**Q6 Call Taxi Booking**

A call taxi operator has a fleet of "n" cars. For simplicity take the count as 5 but the program should work for any number of Taxis. (Let their names be identified as Taxi-1, Taxi-2 .... Taxi-n)

● There are only 5 points in the city for pick-up / drop. Let the points be named A, B, C, D&E

● All the points are in a straight line & the distance between the points are same say 15kms

● Time taken for travel will be 15 minutes between each point.

● Charges for each travel will be calculated as Rs 50 for the first 5 kms and then Rs 10 for subsequent kms of travel.

● During Booking, the following information is given : The Starting point, Destination point & Start time. After dropping customer, the taxi waits there for next customer allotment. Each customer is identified uniquely by a Cust-ID

● Write a program that does the following : During Booking, find out the taxi which will be free at the specified time & which will be nearest to his location and allot it. Assume all Taxis are at Point A initially.

Wait for next set of Inputs once a taxi is allotted

● **Allotment Criteria :**

1. Taxis available on the same location given preference first

2. If more than one taxi present on the same location, preference given to the taxi which 3. earned least during the day

**Questions :**

1. Write a function to handle booking.

2. Write a function that can display travel history of any given call taxi. Travel history should display details like Customer ID, From , To , PickupTime, DropTime, Amount Charged etc.

**Question 1 - Sample data Input 1**

Customer ID: 1

Starting Point : A Destionation Point : D

Time : 9.00 AM

**Output**

Booking ID : 1

Allotted Taxi : Taxi1

------------------------------------------------------------------------------------------------------------------- **Input 2** Customer ID: 2

Starting Point : D

Destination Point : E

Time : 10.00 AM

**Output**

Booking ID : 2

Allotted Taxi : Taxi1

-------------------------------------------------------------------------------------------------------------------**Input3** Customer ID: 3

StartingPoint : B

DestionationPoint : A

Time : 10.00AM

**Output**

Booking ID : 3

AllottedTaxi : Taxi3

------------------------------------------------------------------------------------------------------------------- **Question2-SampleOutput:**

Travel HistoryofTaxi 1:

1 A D 9.00 9.45 450 2 DE9.4510.00150

**…………………SOURCE CODE………………….**

#include<stdio.h>

#include <string.h>

#include <math.h>

#include <stdlib.h>

#define N 5 //Number of taxis

#define M 10//Number of customers

struct CustomerDetails//Details of the Customer

{

int customer\_ID;

char start\_point;

char end\_point;

int hr,min,time\_min;

}cd;

struct Taxi//Taxi Details

{

int c;//number of customers travlled in the taxi

char current\_position;

int current\_time;

int customer\_id[M];

char from[M];

char to[M];

int amount[M];

int pickup\_time[M][2];

int drop\_time[M][2];

}s[N];

int Total\_price(int x)//First preference taxi's allocation based on the minimum amount

{

int tot\_price\_earned=0;

for(int i=0;i<s[x].c;i++)

{

tot\_price\_earned+=s[x].amount[i];

}

return tot\_price\_earned;

}

int calculate\_price()//To Find the available taxi nearest to the customer

{

int min\_distance=6,distance=0;//Minimum distance from A to E

int total\_amount1=0,total\_amount2=0;

int current=0;

for(int i=0;i<N;i++)

{

distance=abs(s[i].current\_position-cd.start\_point);

if(distance<min\_distance&&(s[i].current\_time<=cd.time\_min))

{

min\_distance=distance;

current=i;

}

else if(min\_distance==distance&&(s[i].current\_time<=cd.time\_min))

//if two taxis are available in the same destination,then it must calculate the mininmum earned taxi and make it available for the customer

{

total\_amount1=Total\_price(current);

total\_amount2=Total\_price(i);

if(total\_amount2<total\_amount1)

current=i;

}

}

return current;

}

int customercall()

{

printf("Enter customer id\n");

printf("Enter Starting point\n");

printf("Enter Ending point\n");

scanf("%d",&cd.customer\_ID);

scanf("%s",&cd.start\_point);

scanf("%s",&cd.end\_point);

printf("Enter time hr:min format ");

scanf("%d:%d",&cd.hr,&cd.min);

cd.time\_min=cd.min+(cd.hr\*60);//Total minutes is calculated

int x=calculate\_price();

return x;

}

void taxi\_details\_entry(int n)

{

int x=s[n].c;//n denotes the current taxi number

s[n].c++;

s[n].current\_position=cd.end\_point;//object of the customer details

s[n].pickup\_time[x][0]=cd.hr;

s[n].pickup\_time[x][1]=cd.min;

int q=(abs(cd.start\_point-cd.end\_point)\*15)+cd.min;

if(q>=60)

{

s[n].drop\_time[x][0]=cd.hr+1;

s[n].drop\_time[x][1]=q-60;

}

else

{

s[n].drop\_time[x][0]=cd.hr;

s[n].drop\_time[x][1]=q;

}

int tot\_min=(s[n].drop\_time[x][0]\*60)+s[n].drop\_time[x][1];

s[n].current\_time=tot\_min;

s[n].customer\_id[x]=cd.customer\_ID;

s[n].amount[x]=((abs(cd.start\_point-cd.end\_point)\*15)\*10);

s[n].from[x]=cd.start\_point;

s[n].to[x]=cd.end\_point;

}

void taxi\_details()

{

int i,sum=0,num=0;

for(int j=0;j<N;j++)

{

if(s[j].c>0)

{

num=j;

for(i=0;i<s[num].c;i++)

{

sum=sum+s[num].amount[i];

}

printf("Taxi\_%d details:\n",num+1);

for(i=0;i<s[num].c;i++)

{

printf(" customer %d:\n",i+1);

printf(" 1:customer\_ID %d\n",s[num].customer\_id[i]);

printf(" 2:from position %c to %c\n",s[num].from[i],s[num].to[i]);

printf(" 3:picked up at %d:%d, dropped at %d:%d\n",s[num].pickup\_time[i][0],s[num].pickup\_time[i][1],s[num].drop\_time[i][0],s[num].drop\_time[i][1]);

printf(" 4:Amount earned Rs.%.2d\n",s[num].amount[i]);

}

printf("Total price earned by taxi\_%d is Rs.%.2d",num+1,sum);//total earnings for his ride for a day

printf("\n");

sum=0;

}

}

}

int main() {

int n,a;

for(int i=0;i<N;i++)

{

s[i].c=0;

s[i].current\_position='A';

s[i].current\_time=0;

}

while(1)

{

printf("Enter\n1.To Book a call taxi\n2.To display taxi details\n3.To exit\n");

scanf("%d",&n);

switch(n)

{

case 1:

a=customercall();

taxi\_details\_entry(a);

break;

case 2:

taxi\_details();

break;

case 3:

exit(0);

default:

printf("invalid input\n");

break;

}

}

return 0;

}