ASSIGNMENT 2

NAME: SOURABH PATEL

ADMISSION: U19CS082

1. **In order to access the memory address of a variable, val , prepend it with & sign. For example, &val returns the memory address of val. This memory address is assigned to a pointer and can be shared among functions. For example, int *p = &val assigns the memory address of val to pointer p. To access the content of the memory pointed to, prepend the variable name with a *. For example, *p will return the value stored in val and any modification to it will be performed on val. Create a program with function update having parameters as int *a & int *b Modify the values in memory so that a contains their sum and b contains their absolute difference.**

CODE :

```
#include <iostream>
using namespace std;

void ref_function(int *a, int *b){
    *a += *b;
    *b = abs(*a - 2*(*b));
}

int main()
{
    int val;
    int a,b;
    cout<<"Enter the value: ";
    cin>>a>>b;
    ref_function(&a,&b);
    cout<<"# values are "<<a<<" "<<b;
    return 0;
}
```

OUTPUT:

```
C:\Users\alokp\CLionProjects\practice\cmake-build-debug\practice.exe
Enter the value:12 14
 # values are 26 2
Process finished with exit code 0
```

Q2. Write a program with two classes HotelRoom and HotelApartment denoting respectively a standard hotel room and a hotel apartment. An instance of any of these classes has two parameters: bedrooms and bathrooms denoting respectively the number of bedrooms and the number of bathrooms in the room.

The prices of a standard hotel room and hotel apartment are given as:

- **Hotel Room: 50 x bedrooms + 100 x bathrooms.**
- **Hotel Apartment: The price of a standard room with the same number bedrooms and bathrooms plus 100.**

- **For example, if a standard room costs 200, then an apartment with the same number of bedrooms and bathrooms costs 300.**
- **Write a program to return the correct profit. Make necessary assumptions wherever necessary.**

CODE :

```
#include <iostream>
using namespace std;

class hotelroom{
    int bedroom,bathroom;
public:
    hotelroom(int a, int b){
        this->bedroom=a;
        this->bathroom=b;
    }
    void input(int a,int b){
        this->bedroom=a;
        this->bathroom=b;
    }
    int cost(){
        return this->bedroom*50 + this->bathroom*100;
    }
};

class hotelapartment{
    int bedroom,bathroom;
public:
    hotelapartment(int a, int b){
        this->bedroom=a;
        this->bathroom=b;
    }
    void input(int a,int b){
        this->bedroom=a;
```

```
        this->bathroom=b;
    }
    int cost(){
        return this->bedroom*50 + this->bathroom*100+100;
    }
};
int main()
{
    hotelroom hr(12,14);
    hotelapartment ha(12,14);
    cout<<"room cost: "<<hr.cost()<<" apartment cost: "<<ha.cost()<<endl;
    return 0;
}
```

OUTPUT:

```
C:\Users\alokp\CLionProjects\practice\cmake-build-debug\practice.exe
room cost: 2000 apartment cost: 2100

Process finished with exit code 0
```

Q3. Write a class to represent a vector (a series of float values). Include member functions to perform the following tasks:

- **To create the vector**
- **To modify the value of a given element.**
- **To multiply by a scalar value.**
- **To display the vector in the form (10, 20, 30,…)**

CODE:

```
#include <iostream>
using namespace std;

class my_vector{
    int n;
    float *pt;
public:
    my_vector(int size){
        n=size;
        pt = new float[n];
    }
    void modify(int pos, float val){
        pt[pos] = val;
    }
    void operator *(float a){
        for(int i=0;i<n;i++)
            pt[i] *=a;
```

```
    }
    void vector_view(){
        cout<<"(";
        for(int i=0;i<n;i++){
            cout<<pt[i];
            if(i!=n-1)
                cout<<", ";
        }
        cout<<")"<<endl;
    }
};


int main()
{
    my_vector V(10);
    for(int i=0;i<10;i++)
        V.modify(i,1.2*i);
    V.vector_view();
    V*2;
    V.vector_view();
    return 0;
}
```

OUTPUT:

```
C:\Users\alokp\CLionProjects\practice\cmake-build-debug\practice.exe
(0, 1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8, 9.9)
(0, 2.2, 4.4, 6.6, 8.8, 11, 13.2, 15.4, 17.6, 19.8)

Process finished with exit code 0
```

Q4. A book shop maintains the inventory of books that are being sold at the shop. The list includes details such as author, title, price, publisher and stock position. Whenever a customer wants a book, the sales person inputs the title and author and the system searches the list and displays whether it is available or not. If it is not, an appropriate message is displayed. If it is, then the system displays the book details and requests for the number of copies required. If the requested copies book details and requests for the number of copies required. If the requested copies are available, the total cost of the requested copies is displayed; otherwise the message "Required copies not in stock" is displayed. Design a system using a class called books with suitable member functions and Constructors. Use new operators in constructors to allocate memory space required. Implement C++ program for the system.

Improve the system design to incorporate the following features:

- **The price of the books should be updated as and when required. Use a member function to implement this.**
- **The stock value of each book should be automatically updated as soon as a transaction is completed.**
- **The number of successful and unsuccessful transactions should be recorded for the purpose of statistical analysis. Use static data members to keep count of transactions.**
- **Also demonstrate the use of pointers to access the members.**

CODE:

```
#include <bits/stdc++.h>
using namespace std;

typedef struct transaction{
    string name;
    string author;
    float price,total;
    int nos;
    bool sucess;
};

class books{
    string name,author,publisher;
    float price;
    int stock;
public:
    books();
    void show();
    void update_price();
    bool update_stock(int a);
    void add_record(int req, bool success);
    pair<string, string> connect(){
        return {name, author};
    }
};
map<pair<string,string>,books *> database;
vector<transaction> records;

books::books(){
    price = -1;
    cout<<"Enter the name of the book: ";
    while(!name.size()) getline(cin,name);
    cout<<"Enter the author's name: ";
    while(!author.size()) getline(cin,author);
    cout<<"Enter the publisher's name: ";
    while(!publisher.size()) getline(cin,publisher);
```

```
    cout<<"Enter the price: ";
    while(price == -1) cin>>price;
    cout<<"Enter stock: ";cin>>stock;
}

void books::show(){
    cout<<"***************BOOK DETAILS*******************"<<endl;
    cout<<"book name: "<<name<<endl;
    cout<<"author name: "<<author<<endl;
    cout<<"publisher name: "<<publisher<<endl;
    cout<<"price: "<<price<<endl;
    cout<<"stock: "<<stock<<endl;
    cout<<"*****************END*****************"<<endl;
}

void books::update_price(){
    cout<<"Enter the new price of book: ";
    cin>>price;
    cout<<"price updated!!"<<endl;
}
bool books::update_stock(int req){
    if(req>stock){
        cout<<"Don't have the required stock"<<endl;
        return false;
    }
    stock -= req;
    return true;
}

void books::add_record(int req, bool sucess = false){
    transaction transactions;
    transactions.name = name;
    transactions.author = author;
    transactions.price = price;
    transactions.nos = req;
    transactions.total = req*price;
    cout<<"bill amount: "<<transactions.total<<endl;
    if(sucess){
        cout<<"paid?(0/1) ";
        cin>>transactions.sucess;
    }
    if(!transactions.sucess)
        stock += req;
    records.push_back(transactions);
};

void record_display(){
    cout<<"********** Transcation detail**********";
    for(int i=0;i<records.size();i++){
        cout<<"***********************************"<<endl;
        transaction ts=records[i];
        cout<<"book name:"<<ts.name<<endl;
        cout<<"author name: "<<ts.author<<endl;
        cout<<"no of books ordered: "<<ts.nos<<endl;
        cout<<"price per piece: "<<ts.price<<endl;
        cout<<"total bill: "<<ts.total<<endl;
        if(ts.sucess)cout<<"sucessfull"<<endl;
```

```
            else cout<<"transaction failed"<<endl;
            cout<<"***********************************"<<endl;
        }
        cout<<"***************END******************"<<endl;
    }

    books *search(){
        string r_name, r_author;
        cout<<"Enter book name: ";
        while(!r_name.size())getline(cin, r_name);
        cout<<"Enter author name: ";
        while(!r_author.size())getline(cin, r_author);
        pair<string, string> key = make_pair(r_name, r_author);
        map<pair<string,string>, books*>::iterator mp = database.find(key);
        if(mp == database.end())
            return NULL;
        return mp->second;
    }
    void transaction(){
        books *ptr = search();
        if(!ptr){
            cout<<"Book not in database"<<endl;
            return;
        }
        ptr->show();
        int req;
        cout<<"Enter the required book nos: ";
        cin>>req;
        if(ptr->update_stock(req)== false)
            ptr->add_record(0, false);
        else
            ptr->add_record(req,true);
        cout<<"transaction sucessfull"<<endl;
    }

    int main()
    {
        int ch;
        books *ptr;
        cout<<"1. Transaction"<<endl;
        cout<<"2. Add book to database"<<endl;
        cout<<"3. Update price"<<endl;
        cout<<"4. Record_display"<<endl;
        cout<<"0. Exit"<<endl;
        do{
            cout<<"ENTER CHOICE: ";
            cin>>ch;
            switch(ch){
                case 1:
                    transaction();
                    break;
                case 2:
                    ptr = new books;
                    database[ptr->connect()] = ptr;
                    break;
                case 3:
                    ptr = search();
```

```
                    if(!ptr)
                        cout<<"book not found"<<endl;
                    else
                        ptr->update_price();
                    break;
                case 4:
                    record_display();
                case 0:
                    break;
                default:
                    cout<<"wrong choice"<<endl;
            }
        }while(ch);
        return 0;
    }
```

OUTPUT:

```
C:\Users\alokp\CLionProjects\practice\cmake-build-debug\practice.exe
1. Transaction
2. Add book to database
3. Update price
4. Record_display
0. Exit
ENTER CHOICE:2
 Enter the name of the book:Book_A
 Enter the author's name:Alok
 Enter the publisher's name:119
 Enter the price:100
 Enter st
ock:10
 ENTER CHOICE:2
 Enter the name of the book:Book_B
 Enter the author's name:Mr.x
 Enter the publisher's name:120
 Enter the price:499
 Ent
er stock:100
 ENTER CHOICE:1
 Enter book name:Book_B
 Enter author name:Mr.x
```

```
 ***************BOOK DETAILS********************
book name: Book_B
author name: Mr.x
publisher name: 120
price: 499
stock: 100
*****************END*****************
Enter the required book nos:5
 bill amount: 2495
paid?(0/1)1
 transaction sucessfull
ENTER CHOICE:3
 Enter book name:Book_B
 Enter author name:Mr.x
 Enter the new price of book:230
 price updated!!
 ********** Transcation detail**********************************************
book name:Book_B
author name: Mr.x
no of books ordered: 5
price per piece: 499
total bill: 2495
sucessfull
**********************************
****************END*******************
ENTER CHOICE:0
```
