ASSINMENT 1:-

```
//Class Definition (hashtable)
class hashtable:
    def __init__(self): //constructor of the hashtable class.
        self.m= (int(input("enter size of hash table")))
        self.hashTable = [None] *self.m  // nitializes the hash table as a list of size m, with all elements set to None
        self.elecount=0
        self.comparions=0//he number of comparisons made during insertion or searching
        print(self.hashTable)  //Prints the initial empty hash table


//Hash Function
    def hashFunction(self,key):
        return key % self.m
//Hash tableFull
    def isfull(self):
        if self.elecount== self.m:
            return True
        else:
            return False


//Linear Probing Insert Method (linearprobr)
    def linearprobr(self,key,data):
            index=self.hashFunction(key)
            compare=0
            while(self.hashTable[index]!=None):

                    index=index+1
```

```
                compare=compare+1
                if(index==self.m):
                        index=0
        self.hashTable[index] = [key,data]
        self.elecount +=1
        print("data inserted at",index)
        print(self.hashTable)
        print("no of cpmparisms= ",compare)
```

```
// Linear Probing Search Method (getlinear)
  def getlinear(self, key,data):
    index = self.hashFunction(key)

    while self.hashTable[index] is not None:
      if self.hashTable[index] == [key,data]:
        return index
      # Linear probing to search for the key
      index = (index + 1) % self.m

    # Key not found
    return None
```

```
 //Quadratic Probing Insert Method (quadraticprobr)
  def quadraticprobr(self,key,data):
        index=self.hashFunction(key)
        compare=0
        i=0
        while(self.hashTable[index]!=None):
```

```
                index=(index+i*i)% self.m
                compare=compare+1
                i=i+1

        self.hashTable[index] = [key,data]
        self.elecount +=1
        print("data inserted at",index)
        print(self.hashTable)
        print("no of cpmparisms= ",compare)
  //    Quadratic Probing Search Method (getQuadratic)
python
Copy code

   def getQuadratic(self, key,data):
      index = self.hashFunction(key)
      i=0
      while self.hashTable[index] is not None:

         if self.hashTable[index] == [key,data]:
            return index
         # Quadractic probing to search for the key
         i=i+1
         index = (index + i*i) % self.m

      # Key not found
      return None
```

```
    def insertvialinear(self,key, data):

        if self.isfull():
            print("table is full")
            return False
        index = self.hashFunction(key)



        if self.hashTable[index]== None:

            self.hashTable[index] = [key, data]
            self.elecount +=1
            print("data inserted at",index)
            print(self.hashTable)

        else:
             print("collision occured apply Linear method")
             self.linearprobr(key,data) # Corrected line

  //Insert Using Quadratic Probing (insertviaQuadratic)
def insertviaQuadratic(self,key, data):

        if self.isfull():
            print("table is full")
            return False
```

```
        index = self.hashFunction(key)



        if self.hashTable[index]== None:

            self.hashTable[index] = [key, data]
            self.elecount +=1
            print("data inserted at",index)
            print(self.hashTable)

        else:
             print("collision occured apply quadratic method")
             self.quadraticprobr(key,data) # Corrected line




def menu():
    obj=hashtable()

    ch=0
    while( ch!=3):
        print("************************")
        print("1. Linear Probe    *")
        print("2. Quadratic Probe  *")
        print("3.Exit")
        print("************************")
```

```
ch = int(input("Enter Choice"))


if ch==1:


    ch2=0
    while(ch2!=3):
     print("** Insert **")
     print("** Search **")
     print("** Exit **")
     ch2=int(input("enter your choice"))
     if ch2==1:
                a=int(input("enter phone number"))
                b=str(input("enter name"))
                obj.insertvialinear(a,b) # Corrected line
     elif ch2==2:
                k=int(input("enter key to be searched"))
                b=str(input("enter name"))
                f=obj.getlinear(k,b)
                if (f==None):
                        print("Key not found")
                else:
                        print("key found at",f)
elif ch==2:
    ch2=0
    obj1=hashtable()
    while(ch2!=3):
     print("** Insert **")
     print("** Search **")
```

```
        print("** Exit **")
        ch2=int(input("enter your choice"))
        if ch2==1:
                a=int(input("enter phone number"))
                b=str(input("enter name"))
                obj1.insertviaQuadratic(a,b) # Corrected line
        elif ch2==2:
                k=int(input("enter key to be searched"))
                b=str(input("enter name"))
                f=obj1.getQuadratic(k,b)
                if (f==None):
                        print("Key not found")
                else:
                        print("key found at",f)


menu()
```