DAY-03

Question 09 : Problem Statement: Implement a simplified e-commerce cart system. You need to write functions to add an item to the cart, calculate the total cost of items in the cart including a fixed tax rate, and remove an item from the cart. Assume each item is represented by a struct containing a name, price, and quantity.

Sol :

```
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>
using namespace std;

struct Item {
    string name;
    double price;
    int quantity;
};
class Cart {
private:
    vector<Item> items;
    double taxRate;

public:
    Cart(double tax) : taxRate(tax) {}

    void addItem(const Item& item) {
        items.push_back(item);
    }
    double calculateTotalCost() {
        double total = 0.0;
        for (const auto& item : items) {
            total += item.price * item.quantity;
        }
        return total * (1 + taxRate);
    }
    void removeItem(const string& itemName) {
        items.erase(remove_if(items.begin(), items.end(), [&itemName](const Item& item) {
            return item.name == itemName;
        }), items.end());
    }
};
```

```
int main() {
    Cart cart(0.1);
    Item item1, item2;
    cout << "Enter the name, price, and quantity for item 1: ";
    cin >> item1.name >> item1.price >> item1.quantity;
    cout << "Enter the name, price, and quantity for item 2: ";
    cin >> item2.name >> item2.price >> item2.quantity;

    cart.addItem(item1);
    cart.addItem(item2);
    cout << "Total cost including tax: " << cart.calculateTotalCost() << endl;
    string itemToRemove;
    cout << "Enter the name of the item to remove: ";
    cin >> itemToRemove;
    cart.removeItem(itemToRemove);
    cout << "Total cost including tax after removing an item: " << cart.calculateTotalCost() <<
endl;

    return 0;
}
```

```
O/P : Enter the name, price, and quantity for item 1: cola 40 2
Enter the name, price, and quantity for item 2: sprite 40 3
Total cost including tax: 220
Enter the name of the item to remove: cola
Total cost including tax after removing an item: 132

--------------------------------
Process exited after 16.05 seconds with return value 0
Press any key to continue . .
```

Question-10 : Problem Statement: Develop a set of functions as part of a matrix operations library. Include functions for creating a matrix, adding two matrices, multiplying a matrix by a scalar, and displaying a matrix. Assume matrices are represented by vectors of vectors.

Sol :

```
#include <iostream>
#include <vector>
using namespace std;
typedef vector<vector<int>> Matrix;
Matrix create_matrix(int rows, int cols) {
    return Matrix(rows, vector<int>(cols, 0));
}
Matrix add_matrices(const Matrix& A, const Matrix& B) {
```

```
    int rows = A.size();
    int cols = A[0].size();
    Matrix result = create_matrix(rows, cols);
    for (int i = 0; i < rows; ++i) {
        for (int j = 0; j < cols; ++j) {
            result[i][j] = A[i][j] + B[i][j];
        }
    }
    return result;
}
Matrix scalar_multiply(const Matrix& A, int scalar) {
    int rows = A.size();
    int cols = A[0].size();
    Matrix result = create_matrix(rows, cols);
    for (int i = 0; i < rows; ++i) {
        for (int j = 0; j < cols; ++j) {
            result[i][j] = A[i][j] * scalar;
        }
    }
    return result;
}
void display_matrix(const Matrix& A) {
    int rows = A.size();
    int cols = A[0].size();
    for (int i = 0; i < rows; ++i) {
        for (int j = 0; j < cols; ++j) {
            cout << A[i][j] << " ";
        }
        cout << endl;
    }
}
int main() {
    Matrix A = { {1, 2, 3}, {4, 5, 6}, {7, 8, 9} };
    Matrix B = { {9, 8, 7}, {6, 5, 4}, {3, 2, 1} };
    cout << "Matrix A:" << endl;
    display_matrix(A);
    cout << "Matrix B:" << endl;
    display_matrix(B);
    Matrix sum = add_matrices(A, B);
    cout << "Sum of A and B:" << endl;
    display_matrix(sum);
    Matrix scaled_A = scalar_multiply(A, 2);
    cout << "Scaled matrix A by 2:" << endl;
```

```
    display_matrix(scaled_A);
    return 0;
}
```

O/P : Matrix A:
1 2 3
4 5 6
7 8 9
Matrix B:
9 8 7
6 5 4
3 2 1
Sum of A and B:
10 10 10
10 10 10
10 10 10
Scaled matrix A by 2:
2 4 6
8 10 12
14 16 18

Question-11 :
(1) Write a c++ program to create a class for a bank account with a constructor and a destructor
Sol :

```
#include <iostream>
#include <string>
class BankAccount {
private:
    std::string accountNumber;
    double balance;
public:
    BankAccount(const std::string& accNumber, double initialBalance) :
accountNumber(accNumber), balance(initialBalance) {
        std::cout << "Bank account created\n";
    }
    ~BankAccount() {
        std::cout << "Bank account destroyed\n";
    }
};
int main() {
    BankAccount account("123456789", 1000.0);
    return 0;
}
```

O/P : Bank account created
Bank account destroyed

(2) : Write a c++ program to create a class for a car with a constructor and a destructor

Sol :
```
#include <iostream>
#include <string>
class Car {
private:
    std::string make;
    std::string model;
    int year;
public:
    Car(const std::string& carMake, const std::string& carModel, int carYear) : make(carMake),
model(carModel), year(carYear) {
        std::cout << "Car object created\n";
    }
    ~Car() {
        std::cout << "Car object destroyed\n";
    }
    void display() const {
        std::cout << "Make: " << make << ", Model: " << model << ", Year: " << year << std::endl;
    }
};
int main() {
    Car myCar("Toyota", "Corolla", 2020);
    myCar.display();
    return 0;
}
```

O/P : Car object created
Make: Toyota, Model: Corolla, Year: 2020
Car object destroyed

(3) : Write a c++ program to create a class for a rectangle with a constructor and a destructor

Sol :
```
#include <iostream>
class Rectangle {
private:
    double length;
    double width;
public:
```

```
    Rectangle(double l, double w) : length(l), width(w) {
        std::cout << "Rectangle object created\n";
    }
    ~Rectangle() {
        std::cout << "Rectangle object destroyed\n";
    }
    double area() const {
        return length * width;
    }
    double perimeter() const {
        return 2 * (length + width);
    }
};
int main() {
    Rectangle myRect(5.0, 3.0);
    std::cout << "Area: " << myRect.area() << std::endl;
    std::cout << "Perimeter: " << myRect.perimeter() << std::endl;

    return 0;
}
```

O/P : Rectangle object created
Area: 15
Perimeter: 16
Rectangle object destroyed

(4) : Write a c++ program to create a class for a book with a constructor and a destructor

Sol :

```
#include <iostream>
#include <string>
class Book {
private:
    std::string title;
    std::string author;
    int year;
public:
    Book(const std::string& title, const std::string& author, int year) : title(title), author(author),
year(year) {
        std::cout << "Book object created.\n";
    }
    ~Book() {
        std::cout << "Book object destroyed.\n";
    }
```

```
    void display() const {
        std::cout << "Title: " << title << ", Author: " << author << ", Year: " << year << std::endl;
    }
};
int main() {
    Book book("The Catcher in the Rye", "J.D. Salinger", 1951);
    book.display();

    return 0;
}
```

O/P : Book object created.
Title: The Catcher in the Rye, Author: J.D. Salinger, Year: 1951
Book object destroyed.

(5) : Write a c++ program to create a class for student with a constructor and a destructor

Sol :
```
#include <iostream>
#include <string>
class Student {
private:
    std::string name;
    int age;
public:
    Student(const std::string& name, int age) : name(name), age(age) {
        std::cout << "Student object created.\n";
    }
    ~Student() {
        std::cout << "Student object destroyed.\n";
    }
    void display() const {
        std::cout << "Name: " << name << ", Age: " << age << std::endl;
    }
};
int main() {
    Student student("John Doe", 20);
    student.display();

    return 0;
}
```

O/P : Student object created.
Name: John Doe, Age: 20
Student object destroyed.