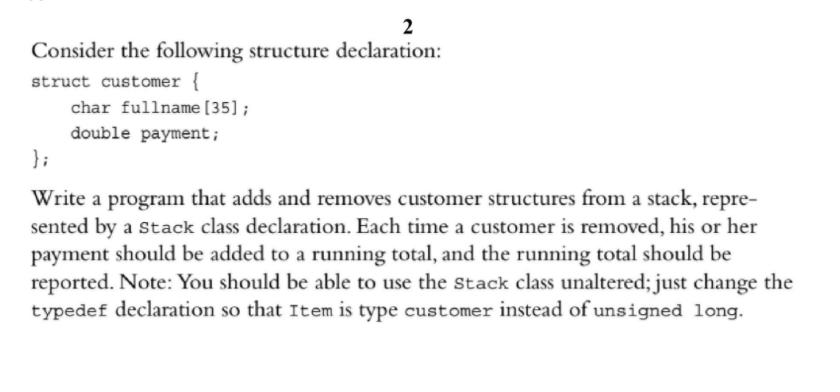
Rudenko Ruslan, SE-TE 2.01

**Task:**

****

**Code:**

#include <iostream>

#include <string>

#include <cstring>

// DEVELOPED BY RUDENKO RUSLAN SE-TE 2.01!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* DOMAIN THINGS

\*/

struct customer {

char fullname[35];

double payment;

};

typedef customer Item;

class Stack {

private:

enum {

MAX = 10

};

// constant specific to class

Item items[MAX];

// holds stack items

int top;

// index for top stack item

public:

Stack();

bool isempty() const;

bool isfull() const;

// push() returns false if stack already is full, true otherwise

bool push(const Item& item);

// add item to stack

// pop() returns false if stack already is empty, true otherwise

bool pop(Item& item);

// pop top into item

};

Stack::Stack()

// create an empty stack

{

top = 0;

}

bool Stack::isempty() const {

return top == 0;

}

bool Stack::isfull() const {

return top == MAX;

}

bool Stack::push(const Item& item) {

if (top < MAX) {

items[top++] = item;

return true;

}

else

return false;

}

bool Stack::pop(Item& item) {

if (top > 0) {

item = items[--top];

return true;

}

else

return false;

}

int main() {

using std::string;

using std::cout;

using std::endl;

using std::cin;

Stack myStack;

int selection = 0;

double total = 0;

cout << "Enter your selection" << endl;

cout << "1 - Add customer" << endl;

cout << "2 - Remove customer" << endl;

cout << "Any other character - quit" << endl;

while (cin >> selection && (selection == 1 || selection == 2)) {

cin.get();

if (selection == 1) {

if (myStack.isfull()) {

cout << "Cannot add customer. Stack is full." << endl;

}

else {

customer newCustomer;

cout << "Enter customer name: ";

cin.getline(newCustomer.fullname, 35);

cout << "Enter payment: ";

(cin >> newCustomer.payment).get();

myStack.push(newCustomer);

cout << "Customer " << newCustomer.fullname << " added with a payment of " << newCustomer.payment << endl;

}

}

else {

if (myStack.isempty()) {

cout << "Cannot remove customer. Stack is empty." << endl;

}

else {

customer aCustomer;

myStack.pop(aCustomer);

cout << "Customer " << aCustomer.fullname << " removed." << endl;

total += aCustomer.payment;

cout << "Running Total: " << total << endl;

}

}

cout << "Make another selection: ";

}

cout << "Done." << endl;

return 0;

}

**Results**

