Name - Manaswi Santosh Kulkarni

Roll No - 47

PRN - F23112054

Q batch Comp-2

Group B-11

a) Write a python program to store roll numbers of student in array who attended

training program in random order. Write function for searching whether particular

student attended training program or not, using Linear search and Sentinel search.

b) Write a python program to store roll numbers of student array who attended training

program in sorted order. Write function for searching whether particular student

attended training program or not, using Binary search and Fibonacci search

CODE :-

class Searching:

def \_\_init\_\_(self):

self.\_\_listLength = 0

self.\_\_SearchList = []

def acceptList(self):

self.\_\_listLength = int(input("Enter the number elements in the list "))

for i in range(self.\_\_listLength):

temp = int(input(f"Enter the element at {i} position "))

self.\_\_SearchList.append(temp)

def LinearSearch(self, key):

for i in range(self.\_\_listLength):

if self.\_\_SearchList[i] == key:

return i

return -1

def SentinelSearch(self, key):

last = self.\_\_SearchList[self.\_\_listLength - 1]

# make a copy of the last element

self.\_\_SearchList[self.\_\_listLength - 1] = key

# set last element to be key

i = 0

while self.\_\_SearchList[i] != key:

# As last element is key it will end for sure

i += 1

self.\_\_SearchList[self.\_\_listLength - 1] = last

if ((i < self.\_\_listLength - 1) or (self.\_\_SearchList[self.\_\_listLength - 1] == key)):

return i

else:

return -1

def BinarySearch(self, key):

print("List must be sorted in order for binary search\nSo we will sort it")

print("New list is ")

sortedList = self.\_\_SearchList.copy()

sortedList.sort()

print(sortedList)

first = 0

last = self.\_\_listLength - 1

while first <= last:

mid = first + (last - first) // 2

if sortedList[mid] == key:

return mid

elif key > sortedList[mid]:

first = mid + 1

else:

last = mid - 1

return -1

def FibonacciSearch(self, key):

print("List must be sorted in order for fibonacci search\nSo we will sort it")

print("New list is ")

sortedList = self.\_\_SearchList.copy()

sortedList.sort()

print(sortedList)

memset(-1, self.\_\_listLength + 2)

fibonacci(self.\_\_listLength + 1)

m = 0

while FibonacciList[m] < self.\_\_listLength:

m += 1

offset = -1

while FibonacciList[m] > 1:

i = min(offset + FibonacciList[m - 2], len(sortedList) - 1)

if key > sortedList[i]:

m = m - 1

offset = i

elif key < sortedList[i]:

m -= 2

else:

return i

# check for the last element

if fibonacci(m - 1) and sortedList[offset + 1] == key:

return offset + 1

return -1

def printList(self):

print(self.\_\_SearchList)

FibonacciList = []

def memset(value, length):

for i in range(length):

FibonacciList.append(value)

# print(FibonacciList[i])

def fibonacci(n):

if FibonacciList[n] == -1:

if n <= 1:

FibonacciList[n] = n

else:

FibonacciList[n] = fibonacci(n - 1) + fibonacci(n - 2)

return FibonacciList[n]

obj = Searching()

obj.acceptList()

while True:

print()

print("\*" \* 10 + "Menu" + "\*" \* 10)

print("""1 . Linear Search

2. Binary Search

3. Sentinel Search

4. Fibonacci Search

5. EXIT """)

print("\*" \* 24)

choice = int(input("Enter your choice for the following operations:"))

if choice == 1:

obj.printList()

key = int(input("Enter the key to be found "))

print("\*" \* 10)

print("Linear Search")

position = obj.LinearSearch(key)

if position == -1:

print("Element not found")

else:

print("Element found at position ", position)

elif choice == 2:

obj.printList()

key = int(input("Enter the key to be found "))

print("\*" \* 10)

print("Binary Search")

position = obj.BinarySearch(key)

if position != -1:

print("Element found at position ", position)

else:

print("Element not found")

elif choice == 3:

obj.printList()

key = int(input("Enter the key to be found "))

print("\*" \* 10)

print("Sentinel Search")

position = obj.SentinelSearch(key)

if position == -1:

print("Element not found")

else:

print("Element found at position ", position)

elif choice == 4:

obj.printList()

key = int(input("Enter the key to be found "))

print("\*" \* 10)

print("Fibonacci Search")

position = obj.FibonacciSearch(key)

if position == -1:

print("Element not found")

else:

print("Element found at position ", position)

print("\*" \* 10)

elif choice == 5:

print("EXIT")

break

else:

print("Enter valid input")

OUTPUT :-

Enter the number elements in the list - 5

Enter the element at 0 position- 1

Enter the element at 1 position- 2

Enter the element at 2 position- 3

Enter the element at 3 position- 4

Enter the element at 4 position- 5

\*\*\*\*\*\*\*\*\*\*Menu\*\*\*\*\*\*\*\*\*\*

1 . Linear Search

2. Binary Search

3. Sentinel Search

4. Fibonacci Search

5. EXIT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter your choice for the following operations:- 1

[1, 2, 3, 4, 5]

Enter the key to be found- 4

\*\*\*\*\*\*\*\*\*\*

Linear Search

Element found at position- 3

\*\*\*\*\*\*\*\*\*\*Menu\*\*\*\*\*\*\*\*\*\*

1 . Linear Search

2. Binary Search

3. Sentinel Search

4. Fibonacci Search

5. EXIT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter your choice for the following operations:- 2

[1, 2, 3, 4, 5]

Enter the key to be found- 4

\*\*\*\*\*\*\*\*\*\*

Binary Search

List must be sorted in order for binary search

So we will sort it

New list is-

[1, 2, 3, 4, 5]

Element found at position- 3

\*\*\*\*\*\*\*\*\*\*Menu\*\*\*\*\*\*\*\*\*\*

1 . Linear Search

2. Binary Search

3. Sentinel Search

4. Fibonacci Search

5. EXIT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter your choice for the following operations:- 3

[1, 2, 3, 4, 5]

Enter the key to be found- 4

\*\*\*\*\*\*\*\*\*\*

Sentinel Search

Element found at position- 3

\*\*\*\*\*\*\*\*\*\*Menu\*\*\*\*\*\*\*\*\*\*

1 . Linear Search

2. Binary Search

3. Sentinel Search

4. Fibonacci Search

5. EXIT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter your choice for the following operations:- 4

[1, 2, 3, 4, 5]

Enter the key to be found- 4

\*\*\*\*\*\*\*\*\*\*

Fibonacci Search

List must be sorted in order for fibonacci search

So we will sort it

New list is-

[1, 2, 3, 4, 5]

Element found at position- 3

\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*Menu\*\*\*\*\*\*\*\*\*\*

1 . Linear Search

2. Binary Search

3. Sentinel Search

4. Fibonacci Search

5. EXIT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter your choice for the following operations:- 1

[1, 2, 3, 4, 5]

Enter the key to be found- 6

\*\*\*\*\*\*\*\*\*\*

Linear Search

Element not found

\*\*\*\*\*\*\*\*\*\*Menu\*\*\*\*\*\*\*\*\*\*

1 . Linear Search

2. Binary Search

3. Sentinel Search

4. Fibonacci Search

5. EXIT

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter your choice for the following operations:- 5

EXIT