2/3/2025

Muhammad Arham Usman (24k-1016)

FAST NUCES

Object Oriented Programming

Lab 3

Question 1

# Source Code

/\*

Create a class called time that has separate int member data for hours, minutes, and seconds. One constructor

should initialize this data to 0, and another should initialize it to fixed values. Another member function should

display it, in 11:59:59 format. The final member function should add two objects of type time passed as

arguments.

A main() program should create two initialized time objects (should they be const?) and one that isn’t initialized.

Then it should add the two initialized values together, leaving the result in the third time variable. Finally it should

display the value of this third variable. Make appropriate member functions const.

\*/

#include <iostream>

using namespace std;

class Time{

    private:

        int hour, minute, second;

    public:

        Time(int h, int m, int s){

            hour=h;

            minute=m;

            second=s;

        }

        Time(){

            hour=0;

            minute=0;

            second=0;

        }

        int getHour() const {

            return hour;

        }

        int getMinute() const {

            return minute;

        }

        int getSecond() const {

            return second;

        }

        void setHour(int h){

            hour=h;

        }

        void setMinute(int m){

            minute=m;

        }

        void setSecond(int s){

            second=s;

        }

        void display() const{

            cout<<hour<<":"<<minute<<":"<<second<<endl;

        }

        Time add(const Time& t) const {

            Time t2;

            t2.second=t.getSecond() +second;

            if (t2.second>=60){

                t2.second-=60;

                t2.minute+=1;

            }

            t2.minute+=t.getMinute() +minute;

            if (t2.minute>=60){

                t2.minute-=60;

                t2.hour+=1;

            }

            t2.hour+=t.getHour() +hour;

            return t2;

        }

};

int main(){

    Time t1;

    Time t2(11,59,59);

    Time t3=t2.add(t1);

    cout << "t3: ";

    t3.display();

    t1.setHour(2);

    t1.setMinute(34);

    t1.setSecond(48);

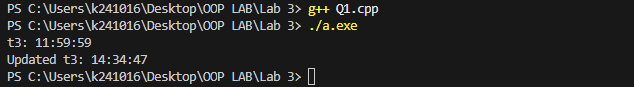
    cout<<"Updated t3: ";

    t3=t2.add(t1);

    t3.display();

}

# Output:



Question 2

# Source Code:

/\*

Imagine a tollbooth at a bridge. Cars passing by the booth are expected to pay a 50 cent toll. Mostly they do, but

sometimes a car goes by without paying. The tollbooth keeps track of the number of cars that have gone by, and of

the total amount of money collected.

Model this tollbooth with a class called tollBooth. The two data items are a type unsigned int to hold the total

number of cars, and a type double to hold the total amount of money collected. A constructor initializes both of

these to 0. A member function called payingCar() increments the car total and adds 0.50 to the cash total. Another

function,

called nopayCar(), increments the car total but adds nothing to the cash total. Finally, a member function called

display() displays the two totals. Make appropriate member functions const.

Include a program to test this class. This program should allow the user to push one key to count a paying car, and

another to count a nonpaying car. Pushing the Esc key should cause the program to print out the total cars and

total cash and then exit.

\*/

#include <iostream>

#include <conio.h>

using namespace std;

class tollBooth{

    private:

    int car;

    double money;

    public:

    tollBooth(){

        car=0;

        money=0;

    }

    void payingCar(){

        car++;

        money+=0.5;

    }

    void nopayCar(){

        car++;

    }

    void display(){

        cout<<"Total no of Cars: "<<car<<endl;

        cout<<"Total Cash collected: "<<money<<endl;

    }

};

int main(){

    tollBooth t1;

    char c;

    cout<<"Press 'a' to count a paying car\n";

    cout<<"Press 'd' to count a non-paying car\n";

    cout<<"Press Esc to exit\n";

    while (c!=27){

        c=\_getch();

        if (c=='a') t1.payingCar();

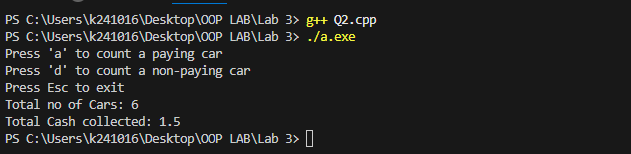
        if (c=='d') t1.nopayCar();

    }

    t1.display();

}

# Output:



Question 3

# Source Code:

/\*

Create a class that includes a data member that holds a “serial number” for each object created from the class.

That is, the first object created will be numbered 1, the second 2, and so on. To do this, you’ll need another data

member that records a count of how many objects have been created so far. (This member should apply to the

class as a whole; not to individual objects. What keyword specifies this?) Then, as each object is created, its

constructor can examine this count member variable to determine the appropriate serial number for the new

object.

Add a member function that permits an object to report its own serial number. Then write a main() program that

creates three objects and queries each one about its serial number. They should respond I am object number 2, and

so on.

\*/

#include <iostream>

using namespace std;

class serialNumberClass{

    private:

    int serialNumber;

    static int count;

    public:

    serialNumberClass(){

        count++;

        serialNumber=count;

    }

    void report\_serialNumber(){

        cout<<"I am object number "<<serialNumber<<endl;

    }

};

int serialNumberClass::count=0;

int main(){

    serialNumberClass o1, o2, o3;

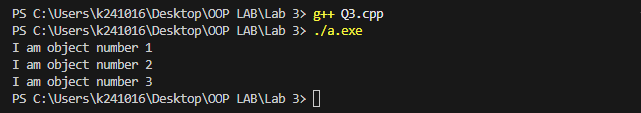
    o1.report\_serialNumber();

    o2.report\_serialNumber();

    o3.report\_serialNumber();

}

# Output:



Question 4

# Source Code:

/\*

In ocean navigation, locations are measured in degrees and minutes of latitude and longitude. Thus if you’re lying

off the mouth of Papeete Harbor in Tahiti, your location is 149 degrees 34.8 minutes west longitude, and 17

degrees 31.5 minutes south latitude. This is written as 149°34.8’ W, 17°31.5’ S. There are 60 minutes in a degree.

(An older system also divided a minute into 60 seconds, but the modern approach is to use decimal minutes

instead.) Longitude is measured from 0 to 180 degrees, east or west from Greenwich, England, to the international

dateline in the Pacific. Latitude is measured from 0 to 90 degrees, north or south from the equator to the poles.

Create a class angle that includes three member variables: an int for degrees, a float for minutes, and a char for the

direction letter (N, S, E, or W). This class can hold either a latitude variable or a longitude variable. Write one

member function to obtain an angle value (in degrees and minutes) and a direction from the user, and a second to

display the angle value in 179°59.9’ E format. Also write a three-argument constructor.

Write a main() program that displays an angle initialized with the constructor, and then, within a loop, allows the

user to input any angle value, and then displays the value. You can use the hex character constant ‘\xF8’, which

usually prints a degree (°) symbol.

Note:

• fixed: This manipulator is used to display floating-point numbers in fixed-point notation (i.e., a set number

of digits after the decimal point).

• setprecision(n): This manipulator sets the number of digits to be displayed after the decimal point. For

example, setprecision(1) ensures that only one digit will be displayed after the decimal point, as seen in

your angle minutes.

\*/

#include <iostream>

#include<iomanip>

using namespace std;

class Angle{

    private:

    int degrees;

    float minutes;

    char direction;

    public:

    Angle(int d, float m, char dir){

        update(d, m, dir);

    }

    void update(int d, float m, char dir){

        while (!(dir=='N'||dir=='S'||dir=='E'||dir=='W')){

            cout<<"\nEnter a valid Direction(N/S/E/W): ";

            cin>>dir;

        }

        direction=dir;

        while (m>60||m<0){

            cout<<"\nEnter valid minutes: ";

            cin>>m;

        }

        minutes=m;

        while (((dir=='N'||dir=='S')&&(d>90))||(d<0)||(d>180)){

            cout<<"\nEnter valid degrees: ";

            cin>>d;

        }

        degrees=d;

    }

    void display(){

        cout<<"\nAngle: "<<degrees<<'\xF8'<<fixed<<setprecision(1)<<minutes<<"'"<<direction<<endl;

    }

    int getDegree(){

        return degrees;

    }

    float getMinute(){

        return minutes;

    }

    char getDirection(){

        return direction;

    }

};

int main(){

    Angle a1(0, 0, 'S');

    int d; float m; char dir;

    while (!(a1.getDegree()==0&&a1.getMinute()==0&&a1.getDirection()=='N')){

        cout<<"\nEnter degrees: 0, minutes: 0 and direction: N to exit\n";

        cout<<"Enter degrees: ";

        cin>>d;

        cout<<"Enter minutes: ";

        cin>>m;

        cout<<"Enter direction: ";

        fflush(stdout);

        cin>>dir;

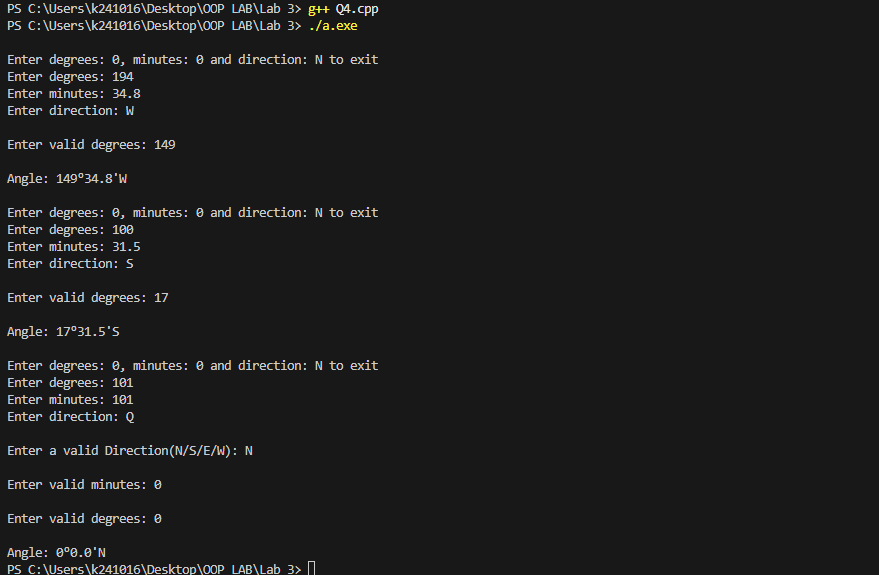
        a1.update(d, m, dir);

        a1.display();

    }

}

# Output:



Question 5

# Source Code:

/\*

Create a class called calendar. The calendar should have 12 arrays for each month of the year, and a variable that

stores information about the current year. The user is allowed to store their tasks to do against each day. Assume

only one entry is needed per day.

Create the following methods for your class:

- Add a task. Thisfunction accepts three parameters: task details, date and month. It should add

- a task on the day specified.

- Remove a task. Accepts the date and month as a parameter to remove the task from.

- Show tasks. This method should go through all of your months and print all the tasks allocated.

Your task is to create one calendar object, then hardcode 5-6 tasks for your calendar. Then demonstrate how

you’ll remove a task, and display the updated task list.

\*/

#include <iostream>

using namespace std;

class calender{

    private:

    string \*\*cal=new string\* [12];

    void check(int &m, int &d){

        while (m<=0 || m>12){

            cout<<"Enter a valid month number(1-12): ";

            cin>>m;

        }

        int max\_days;

        if (m == 2)

            max\_days = 28;

        else if (m == 4 || m == 6 || m == 9 || m == 11)

            max\_days = 30;

        else

            max\_days = 31;

        while (d <= 0 || d > max\_days) {

            cout << "Enter a valid day number: ";

            cin >> d;

        }

    }

    public:

    calender(){

        for (int i=0; i<12; i++){

            cal[i]=new string[31];

            for (int j=0; j<31; j++){

                cal[i][j]=" ";

            }

        }

    }

    void addTask(string t, int d, int m){

        check(m, d);

        cal[m-1][d-1]=t;

    }

    void removeTask(int d, int m){

        check(m, d);

        cal[m-1][d-1]=" ";

    }

    void showTasks(){

        for (int i=0; i<12; i++){

            for (int j=0; j<31; j++){

                if (cal[i][j]!=" "){

                    cout<<"Day: "<<j<<" Month: "<<i+1<<" Task: "<<cal[i][j]<<endl;

                }

            }

        }

    }

};

int main(){

    calender c;

    c.addTask("Do Task 1", 1, 1);

    c.addTask("Do Task 2", 2, 2);

    c.addTask("Do Task 3", 3, 3);

    c.addTask("Do Task 4", 4, 4);

    c.addTask("Do Task 5", 5, 5);

    c.addTask("Do Task 6", 6, 6);

    cout<<"Before Removing:\n";

    c.showTasks();

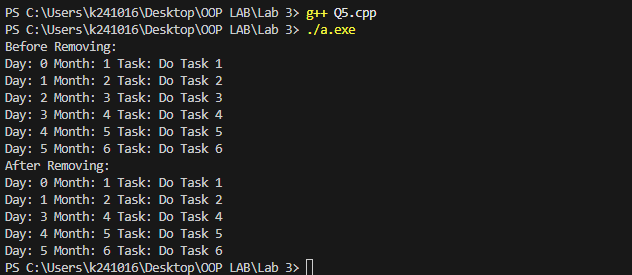
    c.removeTask(3,3);

    cout<<"After Removing:\n";

    c.showTasks();

}

# Output:



Question 6

# Source Code:

/\*

Create a class called Smartphone with the following attributes:

- Company

- Model

- Display Resolution

- RAM

- ROM

- Storage

Create getter and setter methods for your attributes. A smartphone has some specific actions that it can perform.

For example:

1. Make a phone call

2. Send a message

3. Connect to the wifi

4. Browse the internet

Create different smartphone objects. Set their attributes using the setter functions and display their attributes after

using the getter functions to fetch the attributes.

\*/

#include <iostream>

using namespace std;

class Smartphone{

    private:

    string company, model, display;

    int ram, rom, storage;

    public:

    void setcompany(string c){

        company=c;

    }

    void setmodel(string m){

        model=m;

    }

    void setdisplay(string d){

        display=d;

    }

    void setram(int r){

        ram=r;

    }

    void setrom(int r){

        rom=r;

    }

    void setstorage(int s){

        storage=s;

    }

    string getcompany(){

        return company;

    }

    string getmodel(){

        return model;

    }

    string getdisplay(){

        return display;

    }

    int getram(){

        return ram;

    }

    int getrom(){

        return rom;

    }

    int getstorage(){

        return storage;

    }

    void call(){

        cout<<"Making a phone call\n";

    }

    void message(){

        cout<<"Sending a message\n";

    }

    void wifi(){

        cout<<"Connecting to wifi\n";

    }

    void browse(){

        cout<<"Browsing the internet\n";

    }

};

int main() {

    Smartphone phone1, phone2;

    phone1.setcompany("Apple");

    phone1.setmodel("iPhone 14 Pro");

    phone1.setdisplay("1179 x 2556 pixels");

    phone1.setram(6);

    phone1.setrom(128);

    phone1.setstorage(256);

    phone2.setcompany("Samsung");

    phone2.setmodel("Galaxy S23 Ultra");

    phone2.setdisplay("1440 x 3088 pixels");

    phone2.setram(12);

    phone2.setrom(256);

    phone2.setstorage(512);

    cout << "Smartphone 1 Details:\n";

    cout << "Company: " << phone1.getcompany() << endl;

    cout << "Model: " << phone1.getmodel() << endl;

    cout << "Display: " << phone1.getdisplay() << endl;

    cout << "RAM: " << phone1.getram() << "GB\n";

    cout << "ROM: " << phone1.getrom() << "GB\n";

    cout << "Storage: " << phone1.getstorage() << "GB\n";

    phone1.call();

    phone1.message();

    phone1.wifi();

    phone1.browse();

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

    cout << "Smartphone 2 Details:\n";

    cout << "Company: " << phone2.getcompany() << endl;

    cout << "Model: " << phone2.getmodel() << endl;

    cout << "Display: " << phone2.getdisplay() << endl;

    cout << "RAM: " << phone2.getram() << "GB\n";

    cout << "ROM: " << phone2.getrom() << "GB\n";

    cout << "Storage: " << phone2.getstorage() << "GB\n";

    phone2.call();

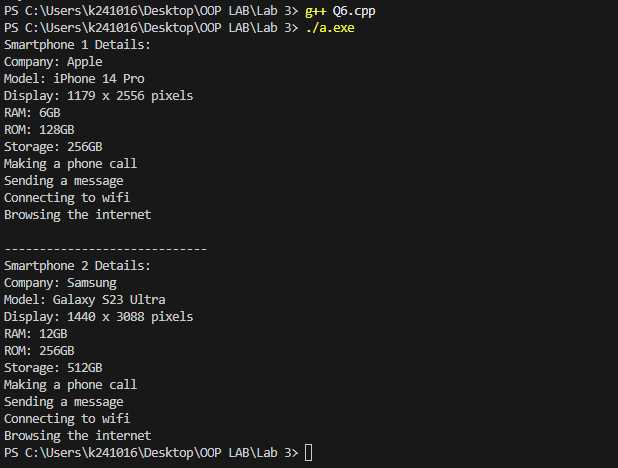
    phone2.message();

    phone2.wifi();

    phone2.browse();

}

# Output:



Question 7

# Source Code:

/\*

Create a class for a stationary shop. The stationary shop maintains a list for all the items that it sells (hint: array of

strings), and another list with the prices of the items (hint: array of prices).

Create a menu-driven program to:

1. Allow the shop owner to add the items and their prices.

2. Fetch the list of items

3. Edit the prices of the items

4. View all the items and their prices

Create a receipt that the shopkeeper can share with their customers. The receipt can only be created after the

shopkeeper inputs the items and their amounts bought by the customer.

\*/

#include <iostream>

using namespace std;

class StationaryShop{

    private:

    string \*item;

    int \*price;

    int count;

    public:

    StationaryShop(){

        count=0;

        item=nullptr;

        price=nullptr;

    }

    ~StationaryShop() {

        delete[] item;

        delete[] price;

    }

    void add(string it, int p){

        string \*arr1=new string[count+1];

        int \*arr2=new int[count+1];

        for (int i=0; i<count; i++){

            arr1[i]=item[i];

            arr2[i]=price[i];

        }

        arr1[count]=it;

        arr2[count]=p;

        count++;

        delete[] item;

        delete[] price;

        item=arr1;

        price=arr2;

    }

    string\* fetchItem(){

        return item;

    }

    void editPrice(string name, int p){

        for (int i=0; i<count; i++){

            if (name==item[i]){

                price[i]=p;

                return;

            }

        }

        cout<<"Item not found\n";

    }

    void display(){

        if (count==0){

            cout<<"Inventory is empty\n";

            return;

        }

        cout<<"\nShop Items:\n";

        for (int i=0; i<count; i++){

            cout<<"Item: "<<item[i]<<" | Price: "<<price[i]<<endl;

        }

    }

    int getIndex(string name){

        for (int i=0; i<count; i++){

            if (name==item[i]){

                return i;

            }

        }

        cout<<"Item not found\n";

        return -1;

    }

    void generate\_reciept(){

        int\* quantity=new int[count];

        for (int i=0; i<count; i++){

            quantity[i]=0;

        }

        string s;

        int index, q, total=0;

        while (1){

            cout<<"Enter item(exit to exit): ";

            cin>>s;

            if (s=="exit") break;

            index=getIndex(s);

            if (index>=0) {

                cout<<"Enter quantity: ";

                cin>>q;

                if (q>0) quantity[index]=q;

                else cout<<"Error\n";

            }

        }

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

        for (int i=0; i<count; i++){

            if (quantity[i]!=0){

                cout<<"Item: "<<item[i]<<" Quantity: "<<quantity[i]<<" Price (per piece): "<<price[i]<<endl;

                total+=quantity[i]\*price[i];

            }

        }

        cout<<"\nTotal bill: "<<total<<endl;

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

        delete[] quantity;

    }

    int getCount(){

        return count;

    }

};

int main() {

    StationaryShop shop;

    int choice;

    do {

        cout << "\n--- Stationary Shop Menu ---\n";

        cout << "1. Add Item\n";

        cout << "2. Fetch List of Items\n";

        cout << "3. Edit Price\n";

        cout << "4. View All Items and Prices\n";

        cout << "5. Generate Receipt\n";

        cout << "6. Exit\n";

        cout << "Enter your choice: ";

        cin >> choice;

        switch (choice) {

            case 1: {

                string item;

                int price;

                cout << "Enter item name: ";

                cin >> item;

                cout << "Enter price: ";

                cin >> price;

                shop.add(item, price);

                break;

            }

            case 2: {

                string\* items = shop.fetchItem();

                cout << "\nList of Items:\n";

                int count=shop.getCount();

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

                    cout << items[i] << endl;

                }

                break;

            }

            case 3: {

                string item;

                int newPrice;

                cout << "Enter item name to edit price: ";

                cin >> item;

                cout << "Enter new price: ";

                cin >> newPrice;

                shop.editPrice(item, newPrice);

                break;

            }

            case 4: {

                shop.display();

                break;

            }

            case 5: {

                shop.generate\_reciept();

                break;

            }

            case 6: {

                cout << "Exiting program...\n";

                break;

            }

            default:

                cout << "Invalid choice. Please try again.\n";

        }

    } while (choice != 6);

}

# Output:

