FACULTY OF COMPUTING

SEMESTER 2 2023/2024

SECJ1023 – PROGRAMMING TECHNIQUE II

SECTION 2

PROJECT REPORT

LECTURER: DR. JOHANNA BINTI AHMAD

GROUP 7

STUDENT NAME	MATRIC NO
NEO LI XIN	A23CS0253
BRENDAN CHIA YAN FEI	A23CS0211
CHUA JIA LIN	A23CS0069

TABLE OF CONTENT

1.0 INTRODUCTION

1.1 SYNOPSIS PROJECT

In this project, we design a pharmacy system called Caring Pharmacy System to speed up the transaction process and generate the sales report. In this project, we will create functionalities for the admin and the customers using C++ programming language.

For the customers, they are required to enter their name, enter the amount of medicine to purchase, and select the payment method they preferred. At the end of the transaction process, the system will auto generate a receipt for the customers.

For the admin, which is the pharmacy workers, they are required to enter the admin password before they select the special operations. The special operations are viewing the number of medicines sold, medicines sales, restock medicine, and view customer list.

In this project, we implement plenty of techniques in our C++ coding for this Caring Pharmacy System. For example, we use encapsulation, data hiding, constructor, destructor, composition, aggregation, accessor and mutator. Besides, we also separate the definition and implementation of the classes in this system into hpp and cpp files.

1.2 OBJECTIVE OF THE PROJECT

The objective of designing this Caring Pharmacy System is to speed up the transaction process because customers can complete the process by themselves on the system without queuing in a long line and waiting the service from the pharmacy workers. Not to mention this automated system will help to reduce potential human errors. Other than that, the reason we designed this system is to generate the sales report. This system enables the admin, which is the pharmacy worker, to view the number of medicines sold and the sales of the medicines thus ease for the analysis and future decision making to increase the pharmacy sales.

2.0 SYSTEM ANALYSIS AND DESIGN (USE CASE AND CLASS DIAGRAM)

2.1 SYSTEM REQUIREMENTS

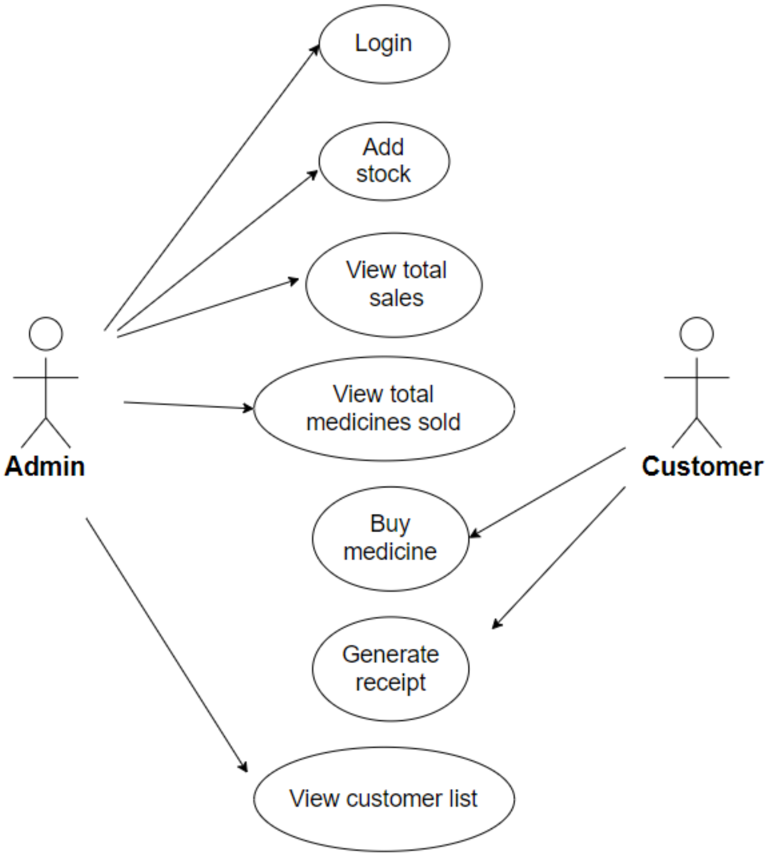


Figure 1: Use Case Diagram for Caring Pharmacy System

Detailed Description for Each Use Cases

The system has 6 main use cases.

Use Case	Purpose
Buy Medicines	Customers can enter their name and the number of medicines they want to buy for three types of medicines, then choose the payment method.
Generate Receipt	After the customers buy the medicines, the system will generate a receipt for customers. The receipt contains information about the customer's name, the number of medicines bought by the customer for three types of medicines, and the customer's payment method.
Log In	Admins need to log in to the system via password "CP123" to undergo several special functions such as view total medicines sold, view total sales, and add stock.
View Total Medicines Sold	Admins can view the number of medicines sold for each type of medicine and the total number of medicines sold.
View Total Sales	Admins can view the sales for each type of medicine and the total sales of the medicines.
Add Stock	Admins can view the current stock for each type of medicine before entering the number of stocks added for each type of medicine. Then, admins can view the latest stock for each type of medicine.
View Customer List	Admins can view the list of customers who purchased medicines in the pharmacy.

2.2 SYSTEM DESIGN

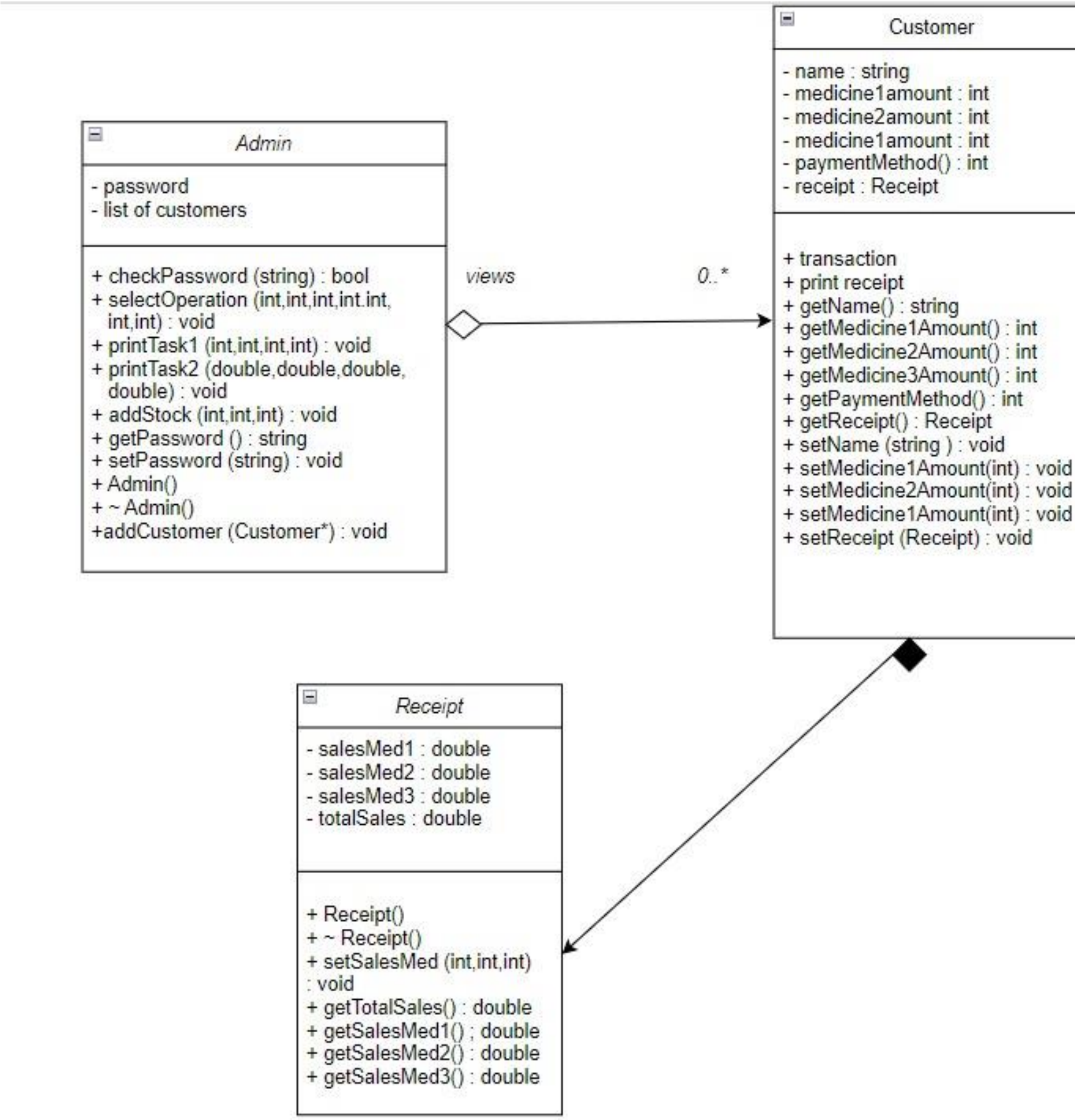
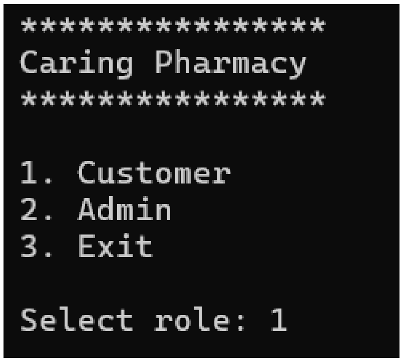

Admin
- password
- list of customers
+ checkPassword (string) : bool
+ selectOperation (int,int,int,int.int, int,int) : void
+ printTask1 (int,int,int,int) : void
+ printTask2 (double,double,double, double) : void
+ addStock (int,int,int) : void
+ getPassword () : string
+ setPassword (string) : void
+ Admin()
+ ~ Admin()
+addCustomer (Customer*) : void
views
0..*
Customer
- name : string
- medicine1amount : int
- medicine2amount : int
- medicine1amount : int
- paymentMethod() : int
- receipt : Receipt
+ transaction
+ print receipt
+ getName() : string
+ getMedicine1Amount() : int
+ getMedicine2Amount() : int
+ getMedicine3Amount() : int
+ getPaymentMethod() : int
+ getReceipt() : Receipt
+ setName (string) : void
+ setMedicine1Amount(int) : void
+ setMedicine2Amount(int) : void
+ setMedicine1Amount(int) : void
+ setReceipt (Receipt) : void
Receipt
- salesMed1 : double
- salesMed2 : double
- salesMed3 : double
- totalSales : double
+ Receipt()
+ ~ Receipt()
+ setSalesMed (int,int,int) : void
+ getTotalSales() : double
+ getSalesMed1() ; double
+ getSalesMed2() : double
+ getSalesMed3() : double

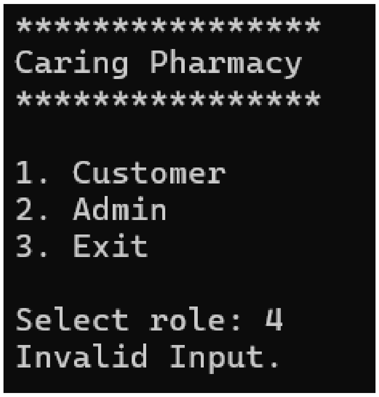
Figure 2: Class Diagram for Caring Pharmacy System

3.0 SYSTEM PROTOTYPE

Screen 1: At the start of the system, people who use it must choose their role, either customer or admin, or exit the system. If the input number is not 1,2 or 3, the system will show invalid input.



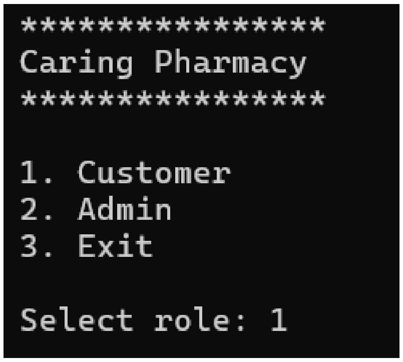
```
****************
Caring Pharmacy
****************

1. Customer
2. Admin
3. Exit

Select role: 1
```



```
****************
Caring Pharmacy
****************

1. Customer
2. Admin
3. Exit

Select role: 4
Invalid Input.
```

3.1 CUSTOMER SYSTEM

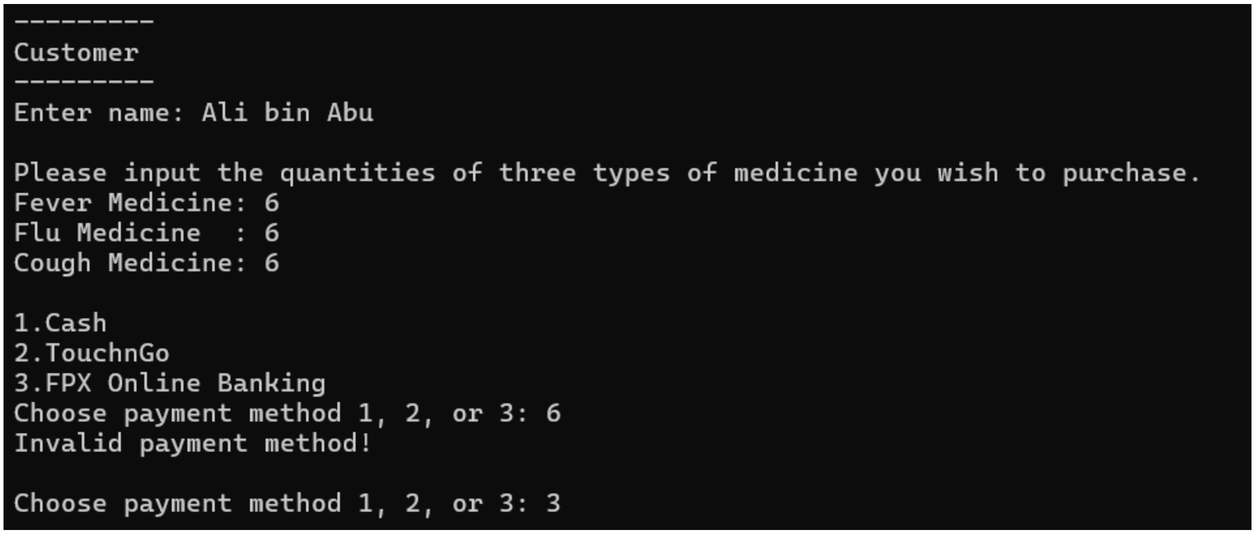
Screen 1: The customer must proceed with the system by entering ‘1’ in the input.



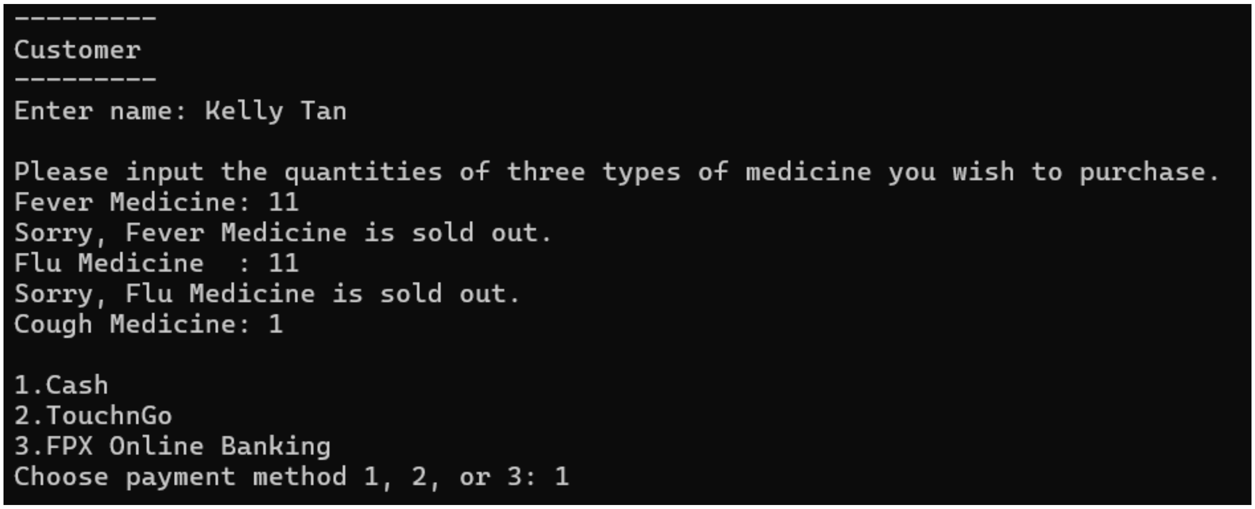
```
****************
Caring Pharmacy
****************

1. Customer
2. Admin
3. Exit

Select role: 1
```

Screen 2: Next, the customer is required to enter his/her name first. Then customers should input the number of medicines they want to buy, if none then they should put 0. If the number of medicines the customer wants to buy is more than the current stock, a notification will pop out to notify that the current stock is insufficient.

After that, the customer must choose a payment method. If the input number is larger than 3, then the system will output invalid payment method.

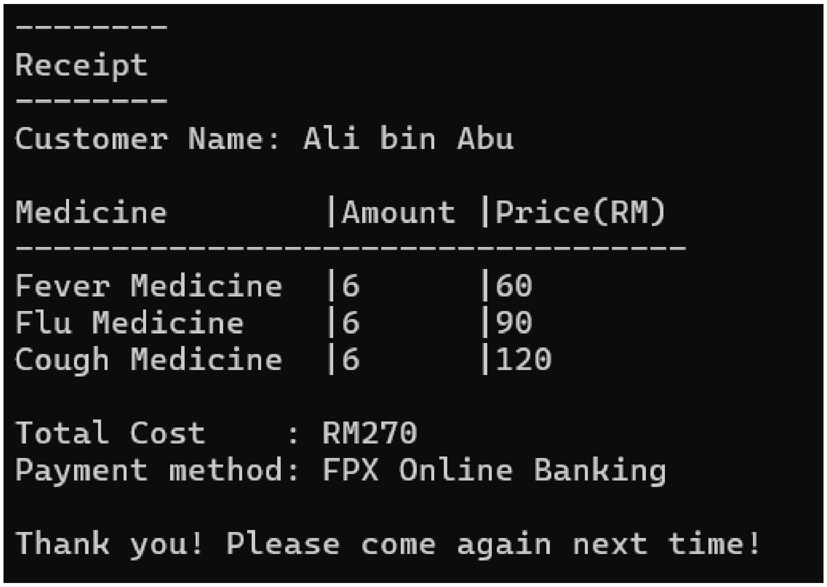


```
---------
Customer
---------
Enter name: Ali bin Abu

Please input the quantities of three types of medicine you wish to purchase.
Fever Medicine: 6
Flu Medicine  : 6
Cough Medicine: 6

1.Cash
2.TouchnGo
3.FPX Online Banking
Choose payment method 1, 2, or 3: 6
Invalid payment method!

Choose payment method 1, 2, or 3: 3
```

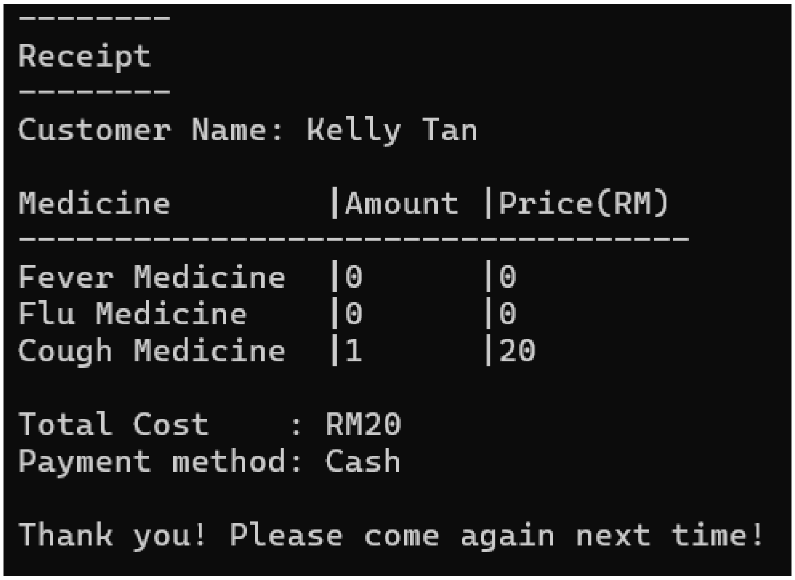


```
---------
Customer
---------
Enter name: Kelly Tan

Please input the quantities of three types of medicine you wish to purchase.
Fever Medicine: 11
Sorry, Fever Medicine is sold out.
Flu Medicine  : 11
Sorry, Flu Medicine is sold out.
Cough Medicine: 1

1.Cash
2.TouchnGo
3.FPX Online Banking
Choose payment method 1, 2, or 3: 1
```

Screen 3: Lastly, a receipt will be printed out. The receipt will show the name, amount of medicines bought, total cost and the chosen payment method.



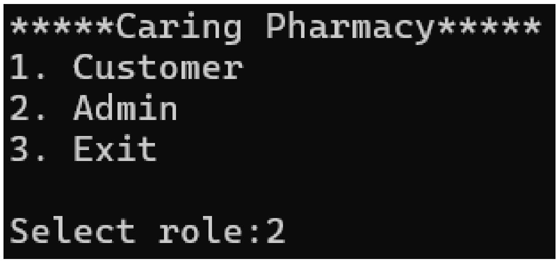
```
--------
Receipt
--------
Customer Name: Ali bin Abu

Medicine        |Amount |Price(RM)
-----------------------------------
Fever Medicine  |6      |60
Flu Medicine    |6      |90
Cough Medicine  |6      |120

Total Cost    : RM270
Payment method: FPX Online Banking

Thank you! Please come again next time!
```



```
--------
Receipt
--------
Customer Name: Kelly Tan

Medicine        |Amount |Price(RM)
-----------------------------------
Fever Medicine  |0      |0
Flu Medicine    |0      |0
Cough Medicine  |1      |20

Total Cost    : RM20
Payment method: Cash

Thank you! Please come again next time!
```

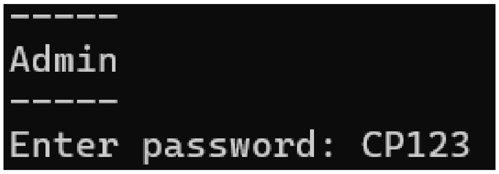
3.2 ADMIN SYSTEM

Screen 1: For the admin, they should access the administrative access page by entering ‘2’ in the input.

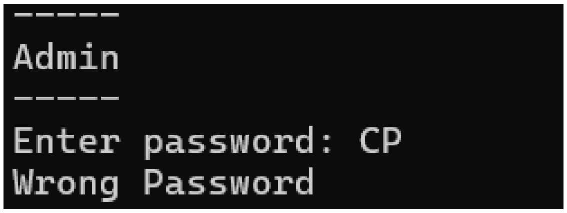


```
*****Caring Pharmacy*****
1. Customer
2. Admin
3. Exit

Select role:2
```

Screen 2: Next, admin must enter the correct password to access the next page. If the password entered is incorrect, an error message will be shown, and the admin will be sent back to the main page.

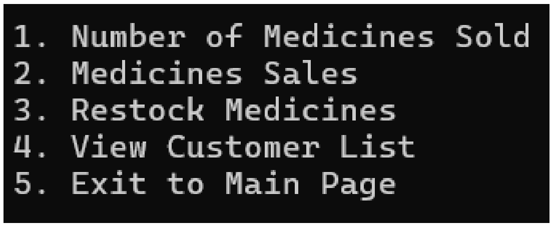


```
-----
Admin
-----
Enter password: CP123
```



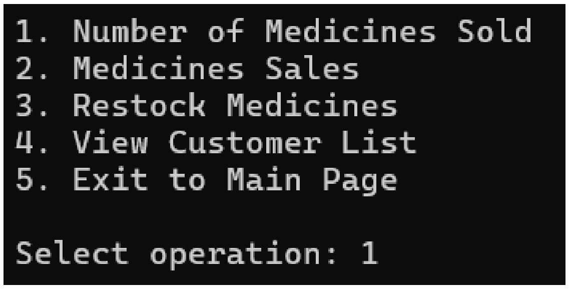
```
-----
Admin
-----
Enter password: CP
Wrong Password
```

Screen 3: In the administrative access page, the admin can choose 5 of the following tasks: Check number of medicines sold, check total medicines sales, restock medicines, view customer list or exit to main page.

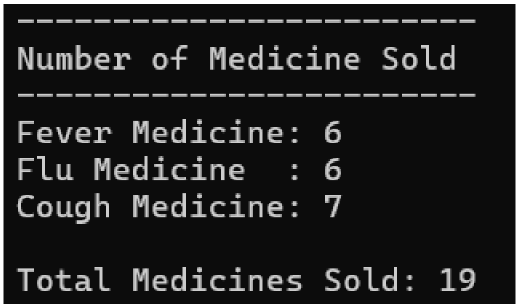


```
1. Number of Medicines Sold
2. Medicines Sales
3. Restock Medicines
4. View Customer List
5. Exit to Main Page
```

Screen 4: After choosing task 1, admin will be able to check the medicine sold for each type of medicine and the total number of medicines sold.

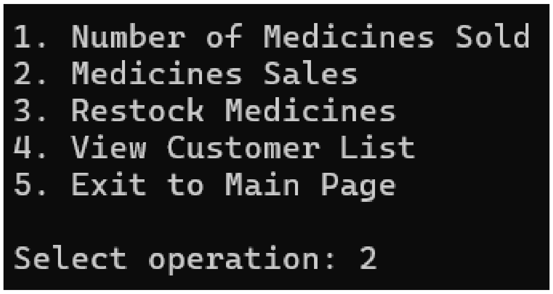


```
1. Number of Medicines Sold
2. Medicines Sales
3. Restock Medicines
4. View Customer List
5. Exit to Main Page

Select operation: 1
```

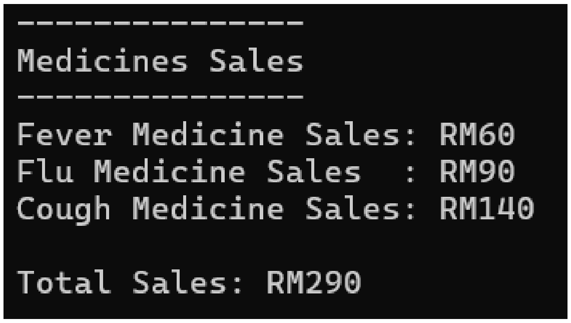


```
------------------------
Number of Medicine Sold
------------------------
Fever Medicine: 6
Flu Medicine  : 6
Cough Medicine: 7

Total Medicines Sold: 19
```

Screen 5: After choosing task 2, admin will be able to check the medicine sales for each type of medicine and the total sales of all medicines.

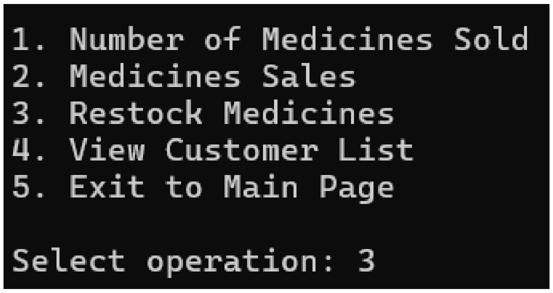


```
1. Number of Medicines Sold
2. Medicines Sales
3. Restock Medicines
4. View Customer List
5. Exit to Main Page

Select operation: 2
```

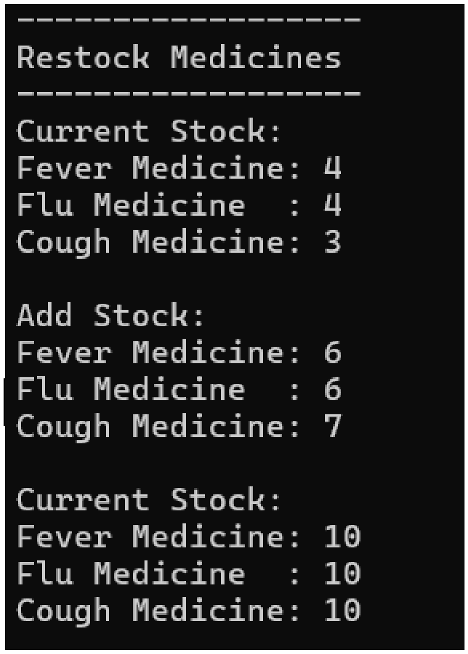


```
---------------
Medicines Sales
---------------
Fever Medicine Sales: RM60
Flu Medicine Sales  : RM90
Cough Medicine Sales: RM140

Total Sales: RM290
```

Screen 6: After choosing task 3, admin will be able to restock each type of medicine and the current stock will be updated.

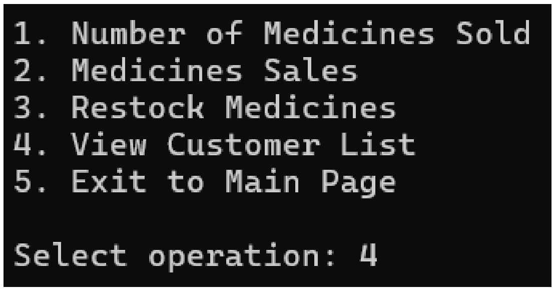


```
1. Number of Medicines Sold
2. Medicines Sales
3. Restock Medicines
4. View Customer List
5. Exit to Main Page

Select operation: 3
```

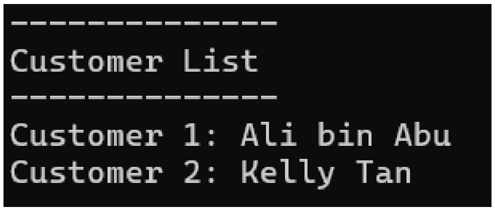


```
------------------
Restock Medicines
------------------
Current Stock:
Fever Medicine: 4
Flu Medicine  : 4
Cough Medicine: 3

Add Stock:
Fever Medicine: 6
Flu Medicine  : 6
Cough Medicine: 7

Current Stock:
Fever Medicine: 10
Flu Medicine  : 10
Cough Medicine: 10
```

Screen 7: After choosing task 4, admin will be able to check the names of the customers that had visited the clinic recently.

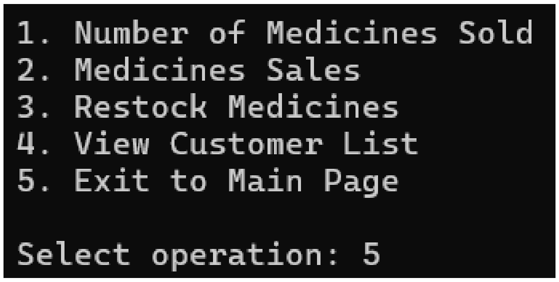


```
1. Number of Medicines Sold
2. Medicines Sales
3. Restock Medicines
4. View Customer List
5. Exit to Main Page

Select operation: 4
```



```
--------------
Customer List
--------------
Customer 1: Ali bin Abu
Customer 2: Kelly Tan
```

Screen 8: When the admin chooses operation 5, the admin will return to the main page.

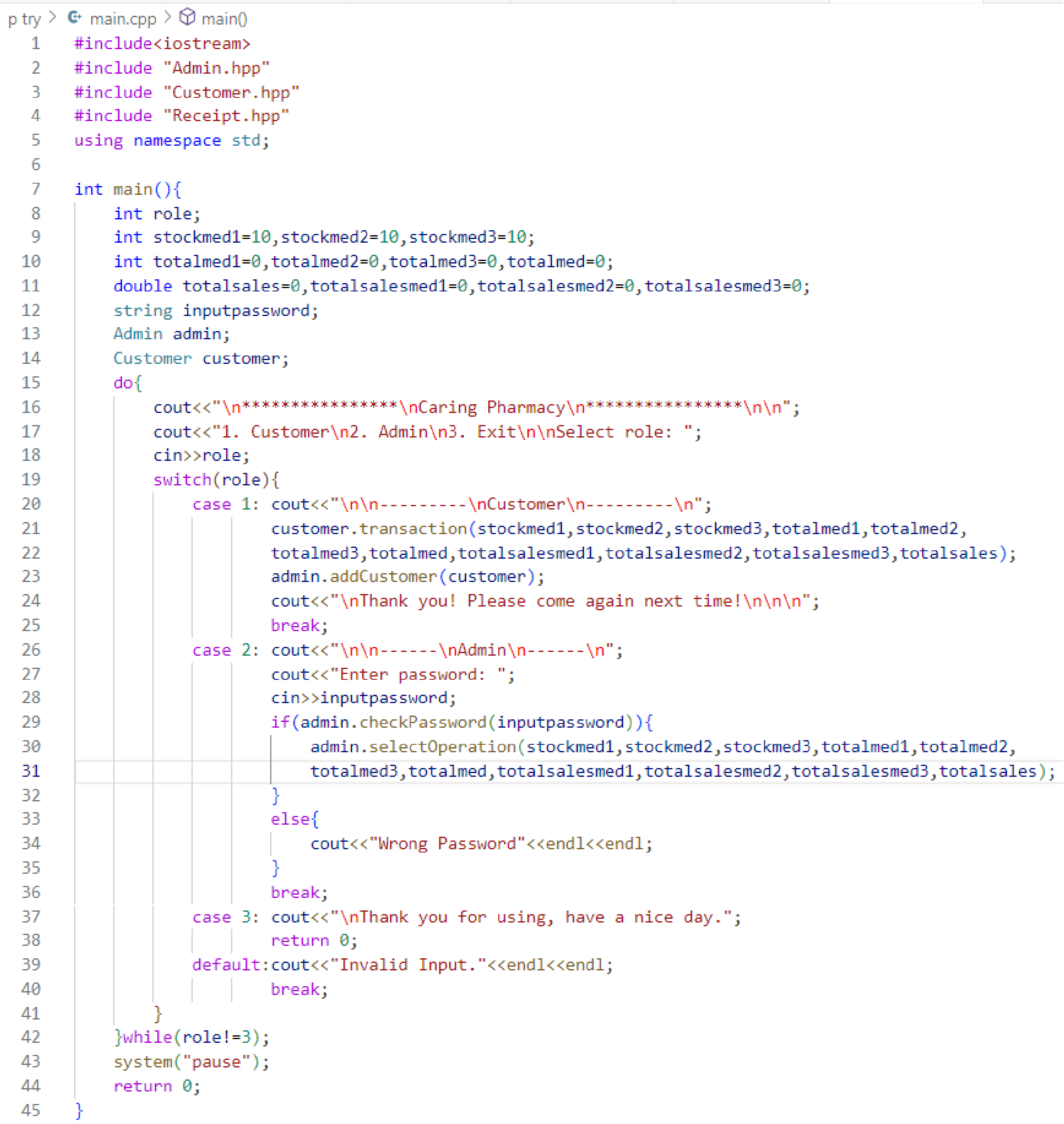


```
1. Number of Medicines Sold
2. Medicines Sales
3. Restock Medicines
4. View Customer List
5. Exit to Main Page

Select operation: 5
```

-------- END OF PROGRAM --------

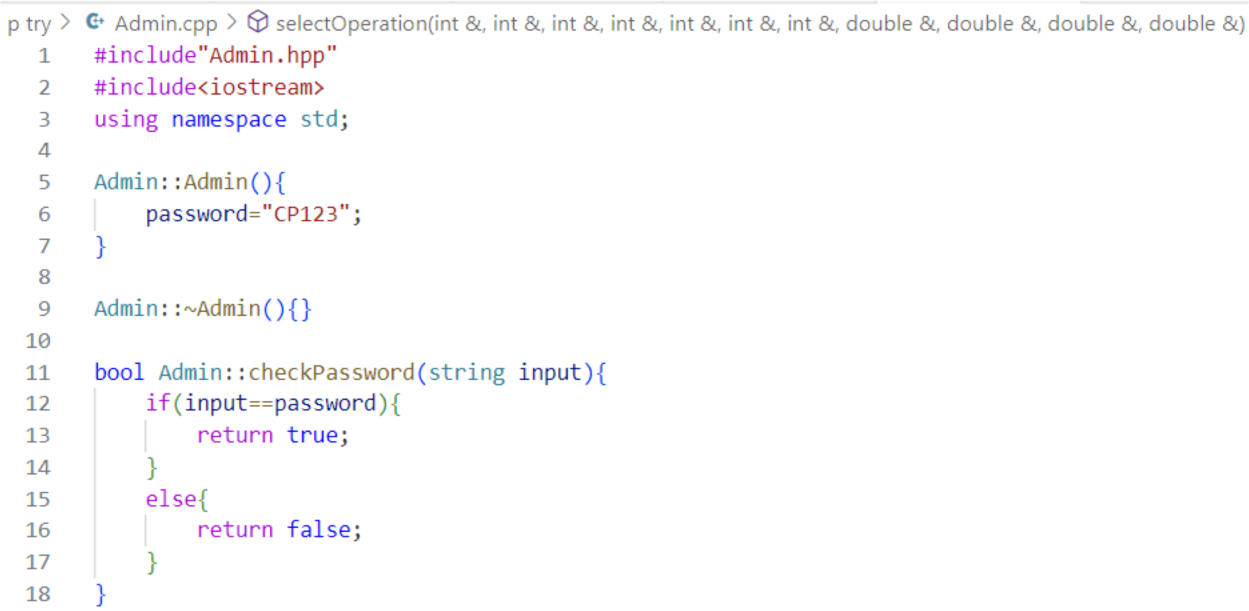
4.0 APPENDIX

main.cpp

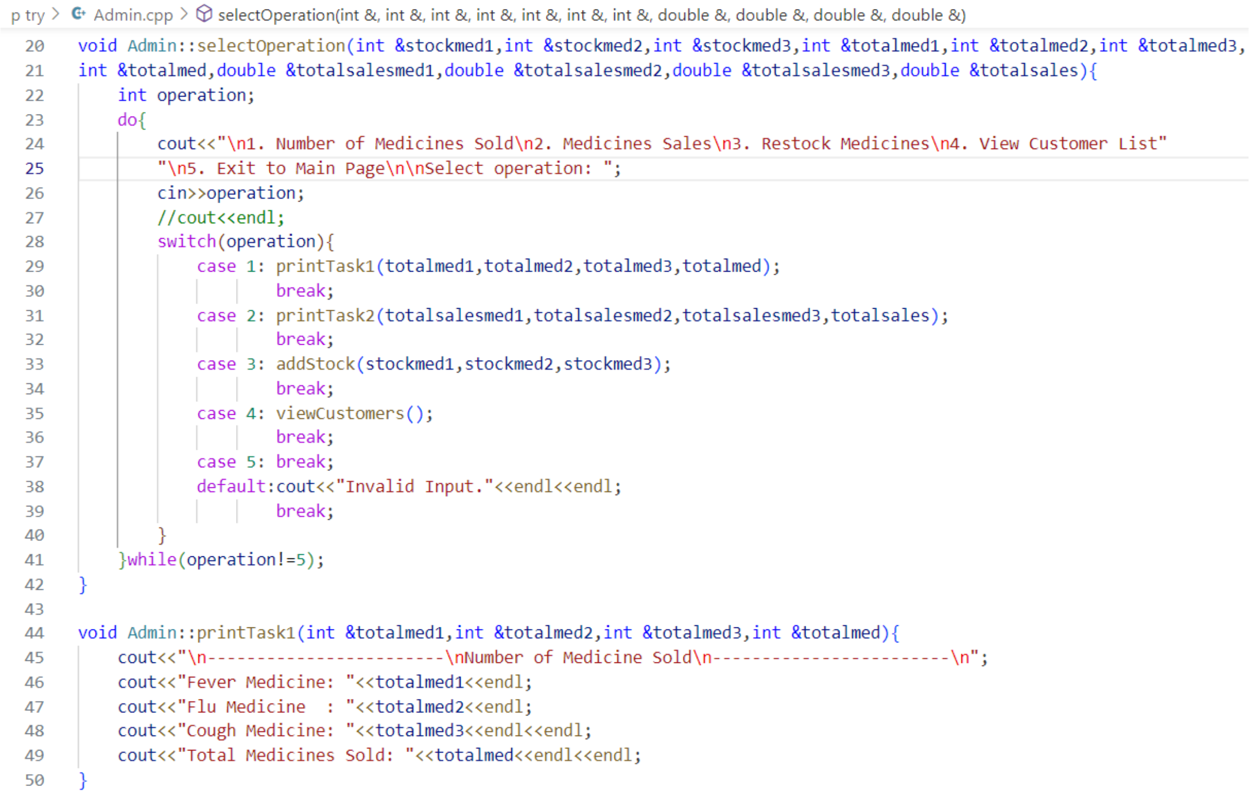


```cpp
#include<iostream>
#include "Admin.hpp"
#include "Customer.hpp"
#include "Receipt.hpp"
using namespace std;

int main(){
    int role;
    int stockmed1=10,stockmed2=10,stockmed3=10;
    int totalmed1=0,totalmed2=0,totalmed3=0,totalmed=0;
    double totalsales=0,totalsalesmed1=0,totalsalesmed2=0,totalsalesmed3=0;
    string inputpassword;
    Admin admin;
    Customer customer;
    do{
        cout<<"\n****************\nCaring Pharmacy\n****************\n\n";
        cout<<"1. Customer\n2. Admin\n3. Exit\n\nSelect role: ";
        cin>>role;
        switch(role){
            case 1: cout<<"\n\n---------\nCustomer\n---------\n";
                    customer.transaction(stockmed1,stockmed2,stockmed3,totalmed1,totalmed2,
                    totalmed3,totalmed,totalsalesmed1,totalsalesmed2,totalsalesmed3,totalsales);
                    admin.addCustomer(customer);
                    cout<<"\nThank you! Please come again next time!\n\n\n";
                    break;
            case 2: cout<<"\n\n------\nAdmin\n------\n";
                    cout<<"Enter password: ";
                    cin>>inputpassword;
                    if(admin.checkPassword(inputpassword)){
                        admin.selectOperation(stockmed1,stockmed2,stockmed3,totalmed1,totalmed2,
                        totalmed3,totalmed,totalsalesmed1,totalsalesmed2,totalsalesmed3,totalsales);
                    }
                    else{
                        cout<<"Wrong Password"<<endl<<endl;
                    }
                    break;
            case 3: cout<<"\nThank you for using, have a nice day.";
                    return 0;
            default:cout<<"Invalid Input."<<endl<<endl;
                    break;
        }
    }while(role!=3);
    system("pause");
    return 0;
}
```

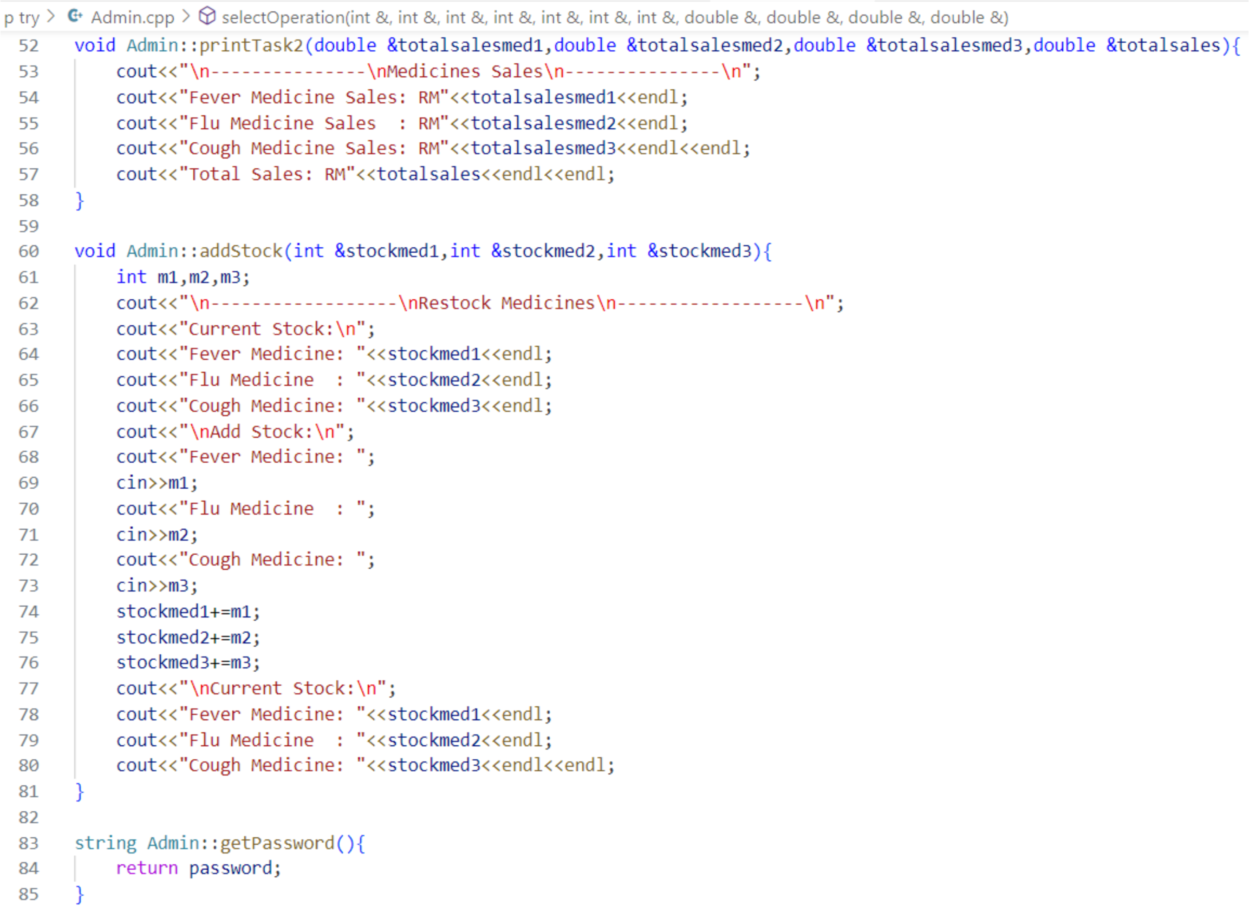
Admin.cpp


p try > Admin.cpp > selectOperation(int &, int &, int &, int &, int &, int &, int &, double &, double &, double &, double &)

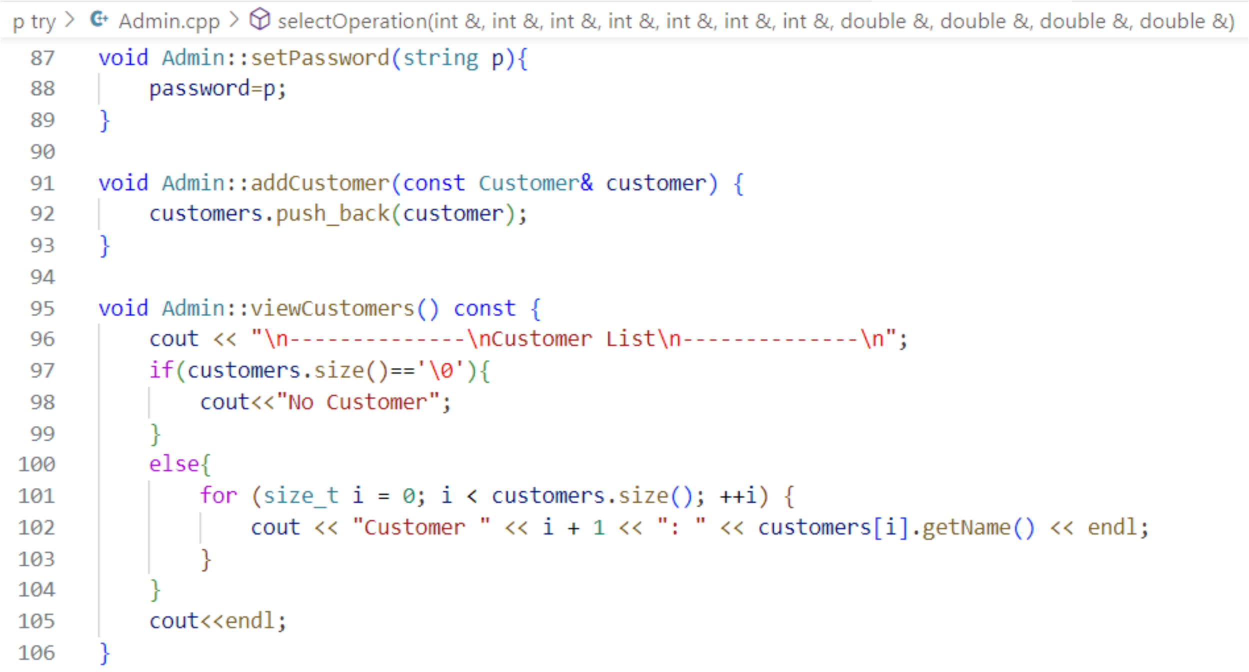
```
#include"Admin.hpp"
#include<iostream>
using namespace std;

Admin::Admin(){
    password="CP123";
}

Admin::~Admin(){}

bool Admin::checkPassword(string input){
    if(input==password){
        return true;
    }
    else{
        return false;
    }
}
```


p try > Admin.cpp > selectOperation(int &, int &, int &, int &, int &, int &, int &, double &, double &, double &, double &)

```
void Admin::selectOperation(int &stockmed1,int &stockmed2,int &stockmed3,int &totalmed1,int &totalmed2,int &totalmed3,
int &totalmed,double &totalsalesmed1,double &totalsalesmed2,double &totalsalesmed3,double &totalsales){
    int operation;
    do{
        cout<<"\n1. Number of Medicines Sold\n2. Medicines Sales\n3. Restock Medicines\n4. View Customer List"
        "\n5. Exit to Main Page\n\nSelect operation: ";
        cin>>operation;
        //cout<<endl;
        switch(operation){
            case 1: printTask1(totalmed1,totalmed2,totalmed3,totalmed);
                    break;
            case 2: printTask2(totalsalesmed1,totalsalesmed2,totalsalesmed3,totalsales);
                    break;
            case 3: addStock(stockmed1,stockmed2,stockmed3);
                    break;
            case 4: viewCustomers();
                    break;
            case 5: break;
            default:cout<<"Invalid Input."<<endl<<endl;
                    break;
        }
    }while(operation!=5);
}

void Admin::printTask1(int &totalmed1,int &totalmed2,int &totalmed3,int &totalmed){
    cout<<"\n------------------------\nNumber of Medicine Sold\n------------------------\n";
    cout<<"Fever Medicine: "<<totalmed1<<endl;
    cout<<"Flu Medicine  : "<<totalmed2<<endl;
    cout<<"Cough Medicine: "<<totalmed3<<endl<<endl;
    cout<<"Total Medicines Sold: "<<totalmed<<endl<<endl;
}
```

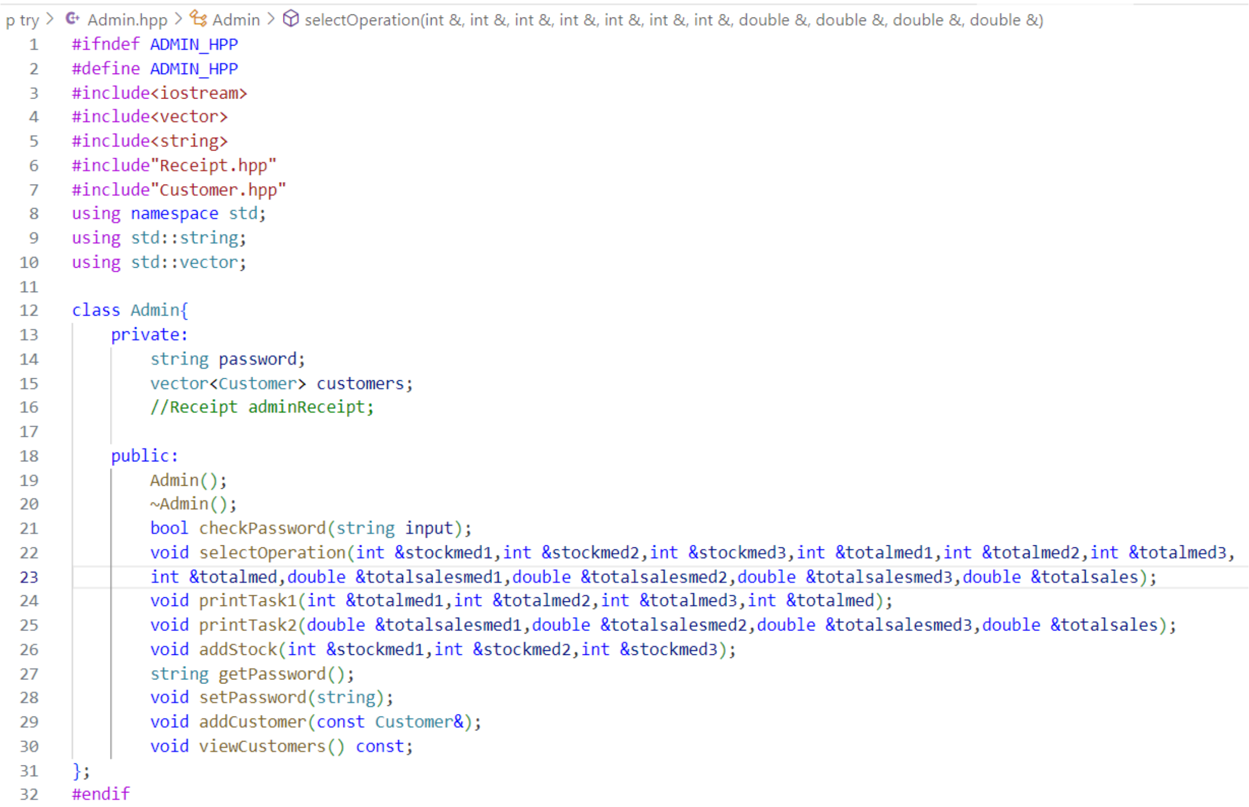

p try > Admin.cpp > selectOperation(int &, int &, int &, int &, int &, int &, int &, double &, double &, double &, double &)

```cpp
void Admin::printTask2(double &totalsalesmed1,double &totalsalesmed2,double &totalsalesmed3,double &totalsales){
    cout<<"\n--------------\nMedicines Sales\n--------------\n";
    cout<<"Fever Medicine Sales: RM"<<totalsalesmed1<<endl;
    cout<<"Flu Medicine Sales  : RM"<<totalsalesmed2<<endl;
    cout<<"Cough Medicine Sales: RM"<<totalsalesmed3<<endl<<endl;
    cout<<"Total Sales: RM"<<totalsales<<endl<<endl;
}

void Admin::addStock(int &stockmed1,int &stockmed2,int &stockmed3){
    int m1,m2,m3;
    cout<<"\n-----------------\nRestock Medicines\n-----------------\n";
    cout<<"Current Stock:\n";
    cout<<"Fever Medicine: "<<stockmed1<<endl;
    cout<<"Flu Medicine  : "<<stockmed2<<endl;
    cout<<"Cough Medicine: "<<stockmed3<<endl;
    cout<<"\nAdd Stock:\n";
    cout<<"Fever Medicine: ";
    cin>>m1;
    cout<<"Flu Medicine  : ";
    cin>>m2;
    cout<<"Cough Medicine: ";
    cin>>m3;
    stockmed1+=m1;
    stockmed2+=m2;
    stockmed3+=m3;
    cout<<"\nCurrent Stock:\n";
    cout<<"Fever Medicine: "<<stockmed1<<endl;
    cout<<"Flu Medicine  : "<<stockmed2<<endl;
    cout<<"Cough Medicine: "<<stockmed3<<endl<<endl;
}

string Admin::getPassword(){
    return password;
}
```


p try > Admin.cpp > selectOperation(int &, int &, int &, int &, int &, int &, int &, double &, double &, double &, double &)

```cpp
void Admin::setPassword(string p){
    password=p;
}

void Admin::addCustomer(const Customer& customer) {
    customers.push_back(customer);
}

void Admin::viewCustomers() const {
    cout << "\n--------------\nCustomer List\n--------------\n";
    if(customers.size()=='\0'){
        cout<<"No Customer";
    }
    else{
        for (size_t i = 0; i < customers.size(); ++i) {
            cout << "Customer " << i + 1 << ": " << customers[i].getName() << endl;
        }
    }
    cout<<endl;
}
```

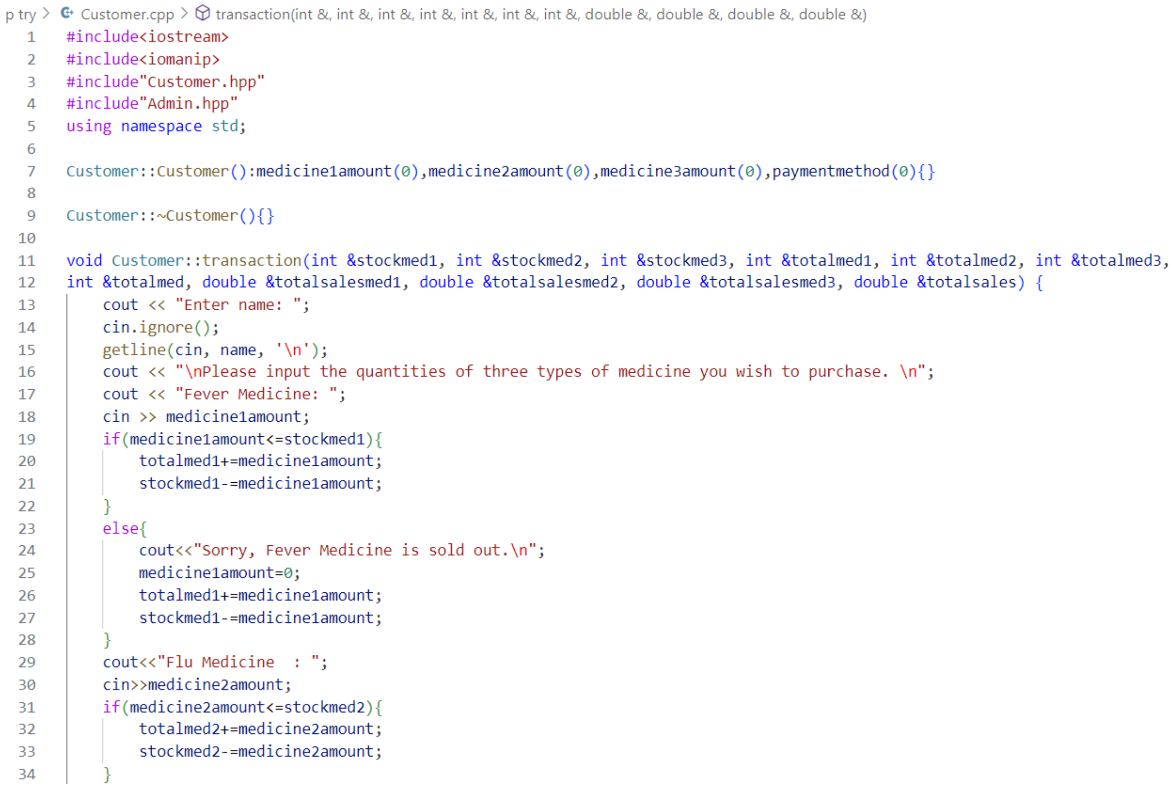
Admin.hpp



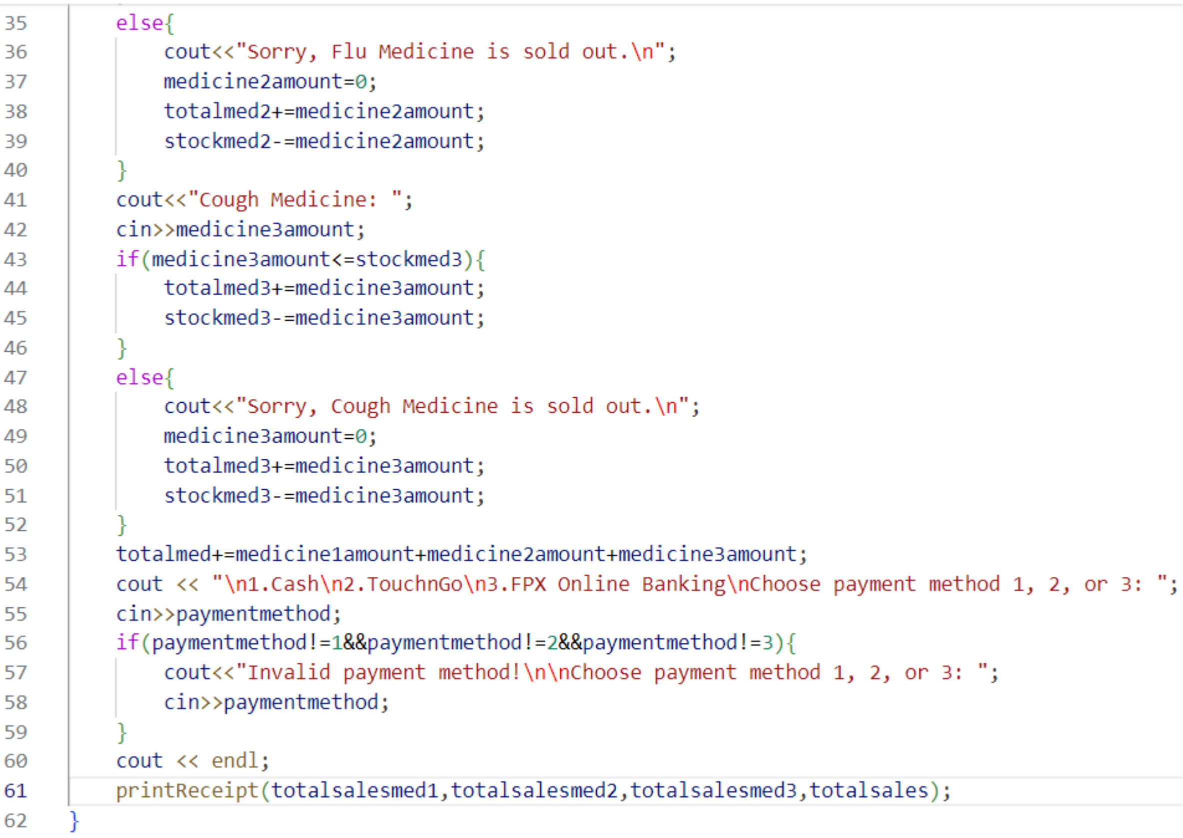
p try > Admin.hpp > Admin > selectOperation(int &, int &, int &, int &, int &, int &, double &, double &, double &, double &)

```
#ifndef ADMIN_HPP
#define ADMIN_HPP
#include<iostream>
#include<vector>
#include<string>
#include"Receipt.hpp"
#include"Customer.hpp"
using namespace std;
using std::string;
using std::vector;

class Admin{
    private:
        string password;
        vector<Customer> customers;
        //Receipt adminReceipt;

    public:
        Admin();
        ~Admin();
        bool checkPassword(string input);
        void selectOperation(int &stockmed1,int &stockmed2,int &stockmed3,int &totalmed1,int &totalmed2,int &totalmed3,
        int &totalmed,double &totalsalesmed1,double &totalsalesmed2,double &totalsalesmed3,double &totalsales);
        void printTask1(int &totalmed1,int &totalmed2,int &totalmed3,int &totalmed);
        void printTask2(double &totalsalesmed1,double &totalsalesmed2,double &totalsalesmed3,double &totalsales);
        void addStock(int &stockmed1,int &stockmed2,int &stockmed3);
        string getPassword();
        void setPassword(string);
        void addCustomer(const Customer&);
        void viewCustomers() const;
};
#endif
```

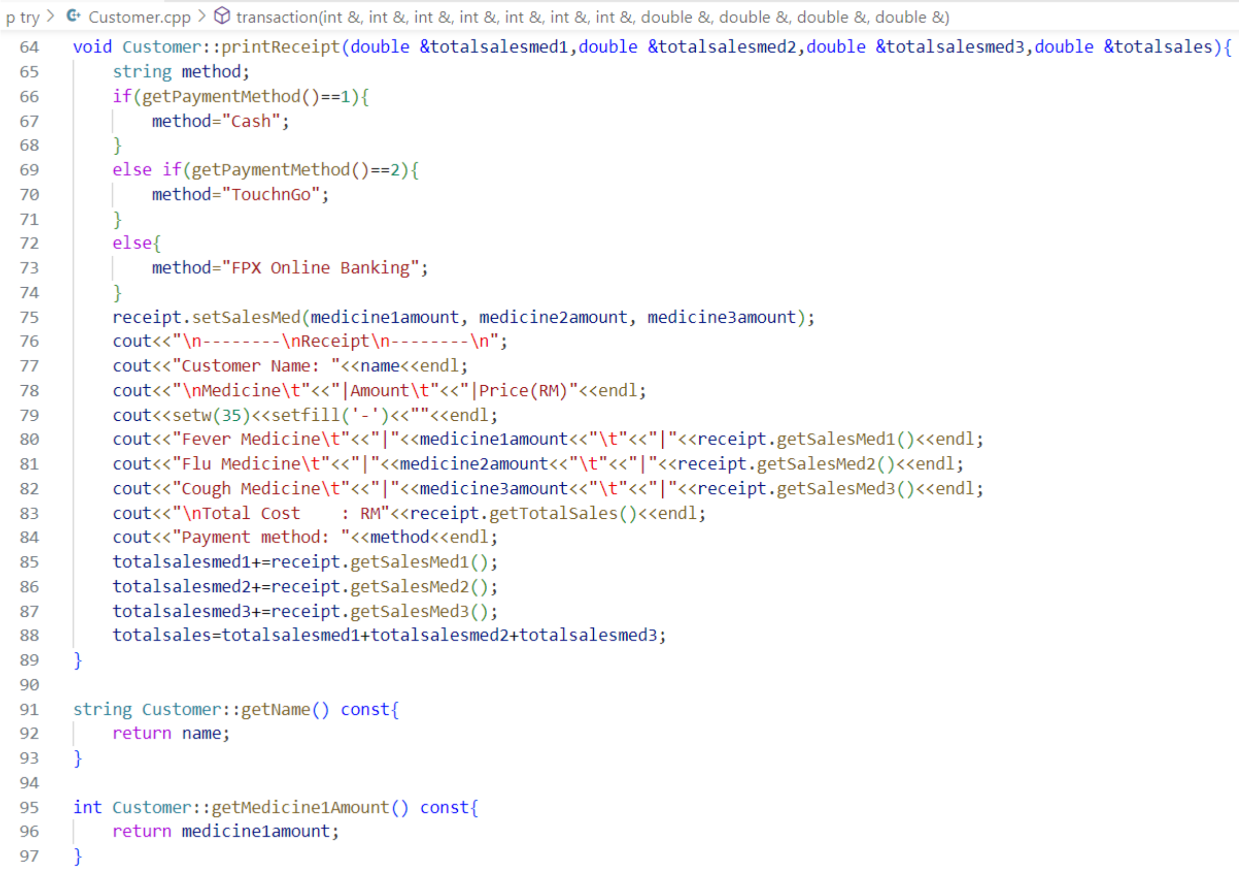
Customer.cpp


p try > Customer.cpp > transaction(int &, int &, int &, int &, int &, int &, int &, double &, double &, double &, double &)

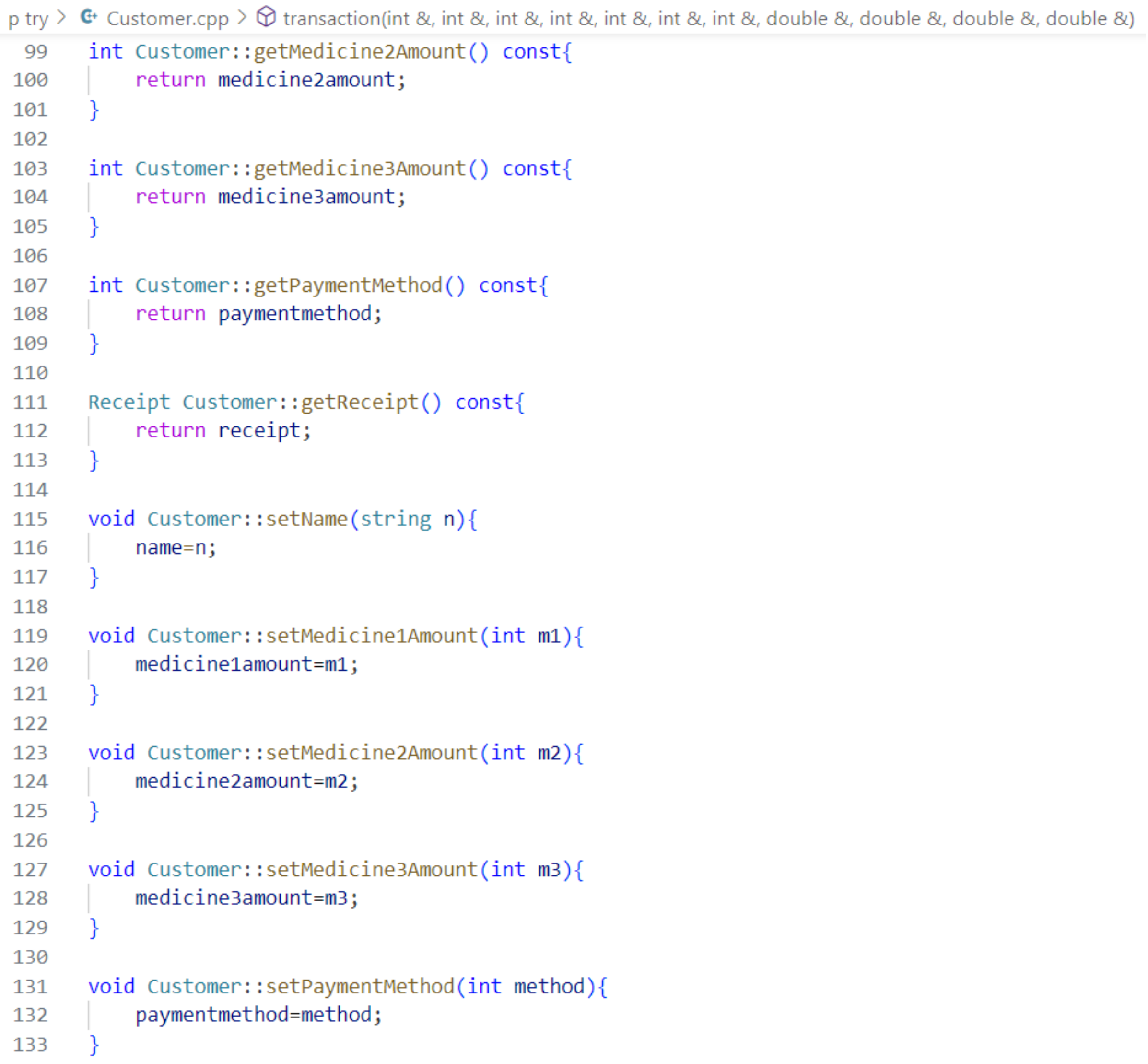
```cpp
#include<iostream>
#include<iomanip>
#include"Customer.hpp"
#include"Admin.hpp"
using namespace std;

Customer::Customer():medicine1amount(0),medicine2amount(0),medicine3amount(0),paymentmethod(0){}

Customer::~Customer(){}

void Customer::transaction(int &stockmed1, int &stockmed2, int &stockmed3, int &totalmed1, int &totalmed2, int &totalmed3,
int &totalmed, double &totalsalesmed1, double &totalsalesmed2, double &totalsalesmed3, double &totalsales) {
    cout << "Enter name: ";
    cin.ignore();
    getline(cin, name, '\n');
    cout << "\nPlease input the quantities of three types of medicine you wish to purchase. \n";
    cout << "Fever Medicine: ";
    cin >> medicine1amount;
    if(medicine1amount<=stockmed1){
        totalmed1+=medicine1amount;
        stockmed1-=medicine1amount;
    }
    else{
        cout<<"Sorry, Fever Medicine is sold out.\n";
        medicine1amount=0;
        totalmed1+=medicine1amount;
        stockmed1-=medicine1amount;
    }
    cout<<"Flu Medicine  : ";
    cin>>medicine2amount;
    if(medicine2amount<=stockmed2){
        totalmed2+=medicine2amount;
        stockmed2-=medicine2amount;
    }
```



```cpp
    else{
        cout<<"Sorry, Flu Medicine is sold out.\n";
        medicine2amount=0;
        totalmed2+=medicine2amount;
        stockmed2-=medicine2amount;
    }
    cout<<"Cough Medicine: ";
    cin>>medicine3amount;
    if(medicine3amount<=stockmed3){
        totalmed3+=medicine3amount;
        stockmed3-=medicine3amount;
    }
    else{
        cout<<"Sorry, Cough Medicine is sold out.\n";
        medicine3amount=0;
        totalmed3+=medicine3amount;
        stockmed3-=medicine3amount;
    }
    totalmed+=medicine1amount+medicine2amount+medicine3amount;
    cout << "\n1.Cash\n2.TouchnGo\n3.FPX Online Banking\nChoose payment method 1, 2, or 3: ";
    cin>>paymentmethod;
    if(paymentmethod!=1&&paymentmethod!=2&&paymentmethod!=3){
        cout<<"Invalid payment method!\n\nChoose payment method 1, 2, or 3: ";
        cin>>paymentmethod;
    }
    cout << endl;
    printReceipt(totalsalesmed1,totalsalesmed2,totalsalesmed3,totalsales);
}
```

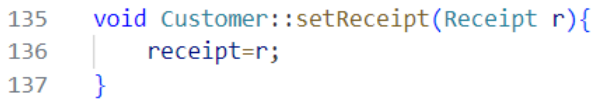

p try > Customer.cpp > transaction(int &, int &, int &, int &, int &, int &, double &, double &, double &, double &)

```cpp
void Customer::printReceipt(double &totalsalesmed1,double &totalsalesmed2,double &totalsalesmed3,double &totalsales){
    string method;
    if(getPaymentMethod()==1){
        method="Cash";
    }
    else if(getPaymentMethod()==2){
        method="TouchnGo";
    }
    else{
        method="FPX Online Banking";
    }
    receipt.setSalesMed(medicine1amount, medicine2amount, medicine3amount);
    cout<<"\n--------\nReceipt\n--------\n";
    cout<<"Customer Name: "<<name<<endl;
    cout<<"\nMedicine\t"<<"|Amount\t"<<"|Price(RM)"<<endl;
    cout<<setw(35)<<setfill('-')<<""<<endl;
    cout<<"Fever Medicine\t"<<"|"<<medicine1amount<<"\t"<<"|"<<receipt.getSalesMed1()<<endl;
    cout<<"Flu Medicine\t"<<"|"<<medicine2amount<<"\t"<<"|"<<receipt.getSalesMed2()<<endl;
    cout<<"Cough Medicine\t"<<"|"<<medicine3amount<<"\t"<<"|"<<receipt.getSalesMed3()<<endl;
    cout<<"\nTotal Cost    : RM"<<receipt.getTotalSales()<<endl;
    cout<<"Payment method: "<<method<<endl;
    totalsalesmed1+=receipt.getSalesMed1();
    totalsalesmed2+=receipt.getSalesMed2();
    totalsalesmed3+=receipt.getSalesMed3();
    totalsales=totalsalesmed1+totalsalesmed2+totalsalesmed3;
}

string Customer::getName() const{
    return name;
}

int Customer::getMedicine1Amount() const{
    return medicine1amount;
}
```



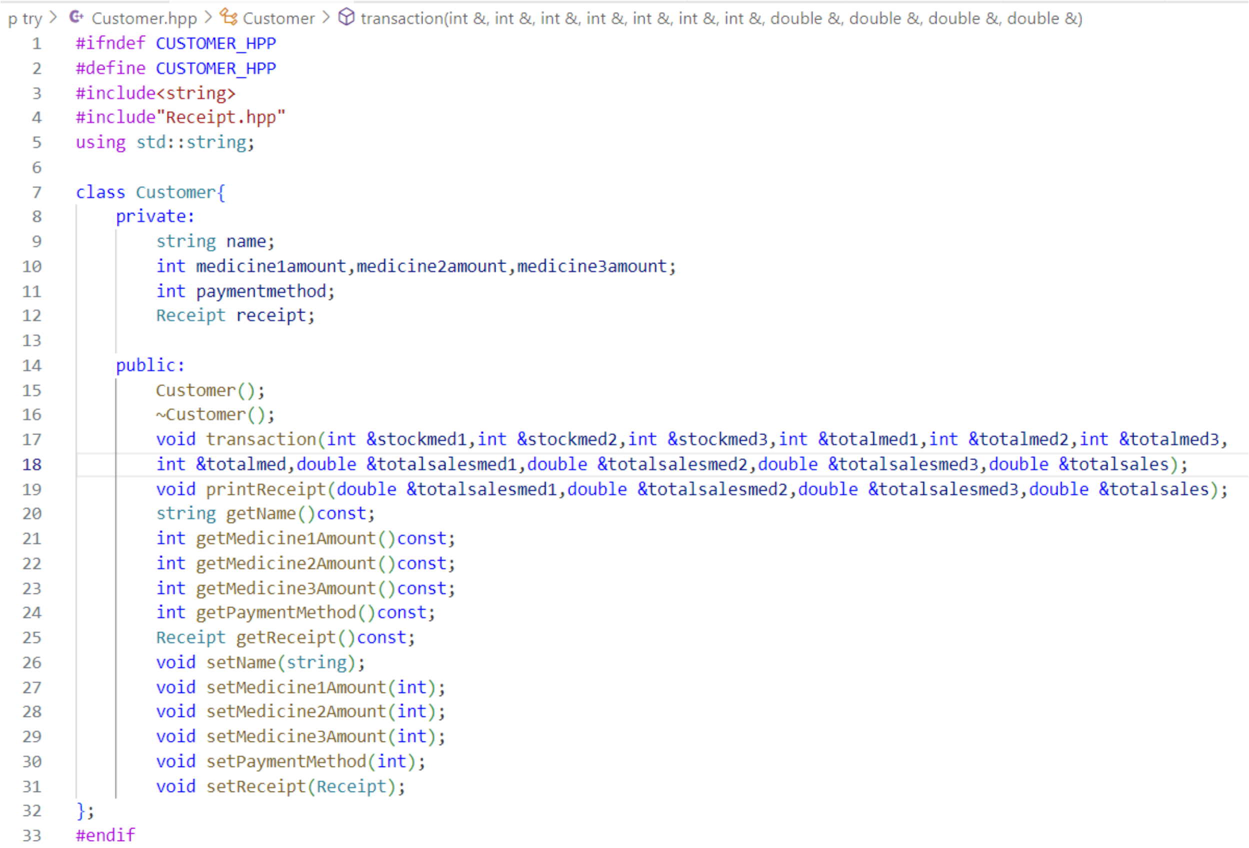
p try > Customer.cpp > transaction(int &, int &, int &, int &, int &, int &, double &, double &, double &, double &)

```cpp
int Customer::getMedicine2Amount() const{
    return medicine2amount;
}

int Customer::getMedicine3Amount() const{
    return medicine3amount;
}

int Customer::getPaymentMethod() const{
    return paymentmethod;
}

Receipt Customer::getReceipt() const{
    return receipt;
}

void Customer::setName(string n){
    name=n;
}

void Customer::setMedicine1Amount(int m1){
    medicine1amount=m1;
}

void Customer::setMedicine2Amount(int m2){
    medicine2amount=m2;
}

void Customer::setMedicine3Amount(int m3){
    medicine3amount=m3;
}

void Customer::setPaymentMethod(int method){
    paymentmethod=method;
}
```



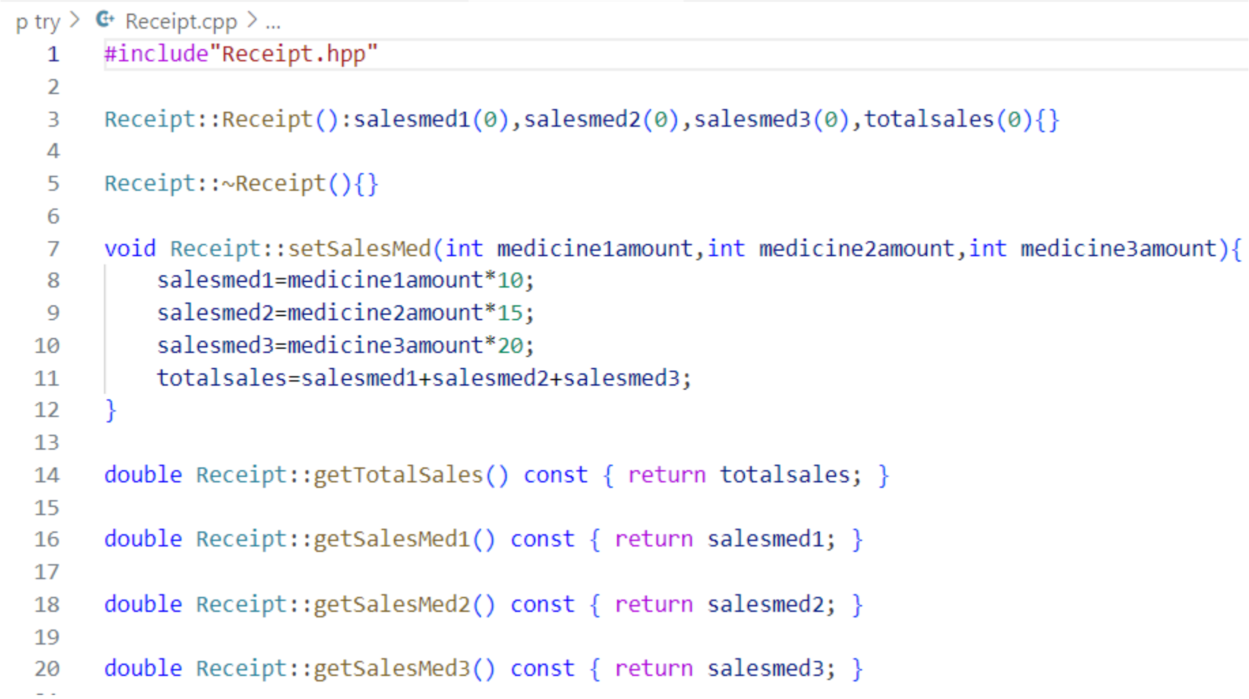
```cpp
void Customer::setReceipt(Receipt r){
    receipt=r;
}
```

Customer.hpp


p try > Customer.hpp > Customer > transaction(int &, int &, int &, int &, int &, int &, double &, double &, double &, double &)

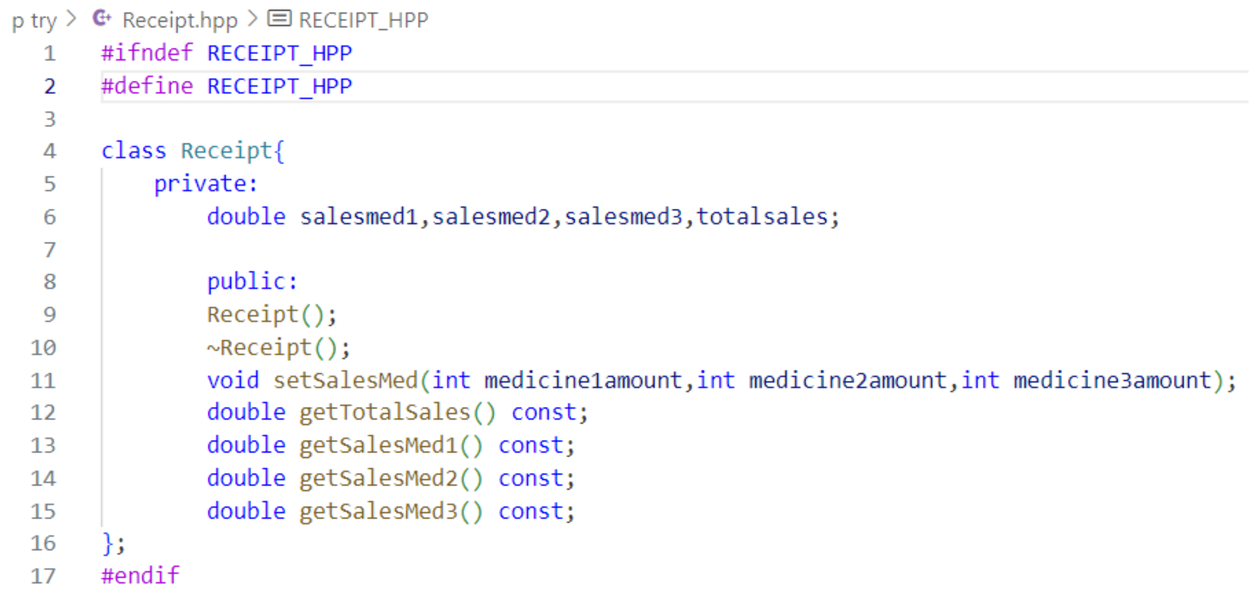
```
#ifndef CUSTOMER_HPP
#define CUSTOMER_HPP
#include<string>
#include"Receipt.hpp"
using std::string;

class Customer{
    private:
        string name;
        int medicine1amount,medicine2amount,medicine3amount;
        int paymentmethod;
        Receipt receipt;

    public:
        Customer();
        ~Customer();
        void transaction(int &stockmed1,int &stockmed2,int &stockmed3,int &totalmed1,int &totalmed2,int &totalmed3,
        int &totalmed,double &totalsalesmed1,double &totalsalesmed2,double &totalsalesmed3,double &totalsales);
        void printReceipt(double &totalsalesmed1,double &totalsalesmed2,double &totalsalesmed3,double &totalsales);
        string getName()const;
        int getMedicine1Amount()const;
        int getMedicine2Amount()const;
        int getMedicine3Amount()const;
        int getPaymentMethod()const;
        Receipt getReceipt()const;
        void setName(string);
        void setMedicine1Amount(int);
        void setMedicine2Amount(int);
        void setMedicine3Amount(int);
        void setPaymentMethod(int);
        void setReceipt(Receipt);
};
#endif
```

Receipt.cpp



```cpp
#include"Receipt.hpp"

Receipt::Receipt():salesmed1(0),salesmed2(0),salesmed3(0),totalsales(0){}

Receipt::~Receipt(){}

void Receipt::setSalesMed(int medicine1amount,int medicine2amount,int medicine3amount){
    salesmed1=medicine1amount*10;
    salesmed2=medicine2amount*15;
    salesmed3=medicine3amount*20;
    totalsales=salesmed1+salesmed2+salesmed3;
}

double Receipt::getTotalSales() const { return totalsales; }

double Receipt::getSalesMed1() const { return salesmed1; }

double Receipt::getSalesMed2() const { return salesmed2; }

double Receipt::getSalesMed3() const { return salesmed3; }
```

Receipt.hpp



```cpp
#ifndef RECEIPT_HPP
#define RECEIPT_HPP

class Receipt{
    private:
        double salesmed1,salesmed2,salesmed3,totalsales;

        public:
        Receipt();
        ~Receipt();
        void setSalesMed(int medicine1amount,int medicine2amount,int medicine3amount);
        double getTotalSales() const;
        double getSalesMed1() const;
        double getSalesMed2() const;
        double getSalesMed3() const;
};
#endif
```