**15B17CI371 – Data Structures Lab**

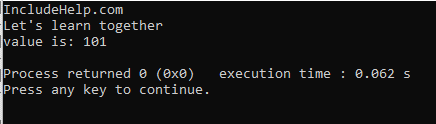
**ODD 2024**

**Week 0-LAB B**

**Practice Lab**

**[CO: C270.1**

1.



Yes, this code can serve as a skeleton that can be modified and reused for solving different questions. It makes it easy to add, modify, or remove functionalities. New member functions can be added to the class as needed. Existing functions can be modified to handle different tasks or process different inputs.

2.

#include <iostream>

using namespace std;

class naturalnumber {

public:

int value;

naturalnumber(int v) : value(v) {}

naturalnumber operator+(const naturalnumber& other) {

return naturalnumber(value + other.value);

}

};

class complex {

public:

double real, imag;

complex(double r = 0, double i = 0) : real(r), imag(i) {}

complex operator+(const complex& other) {

return complex(real + other.real, imag + other.imag);

}

void display() const {

cout << real << " + " << imag << "i" << endl;

}

};

class matrix {

public:

int rows, cols;

int\*\* data;

matrix(int r, int c) : rows(r), cols(c) {

data = new int\*[rows];

for (int i = 0; i < rows; ++i) {

data[i] = new int[cols]();

}

}

void input() {

cout << "enter elements of the matrix (" << rows << "x" << cols << "):" << endl;

for (int i = 0; i < rows; ++i) {

for (int j = 0; j < cols; ++j) {

cin >> data[i][j];

}

}

}

matrix operator+(const matrix& other) {

if (rows != other.rows || cols != other.cols) {

cerr << "matrices dimensions do not match!" << endl;

exit(EXIT\_FAILURE);

}

matrix result(rows, cols);

for (int i = 0; i < rows; ++i) {

for (int j = 0; j < cols; ++j) {

result.data[i][j] = data[i][j] + other.data[i][j];

}

}

return result;

}

void display() const {

for (int i = 0; i < rows; ++i) {

for (int j = 0; j < cols; ++j) {

cout << data[i][j] << " ";

}

cout << endl;

}

}

~matrix() {

for (int i = 0; i < rows; ++i) {

delete[] data[i];

}

delete[] data;

}

};

int main() {

naturalnumber num1(10), num2(20);

naturalnumber sum = num1 + num2;

cout << "sum of natural numbers: " << sum.value << endl;

complex c1(3.4, 5.6), c2(1.2, 4.3);

complex csum = c1 + c2;

cout << "sum of complex numbers: ";

csum.display();

int rows, cols;

cout << "enter the number of rows and columns for matrices: ";

cin >> rows >> cols;

matrix m1(rows, cols), m2(rows, cols);

m1.input();

m2.input();

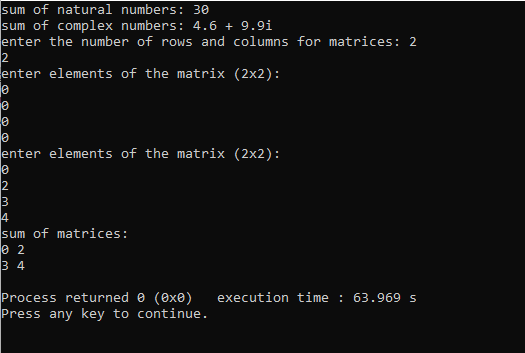
matrix msum = m1 + m2;

cout << "sum of matrices:" << endl;

msum.display();

return 0;

}



3.

#include<iostream>

using namespace std;

class vendor

{

    public:

    string name;

    int liscnumber;

    int lanq;

    int keyboard\_sp;

    void inputdetails()

    {

        cout<<"enter the name of the vendor : ";

        cin>>name;

        cout<<"enter the License number of the vendor : ";

        cin>>liscnumber;

        cout<<"enter the quantity of LAN cables available : ";

        cin>>lanq;

        cout<<"enter the selling price of the keyboard : ";

        cin>>keyboard\_sp;

    }

    void printdetails()

    {

        cout<<"\n\nPrinting Details :\n\n";

        cout<<"name of the vendor : "<<name<<"\n";

        cout<<"License number of the vendor : "<<liscnumber<<"\n";

        cout<<"quantity of LAN cables available : "<<lanq<<"\n";

        cout<<"selling price of the keyboard : "<<keyboard\_sp<<"\n";

    }

    void compareven(vendor v1,vendor v2)

    {

        if(v1.keyboard\_sp>v2.keyboard\_sp)

        {

            cout<<"Vendor "<<[v2.name](http://v2.name/)<<" has lower selling price of the keyboard \n";

        }

        else

        {

            cout<<"Vendor "<<[v1.name](http://v1.name/)<<" has lower selling price of the keyboard \n";

        }

        if(v1.lanq>v2.lanq)

        {

            cout<<"Vendor "<<[v1.name](http://v1.name/)<<" has more LAN cables\n";

        }

        else

        {

            cout<<"Vendor "<<[v2.name](http://v2.name/)<<" has more LAN cables\n";

        }

    }

    void findv(vendor p[],int countr)

    {

        int maxlan= p[0].lanq;

        int index = 0;

        for(int i=0; i<countr; i++)

        {

            if(p[i].lanq>maxlan)

            {

                maxlan=p[i].lanq;

                index=i;

            }

        }

        cout<<"the vendor with maximum quantity of LAN cables is : "<<p[index].name<<"\n\n";

    }

    void findprice(vendor k[],int countr)

    {

        int minprice = k[0].keyboard\_sp;

        int index = 0;

        for(int i=0; i<countr; i++)

        {

            if(k[i].keyboard\_sp<minprice)

            {

                minprice=k[i].keyboard\_sp;

                index=i;

            }

        }

        cout<<"the vendor with maximum quantity of LAN cables is : "<<k[index].name<<"\n\n";

    }

};

int main()

{

    int countr;

    cout<<"enter the number of counter : ";

    cin>>countr;

    vendor \* arr = new vendor[countr];

    for(int i=0;i<countr;i++)

    {

        arr[i].inputdetails();

    }

    for(int i=0;i<countr;i++)

    {

        arr[i].printdetails();

    }

    int index1,index2;

    cout<<"enter the indexes of the vendors to be compared : ";

    cin>>index1>>index2;

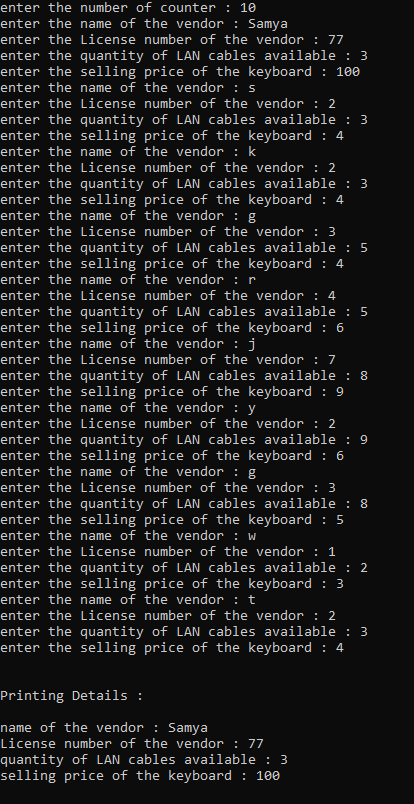
    arr[0].compareven(arr[index1],arr[index2]);

    arr[0].findprice(arr,countr);

    arr[0].findv(arr,countr);

    return 0;

}



4.

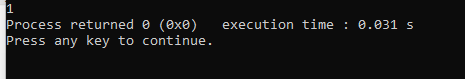
a)

Error.

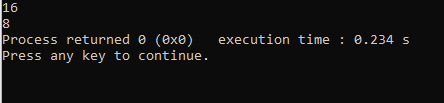
X is private and cannot be accessed outside the class.



b)



c)



d)

#include <iostream>

class Test

{

public:

int i;

void get();

};

void Test::get()

{

std::cout << "Enter the value of i: "<<"\n";

std::cin >> i;

}

Test t;

int main()

{

Test t; // local object

t.get();

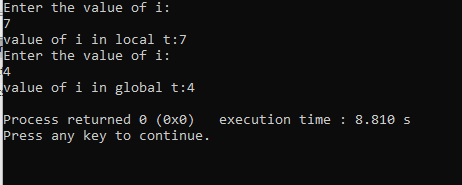
std::cout << "value of i in local t:"<<t.i<<"\n";

::t.get();

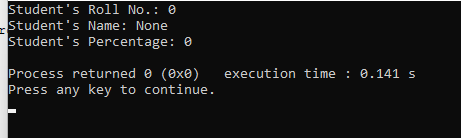
std::cout << "value of i in global t:"<<::t.i<<"\n";

return 0;

}



e)



f)

