

PRAKTIKUM ALGORITMA STRUKTUR DATA

MODUL 4

PENCARIAN



Disusun oleh:

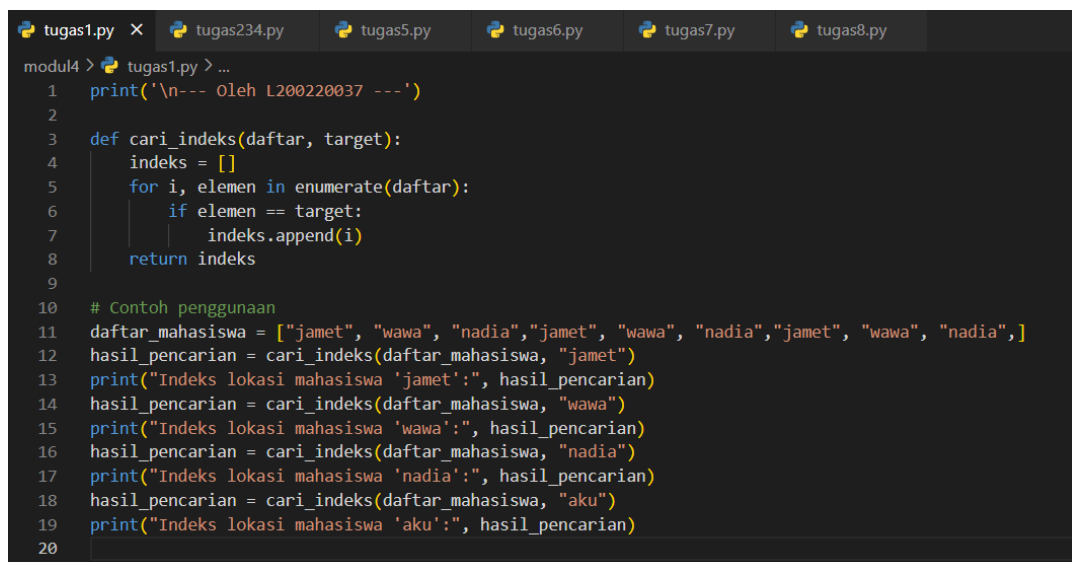
Adinda Aulia Hapsari

L200220037

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS KOMUNIKASI DAN INFORMATIKA
UNIVERSITAS MUHAMMADIYAH SURAKARTA
TAHUN 2024**

Setelah kegiatan selesai, lembar kerja ini dicetak (di-print) dan dikumpulkan ke asisten.	(Diisi oleh Asisten)
NIM : L200220037 Nama : Adinda Aulia Hapsari Nama Asisten : Tanggal Praktikum : 22 Maret 2024	Nilai Praktek : Tanda Tangan :

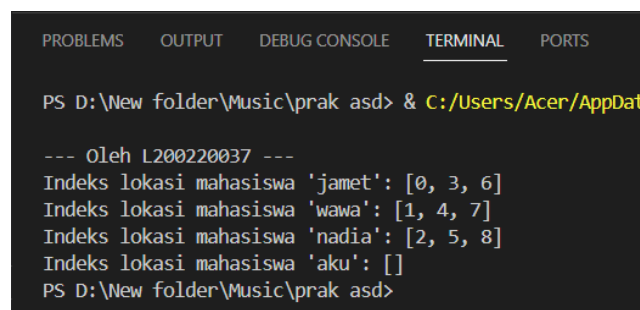
1. Buatlah suatu fungsi pencarian yang, alih-alih mengembalikan True/False, mengembalikan semua index lokasi elemen yang dicari. Jadi, missal pada list daftar mahasiswa di halaman 40 kita mencari mahasiswa yang berasal dari Klaten, kita akan mendapatkan [6,8]. Kalau yang dicari tidak ditemukan, fungsi ini akan mengembalikan list kosong.



```

modul4 > tugas1.py X  tugas234.py  tugas5.py  tugas6.py  tugas7.py  tugas8.py
1  print('\n--- Oleh L200220037 ---')
2
3  def cari_indeks(daftar, target):
4      indeks = []
5      for i, elemen in enumerate(daftar):
6          if elemen == target:
7              indeks.append(i)
8      return indeks
9
10 # Contoh penggunaan
11 daftar_mahasiswa = ["jamet", "wawa", "nadia", "jamet", "wawa", "nadia", "jamet", "wawa", "nadia",]
12 hasil_pencarian = cari_indeks(daftar_mahasiswa, "jamet")
13 print("Indeks lokasi mahasiswa 'jamet':", hasil_pencarian)
14 hasil_pencarian = cari_indeks(daftar_mahasiswa, "wawa")
15 print("Indeks lokasi mahasiswa 'wawa':", hasil_pencarian)
16 hasil_pencarian = cari_indeks(daftar_mahasiswa, "nadia")
17 print("Indeks lokasi mahasiswa 'nadia':", hasil_pencarian)
18 hasil_pencarian = cari_indeks(daftar_mahasiswa, "aku")
19 print("Indeks lokasi mahasiswa 'aku':", hasil_pencarian)
20

```



```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\New folder\Music\prak asd> & C:/Users/Acer/AppDat

--- Oleh L200220037 ---
Indeks lokasi mahasiswa 'jamet': [0, 3, 6]
Indeks lokasi mahasiswa 'wawa': [1, 4, 7]
Indeks lokasi mahasiswa 'nadia': [2, 5, 8]
Indeks lokasi mahasiswa 'aku': []
PS D:\New folder\Music\prak asd>

```

2. Dari list daftar mahasiswa di atas, buatlah fungsi untuk menemukan uang saku yang terkecil diantara mereka.
3. Ubah program diatas agar mengembalikan objek mahasiswa yang mempunyai uang saku terkecil. Jika ada lebih dari satu mahasiswa yang uang sakunya terkecil, semua objek mahasiswa itu dikembalikan.

4. Buatlah suatu fungsi yang mengembalikan semua objek mahasiswa yang uang sakunya kurang dari 250000.

```
tugas1.py tugas234.py X tugas5.py tugas6.py tugas7.py tugas8.py
modul4 > tugas234.py > ...
1 print('\n--- Oleh L200220037 ---')
2
3 class MhsTIF:
4     def __init__(self, nama, nim, alamat, us):
5         self.nama = nama
6         self.nim = nim
7         self.alamat = alamat
8         self.us = us
9
10 #no2
11 def uangSaku_terkecil(daftar):
12     # Menginisialisasi nilai uang saku terkecil dengan uang saku mahasiswa pertama dalam daftar
13     uangSaku_min = daftar[0].us
14     # Melakukan iterasi untuk mencari uang saku terkecil
15     for mhs in daftar:
16         if mhs.us < uangSaku_min:
17             uangSaku_min = mhs.us
18     return uangSaku_min
19
20 #no3
21 def mahasiswa_uang_saku_terkecil(daftar):
22     # Menginisialisasi nilai uang saku terkecil dengan uang saku mahasiswa pertama dalam daftar
23     uangSaku_min = daftar[0].us
24     # Melakukan iterasi untuk mencari nilai uang saku terkecil
25     for mhs in daftar:
26         if mhs.us < uangSaku_min:
27             uangSaku_min = mhs.us
28     # Membuat list untuk menyimpan semua objek mahasiswa yang memiliki uang saku terkecil
29     mahasiswa_terkecil = []
30     # Melakukan iterasi lagi untuk mencari semua mahasiswa dengan uang saku terkecil
31     for mhs in daftar:
32         if mhs.us == uangSaku_min:
33             mahasiswa_terkecil.append(mhs)
34     return mahasiswa_terkecil
35
36 #no4
37 def usKurang(daftar, batas):
38     mahasiswa_kurang_dari = []
39     for mhs in daftar:
40         if mhs.us < batas:
41             mahasiswa_kurang_dari.append(mhs)
42     return mahasiswa_kurang_dari
43
44 c0 = MhsTIF('Adinda Aulia',10,'Sukoharjo', 240000)
45 c1 = MhsTIF('Dian Sasya',51,'Sragen', 230000)
46 c2 = MhsTIF('Memet',2,'Surakarta', 250000)
47 c3 = MhsTIF('Arya Veda',18,'Surakarta', 235000)
48 c4 = MhsTIF('Ganza',4,'Boyolali', 240000)
49 c5 = MhsTIF('Damar Galih',31,'Salatiga', 250000)
50 c6 = MhsTIF('Izzat',13,'Klaten', 245000)
51 c7 = MhsTIF('Adz Dzaka',5,'Wonogiri', 245000)
52 c8 = MhsTIF('Ahmad Taslim',23,'Klaten', 245000)
53 c9 = MhsTIF('Yanto',64,'Karanganyar', 270000)
54 c10 = MhsTIF('Sugeng',29,'Purwodadi', 265000)
55 ## Lalu kita membuat daftar mahasiswa dalam bentuk list seperti ini:
56 Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]
57
58 #lat
59 target = 'Surakarta'
60 for i in Daftar:
61     if i.alamat == target:
62         print(i.nama + ' tinggal di ' + target)
63
64 #panggil no2
65 uangSaku_min = uangSaku_terkecil(Daftar)
66 print("Uang saku terkecil:", uangSaku_min)
67
```

```

68 #panggil no3
69 mahasiswa_terkecil = mahasiswa_uang_saku_terkecil(Daftar)
70 print("Mahasiswa dengan uang saku terkecil:")
71 for mhs in mahasiswa_terkecil:
72     print("Nama:", mhs.nama, "\nUang Saku:", mhs.us)
73
74 #panggil no4
75 batas = 250000
76 mahasiswa_kurang_dari = usKurang(Daftar, batas)
77
78 # Mencetak informasi mahasiswa yang uang sakunya kurang dari 250.000
79 print("Mahasiswa dengan uang saku kurang dari", batas, ":")
80 for mhs in mahasiswa_kurang_dari:
81     print("Nama:", mhs.nama, "\nUang Saku:", mhs.us)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS D:\New folder\Music\prak asd> & C:/Users/Acer/AppData/Local/Mic

--- Oleh L200220037 ---
Memet tinggal di Surakarta
Arya Veda tinggal di Surakarta
Uang saku terkecil: 230000
Mahasiswa dengan uang saku terkecil:
Nama: Dian Sasya
Uang Saku: 230000
Mahasiswa dengan uang saku kurang dari 250000 :
Nama: Adinda Aulia
Uang Saku: 240000
Nama: Dian Sasya
Uang Saku: 230000
Nama: Arya Veda
Uang Saku: 235000
Nama: Ganza
Uang Saku: 240000
Nama: Izzat
Uang Saku: 245000
Nama: Adz Dzaka
Uang Saku: 245000
Nama: Ahmad Taslim
Uang Saku: 245000
PS D:\New folder\Music\prak asd>

```

5. Buatlah suatu program untuk mencari suatu item disebuah linkedlist.

```
tugas1.py tugas234.py tugas5.py X tugas6.py tugas7.py tugas8.py
modul4 > tugas5.py > LinkedList
1 print('\n--- Oleh L200220037 ---')
2
3 class Node:
4     def __init__(self, data):
5         self.data = data
6         self.next = None
7
8 class LinkedList:
9     def __init__(self):
10        self.head = None
11
12    def tambah_di_depan(self, data):
13        new_node = Node(data)
14        new_node.next = self.head
15        self.head = new_node
16
17    def cari_item(self, item):
18        current = self.head
19        while current:
20            if current.data == item:
21                return True
22            current = current.next
23        return False
24
25    def cetak_list(self):
26        current = self.head
27        while current:
28            print(current.data, end=" -> ")
29            current = current.next
30        print("None")
31
32 # Membuat linked list
33 linked_list = LinkedList()
34 linked_list.tambah_di_depan(3)
35 linked_list.tambah_di_depan(7)
36 linked_list.tambah_di_depan(9)
37 linked_list.tambah_di_depan(2)
38
39 # Mencetak linked list
40 print("Linked List:")
41 linked_list.cetak_list()
42
43 # Mencari item di dalam linked list
44 item_dicari = 7
45 if linked_list.cari_item(item_dicari):
46     print(f"Item {item_dicari} ditemukan di dalam linked list.")
47 else:
48     print(f"Item {item_dicari} tidak ditemukan di dalam linked list.")
49
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\New folder\Music\prak asd> & C:/Users/Acer/AppData/L
--- Oleh L200220037 ---
Linked List:
2 -> 9 -> 7 -> 3 -> None
Item 7 ditemukan di dalam linked list.
PS D:\New folder\Music\prak asd>
```

6. Binary search. Ubahlah fungsi binSedi halaman43 agar mengembalikan index lokasi elemen yang ditemukan. Kalau tidak ketemu, akan mengembalikan False.

```
tugas1.py tugas234.py tugas5.py tugas6.py X tugas7.py tugas8.py
modul4 > tugas6.py > binSe
1 print('\n--- Oleh L200220037 ---')
2
3 def binSe(kumpulan, target):
4     # Menginisialisasi indeks awal, indeks akhir, dan status temuan
5     i = 0
6     j = len(kumpulan) - 1
7     found = False
8
9     # Melakukan pencarian biner selama indeks awal tidak melebihi indeks akhir dan elemen belum ditemukan
10    while i <= j and not found:
11        # Menghitung indeks tengah
12        tengah = (i + j) // 2
13        # Jika elemen di tengah adalah target, set status temuan menjadi True
14        if kumpulan[tengah] == target:
15            found = True
16        # Jika target kurang dari elemen di tengah, cari di bagian kiri
17        elif target < kumpulan[tengah]:
18            j = tengah - 1
19        # Jika target lebih besar dari elemen di tengah, cari di bagian kanan
20        else:
21            i = tengah + 1
22
23    # Jika elemen ditemukan, kembalikan indeksnya
24    if found:
25        return tengah
26    # Jika elemen tidak ditemukan, kembalikan False
27    else:
28        return False
29
30 # Contoh penggunaan
31 kumpulan = [2, 3, 5, 7, 11, 13, 17, 19, 23]
32 target = 11
33 hasil = binSe(kumpulan, target)
34 if hasil is not False:
35     print("Elemen", target, "ditemukan pada indeks:", hasil)
36 else:
37     print("Elemen", target, "tidak ditemukan.")
38
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\New folder\Music\prak asd> & C:/Users/Acer/AppData/Local/Programs/Python/Python39-6/Python.exe tugas6.py
--- Oleh L200220037 ---
Elemen 11 ditemukan pada indeks: 4
PS D:\New folder\Music\prak asd>
```

7. Binary search. Ubahlah fungsi binSe itu agar mengembalikan semua index lokasi elemen yang ditemukan. Contoh: mencari angka 6 pada list [2,3,5,6, 6,6,8,9,9,10,11,12,13,13,14] akan mengembalikan [3,4,5]. Karena sudah urut, “tinggal melihat kiri dan kanannya”.

```
tugas1.py tugas234.py tugas5.py tugas6.py tugas7.py X tugas8.py
modul4 > tugas7.py > ...
1 print('\n--- Oleh L200220037 ---')
2 def binSe(kumpulan, target):
3     indeks = []
4     # Menginisialisasi indeks awal, indeks akhir, dan status temuan
5     i = 0
6     j = len(kumpulan) - 1
7
8     # Melakukan pencarian biner selama indeks awal tidak melebihi indeks akhir
9     while i <= j:
10        # Menghitung indeks tengah
11        tengah = (i + j) // 2
12        # Jika elemen di tengah adalah target, tambahkan indeksnya ke dalam list indeks
13        if kumpulan[tengah] == target:
14            indeks.append(tengah)
15            # Cek ke kiri dari indeks tengah untuk mencari kemungkinan elemen yang sama
16            kiri = tengah - 1
17            while kiri >= 0 and kumpulan[kiri] == target:
18                indeks.append(kiri)
19                kiri -= 1
20            # Cek ke kanan dari indeks tengah untuk mencari kemungkinan elemen yang sama
21            kanan = tengah + 1
22            while kanan < len(kumpulan) and kumpulan[kanan] == target:
23                indeks.append(kanan)
24                kanan += 1
25            # Setelah menemukan semua kemungkinan indeks, keluar dari loop
26            break
27        # Jika target kurang dari elemen di tengah, cari di bagian kiri
28        elif target < kumpulan[tengah]:
29            j = tengah - 1
30        # Jika target lebih besar dari elemen di tengah, cari di bagian kanan
31        else:
32            i = tengah + 1
33
34    # Mengembalikan list indeks lokasi elemen yang ditemukan
35    return indeks
36
37 # Contoh penggunaan
38 kumpulan = [2, 3, 5, 6, 6, 6, 8, 9, 9, 10, 11, 12, 13, 13, 14]
39 target = 6
40 hasil = binSe(kumpulan, target)
41 if hasil:
42     print("Elemen", target, "ditemukan pada indeks:", hasil)
43 else:
44     print("Elemen", target, "tidak ditemukan.")
45
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS D:\New folder\Music\prak asd> & C:/Users/Acer/AppData/Local/Python/Python39-64/Python.exe -i
--- Oleh L200220037 ---
Elemen 6 ditemukan pada indeks: [3, 4, 5]
PS D:\New folder\Music\prak asd>
```

8. Pada permainan tebak angka yang sudah kamu buat di Modul 1 (soal nomer 12, halaman 16), kalau angka yang harus ditebak berada diantara 1 dan 100, seharusnya maksimal jumlah tebakan adalah 7. Kalau antara 1 dan 1000, maksimal jumlah tebakan adalah 10. Mengapa seperti itu? Bagaimanakah konsepnya?

100

1-50 : 51-100

1-25 : ...-... : ...-... : ...-...

1-13 : ...-... : ...-... : ...-... : ...-...

1-7 : ...-... : ...-... : ...-... : ...-... : ...-...

1-4 : ...-... : ...-... : ...-... : ...-... : ...-... : ...-...

1-2 : ...-... : ...-... : ...-... : ...-... : ...-... : ...-... : ...-...

2 : ...

Kira-kira seperti itu kak, semoga bisa dipahami, terima kasih.