DATA STRUCTURES AND ALGORITHMS LABORATORY

EndSem Practical Examination
SPPU AY 2020-21
Semester :- 2

SUBMITTED BY :-

NAME :- OJUS PRAVIN JAISWAL
SEAT NO. :- S191094290
ROLL NO. :- SACO19108
DIVISION :- A

Group : A

Practical No. : 2

Problem Statement :- For given set of elements create skip list. Find the element in the set that is closest to some given value. (note: Decide the level of element in the list Randomly with some upper limit)

Solution :-

Program :

```
import random


class Node(object):
    '''
    Class to implement node
    '''
    def __init__(self, key, level):
        self.key = key


        # list to hold references to node of different level
        #Allocate memory to forward & Fill forward with NULL
        self.forward = [None]*(level+1)


class SkipList(object):
    '''
    Class for Skip list
```

```
'''
def __init__(self, max_lvl, P):
        # Maximum level for this skip list
        self.MAXLVL = max_lvl

        # P is the fraction of the nodes with level
        # i references also having level i+1 references
        self.P = P

        # create header node and initialize key to -1
        self.header = self.createNode(self.MAXLVL, -1)

        # current level of skip list
        self.level = 0

# create new node
def createNode(self, lvl, key):
        n = Node(key, lvl)
        return n

# create random level for node
def randomLevel(self):
        lvl = 0
        while random.random()<self.P and \
                lvl<self.MAXLVL:lvl += 1
        return lvl
```

```
# insert given key in skip list
def insertElement(self, key):
        # create update array and initialize it
        update = [None]*(self.MAXLVL+1)
        current = self.header

        '''
        start from highest level of skip list
        move the current reference forward while key
        is greater than key of node next to current
        Otherwise inserted current in update and
        move one level down and continue search
        '''
        for i in range(self.level, -1, -1):
                while current.forward[i] and \
                        current.forward[i].key < key:
                        current = current.forward[i]
                update[i] = current

        '''
        reached level 0 and forward reference to
        right, which is desired position to
        insert key.
        '''
        current = current.forward[0]

        '''
```

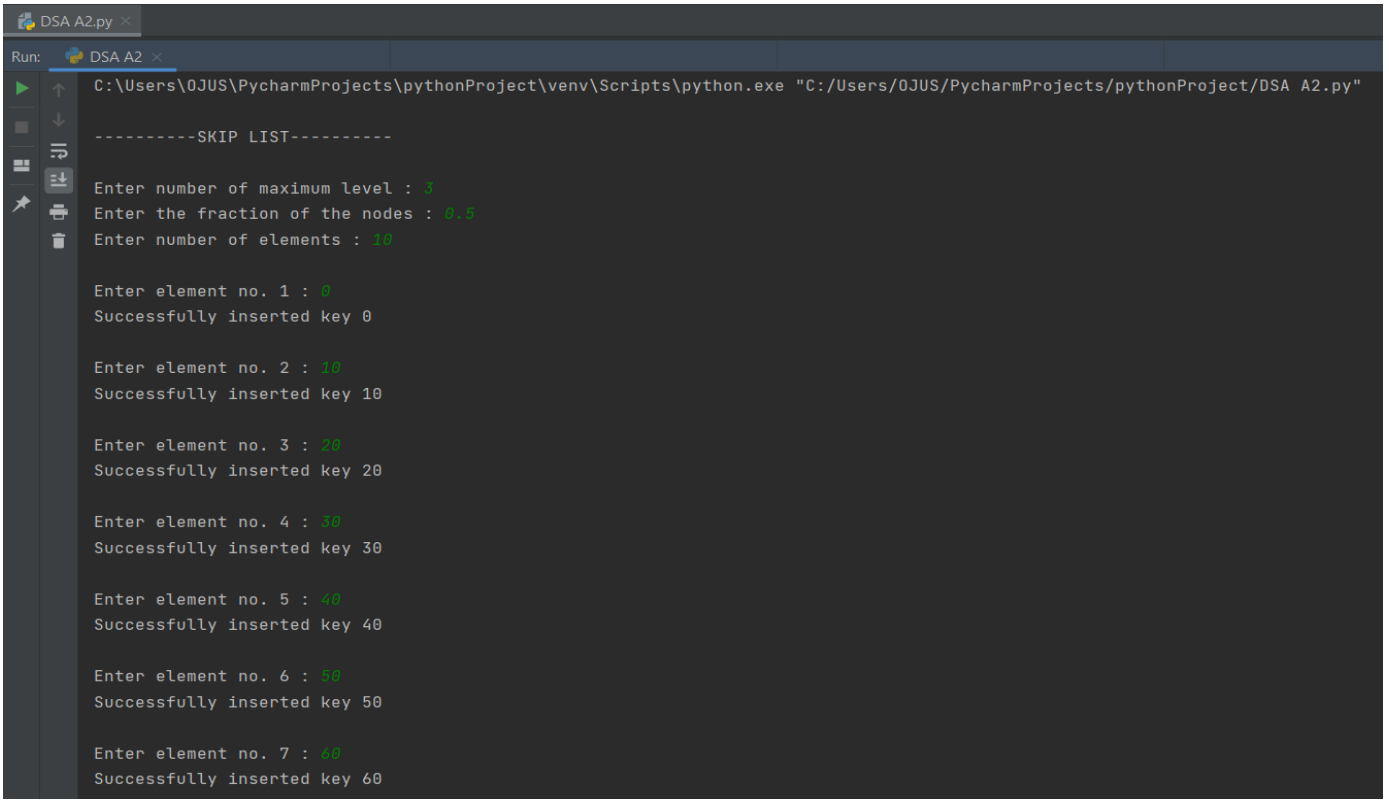
```
        if current is NULL that means we have reached
        to end of the level or current's key is not equal
        to key to insert that means we have to insert
        node between update[0] and current node
'''
        if current == None or current.key != key:
                # Generate a random level for node
                rlevel = self.randomLevel()

                '''
                If random level is greater than list's current
                level (node with highest level inserted in
                list so far), initialize update value with reference
                to header for further use
                '''
                if rlevel > self.level:
                        for i in range(self.level+1, rlevel+1):
                                update[i] = self.header
                        self.level = rlevel

                # create new node with random level generated
                n = self.createNode(rlevel, key)

                # insert node by rearranging references
                for i in range(rlevel+1):
                        n.forward[i] = update[i].forward[i]
                        update[i].forward[i] = n
```

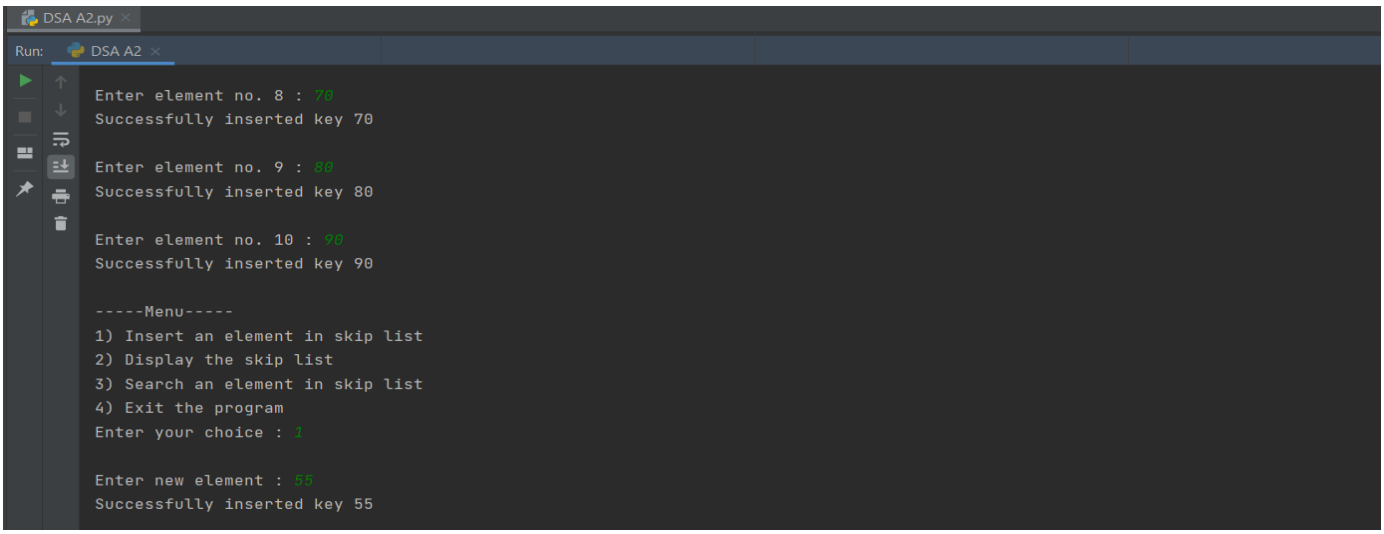
```
                print("Successfully inserted key {}".format(key))


def searchElement(self, key):
        current = self.header

        '''
        start from highest level of skip list
        move the current reference forward while key
        is greater than key of node next to current
        Otherwise inserted current in update and
        move one level down and continue search
        '''
        for i in range(self.level, -1, -1):
                while(current.forward[i] and\
                        current.forward[i].key < key):
                        current = current.forward[i]

        # reached level 0 and advance reference to
        # right, which is prssibly our desired node
        current = current.forward[0]

        # If current node have key equal to
        # search key, we have found our target node
        if current and current.key == key:
                print("Found value ", key)
```

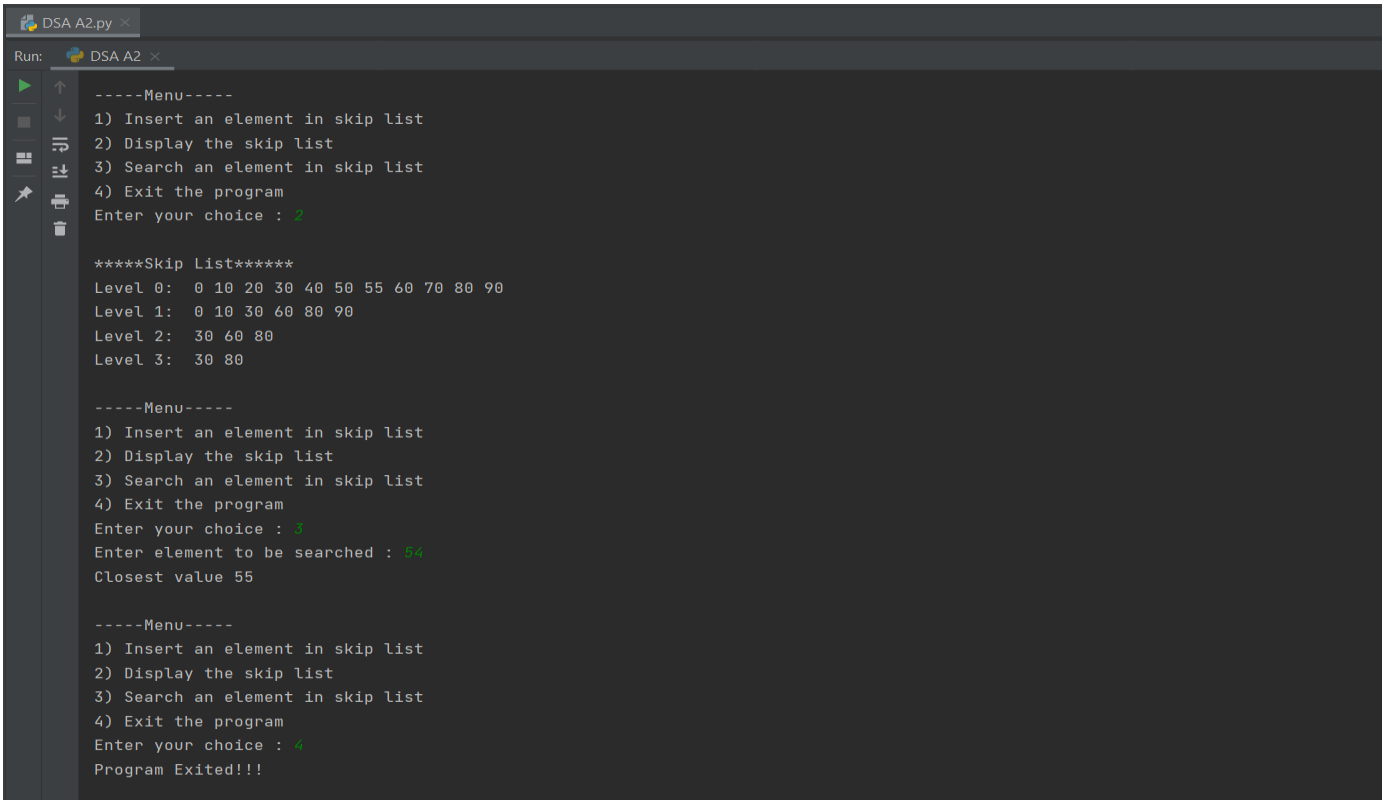
```
            else:
                print ("Closest value",current.key)



    # Display skip list level wise
    def displayList(self):
            print("\n*****Skip List******")
            head = self.header
            for lvl in range(self.level+1):
                    print("Level {}: ".format(lvl), end=" ")
                    node = head.forward[lvl]
                    while(node != None):
                            print(node.key, end=" ")
                            node = node.forward[lvl]
                    print("")

# Driver to test above code
def main():
    print("----------SKIP LIST----------\n")
    l=int(input("Enter number of maximum level : "))
    f=float(input("Enter the fraction of the nodes : "))
    lst = SkipList(l,f)
    n=int(input("Enter number of elements : "))
    for i in range(1, n+1):
            e=int(input("\nEnter element no. %d : " %i))
            lst.insertElement((e))
    while True:
```

```
        print("\n-----Menu-----\n1) Insert an element in skip list\n2) Display the skip list\n3) Search an element in skip list\n4) Exit the program")
        c=int(input("Enter your choice : "))
        if c==1:
            e=int(input("\nEnter new element : "))
            lst.insertElement(e)
        elif c==2:
            lst.displayList()
        elif c==3:
            s=int(input("Enter element to be searched : "))
            lst.searchElement(s)
        elif c==4:
            print("Program Exited!!!")
            exit(0)
        else:
            print("Wrong choice entered!!!")

main()
```

Output :


DSA A2.py
Run:
DSA A2
C:\Users\OJUS\PycharmProjects\pythonProject\venv\Scripts\python.exe "C:/Users/OJUS/PycharmProjects/pythonProject/DSA A2.py"
----------SKIP LIST----------
Enter number of maximum level : 3
Enter the fraction of the nodes : 0.5
Enter number of elements : 10
Enter element no. 1 : 0
Successfully inserted key 0
Enter element no. 2 : 10
Successfully inserted key 10
Enter element no. 3 : 20
Successfully inserted key 20
Enter element no. 4 : 30
Successfully inserted key 30
Enter element no. 5 : 40
Successfully inserted key 40
Enter element no. 6 : 50
Successfully inserted key 50
Enter element no. 7 : 60
Successfully inserted key 60


DSA A2.py
Run:
DSA A2
Enter element no. 8 : 70
Successfully inserted key 70
Enter element no. 9 : 80
Successfully inserted key 80
Enter element no. 10 : 90
Successfully inserted key 90
-----Menu-----
1) Insert an element in skip list
2) Display the skip list
3) Search an element in skip list
4) Exit the program
Enter your choice : 1
Enter new element : 55
Successfully inserted key 55


DSA A2.py
Run:
DSA A2
-----Menu-----
1) Insert an element in skip list
2) Display the skip list
3) Search an element in skip list
4) Exit the program
Enter your choice : 2
*****Skip List******
Level 0: 0 10 20 30 40 50 55 60 70 80 90
Level 1: 0 10 30 60 80 90
Level 2: 30 60 80
Level 3: 30 80
-----Menu-----
1) Insert an element in skip list
2) Display the skip list
3) Search an element in skip list
4) Exit the program
Enter your choice : 3
Enter element to be searched : 54
Closest value 55
-----Menu-----
1) Insert an element in skip list
2) Display the skip list
3) Search an element in skip list
4) Exit the program
Enter your choice : 4
Program Exited!!!