‘’’

Consider the telephone book database of N clients. Make use of a hash table

implementation to quickly look up a client's telephone number. Make use of two

collision handling techniques and compare them using number of comparisons

required to find a set of telephone number

‘’’

#Code

class HashMapTable :

def HashFunction(ele , size) :

return ele % size

numbers = {}

size= numbers.keys()

def PrintStoreTelNumber():

print("Telephone Numbers:")

HashMapTable(numbers,size)

for x in numbers.keys():

print("Name: " , x , "\tNumber:", numbers[x])

print()

def addTelNumber():

print("Add Name and Number")

name = input("Name: ")

phone = input("Number: ")

numbers[name] = phone

def delTelNumber():

print("Remove Name and Number")

name = input("Name: ")

if name in numbers:

del numbers[name]

else:

print(name, "was not found")

def SearchTelNumber():

print("Lookup Number")

name = input("Name: ")

if name in numbers:

print("The number is", numbers[name])

else:

print(name, "was not found")

def Menu():

print('\n Telephone Directory')

print(' Menu\n')

print(' 1. Print Phone Numbers')

print(' 2. Add a Phone Number')

print(' 3. Remove a Phone Number')

print(' 4. Lookup a Phone Number')

print(' 5. Quit')

print()

menu\_choice = 0

Menu()

while menu\_choice != 5:

menu\_choice = int(input("Your\_Choice(1-5): "))

if menu\_choice == 1:

PrintStoreTelNumber()

elif menu\_choice == 2:

addTelNumber()

elif menu\_choice == 3:

delTel Number()

elif menu\_choice == 4:

SearchTelNumber()

elif menu\_choice != 5:

print("Invalid Choice!! Try Again")

Menu()

Output

