Customized Bus Ticket Reservation System

PROJECT FOR:

CSE 2003 DATA STRUCTURES AND ALGORITHM

MADE BY:

- **Aaditya Pareek 19BCE0866**
- **Tanmay Bansal 19BCE0421**
- **Devjyoti Karan 19BCE0690**

- Under the guidance of: **Prof. Delhi Babu**

Abstract:

- Traveling is a large growing business across all countries. Bus reservation system deals with maintenance of records of details of each passenger who had reserved a seat for a journey. It also includes maintenance of information like schedule and details of each bus.
- We believe that we can use computer science concepts to help in the operations of bus ticket reservation by automating and simplifying tasks which are done physically and manually and might cause some error or take more time.
- By using this project, we can book tickets from any part of the world. User can check availability of bus and select seats. The project is useful as it makes the process of ticket booking effective.

Introduction:

We have used computer science concepts like

- OOP Object Oriented Programming
- Shortest Path Finding Algorithms
- Functions and Modules
- Hashing

to build a software for bus ticket reservation in Python 3.

We have used OOP to create classes of type Bus which has several features like bus number, arrival time etc.

We use Shortest Path Finding Algorithms to help user find the shortest path using buses from one location to other.

We use various functions such as bus_avail, updates etc. to perform various task which help in the process of ticket reservation. We also use various python modules like math,os etc. to make our software better and get more functions.

We have used hashing technique to make login system better.

Literature Review:

- We have studied the operations involved in Bus reservation process like selecting location, selecting bus etc. we have made many of these processes automated by use of programming.
- Sometimes these operations if done manually or physically can be performed wrongly or can take time so our program can help in such cases.
- We studied concepts like OOP, Dijkstra's algorithm, Python modules and hashing from various sources to implement them in our program to make our program better.

Problem:

- Most of Bus ticket reservation is done through calls or through using paper ticket system. We think that these systems have a chance of making mistakes and maybe take more time.
- These physical systems of ticket booking can lead to lot of problems as if it takes time some people might miss the bus or theres might be a lot of crowd near the booking system.
- So our solution is a Python program which helps a user/customer to book a ticket. Also there is option to login for the system controller who can edit bus timings, numbers etc.


C 355370
BEST
4 as.
L 842899
BEST
3
as.
S 475259
BEST
1 an.
Ha 810940
BEST
2 as.
BUS


LATE

Solution:

- So our solution is a Python program which helps a user/customer to book a ticket.
- There is an option to login for the system controller who can edit bus timings, numbers etc.
- These systems can function better an faster and are more convenient.
- We have used OOPs concept to make Buses with same properties but different values of those properties.
- We have used Dijkstra's algorithm to find shortest path.
- We have used hashing techniques to make the login more secure.

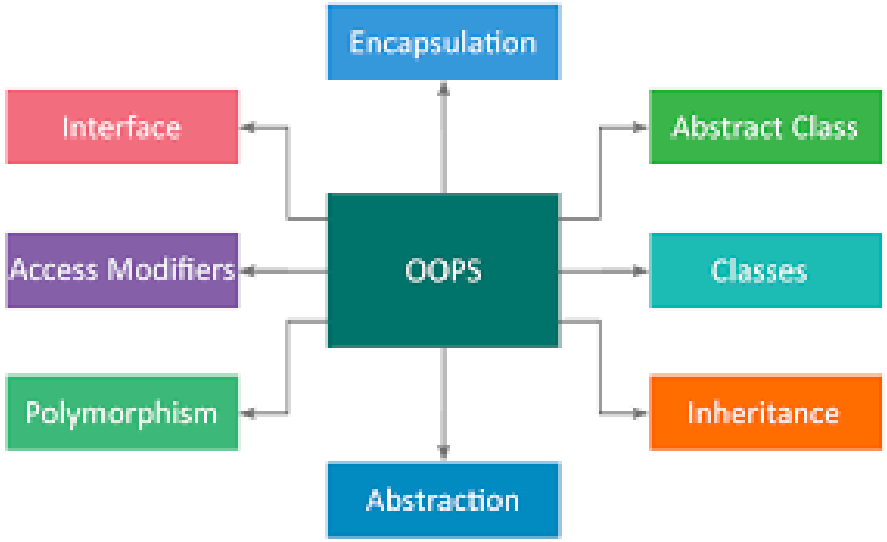
OOP: Object Oriented Programming

Python is a multi-paradigm programming language which means that it supports different programming approach. One approach is to solve a programming problem is by creating objects. This is known as Object-Oriented Programming(OOP).

An object has two characteristics:

- attributes
- behavior

A class is a blueprint for the object. An object (instance) is an instantiation of a class.



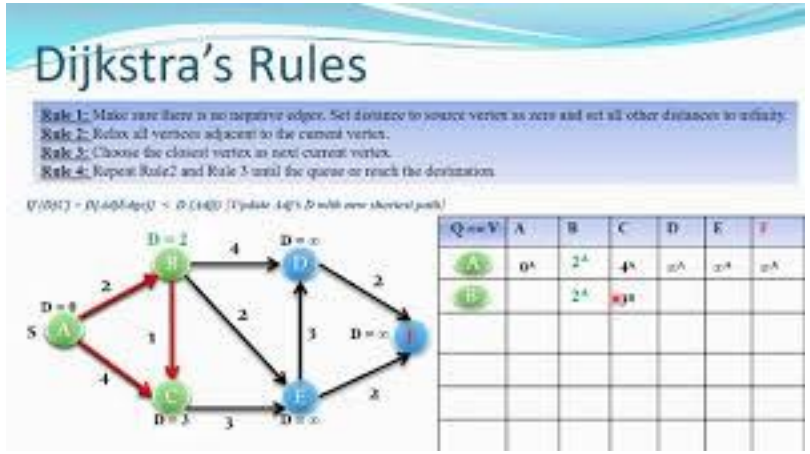
We have used OOPS concepts to make class Bus and Graph.

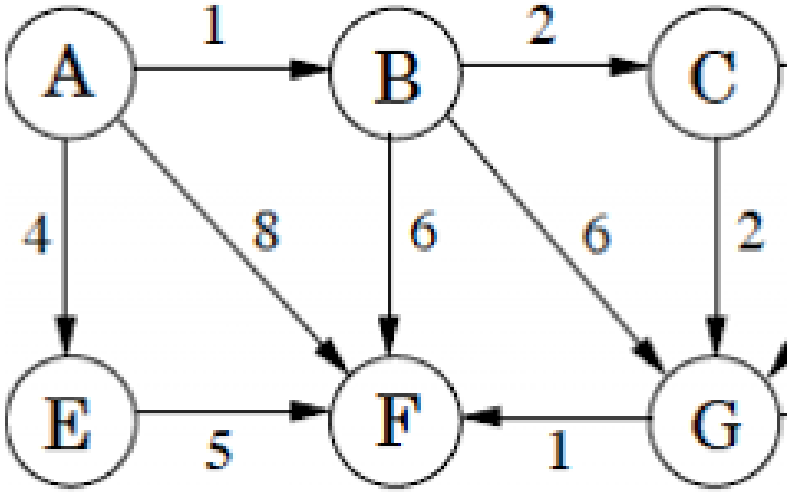
```python
class Graph():

    def _init_(self, vertices):
        self.V = vertices
        self.graph = [[0 for column in range(vertices)] for row in range(vertices)]

    def printSolution(self, dist, des):
        print("The minimum distance between the souce and and destination entered is ", dist[des], " kilometers")

    def minDistance(self, dist, sptSet):
        min = 999999
        for v in range(self.V):
            if dist[v] < min and sptSet[v] == False:
                min = dist[v]
                min_index = v
        return min_index
```

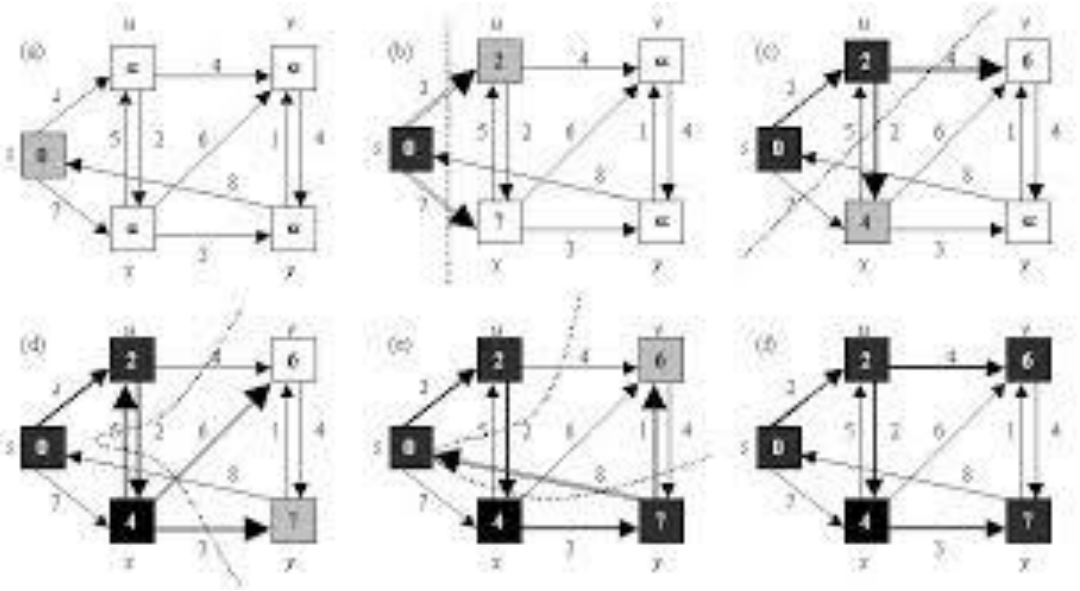
```python
class Bus:

    def _init_(self, bus_no, source, destination, pickup_time, dropoff_time, driv_nam, seats_left, tic_price, seats):
        self.bus_no = bus_no
        self.source = source
        self.destination = destination
        self.pickup_time = pickup_time
        self.dropoff_time = dropoff_time
        self.driv_name = driv_nam
        self.seats_left = seats_left
        self.tic_price = tic_price
        self.seats = seats
```

Shortest Path Finding Algorithms :

- There are a lot of different algorithms used for path finding or shortest path. We use these algorithm to help the user go from one place to other in shortest path by bus.
- We use Dijkstra's algorithm to do so in our program. Dijkstra's algorithm (or Dijkstra's Shortest Path First algorithm, SPF algorithm) is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. It was conceived by computer scientist Edsger W. Dijkstra in 1956 and published three years later.







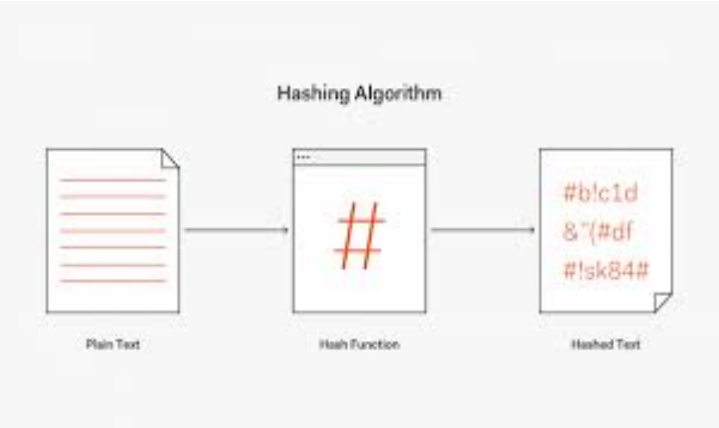
We consider nodes on graph as places where there are bus stops in cities
We help user find the shortest path for the journey

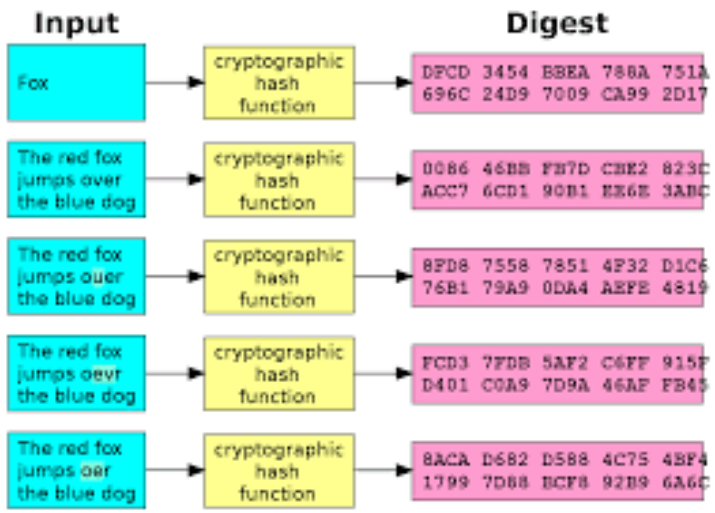
```python
class Graph():

    def _init_(self, vertices):
        self.V = vertices
        self.graph = [[0 for column in range(vertices)] for row in range(vertices)]

    def printSolution(self, dist, des):
        print("The minimum distance between the souce and and destination entered is ", dist[des], " kilometers")

    def minDistance(self, dist, sptSet):
        min = 999999
        for v in range(self.V):
            if dist[v] < min and sptSet[v] == False:
                min = dist[v]
                min_index = v
        return min_index

    def dijkstra(self, src, des):
        dist = [99999] * self.V
        dist[src] = 0
        sptSet = [False] * self.V
        for cout in range(self.V):
            u = self.minDistance(dist, sptSet)
            sptSet[u] = True
            for v in range(self.V):
                if self.graph[u][v] > 0 and sptSet[v] == False and dist[v] > dist[u] + self.graph[u][v]:
                    dist[v] = dist[u] + self.graph[u][v]
        self.printSolution(dist, des)
```

```
def info():
    print("1.Mumbai")
    print("2.Jaipur")
    print("3.Bangalore")
    print("4.New Delhi")
    print("5.Kolkata")
    print("6.Chennai")
    print("7.Bhopal")
    print("8.Ranchi")
    print("9.Patna")
    print("10.Chandigarh")
    g = Graph(10)

    # All the data below has been taken from the site https://www.distancecalculator.net/country/india and the units of the data are in kilometres

    # This is the original data

    # g.graph = [
    # [0,922,987,1153,1654,1033,660,1375,1452,1355],
    # [922,0,1562,239,1358,1607,439,1040,942,435],
    # [987,1562,0,1744,1560,291,1144,1414,1610,1977],
    # [1153,239,1744,0,1305,1759,601,1002,851,235],
    # [1654,1358,1560,1305,0,1356,1125,322,472,1464],
    # [1033,1607,291,1759,1356,0,1170,1259,1481,1995],
    # [660,439,1144,601,1125,1170,0,810,823,834],
    # [1375,1040,1414,1002,322,1259,810,0,252,1179],
    # [1452,942,1610,851,472,1481,823,252,0,996],
    # [1355,435,1977,235,1464,1995,834,1179,996,0]
    # ]

    # Many values are changed to zero below to show the implementation of dijsktra's algorithm
```

Hashing:

- Hashing is the transformation of a string of characters into a usually shorter fixed-length value or key that represents the original string. Hashing is used to index and retrieve items in a database because it is faster to find the item using the shorter hashed key than to find it using the original value. It is also used in many encryption algorithms.






```python
def login():
    flag = 0
    while (flag == 0):

        print("\n1.To login")
        print("2.To signup")
        choice = int(input())

        if (choice == 1):

            with open('data1.txt', 'r') as f:

                print("Enter the username: ", end=" ")
                username = input()
                print("Enter the password: ", end=" ")
                password = input()

                for line in f:

                    l = line.split()

                    if (username == l[0]):

                        decrypt_pass = ""

                        for i in l[1]:
                            decrypt_pass += chr(ord(i) + 1)

                        if (decrypt_pass == password):
                            print("Successfully loged in")
                            flag = 1

                        else:
                            print("Wrong Password")
                            flag = 0
```

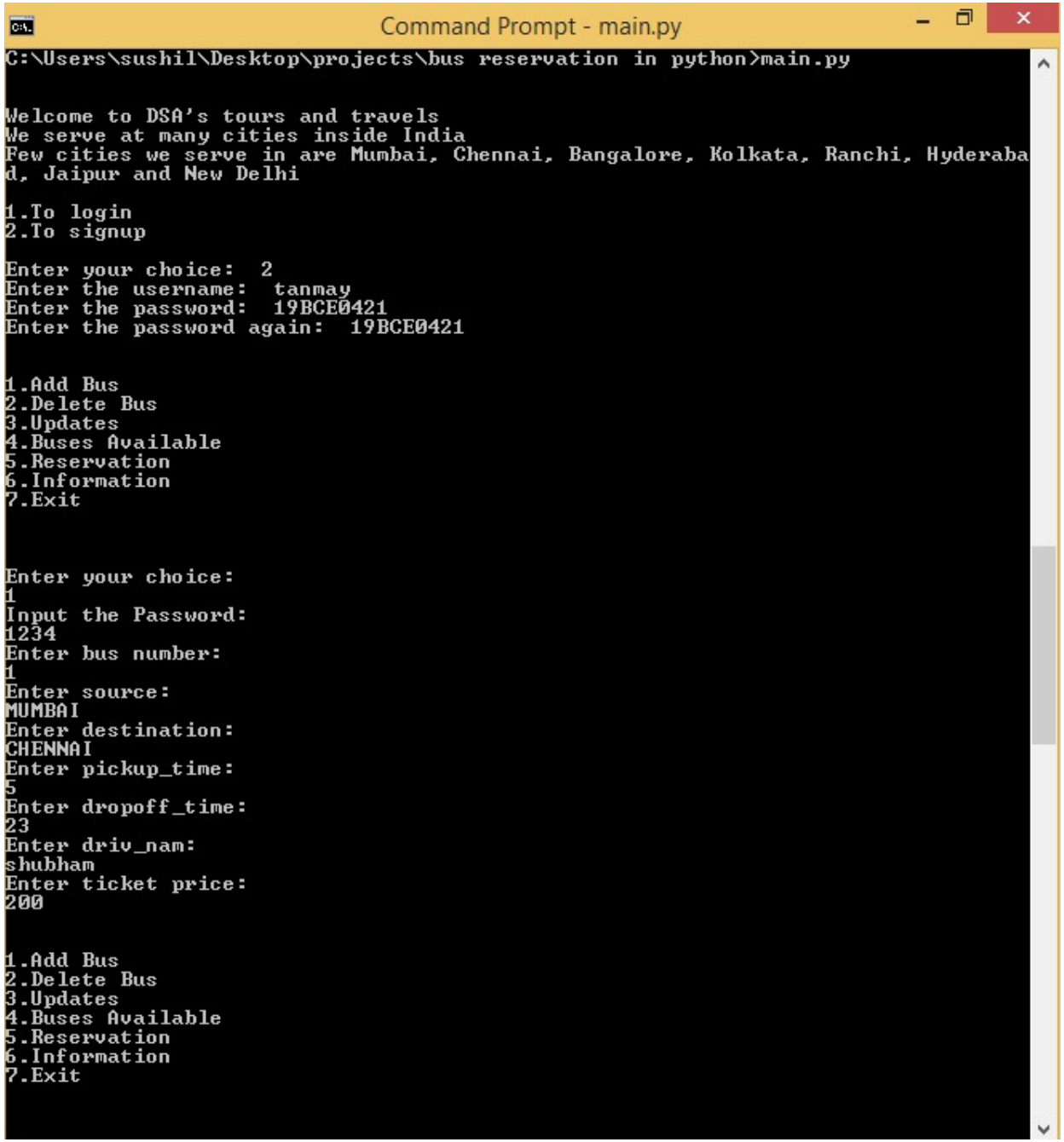
Functions used:

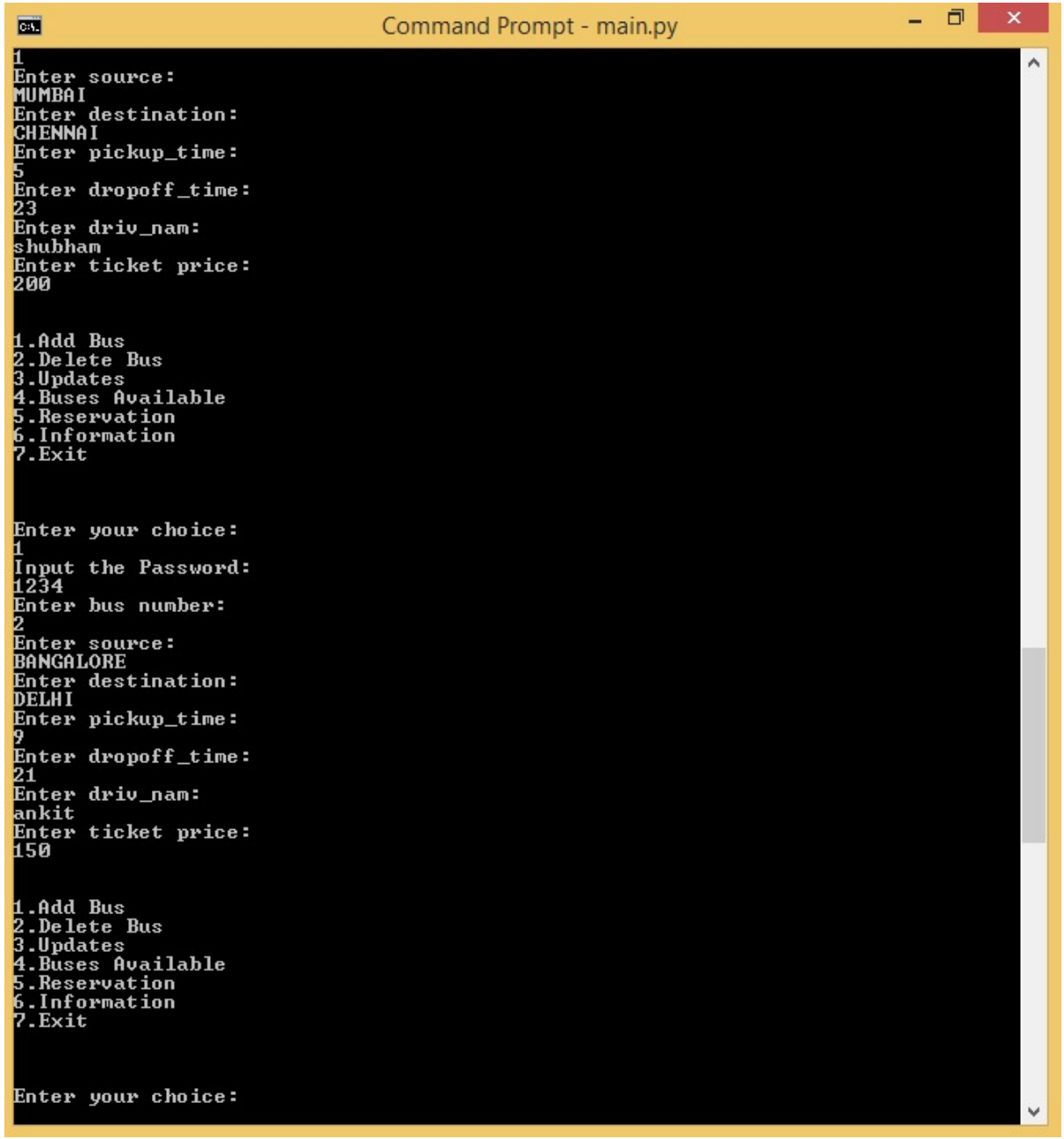
```
def bus_avail():
    print("\n\n")
    line1()
    with open("data.txt", "r") as f:
        data = f.readlines()
        for line in data:
            print("\n")
            words = line.split()
            print(
                "\n\nbus number: {:10s}   \nsource: {:10s}   destination: {:30s} \npickup time:{:10s}   dropoff time: {:30s}
                    words.pop(0), words.pop(0), words.pop(0), words.pop(0), words.pop(0), words.pop(0), words.pop(0)))
            words.pop(0)
            print("\nSeats:\n")
            for i in range(len(words)):
                print(words[i], end="   ")
                if ((i + 1) % 4 == 0):
                    print()
            print()
            print("\n")
            line1()
```

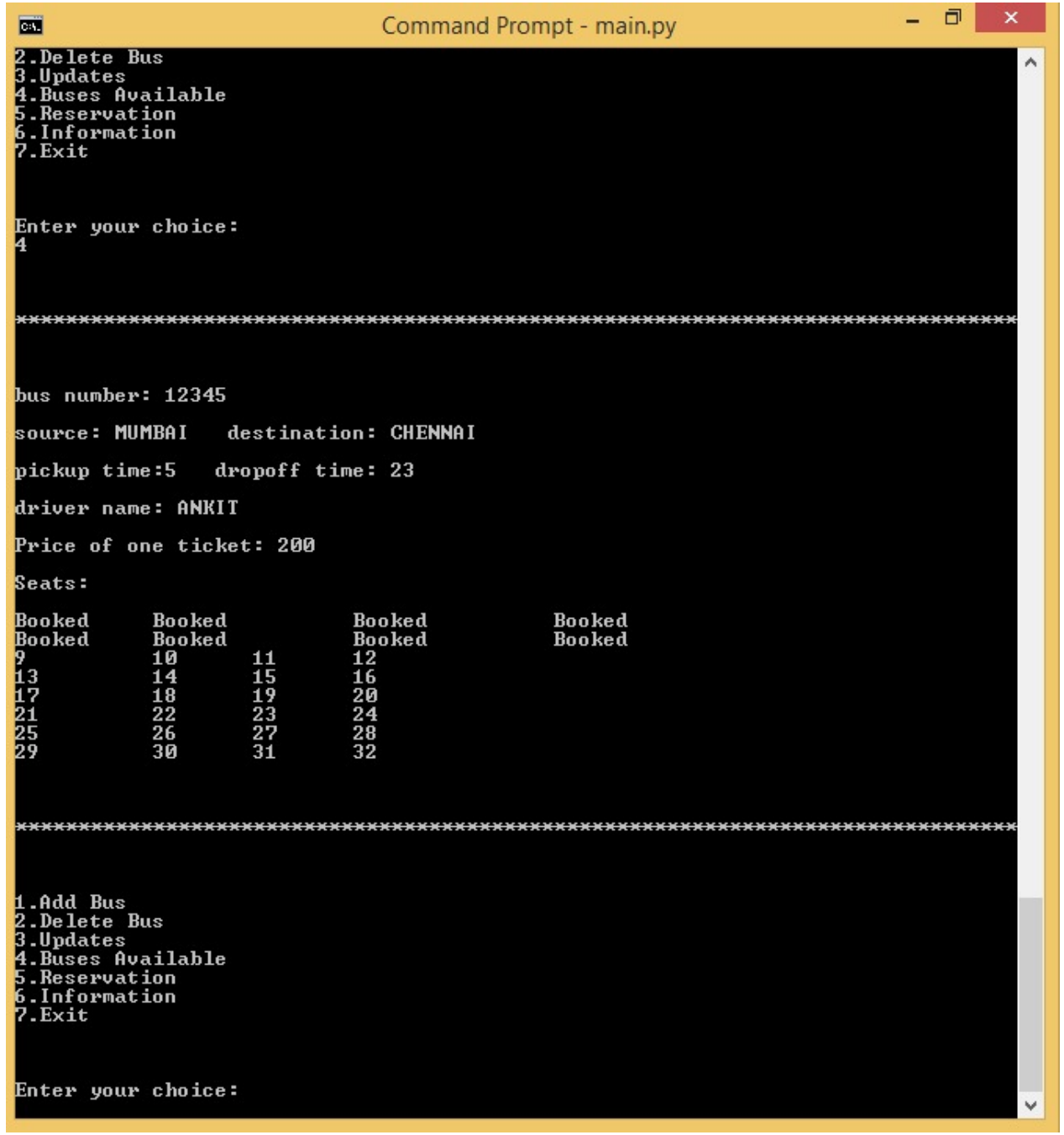
```python
def reservation():
    print("\n\nEnter the Bus Number: ")
    bus_number = input()
    with open("data.txt", "r") as f:
        data = f.readlines()
        for line in data:
            words = line.split()
            if (words[0] == bus_number):
                break
    temp = words
    print("\nDiscount Detail:\n\nFor every 5 tickets 5'%' extra discount will be provided")
    print("\nNumber of tickets to Book: ")
    tickets = int(input())
    if (tickets > int(words[7])):
        print(f"\nOnly {words[7]} tickets left")
    else:
        words[7] = str(int(words[7]) - tickets)
        for i in range(8, len(words)):
            print(words[i], end="   ")
            if ((i + 1) % 4 == 0):
                print()
        print("\n\nEnter the seat numbers :")
        i = 0
        seats_booked = []
        while (i < tickets):
            seat_no = int(input())
            if (words[7 + seat_no] == "Booked"):
                print("\nThe seat is occupied")
                print("Please choose a different seat\n")
            else:
                words[7 + seat_no] = "Booked"
                i += 1
                seats_booked.append(str(seat_no))
```

```python
def info():
    print("1.Mumbai")
    print("2.Jaipur")
    print("3.Bangalore")
    print("4.New Delhi")
    print("5.Kolkata")
    print("6.Chennai")
    print("7.Bhopal")
    print("8.Ranchi")
    print("9.Patna")
    print("10.Chandigarh")
    g = Graph(10)

    # All the data below has been taken from the site https://www.distancecalculator.net/countr

    # This is the original data

    # g.graph = [
    # [0,922,987,1153,1654,1033,660,1375,1452,1355],
    # [922,0,1562,239,1358,1607,439,1040,942,435],
    # [987,1562,0,1744,1560,291,1144,1414,1610,1977],
    # [1153,239,1744,0,1305,1759,601,1002,851,235],
    # [1654,1358,1560,1305,0,1356,1125,322,472,1464],
    # [1033,1607,291,1759,1356,0,1170,1259,1481,1995],
    # [660,439,1144,601,1125,1170,0,810,823,834],
    # [1375,1040,1414,1002,322,1259,810,0,252,1179],
    # [1452,942,1610,851,472,1481,823,252,0,996],
    # [1355,435,1977,235,1464,1995,834,1179,996,0]
    # ]

    # Many values are changed to zero below to show the implementation of dijsktra's algorithm
```

OUTPUT


Command Prompt - main.py
C:\Users\sushil\Desktop\projects\bus reservation in python>main.py
Welcome to DSA's tours and travels
We serve at many cities inside India
Few cities we serve in are Mumbai, Chennai, Bangalore, Kolkata, Ranchi, Hyderaba
d, Jaipur and New Delhi
1.To login
2.To signup
Enter your choice: 2
Enter the username: tanmay
Enter the password: 19BCE0421
Enter the password again: 19BCE0421
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:
1
Input the Password:
1234
Enter bus number:
1
Enter source:
MUMBAI
Enter destination:
CHENNAI
Enter pickup_time:
5
Enter dropoff_time:
23
Enter driv_nam:
shubham
Enter ticket price:
200
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit


Command Prompt - main.py
1
Enter source:
MUMBAI
Enter destination:
CHENNAI
Enter pickup_time:
5
Enter dropoff_time:
23
Enter driv_nam:
shubham
Enter ticket price:
200
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:
1
Input the Password:
1234
Enter bus number:
2
Enter source:
BANGALORE
Enter destination:
DELHI
Enter pickup_time:
9
Enter dropoff_time:
21
Enter driv_nam:
ankit
Enter ticket price:
150
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:


Command Prompt - main.py
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:
4
****
bus number: 12345
source: MUMBAI destination: CHENNAI
pickup time:5 dropoff time: 23
driver name: ANKIT
Price of one ticket: 200
Seats:
Booked Booked Booked Booked
Booked Booked Booked Booked
9 10 11 12
13 14 15 16
17 18 19 20
21 22 23 24
25 26 27 28
29 30 31 32
****
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:


Command Prompt - main.py
Enter the username: tanmay
Enter the password: 19bce0421
Successfully loged in
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:
1
Input the Password:
1234
Enter bus number:
12345
Enter source:
MUMBAI
Enter destination:
CHENNAI
Enter pickup_time:
5
Enter dropoff_time:
23
Enter driv_nam:
ANKIT
Enter ticket price:
200
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:


Command Prompt - main.py
C:\Users\sushil\Desktop\projects\bus reservation in python>main.py
Welcome to DSA's tours and travels
We serve at many cities inside India
Few cities we serve in are Mumbai, Chennai, Bangalore, Kolkata, Ranchi, Hyderaba
d, Jaipur and New Delhi
1.To login
2.To signup
Enter your choice: 2
Enter the username: tanmay
Enter the password: pass
Enter password of length between 8-32 characters
Enter the password again: 19bce0421
Enter the password again: pass
Password doesn't match.
Enter the password again: 19bce0421
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:
7
C:\Users\sushil\Desktop\projects\bus reservation in python>main.py
Welcome to DSA's tours and travels
We serve at many cities inside India
Few cities we serve in are Mumbai, Chennai, Bangalore, Kolkata, Ranchi, Hyderaba
d, Jaipur and New Delhi
1.To login
2.To signup
Enter your choice: 1
Enter the username: tanmay
Enter the password: 19bce0421
Successfully loged in
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information

CONCLUSION:

- This facility is very useful for both the customer and the organization. This is a simple yet effective technology which helps the users to access the service from different places easily.
- Advantages:

a) It reduces the burden of the administrator to maintain numerous data of passengers.

b) It reduces time taken as it is fast and simple.

c) It is very less in price.

d) Improves the efficiency of the organization.

e)Reduces number of errors made which booking tickets.

THANK YOU

Customized Bus Ticket Reservation System

Submitted in partial fulfilment of the requirements for

CSE2003 Data Structures and Algorithms

By:

Aaditya Pareek

19BCE0866

Tanmay Bansal
19BCE0421

Devjyoti Karan

19BCE0690

Under the guidance of:

Prof. Delhi Babu

SCOPE

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CONTENTS:

Abstract:

- Traveling is a large growing business across all countries. Bus reservation system deals with maintenance of records of details of each passenger who had reserved a seat for a journey. It also includes maintenance of information like schedule and details of each bus.
- We believe that we can use computer science concepts to help in the operations of bus ticket reservation by automating and simplifying tasks which are done physically and manually and might cause some error or take more time.
- By using this project, we can book tickets from any part of the world. User can check availability of bus and select seats. The project is useful as it makes the process of ticket booking effective.

Introduction:

We have used computer science concepts like

- OOP Object Oriented Programming
- Shortest Path Finding Algorithms
- Functions and Modules
- Hashing

to build a software for bus ticket reservation in Python 3.

We have used OOP to create classes of type Bus which has several features like bus number, arrival time etc.We use Shortest Path Finding Algorithms to help user find the shortest path using buses from one location to other.

We use various functions such as bus_avail, updates etc. to perform various task which help in the process of ticket reservation. We also use various python modules like math,os etc. to make our software better and get more functions.We have used hashing technique to make login system better.

Literature review:

- We have studied the operations involved in Bus reservation process like selecting location, selecting bus etc. we have made many of these processes automated by use of programming.
- Sometimes these operations if done manually or physically can be performed wrongly or can take time so our program can help in such cases.
- We studied concepts like OOP, Dijkstra's algorithm, Python modules and hashing from various sources to implement them in our program to make our program better.

Problem:

- Most of Bus ticket reservation is done through calls or through using paper ticket system. We think that these systems have a chance of making mistakes and maybe take more time.
- These physical systems of ticket booking can lead to lot of problems as if it takes time some people might miss the bus or theres might be a lot of crowd near the booking system.
- So our solution is a Python program which helps a user/customer to book a ticket. Also there is option to login for the system controller who can edit bus timings, numbers etc.

Solution:

- So our solution is a Python program which helps a user/customer to book a ticket.
- There is an option to login for the system controller who can edit bus timings, numbers etc.
- These systems can function better an faster and are more convenient.
- We have used OOPs concept to make Buses with same properties but different values of those properties.
- We have used Dijkstra's algorithm to find shortest path.
- We have used hashing techniques to make the login more secure

OOPs:

Python is a multi-paradigm programming language which means that it supports different programming approach. One approach is to solve a programming problem is by creating objects. This is known as Object-Oriented Programming(OOP).

An object has two characteristics:

- attributes
- behavior

A class is a blueprint for the object. An object (instance) is an instantiation of a class.

Shortest Path Finding Algorithms:

- There are a lot of different algorithms used for path finding or shortest path. We use these algorithm to help the user go from one place to other in shortest path by bus.
- We use Dijkstra's algorithm to do so in our program. Dijkstra's algorithm (or Dijkstra's Shortest Path First algorithm, SPF algorithm) is an algorithm for finding the shortest paths between nodes in a graph, which may represent, for example, road networks. It was conceived by computer scientist Edsger W. Dijkstra in 1956 and published three years later.

```python
class Graph():

    def __init__(self, vertices):
        self.V = vertices
        self.graph = [[0 for column in range(vertices)] for row in range(vertices)]

    def printSolution(self, dist, des):
        print("The minimum distance between the souce and and destination entered is ", dist[des], " kilometers")

    def minDistance(self, dist, sptSet):
        min = 999999
        for v in range(self.V):
            if dist[v] < min and sptSet[v] == False:
                min = dist[v]
                min_index = v
        return min_index

    def dijkstra(self, src, des):
        dist = [99999] * self.V
        dist[src] = 0
        sptSet = [False] * self.V
        for cout in range(self.V):
            u = self.minDistance(dist, sptSet)
            sptSet[u] = True
            for v in range(self.V):
                if self.graph[u][v] > 0 and sptSet[v] == False and dist[v] > dist[u] + self.graph[u][v]:
                    dist[v] = dist[u] + self.graph[u][v]
        self.printSolution(dist, des)
```

Hashing:

Hashing is the transformation of a string of characters into a usually shorter fixed-length value or key that represents the original string. Hashing is used to index and retrieve items in a database because it is faster to find the item using the shorter hashed key than to find it using the original value. It is also used in many encryption algorithms

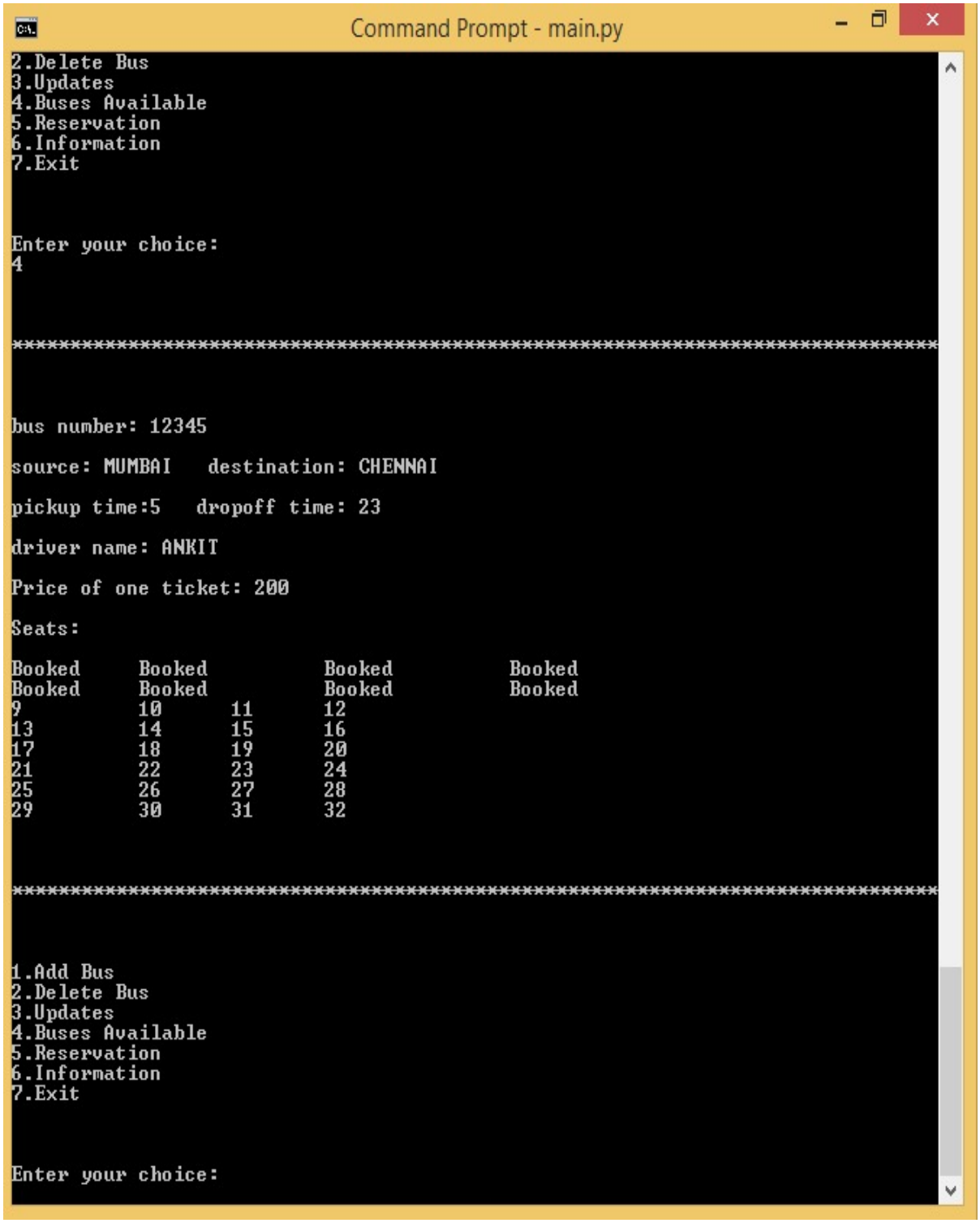
Functions:

We have used functions to perform different tasks like reservation , giving updates, login, signup, shortest path, delete bus, add bus etc.

We have also used some python modules like math, sys etc.

OUTPUT:


Command Prompt - main.py
C:\Users\sushil\Desktop\projects\bus reservation in python>main.py
Welcome to DSA's tours and travels
We serve at many cities inside India
Few cities we serve in are Mumbai, Chennai, Bangalore, Kolkata, Ranchi, Hyderaba
d, Jaipur and New Delhi
1.To login
2.To signup
Enter your choice: 2
Enter the username: tanmay
Enter the password: 19BCE0421
Enter the password again: 19BCE0421
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:
1
Input the Password:
1234
Enter bus number:
1
Enter source:
MUMBAI
Enter destination:
CHENNAI
Enter pickup_time:
5
Enter dropoff_time:
23
Enter driv_nam:
shubham
Enter ticket price:
200
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit


Command Prompt - main.py
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:
4
****
bus number: 12345
source: MUMBAI destination: CHENNAI
pickup time:5 dropoff time: 23
driver name: ANKIT
Price of one ticket: 200
Seats:
Booked Booked Booked Booked
Booked Booked Booked Booked
9 10 11 12
13 14 15 16
17 18 19 20
21 22 23 24
25 26 27 28
29 30 31 32
****
1.Add Bus
2.Delete Bus
3.Updates
4.Buses Available
5.Reservation
6.Information
7.Exit
Enter your choice:

Conclusion:

- This facility is very useful for both the customer and the organization. This is a simple yet effective technology which helps the users to access the service from different places easily.
- Advantages:

a) It reduces the burden of the administrator to maintain numerous data of passengers.

b) It reduces time taken as it is fast and simple.

c) It is very less in price.

d) Improves the efficiency of the organization.

e) Reduces number of errors made which booking tickets