Modul 6 – Searching Praktikum Struktur Data – 2024

Tugas Praktikum

No.1

Buatlah fungsi sequential search (dapat juga menemukan posisi-posisi dari data yang sama), dengan argument atau parameter berupa:

a. data yang akan dicari

b. list data dari data yang akan dicari

Sedangkan return value dari fungsi ini berupa:

a. indeks-indeks atau posisi dari data yang dicari pada list (jika data ditemukan), dan 'Data tidak ada' jika data tidak ditemukan.

b. Jumlah iterasi yang diperlukan selama proses pencarian

Contoh hasil eksekusi dapat dilihat sebagai berikut:

```
In [2]: a=[1,5,9,8,1,5,10,26,5,12]

In [3]: [hasil,jumlahIterasi]=seqSearch(a,0)
        print('Posisi Data=',hasil)
        print('Jumlah Iterasi=',jumlahIterasi)

        Posisi Data= Data tidak ada
        Jumlah Iterasi= 10
```

Pada contoh diatas, data '0' tidak terdapat pada list *a*, dan jumlah iterasi pencarian yang dilakukan sebanyak 10x.

```
In [2]: a=[1,5,9,8,1,5,10,26,5,12]

In [5]: [hasil,jumlahIterasi]=seqSearch(a,5)
        print('Posisi Data=',hasil)
        print('Jumlah Iterasi=',jumlahIterasi)

        Posisi Data= [1, 5, 8]
        Jumlah Iterasi= 10
```

Data '5' berada pada indeks ke-1, 5, dan 8. Dan jumlah iterasi tetaplah 10 kali iterasi.

```
def sequentialSearch(listData, data):
  hasil = []
  jumlahIterasi = 0
  for ind in range(len(listData)):
    jumlahIterasi += 1
    if listData[ind] == data:
      hasil.append(ind)
  if not hasil:
    hasil = 'Data tidak ada'
  return hasil, jumlahIterasi

a = [1, 5, 9, 8, 1, 5, 10, 26, 5, 12]
[hasil, jumlahIterasi] = sequentialSearch(a, 5)
print(f'Posisi Data = {hasil}')
print(f'Jumlah Iterasi = {jumlahIterasi}')
```

```
Posisi Data = [1, 5, 8]
Jumlah Iterasi = 10
```

No.2

Lakukan modifikasi sequential search tersebut agar tetap meneruskan pencarian sampai data terakhir, dan kembalikan index data semua data yang ditemukan (asumsi, ada data yang sama).

```
def sequentialSearch(listData, data):
    hasil = []
    jumlahIterasi = 0
    for ind in range(len(listData)):
        jumlahIterasi += 1
        if listData[ind] == data:
            hasil.append(ind)
    if not hasil:
        hasil = 'Data tidak ada'
    return hasil, jumlahIterasi

a = [1, 5, 9, 8, 1, 5, 10, 26, 5, 12]
[hasil, jumlahIterasi] = sequentialSearch(a, 5)
print(f'Posisi Data = {hasil}')
print(f'Jumlah Iterasi = {jumlahIterasi}')
```

```
Posisi Data = [1, 5, 8]
Jumlah Iterasi = 10
```

No.3

Buatlah fungsi ordered sequential search (dapat juga menemukan posisi-posisi dari data yang sama), yaitu pencarian pada list data dimana semua data-data pada list tersebut dalam keadaan terurut (ascending). Argument atau parameter pada fungsi tersebut berupa :

a. data yang akan dicari

b. list data dari data yang akan dicari

Sedangkan return value dari fungsi ini berupa:

a. indeks-indeks atau posisi dari data yang dicari pada list (jika data ditemukan), dan 'Data tidak ada' jika data tidak ditemukan.

b. Jumlah iterasi yang diperlukan selama proses pencarian

Contoh hasil eksekusi (dengan data pada list a sama dengan sebelumnya, hanya saja sudah dalam keaadaan terurut) dapat dilihat sebagai berikut :

```
In [7]: a=[1,1,5,5,5,8,9,10,12,26]

In [8]: [hasil,iterasi]=orderedSeqSch(a,0)
        print('Posisi data=',hasil)
        print('jumlah iterasi=',iterasi)

        Posisi data= Data tidak ada
        jumlah iterasi= 1
```

Dapat dilihat bahwa data '0' tidak terdapat pada list *a*, dan jumlah iterasi yang dilakukan hanya 1x saja (dibandingkan dengan fungsi sequential search sebelumnya, yang membutuhkan 10x iterasi)

```
In [7]: a=[1,1,5,5,5,8,9,10,12,26]

In [9]: [hasil,iterasi]=orderedSeqSch(a,9)
        print('Posisi data=',hasil)
        print('jumlah iterasi=',iterasi)

        Posisi data= [6]
        jumlah iterasi= 8
```

Jika dilakukan pencarian data '9', maka data ditemukan pada indeks ke-6 dan hanya dilakukan 8 x iterasi pada proses pencarian.

```
In [7]: a=[1,1,5,5,5,8,9,10,12,26]

In [10]: [hasil,iterasi]=orderedSeqSch(a,5)
         print('Posisi data=',hasil)
         print('jumlah iterasi=',iterasi)

         Posisi data= [2, 3, 4]
         jumlah iterasi= 6
```

Jika dilakukan pencarian data '5' maka akan dihasilkan tiga indeks yaitu, indeks ke-2,3, dan 4. Sedangkan jumlah iterasi yang dilakukan hanyalah sebanyak 6x.

```
def orderedSequential(listData, data):
    hasil = []
    jumlahIterasi = 0
    while jumlahIterasi < len(listData):
        if listData[jumlahIterasi] > data:
            jumlahIterasi += 1
            break
        else:
            if listData[jumlahIterasi] == data:
                hasil.append(jumlahIterasi)
        jumlahIterasi += 1
    if hasil == []:
        hasil = 'Data tidak ada'
    return hasil, jumlahIterasi

arr = [1,1,5,5,5,8,9,10,12,26]
[hasil, jumlahIterasi] = orderedSequential(arr,9)
print(f'Posisi listData: {hasil}')
print(f'Jumlah jumlahIterasi: {jumlahIterasi}')
```

```
Posisi data: [6]
Jumlah iterasi: 8
```

No.4

Buatlah fungsi binary search yang sudah dimodifikasi, sehingga dapat mencari data yang sama dan mengembalikan indeks-indeks dari data yang sama tersebut. Argument atau parameter pada fungsi tersebut berupa :

a. data yang akan dicari

b. list data dari data yang akan dicari

Sedangkan return value dari fungsi ini berupa:

c. indeks-indeks atau posisi dari data yang dicari pada list (jika data ditemukan), dan 'Data tidak ada' jika data tidak ditemukan.

d. Jumlah iterasi yang diperlukan selama proses pencarian

Contoh hasil eksekusi dari fungsi tersebut adalah :

Data yang dicari adalah ‘5’ :

```
In [4]: a=[1,1,5,5,5,8,9,10,12,26]
```

```
In [5]: [hasil,iterasi]=binSearch(a,5)
        print('Posisi data=',hasil)
        print('jumlah iterasi=',iterasi)
```

```
Posisi data= [4, 2, 3]
jumlah iterasi= 4
```

Data yang dicari adalah ‘10’ :

```
In [4]: a=[1,1,5,5,5,8,9,10,12,26]
```

```
In [6]: [hasil,iterasi]=binSearch(a,10)
        print('Posisi data=',hasil)
        print('jumlah iterasi=',iterasi)
```

```
Posisi data= [7]
jumlah iterasi= 2
```

Data yang dicari adalah ‘1’ :

```
In [4]: a=[1,1,5,5,5,8,9,10,12,26]
```

```
In [7]: [hasil,iterasi]=binSearch(a,1)
        print('Posisi data=',hasil)
        print('jumlah iterasi=',iterasi)
```

```
Posisi data= [1, 0]
jumlah iterasi= 3
```

Data yang dicari adalah ‘20’

```
In [4]: a=[1,1,5,5,5,8,9,10,12,26]
```

```
In [8]: [hasil,iterasi]=binSearch(a,20)
        print('Posisi data=',hasil)
        print('jumlah iterasi=',iterasi)
```

```
Posisi data= data tidak ada
jumlah iterasi= 4
```

```
def binarySearch(listData, data):
  hasil = []
  iterasi = 0
  indAwal = 0
  indAkhir = len(listData) - 1
  while indAwal <= indAkhir:
    indTengah = (indAwal + indAkhir) // 2
    iterasi += 1
    if listData[indTengah] == data:
      hasil.append(indTengah)
      kiri = indTengah - 1
      while kiri >= 0 and listData[kiri] == data:
        hasil.append(kiri)
        kiri -= 1
        iterasi += 1
      kanan = indTengah + 1
      while kanan < len(listData) and listData[kanan] == data:
        hasil.append(kanan)
        kanan += 1
        iterasi += 1
      break
    elif listData[indTengah] > data:
      indAkhir = indTengah - 1
```