <<endl;

}

in.close();

}

int main()

{

File f1;

f1.writefile();

f1.readfile();

return 0;

}

Grp B-5 Write a function template for selection sort 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];

}

}

int main()

{

int A[size];

float B[size];

cout<<"\nEnter total no of int elements:";

cin>>n;

cout<<"\nEnter int elements:";

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

{

cin>>A[i];

}

sel(A);

cout<<"\nEnter total no of float elements:";

cin>>n;

cout<<"\nEnter float elements:";

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

{

cin>>B[i];

}

sel(B);

return 0;

}

Grp C – 1 Write C++ program using STL for sorting and searching user defined records such as Item

records (Item code, name, cost, quantity etc) using vector container.

#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)

{

if (i1.code == i2.code)

return 1;

else if(i1.code < i2.code)

return 1;

else

return 0;

}

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:

exit(0);

}

}while(ch!=6);

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."<<endl;

cout<<"Item Name : "<<p ->name<<endl;

cout<<"Item Quantity : "<<p ->quantity<<endl;

cout<<"Item Cost : "<<p ->cost<<endl;

cout<<"Item Code: "<<p ->code<<endl;

}

}

Group- C -2 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>

using namespace std;

int main () {

map<string, int> m;

map<string, int>::iterator it;

m.insert (pair<string, int>("MH", 10));

m.insert (pair<string, int>("MP", 20));

m.insert (pair<string, int>("GJ", 30));

m.insert (pair<string, int>("OD", 40));

cout << "Total state and UT of India with Size of populationMap: " << m.size() <<endl;

for (it = m.begin(); it != m.end(); ++it)

{

cout << it->first <<":" << it->second << " million"<<endl;

}

char c;

do{

string state;

cout<<"\nEnter that state you want to know the population of: ";

cin>>state;

it = m.find(state);

if( it != m.end() )

cout << state <<" = "<< it->second << " million" <<endl;

else

cout<<"State is not in Map"<<endl;

cout<<"Do you wish to continue?(y/n):";

cin>>c;

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

return 0;

}