//Constructors and destructors

//Q1

#include<iostream>

using namespace std;

class Cube {

private:

double sideLength;

public:

Cube(double length) : sideLength(length) {

cout << "Cube object created with side length: " << sideLength << endl;

}

~Cube() {

cout << "Cube object with side length " << sideLength << " is destroyed" << endl;

}

double calculateVolume() {

return sideLength \* sideLength \* sideLength;

}

double getSideLength() const {

return sideLength;

}

};

int main() {

double len;

cout << "Enter length: ";

cin >> len;

Cube myCube(len);

double volume = myCube.calculateVolume();

cout << "Volume of the cube with side length " << myCube.getSideLength() << ": " << volume << endl;

return 0;

}

//Q2

#include<iostream>

using namespace std;

class Clicker {

private:

static int clickCount;

public:

Clicker() {

clickCount++;

}

static void displayClickCount() {

cout << "Total Clicks = " << clickCount << endl;

}

~Clicker() {

cout << "A Clicker is retired" << endl;

}

};

int Clicker::clickCount = 0;

int main() {

cout << "Before using Clicker Class: ";

Clicker::displayClickCount();

Clicker clicker1;

cout << "Clicked a new Clicker" << endl;

Clicker clicker2;

cout << "Clicked another Clicker" << endl;

cout << "After using Clicker Class: ";

Clicker::displayClickCount();

return 0;

}

//Q3

#include <iostream>

using namespace std;

class Student {

public:

int studentID;

int testScore;

char letterGrade;

Student(int ID, int score, char grade) {

this->studentID = ID;

this->testScore = score;

this->letterGrade = grade;

}

void display() {

cout << "Record of Student with ID " << studentID << endl;

cout << "Test Score: " << testScore << endl;

cout << "Letter Grade: " << letterGrade << endl;

}

};

int main() {

int ID1, ID2, score1, score2;

char grade1, grade2;

// Student 1

cout << "Enter Student 1 ID: ";

cin >> ID1;

cout << "Enter Student 1 Test Score: ";

cin >> score1;

cout << "Enter Student 1 Letter Grade: ";

cin >> grade1;

// Student 2

cout << "Enter Student 2 ID: ";

cin >> ID2;

cout << "Enter Student 2 Test Score: ";

cin >> score2;

cout << "Enter Student 2 Letter Grade: ";

cin >> grade2;

Student student1(ID1, score1, grade1);

Student student2(ID2, score2, grade2);

cout << endl;

student1.display();

cout << endl;

student2.display();

return 0;

}

//Q4

#include <iostream>

#include <string>

using namespace std;

class Book {

public:

string title;

string author;

double price;

string publisher;

int stock;

Book() {

title = "";

author = "";

price = 0.0;

publisher = "";

stock = 0;

}

Book(string t, string a, double p, string pub, int s) {

title = t;

author = a;

price = p;

publisher = pub;

stock = s;

}

void processOrder(int requiredCopies) {

if (requiredCopies <= stock) {

cout << "\*\*\*\*\*\*\*\*\* Book Details \*\*\*\*\*\*\*\*\*" << endl;

cout << "Title\t\t\tAuthor\t\t\tStock Copies" << endl;

cout << title << "\t\t" << author << "\t\t" << stock << endl;

double totalPrice = price \* requiredCopies;

cout << "Number of Copies Required: " << requiredCopies << endl;

cout << "Total Price = " << totalPrice << " RS" << endl;

} else {

cout << "Required copies not in stock" << endl;

}

}

};

int main() {

const int numBooks = 5;

Book library[numBooks];

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

string title, author, publisher;

double price;

int stock;

cout << "Enter Book " << (i + 1) << " Details:" << endl;

cout << "Title: ";

cin >> title;

cout << "Author: ";

cin >> author;

cout << "Price: ";

cin >> price;

cout << "Publisher: ";

cin >> publisher;

cout << "Stock: ";

cin >> stock;

library[i] = Book(title, author, price, publisher, stock);

}

string inputAuthor, inputTitle;

cout << "Enter Author's Name: ";

cin >> inputAuthor;

cout << "Enter Title of Book (in lowercase): ";

cin >> inputTitle;

bool bookFound = false;

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

if (inputAuthor == library[i].author && inputTitle == library[i].title) {

int requiredCopies;

cout << "How many copies of this book are required: ";

cin >> requiredCopies;

library[i].processOrder(requiredCopies);

bookFound = true;

break;

}

}

if (!bookFound) {

cout << "Book not found" << endl;

}

return 0;

}

//Q5

#include <iostream>

#include <string>

using namespace std;

class Book {

public:

string title;

string author;

double price;

string publisher;

int stock;

int successfulTransactions;

int unsuccessfulTransactions;

Book(){

this->title = "";

this->author = "";

this->price = 0.0;

this->publisher = "";

this->stock = 0;

this->successfulTransactions = 0;

this->unsuccessfulTransactions = 0;

}

Book(string title, string author, double price, string publisher, int stock) {

this->title = title;

this->author = author;

this->price = price;

this->publisher = publisher;

this->stock = stock;

this->successfulTransactions = 0;

this->unsuccessfulTransactions = 0;

}

void processOrder(int requiredCopies) {

if (requiredCopies <= stock) {

cout << "\*\*\*\*\*\*\*\*\* Book Details \*\*\*\*\*\*\*\*\*" << endl;

cout << "Title\t\t\tAuthor\t\t\tStock Copies" << endl;

cout << title << "\t\t" << author << "\t\t" << stock << endl;

double totalPrice = price \* requiredCopies;

cout << "How many copies of this book is required: " << requiredCopies << endl;

cout << "Total price = " << totalPrice << " TK" << endl;

stock -= requiredCopies;

successfulTransactions++;

} else {

cout << "Required copies not in stock" << endl;

unsuccessfulTransactions++;

}

}

void displayBookInfo() {

cout << "Title\t\tStock Copies\tPrice per Book" << endl;

cout << title << "\t\t" << stock << "\t\t" << price << " TK" << endl;

cout << "Successful Transactions: " << successfulTransactions << endl;

cout << "Unsuccessful Transactions: " << unsuccessfulTransactions << endl;

}

private:

void updatePrice(double newPrice) {

price = newPrice;

}

};

int main() {

int numBooks;

cout << "Enter the number of available books: ";

cin >> numBooks;

Book bookCollection[numBooks];

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

string title, author, publisher;

double price;

int stock;

cout << "Enter book " << (i + 1)