

DATA STRUCTURES AND

ALGORITHMS

FINAL PROJECT

SUBMITTED TO: MADAM SOBIA

SUBMITTED BY: SANA AKBAR 026

UBAIDA WAHEED 034

TAYYBA SALAMAT 032

GROUP:  BSE-III A

CAR RENTAL SYSTEM

CODE:

// project.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

#include<string>

using namespace std;

class CarRent

{

private:

struct node

{

string make;

string model;

string color;

int pricePerDay;

int carNo;

bool available;

struct node\* next; // Pointer to next node in DLL

struct node\* prev; // Pointer to previous node in DLL

}\*head;

public:

CarRent()

{

head = NULL;

}

bool empty() {

if (head == NULL)

{

return true;

}

else

return false;

}

void traverse()

{

if (empty())

{

cout << "\n\t\tError! No Records Yet" << endl;

}

else

{

node\* ptr;

ptr = head;

cout << "\n\t\t\t CAR'S DETAIL \n";

cout << " --------------------------------------------------------------\n";

cout << "Car No\tMakeType\tModel\t\tPricePerDay\tColor\tSTATUS" << endl << endl;

while (ptr != NULL)

{

cout << ptr->carNo << "\t" << ptr->make << "\t\t" << ptr->model << "\t\t" << ptr->pricePerDay << "\t\t" << ptr->color << "\t";

if (ptr->available == false)

{

cout << "Available";

}

else

{

cout << "Booked";

}

cout << endl;

ptr = ptr->next;

}

}

cout << endl;

}

void insert\_car\_detail(int n, string mk, string md, string c, int p)

{int flag=0;

node \*temp,\*search;

if(head==NULL)

{

temp=new node;

temp->available=false;

temp->color=c;

temp->carNo=n;

temp->make=mk;

temp->model=md;

temp->pricePerDay=p;

temp->next=head;

temp->prev=NULL;

head=temp;

flag=1;

}

//Insertion At The Start

else if(head->pricePerDay>=p)

{

temp=new node;

temp->next=head;

temp->available=false;

temp->color=c;

temp->carNo=n;

temp->make=mk;

temp->model=md;

temp->pricePerDay=p;

head=temp;

temp->prev=head;

flag=1;

}

//Insertion At The End

else

{

temp=new node;

search=head;

while(search->next!=NULL)

{

search=search->next;

}

if(search->pricePerDay<p)

{

search->next=temp;

temp->prev=search;

temp->next=NULL;

temp->available=false;

temp->color=c;

temp->carNo=n;

temp->make=mk;

temp->model=md;

temp->pricePerDay=p;

flag=1;

}

}

//Insertion In the Middle

if(flag==0)

{

temp=head;

node\* s1=head;

node\* s2=head->next;

while(temp!=NULL)

{

if((s1->pricePerDay<p)&&(s2->pricePerDay>p))

{

temp=new node;

temp->next=s2;

s2->prev=temp;

s1->next=temp;

temp->prev=s1;

temp->available=false;

temp->color=c;

temp->carNo=n;

temp->make=mk;

temp->model=md;

temp->pricePerDay=p;

flag=1;

break;

}

s1=s1->next;

s2=s2->next;

}

}

}

void delete\_car\_detail(int n)

{

int flag = 0;

node\* temp;

temp = head;

if (temp->available == true)

{

while (temp!= NULL)

{

temp = temp->next;

cout << "Can't Delete record Yet Bcz this car is booked" << endl;

break;

}

}

else

{

if (head == NULL)

{

cout << "underflow" << endl;

flag = 1;

}

//deletion At The Start

else if (head->carNo==n)

{

node\* p;

p = head;

head = p->next;

cout << "Deleted " << p->make << "'s Record Successfully" << endl;

delete p;

flag = 1;

}

//deletion At The End

else

{

node\* p1 = head;

node\* p2 = head->next;

while (p2->next != NULL)

{

p1 = p2;

p2 = p2->next;

}

if (p2->carNo==n)

{

p1->next = NULL;

p2->prev = NULL;

cout << "Deleted " << p2->make << "'s Record Successfully" << endl;

delete p2;

flag = 1;

}

}

//deletion In the Middle

if (flag == 0)

{

temp = head;

node\* s1 = head;

node\* s2 = head->next;

while (temp != NULL)

{

if (s2->carNo==n)

{

s1->next = s2->next;

s2->next->prev = s1;

cout << "Deleted " << s2->make << "'s Record Successfully" << endl;

delete s2;

flag = 1;

break;

}

s1 = s1->next;

s2 = s2->next;

}

}

if (flag == 0)

{

cout << "DOES NOT EXIST" << endl;

}

}

}

void rent(int n)

{ int flag = 0;

node\* ptr;

ptr = head;

if (empty())

{

return;

}

if (!empty())

{

while (ptr != NULL)

{

if (ptr->available == false && ptr->carNo==n)

{

string name, address, phone, cnic;

int day;

ptr->available = true;

cout << "Enter Your Name : ";

cin >> name;

cout << "Enter Your Address : ";

cin >> address;

cout << "Enter Your Phone Number : ";

cin >> phone;

while (phone.length() != 11)

{

cout << "Enter Valid Phone Number : ";

cin >> phone;

}

cout << "Enter Your CNIC : ";

cin >> cnic;

while (cnic.length() != 15)

{

cout << "Enter Valid CNIC Number '15' digits including 'spaces'" << endl;

cin >> cnic;

}

cout << "Enter Days : ";

cin >> day;

cout << "--------------Your car is Booked Successfully--------------" << endl << endl;

cout << "\n\n\n\n\n\t\t-----------------------------------------------\n";

cout << "\t\t| CUSTOMER INVOICE |\n";

cout << "\t\t-----------------------------------------------\n";

cout << "\t\t| Name : " << name << endl;

cout << "\t\t| Address : " << address << endl;

cout << "\t\t| Phone # : " << phone << endl;

cout << "\t\t| CNIC # : " << cnic << endl;

cout << "\t\t| Car : " << ptr->make << endl;

cout << "\t\t| Moodel : " << ptr->model << endl;

cout << "\t\t| Days of Rent : " << day << endl;

cout << "\t\t| Price per day : " << ptr->pricePerDay << endl;

cout << "\t\t| Total Bill : " << (ptr->pricePerDay \* day) << endl;

flag = 1;

}

else if (ptr->available == true && ptr->carNo==n)

{

cout << "This car is already Booked,Try another one" << endl << endl;

ptr = ptr->next;

flag = 1;

}

ptr = ptr->next;

}

}

if (flag==0)

{

cout << "This Car is Not Present in our list." << endl << "Enter Valid Car Details" << endl << endl;

}

}

void unbook(int n) {

int flag = 0;

node\* ptr;

ptr = head;

if (empty())

{

return;

}

if (!empty())

{

while (ptr != NULL)

{

if (ptr->available == true && ptr->carNo==n)

{

ptr->available = false;

cout << "--------------Your car is Unbooked Successfully--------------" << endl << endl;

cout << "\n\n\n\n\n\t\t-----------------------------------------------\n";

flag = 1;

}

else if (ptr->available == false && ptr->carNo==n)

{

cout << "This car is already Unbooked,Try another one" << endl << endl;

flag = 1;

}

ptr = ptr->next;

}

}

if (flag == 0)

{

cout << "This Car is Not Present in our list." << endl << "Enter Valid Car Details" << endl << endl;

}

}

void update\_car\_detail(int n) {

node\* ptr;

ptr = head;

if (empty())

{

return;

}

if (!empty())

{

while (ptr != NULL)

{

if (ptr->carNo==n)

{

cout << "Enter PricePerDay for Car : ";

cin >> ptr->pricePerDay;

cout << "--------------Price Updated Successfully--------------" << endl << endl;

break;

}

ptr = ptr->next;

}

}

}

};

int \_tmain(int argc, \_TCHAR\* argv[])

{

CarRent c;

c.insert\_car\_detail(1987, "Hundai", "i20", "Black", 5000000);

c.insert\_car\_detail(7266, "Ford", "Fusion", "White", 5500);

c.insert\_car\_detail(1990, "Toyota", "Corrola", "Silver", 7500);

c.insert\_car\_detail(2000, "Audi", "Q5 e", "Grey", 35000);

c.insert\_car\_detail(2010, "Tesla", "Model S", "Blue", 85000);

int op;

do

{

cout << "\n\n\n\t\t-----------------------------------------------\n";

cout << "\t\t| WELCOME TO 'UST' INVISION CAR RENTAL SYSTEM|\n";

cout << "\t\t-----------------------------------------------\n";

cout << "\t\t| 1.See Car Details |\n";

cout << "\t\t| 2.Rent a Car |\n";

cout << "\t\t| 3.Return a Car |\n";

cout << "\t\t| 4.Update Car Details |\n";

cout << "\t\t| 5.Exit |\n";

cout << "\t\t-----------------------------------------------\n";

cout << "\t\tPlease Enter Your choice in Digits \n\t\t";

cin >> op;

string ma, mo, co;

int N;

int pr;

if (op == 1)

{

system("CLS");

c.traverse();

}

if (op == 2)

{ system("CLS");

c.traverse();

cout << "Enter Car Number of car to Book : ";

cin >> N;

c.rent(N);

}

if (op == 3)

{ system("CLS");

c.traverse();

cout << "Enter Car Number of car to Unbook : ";

cin >> N;

c.unbook(N);

}

if (op == 4)

{

system("CLS");

string pass;

cout << "\n\t\tAre you MANAGER" << endl << "\t\tEnter Pasword to prove your ID : ";

cin >> pass;

if (pass == "UST")

{

cout << "-----------------------------------------------\n";

cout << "Press '1' to Add car Details \nPress '2' to Delete Car Details\nPress '3' to Update Car Price Per Day \n";

cout << "-----------------------------------------------\n";

int ch;

cin >> ch;

if (ch == 1)

{ cout << "Enter Car Number : ";

cin >> N;

cout << "Enter Make Type : ";

cin >> ma;

cout << "Enter Model : ";

cin >> mo;

cout << "Enter Color of Car : ";

cin >> co;

cout << "Enter PricePerDay for Car : ";

cin >> pr;

c.insert\_car\_detail(N, ma, mo, co, pr);

}

if (ch == 2)

{

c.traverse();

cout << "Enter Car Number To Delete Car's Detail : ";

cin >> N;

c.delete\_car\_detail(N);

}

if (ch == 3)

{

c.traverse();

cout << "Enter Car Number To Update Car's Price Per Day : ";

cin >> N;

c.update\_car\_detail(N);

}

c.traverse();

if (ch != 1 && ch != 2 && ch != 3)

{

cout << "Invalid Choice! Please press '1','2' OR '3'" << endl;

}

}

else

{

cout << "YOU HAVE ENTERED INVALID PASSWORD" << endl;

}

}

if (op != 1 && op != 2 && op != 3 && op != 4 && op != 5)

{

system("CLS");

cout << "Invalid Choice !,Try Again " << endl;

}

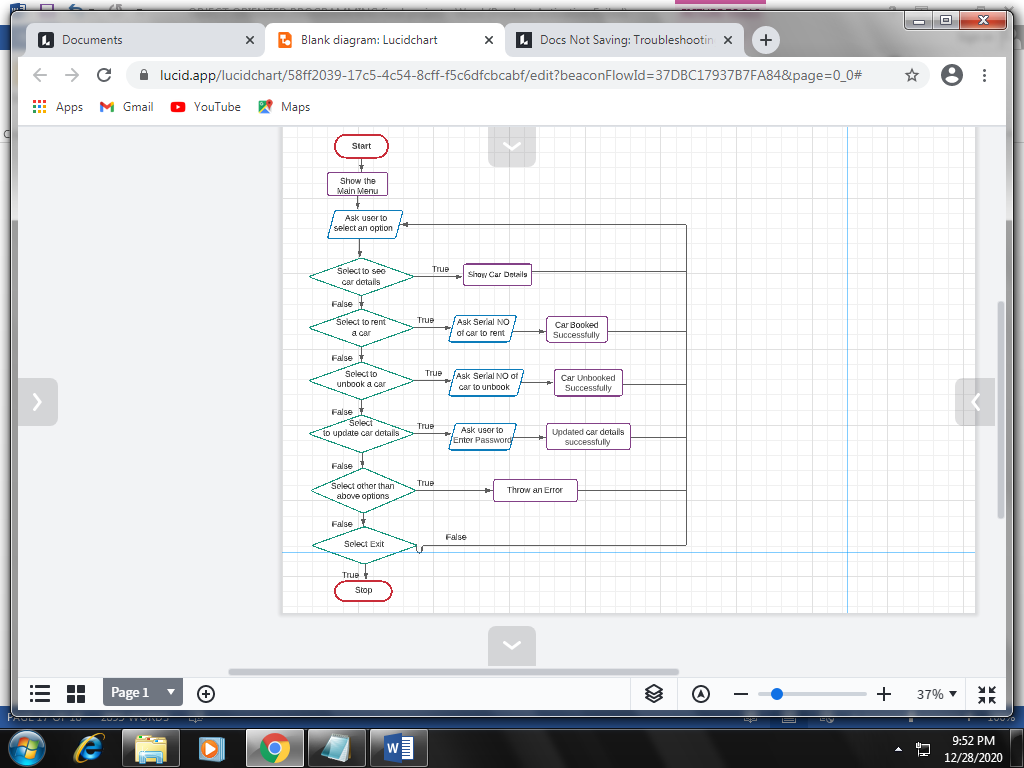
} while (op != 5);

system("pause");

return 0;

}

FLOW CHART:



REPORT:

Car rental system is the system in which the user can book the car according to his/her choice by entering some of their **personal details**.

This system provides the **services to the user as well the managers** who manages the system and keep the record of the **cars that is booked as well as the payment details of the user**.

In this system we had used **our major concepts of data structures like Insertion, Deletion, Traversal and Sorting** which we had implemented through **double linked list**.

In this system we had used **many of the functions** through which the system runs smoothly and is better and easy to understand.

The **functions** used in this project/system are:

* empty()
* traverse()
* insert\_car\_details()
* delete\_car\_details()
* rent()
* unbook()
* update\_car\_details()

In main we had made the **search-based menu** through which the **user and the manager** can chose whatever they want according to their choice.

The **car details can be added, deleted or price of rent can be updated** only by the **manager** after he enters the **authentic password**.

On **run time** all the cars shown **available** and as soon as the user rent a car the status will be presented as **booked** and the other user cannot book the same car and everything gets updated with it.