Tab 1

**Assignment No:** 01

**Assignment Name:** Toll Management Program

**Source Code:**

#include <bits/stdc++.h> // 2203152

#include <conio.h>

#include <fstream>

using namespace std;

string getPassword () {

string pass;

char ch = getch();

while (ch != 13) {

pass.push\_back(ch);

cout << "\*";

ch = getch();

}

return pass;

}

string setDate () {

char Date[10];

time\_t currentTime = time(nullptr);

tm\* localTime = localtime(&currentTime);

int DD = localTime -> tm\_mday;

int MM = localTime -> tm\_mon + 1;

int YYYY = localTime -> tm\_year + 1900;

if (DD < 10) {

Date[0] = '0';

Date[1] = DD + '0';

} else {

string dd = to\_string(DD);

Date[0] = dd[0];

Date[1] = dd[1];

}

Date[2] = '/';

if (MM < 10) {

Date[3] = '0';

Date[4] = MM + '0';

} else {

string mm = to\_string(MM);

Date[3] = mm[0];

Date[4] = mm[1];

}

Date[5] = '/';

string yyyy = to\_string(YYYY);

Date[6] = yyyy[0];

Date[7] = yyyy[1];

Date[8] = yyyy[2];

Date[9] = yyyy[3];

string date = Date;

return date;

}

string setTime () {

char Time[6];

time\_t currentTime = time(nullptr);

tm\* localTime = localtime(&currentTime);

int min = localTime -> tm\_min;

int hour = localTime -> tm\_hour;

if (min < 10) {

Time[3] = '0';

Time[4] = min + '0';

} else {

string MIN = to\_string(min);

Time[3] = MIN[0];

Time[4] = MIN[1];

}

Time[2] = ':';

if (hour < 10) {

Time[0] = '0';

Time[1] = hour + '0';

} else {

string HOUR = to\_string(hour);

Time[0] = HOUR[0];

Time[1] = HOUR[1];

}

string time = Time;

return time;

}

bool isLeapYear (int year) {

if (year % 4 == 0) {

if (year % 100 == 0) {

if (year % 400 == 0) {

return true; // Divisible by 400 -> Leap year

} else {

return false; // Divisible by 100 but not 400 -> Not a leap year

}

} else {

return true; // Divisible by 4 but not 100 -> Leap year

}

} else {

return false; // Not divisible by 4 -> Not a leap year

}

}

bool dateCheck (string date) {

if (date[2] != '/' || date[5] != '/') return false;

int DD = (date[0] - '0') \* 10 + (date[1] - '0');

int MM = (date[3] - '0') \* 10 + (date[4] - '0');

int YYYY = (date[6] - '0') \* 1000 + (date[7] - '0') \* 100 + (date[8] - '0') \* 10 + (date[9] - '0');

if (MM > 12 || DD > 31) return false;

switch (MM) {

case 2:

if (isLeapYear(YYYY)) {

if (DD > 29) return false;

else return true;

} else {

if (DD > 28) return false;

else return true;

}

case 4:

case 6:

case 9:

case 11:

if (DD > 30) return false;

else return true;

default:

return true;

}

}

class Operator {

public:

string name, password;

}; vector <Operator> vo;

class Vehicle {

public:

string Reg\_No, Operator\_ID, Date, Time;

string vehicle;

static int tollBus, tollTruck, tollCar;

void virtual getInput() {};

}; vector <Vehicle> v;

int Vehicle::tollBus = 500;

int Vehicle::tollTruck = 400;

int Vehicle::tollCar = 150;

bool comparator (Vehicle &a, Vehicle &b) { // sorts by date then time

if (a.Date == b.Date) return a.Time < b.Time;

else return a.Date < b.Date;

}

void searchVehicle() {

cout << "Enter Vehicle Registration No: ";

string reg; cin >> reg; int flag = 0;

vector <Vehicle> ob;

for (auto &i : v) {

if (reg == i.Reg\_No) {

flag = 1;

ob.push\_back(i);

}

} if (flag == 0) {

cout << "Vehicle NOT Found" << endl;

} else {

sort(ob.begin(), ob.end(), comparator);

cout << "Date Time Amount Operator" << endl;

for (auto &i : ob) {

if (i.vehicle == "Bus") {

cout << i.Date << " " << i.Time << " " << i.tollBus << " " << i.Operator\_ID << endl;

} else if (i.vehicle == "Truck") {

cout << i.Date << " " << i.Time << " " << i.tollTruck << " " << i.Operator\_ID << endl;

} else {

cout << i.Date << " " << i.Time << " " << i.tollCar << " " << i.Operator\_ID << endl;

}

}

}

}

void searchDatetoDate() {

cout << "Date Format: DD/MM/YYYY" << endl;

cout << "Enter Start Date: ";

string date1; cin >> date1;

cout << "Enter End Date: ";

string date2; cin >> date2;

if (dateCheck(date1) && dateCheck(date2)) {

sort(v.begin(), v.end(), comparator);

cout << "Date Time Vehicle Amount Operator" << endl;

for (auto &i : v) {

if (i.Date >= date1 && i.Date <= date2) {

if (i.vehicle == "Bus") {

cout << i.Date << " " << i.Time << " " << i.vehicle << " " << i.tollBus << " " << i.Operator\_ID << endl;

} else if (i.vehicle == "Truck") {

cout << i.Date << " " << i.Time << " " << i.vehicle << " " << i.tollTruck << " " << i.Operator\_ID << endl;

} else {

cout << i.Date << " " << i.Time << " " << i.vehicle << " " << i.tollCar << " " << i.Operator\_ID << endl;

}

}

}

} else {

// system("cls");

cout << "Invalid Date" << endl;

}

}

void searchOperator() {

cout << "Enter Operator Name: ";

string op; cin >> op;

vector <Vehicle> ob; int flag = 0;

for (auto &i : v) {

if (op == i.Operator\_ID) {

flag = 1;

ob.push\_back(i);

}

} if (flag == 0) {

// system("cls");

cout << "No Vehicle was Registered under this Operator" << endl;

} else {

sort(ob.begin(), ob.end(), comparator);

cout << "Date Time Vehicle Amount Operator" << endl;

for (auto &i : ob) {

if (i.vehicle == "Bus") {

cout << i.Date << " " << i.Time << " " << i.vehicle << " " << i.tollBus << " " << i.Operator\_ID << endl;

} else if (i.vehicle == "Truck") {

cout << i.Date << " " << i.Time << " " << i.vehicle << " " << i.tollTruck << " " << i.Operator\_ID << endl;

} else {

cout << i.Date << " " << i.Time << " " << i.vehicle << " " << i.tollCar << " " << i.Operator\_ID << endl;

}

}

}

}

void tollStatistics() {

cout << "Date Format: DD/MM/YYYY" << endl;

cout << "Enter Date: ";

string date; cin >> date;

if (dateCheck(date)) {

int totalBus, totalTruck, totalCar;

totalBus = totalTruck = totalCar = 0;

int tollBus, tollTruck, tollCar;

for (auto &i : v) {

if (i.Date == date) {

if (i.vehicle == "Bus") {

totalBus++; tollBus = i.tollBus;

} else if (i.vehicle == "Truck") {

totalTruck++; tollTruck = i.tollTruck;

} else {

totalCar++; tollCar = i.tollCar;

}

}

}

cout << "Vehicle" << " " << "Number" << " " << "Amount" << endl;

cout << "Bus " << " " << totalBus << " " << totalBus\*tollBus << endl;

cout << "Truck " << " " << totalTruck << " " << totalTruck\*tollTruck << endl;

cout << "Car " << " " << totalCar << " " << totalCar\*tollCar << endl;

cout << "Total Amount: " << totalBus\*tollBus + totalTruck\*tollTruck + totalCar\*tollCar << endl;

} else {

// system("cls");

cout << "Invalid Date" << endl;

}

}

void tollSettings() {

cout << "Enter Admin username: ";

string name; cin >> name;

Operator ob; int flag = 0;

for (auto &i : vo) {

if (name == i.name) {

ob = i; flag = 1; break;

}

}

if (flag == 0) {

// system("cls");

cout << "Operator Doesn't Exist. Please Register." << endl;

} else {

cout << "Enter Admin password: ";

string pw; pw = getPassword();

cout << endl;

if (ob.password != pw) {

cout << "Password Incorrect" << endl;

} else {

cout << "Login is Successful" << endl;

int toll; labelB:

cout << "Enter Toll for Bus: ";

cin >> toll;

if (toll >= 0) {

Vehicle::tollBus = toll;

} else {

cout << "Invalid Toll" << endl;

goto labelB;

} labelT:

cout << "Enter Toll for Truck: ";

cin >> toll;

if (toll >= 0) {

Vehicle::tollTruck = toll;

} else {

cout << "Invalid Toll" << endl;

goto labelT;

} labelC:

cout << "Enter Toll for Car: ";

cin >> toll;

if (toll >= 0) {

Vehicle::tollCar = toll;

} else {

cout << "Invalid Toll" << endl;

goto labelC;

}

cout << "Toll Change is Successful" << endl;

}

}

}

class Bus : public Vehicle {

int Seats;

public:

void getInput() {

cout << "Enter Bus Registration No: ";

cin >> Reg\_No;

tollCalculate();

Date = setDate();

Time = setTime();

}

void tollCalculate() {

int current = 0;

cout << "Enter Amount (Tk. " << tollBus << "): ";

int amount; cin >> amount; current += amount;

while (current < tollBus) {

cout << "Give " << tollBus-current << " Tk. more: ";

cin >> amount; current += amount;

}

cout << "Return Amount: " << current-tollBus << endl;

cout << "Toll Collection is Successful" << endl;

}

};

class Truck : public Vehicle {

int Weight, Height;

public:

void getInput() {

cout << "Enter Truck Registration No: ";

cin >> Reg\_No;

tollCalculate();

Date = setDate();

Time = setTime();

}

void tollCalculate() {

int current = 0;

cout << "Enter Amount (Tk. " << tollTruck << "): ";

int amount; cin >> amount; current += amount;

while (current < tollTruck) {

cout << "Give " << tollTruck-current << " Tk. more: ";

cin >> amount; current += amount;

}

cout << "Return Amount: " << current-tollTruck << endl;

cout << "Toll Collection is Successful" << endl;

}

};

class Car : public Vehicle {

string Owner;

public:

void getInput() {

cout << "Enter Car Registration No: ";

cin >> Reg\_No;

tollCalculate();

Date = setDate();

Time = setTime();

}

void tollCalculate() {

int current = 0;

cout << "Enter Amount (Tk. " << tollCar << "): ";

int amount; cin >> amount; current += amount;

while (current < tollCar) {

cout << "Give " << tollCar-current << " Tk. more: ";

cin >> amount; current += amount;

}

cout << "Return Amount: " << current-tollCar << endl;

cout << "Toll Collection is Successful" << endl;

}

};

int main () {

fstream file;

file.open("2203152\_Vehicles.txt");

while (file) {

Vehicle ob; string space;

getline(file, ob.Reg\_No);

if (ob.Reg\_No == "") {

break;

}

getline(file, ob.Operator\_ID);

getline(file, ob.Date);

getline(file, ob.Time);

getline(file, ob.vehicle);

getline(file, space);

v.push\_back(ob);

} file.close();

fstream file2;

file2.open("2203152\_Operators.txt");

while (file2) {

Operator ob; string space;

getline(file2, ob.name);

if (ob.name == "") break;

getline(file2, ob.password);

getline(file2, space);

vo.push\_back(ob);

} file2.close();

fstream file3;

file3.open("2203152\_Tolls.txt");

if (file) {

string tb, tt, tc;

getline(file3, tb);

getline(file3, tt);

getline(file3, tc);

Vehicle::tollBus = stoi(tb);

Vehicle::tollTruck = stoi(tt);

Vehicle::tollCar = stoi(tc);

} file3.close();

while (true) {

cout << "\* Toll Plaza: Operator \*" << endl << endl;

cout << " 1. Login" << endl;

cout << " 2. Register" << endl;

cout << " 3. Exit" << endl;

cout << " Enter Your Option: ";

int option; cin >> option; int flagMain = 0;

switch (option) {

case 1: { // Login

cout << "Enter Username: ";

string name; cin >> name;

Operator ob; int flag = 0;

for (auto &i : vo) {

if (name == i.name) {

ob = i; flag = 1; break;

}

}

if (flag == 0) {

// system("cls");

cout << "Operator Doesn't Exist. Please Register." << endl;

} else {

cout << "Enter Password: ";

string pw; pw = getPassword();

cout << endl;

if (ob.password != pw) {

// system("cls");

cout << "Password Incorrect" << endl;

} else {

// system("cls");

cout << "Login is Successful" << endl;

while (true) {

cout << "\*\*\*\*\*\*\* Toll Plaza Menu \*\*\*\*\*\*\*" << endl << endl;

cout << " 1. Bus" << endl;

cout << " 2. Truck" << endl;

cout << " 3. Car" << endl;

cout << " 4. Search" << endl;

cout << " 5. Statistics" << endl;

cout << " 6. Toll Settings (Admin)" << endl;

cout << " 7. Save and Logout" << endl;

cout << " Enter Your Option (1-7): ";

int option; cin >> option; int flagLogin = 0;

switch (option) {

case 1: { // Bus

Bus b;

b.getInput();

v.push\_back(b);

auto i = v.end()-1;

i -> vehicle = "Bus";

i -> Operator\_ID = ob.name;

cout << endl;

cout << "Press any key to go to main menu...";

getch(); system("cls");

} break;

case 2: { // Truck

Truck t;

t.getInput();

v.push\_back(t);

auto i = v.end()-1;

i -> vehicle = "Truck";

i -> Operator\_ID = ob.name;

cout << endl;

cout << "Press any key to go to main menu...";

getch(); system("cls");

} break;

case 3: { // Car

Car c;

c.getInput();

v.push\_back(c);

auto i = v.end()-1;

i -> vehicle = "Car";

i -> Operator\_ID = ob.name;

cout << endl;

cout << "Press any key to go to main menu...";

getch(); system("cls");

} break;

case 4: { // Search

while (true) {

cout << "\*\*\* Toll Plaza: Search \*\*\*" << endl << endl;

cout << " 1. Vehicle" << endl;

cout << " 2. Date to Date" << endl;

cout << " 3. Operator" << endl;

cout << " 4. Back" << endl;

cout << " Enter Your Option (1-4): ";

int option; cin >> option; int flagSearch = 0;

switch (option) {

case 1: { // vehicle

searchVehicle();

cout << endl;

cout << "Press any key to go to main menu...";

getch(); system("cls"); break;

}

case 2: { // date to date

searchDatetoDate();

cout << endl;

cout << "Press any key to go to main menu...";

getch(); system("cls"); break;

}

case 3: { // operator

searchOperator();

cout << endl;

cout << "Press any key to go to main menu...";

getch(); system("cls"); break;

}

case 4: {

system("cls"); flagSearch = 1; break;

}

default: {

// system("cls");

cout << "Invalid Option" << endl;

}

} if (flagSearch) {

break;

}

}

} break;

case 5: { // Statistics

tollStatistics();

cout << endl;

cout << "Press any key to go to main menu...";

getch(); system("cls");

} break;

case 6: { // Toll Settings (Admin)

tollSettings();

cout << endl;

cout << "Press any key to go to main menu...";

getch(); system("cls");

} break;

case 7: { // Save and Logout

flagLogin = 1;

// This option does not save actually, just logs out

// Actual saving is done when exiting the program

} break;

default: {

// system("cls");

cout << "Invalid Option" << endl;

}

}

if (flagLogin) {

// system("cls");

cout << "Saved Successfully" << endl;

break;

}

}

}

}

break;

}

case 2: { // Register

cout << "Username: ";

string name; cin >> name;

int flag = 0;

for (auto &i : vo) {

if (name == i.name) {

// system("cls");

cout << "Operator Already Exists. Please Login." << endl;

flag = 1; break;

}

}

if (flag) break;

else { label1:

cout << "Password: ";

string pw; pw = getPassword();

cout << endl;

if (pw.empty()) {

cout << "Password cannot be Empty" << endl;

goto label1;

}

cout << "Reconfirm Password: ";

string rpw; rpw = getPassword();

cout << endl;

if (pw != rpw) {

// system("cls");

cout << "Passwords didn't match" << endl;

goto label1;

} else {

cout << "Registration is Successful" << endl;

Operator ob; ob.name = name; ob.password = pw;

vo.push\_back(ob);

cout << "Press any key to go to Main Menu...";

getch(); system("cls");

}

}

} break;

case 3: { // Exit

flagMain = 1;

fstream veh;

veh.open("2203152\_Vehicles.txt", ios::trunc | ios::out | ios::in);

for (auto i = v.begin(); i < v.end(); i++) {

veh << i -> Reg\_No << endl;

veh << i -> Operator\_ID << endl;

veh << i -> Date << endl;

veh << i -> Time << endl;

veh << i -> vehicle << endl;

if (i != v.end()-1) veh << endl;

}

veh.close();

fstream toll;

toll.open("2203152\_Tolls.txt", ios::trunc | ios::out | ios::in);

toll << Vehicle::tollBus << endl;

toll << Vehicle::tollTruck << endl;

toll << Vehicle::tollCar << endl;

toll.close();

fstream oper;

oper.open("2203152\_Operators.txt", ios::trunc | ios::out | ios::in);

for (auto i = vo.begin(); i < vo.end(); i++) {

oper << i -> name << endl;

oper << i -> password << endl;

if (i != vo.end()-1) oper << endl;

}

oper.close();

break;

}

default: {

// system("cls");

cout << "Invalid Option" << endl;

}

} if (flagMain) {

// system("cls");

cout << "Exit Successful" << endl;

break;

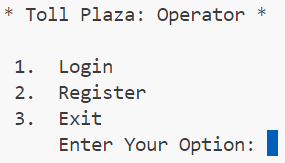
}

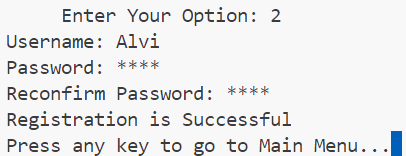
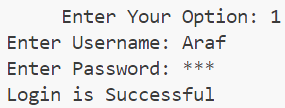
}

}

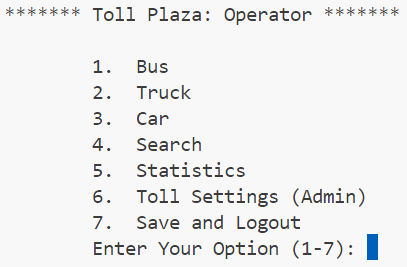
**Input-Output:**

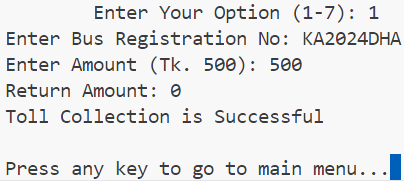
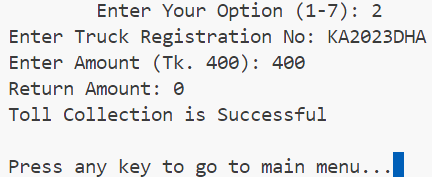
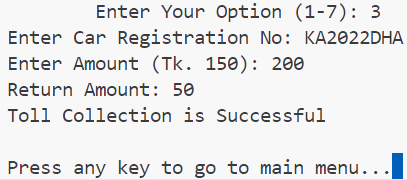
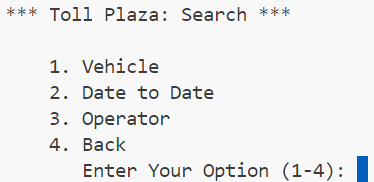
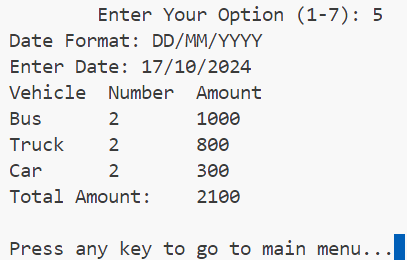
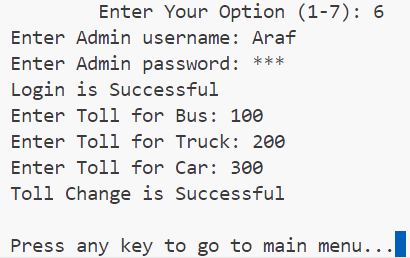
**“**Toll Plaza: Operator” Menu (At Start of Program):



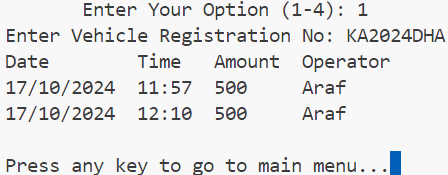
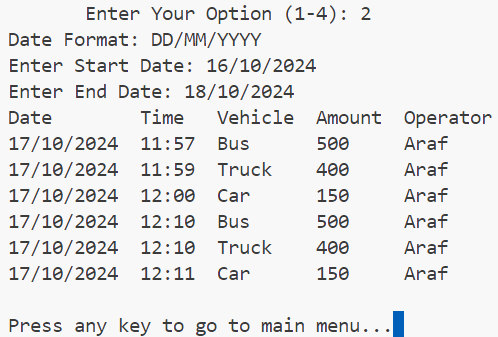
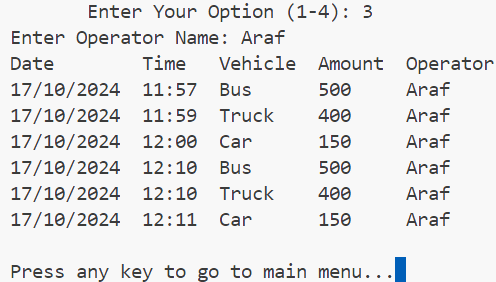
1. Register:  
   
2. Login:  
   
3. Exit:  
   

“Toll Plaza Menu” Menu (After Login):



1. Bus:  
   
2. Truck:  
   
3. Car:  
   
4. Search:  
   
5. Statistics:  
   
6. Toll Settings (Admin):  
   
7. Save and Logout:  
   

“Toll Plaza: Search” Menu (After 4. Search):

1. Vehicle:  
   
2. Date to Date:  
   
3. Operator:  
   
4. Back: Takes to the previous menu (Toll Plaza Menu).