

Nama : Annisa Nugraheni

NIM : L200180066

Kelas : C

PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

MODUL 4

PENCARIAN

Soal - Soal untuk Mahasiswa

1. Buatlah suatu fungsi pencarian yang alih-alih mengembalikan True/False, mengembalikan semua index lokasi elemen yang dicari. Jadi, misal 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.

Jawab:

Berikut adalah screenshot dari program:

```
L200180066_Algostruk_Modul 4_Tugas.py - D:\Informatika\SMT 4\Prak Algoritma dan Struktur Data\Modul 4\L200180066_Algostruk_Modul 4_Tugas.py (3.7.0)
File Edit Format Run Options Window Help

##Kelas : C

##Membuat kelas MhsTIF

class MhsTIF(object):
    def __init__(self, nama, nim, kota, uangsaku):
        self.nama = nama
        self.nim = nim
        self.kotaTinggal = kota
        self.uangSaku = uangsaku

c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)
c1 = MhsTIF("Budi", 51, "Sragen", 230000)
c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)
c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)
c4 = MhsTIF("Eka", 4, "Boyolali", 240000)
c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)
c6 = MhsTIF("Deni", 13, "Klaten", 245000)
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 29, "Purwodadi", 265000)

Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]

## 1. Fungsi pencarian yang mengembalikan semua index lokasi elemen yang dicari
##      mengembalikan list kosong bila tidak ditemukan

def cariKotaTinggal(list, target):
    a = []
    for i in list:
        if i.kotaTinggal == target:
            a.append(list.index(i))
    return a

##cara pemanggilan
a = cariKotaTinggal(Daftar, "Klaten")
print(a)

****
Ln: 45 Col: 8
```

Berikut adalah program yang saya buat:

```
##Membuat kelas MhsTIF
```

```
class MhsTIF(object):
```

```
    def __init__(self, nama, nim, kota, uangsaku):
```

```
        self.nama = nama
```

```
self.nim = nim
self.kotaTinggal = kota
self.uangSaku = uangsaku
```

```
c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)
c1 = MhsTIF("Budi", 51, "Sragen", 230000)
c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)
c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)
c4 = MhsTIF("Eka", 4, "Boyolali", 240000)
c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)
c6 = MhsTIF("Deni", 13, "Klaten", 245000)
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 29, "Purwodadi", 265000)
```

```
Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]
```

```
## 1. Fungsi pencarian yang mengembalikan semua index lokasi elemen yang dicari
##  mengembalikan list kosong bila tidak ditemukan
```

```
def cariKotaTinggal(list, target):
```

```
    a = []
    for i in list :
        if i.kotaTinggal == target:
            a.append(list.index(i))
    return a
```

```
##cara pemanggilan
```

```
a = cariKotaTinggal(Daftar, "Klaten")
print(a)
```

Berikut adalah screenshot dari hasil ketika program diatas dijalankan:

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (tags/v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: D:\Informatika\SMT 4\Prak Algoritma dan Struktur Data\Modul 4\L200180066_Algostruk_Modul 4_Tugas.py
>>> a = cariKotaTinggal(Daftar, "Klaten")
>>> print(a)
[6, 8]
>>> a = cariKotaTinggal(Daftar, "Surakarta")
>>> print(a)
[2, 3]
>>> a = cariKotaTinggal(Daftar, "Jakarta")
>>> print(a)
[1]
>>> |
```

2. Dari list daftar mahasiswa diatas, buatlah fungsi untuk menemukan uang saku yang terkecil di antara mereka

Jawab:

Berikut adalah screenshot dari program:

```
*L200180066_Algostruk_Modul 4_Tugas.py - D:\Informatika\SMT 4\Prak Algoritma dan Struktur Data\Modul 4\L200180066_Algostruk_Modul 4_Tugas.py (3.7.0)*
File Edit Format Run Options Window Help

##Praktikum Algoritma dan Struktur Data
##Modul 4 Pencarian
##Tugas
##Nama      : Annisa Nugraheni
##NIM       : L200180066
##Kelas    : C

##Membuat kelas MhsTIF
##
class MhsTIF(object):
    def __init__(self, nama, nim, kota, uangsaku):
        self.nama = nama
        self.nim = nim
        self.kotaTinggal = kota
        self.uangSaku = uangsaku

c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)
c1 = MhsTIF("Budi", 51, "Sragen", 230000)
c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)
c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)
c4 = MhsTIF("Eka", 4, "Boyolali", 240000)
c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)
c6 = MhsTIF("Deni", 13, "Klaten", 245000)
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 29, "Purwodadi", 265000)

Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]

#### 2. Fungsi untuk menemukan uang saku yang terkecil
def cariUangSakuTerkecil(list):
    temp = list[0].uangSaku
    for i in list[1:]:
        if i.uangSaku < temp:
            temp = i.uangSaku
    return temp

##cara pemanggilan
a = cariUangSakuTerkecil(Daftar)
print(a)|
```

Berikut adalah program yang saya buat:

##Membuat kelas MhsTIF

class MhsTIF(object):

def __init__(self, nama, nim, kota, uangsaku):

self.nama = nama

self.nim = nim

self.kotaTinggal = kota

self.uangSaku = uangsaku

c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)

```

c1 = MhsTIF("Budi", 51, "Sragen", 230000)
c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)
c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)
c4 = MhsTIF("Eka", 4, "Boyolali", 240000)
c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)
c6 = MhsTIF("Deni", 13, "Klaten", 245000)
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 29, "Purwodadi", 265000)

```

```
Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]
```

2. Fungsi untuk menemukan uang saku yang terkecil

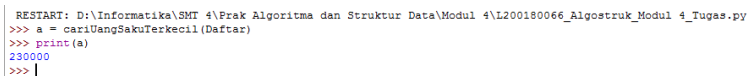
```

def cariUangSakuTerkecil(list):
    temp = list[0].uangSaku
    for i in list[1:]:
        if i.uangSaku < temp:
            temp = i.uangSaku
    return temp

##cara pemanggilan
a = cariUangSakuTerkecil(Daftar)
print(a)

```

Berikut adalah screenshot dari hasil ketika program diatas dijalankan:



```

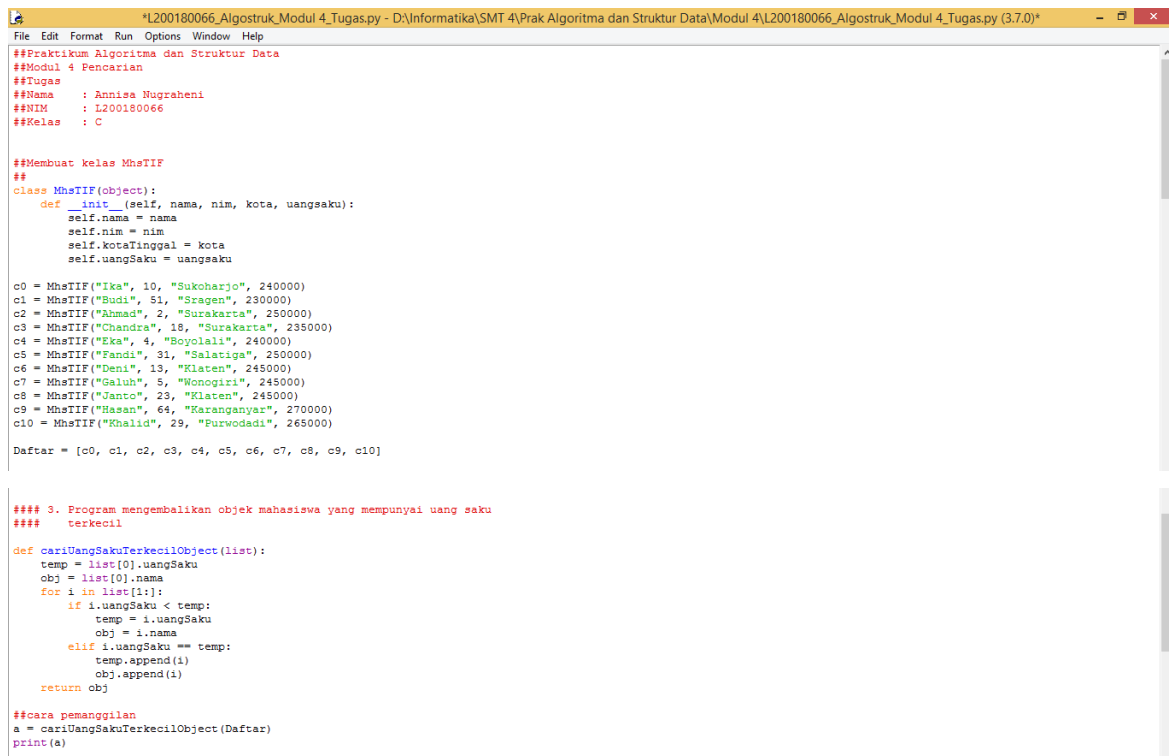
RESTART: D:\Informatika\SMT 4\Prek Algoritma dan Struktur Data\Modul 4\L200180066_Algostruk_Modul 4_Tugas.py
>>> a = cariUangSakuTerkecil(Daftar)
>>> print(a)
230000
>>> |

```

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 objek mahasiswa itu dikembalikan

Jawab:

Berikut adalah screenshot dari program:



```
##Praktikum Algoritma dan Struktur Data
##Modul 4 Pencarian
##Tugas
##Nama : Annisa Nugraheni
##NIM : L200180066
##Kelas : C

##Membuat kelas MhsTIF
##
class MhsTIF(object):
    def __init__(self, nama, nim, kota, uangsaku):
        self.nama = nama
        self.nim = nim
        self.kotaTinggal = kota
        self.uangSaku = uangsaku

c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)
c1 = MhsTIF("Budi", 51, "Sragen", 230000)
c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)
c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)
c4 = MhsTIF("Eka", 4, "Boyolali", 240000)
c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)
c6 = MhsTIF("Deni", 13, "Klaten", 245000)
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 29, "Purwodadi", 265000)

Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]

#### 3. Program mengembalikan objek mahasiswa yang mempunyai uang saku
#### terkecil

def cariUangSakuTerkecilObject(list):
    temp = list[0].uangSaku
    obj = list[0].nama
    for i in list[1:]:
        if i.uangSaku < temp:
            temp = i.uangSaku
            obj = i.nama
        elif i.uangSaku == temp:
            temp.append(i)
            obj.append(i)
    return obj

##cara pemanggilan
a = cariUangSakuTerkecilObject(Daftar)
print(a)
```

Berikut adalah program yang saya buat:

##Membuat kelas MhsTIF

class MhsTIF(object):

def __init__(self, nama, nim, kota, uangsaku):

self.nama = nama

self.nim = nim

self.kotaTinggal = kota

self.uangSaku = uangsaku

c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)

c1 = MhsTIF("Budi", 51, "Sragen", 230000)

c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)

c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)

c4 = MhsTIF("Eka", 4, "Boyolali", 240000)

```

c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)
c6 = MhsTIF("Deni", 13, "Klaten", 245000)
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 29, "Purwodadi", 265000)

```

```
Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]
```

3. Program mengembalikan objek mahasiswa yang mempunyai uang saku

terkecil

```
def cariUangSakuTerkecilObject(list):
```

```
    temp = list[0].uangSaku
```

```
    obj = list[0].nama
```

```
    for i in list[1:]:
```

```
        if i.uangSaku < temp:
```

```
            temp = i.uangSaku
```

```
            obj = i.nama
```

```
        elif i.uangSaku == temp:
```

```
            temp.append(i)
```

```
            obj.append(i)
```

```
    return obj
```

##cara pemanggilan

```
a = cariUangSakuTerkecilObject(Daftar)
```

```
print(a)
```

Berikut adalah screenshot dari hasil ketika program diatas dijalankan:

```

RESTART: D:\Informatika\SMT 4\Prak Algoritma dan Struktur Data\Modul 4\L200180066_Algostruk_Modul 4_Tugas.py
>>> a = cariUangSakuTerkecilObject(Daftar)
>>> print(a)
Budi
>>> |

```

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

Jawab:

Berikut adalah screenshot dari program:



```
##Praktikum Algoritma dan Struktur Data
##Modul 4 Pencarian
##Tugas
##Nama      : Annisa Nugraheni
##NIM       : L200180066
##Kelas    : C

##Membuat kelas MhsTIF
class MhsTIF(object):
    def __init__(self, nama, nim, kota, uangsaku):
        self.nama = nama
        self.nim = nim
        self.kotaTinggal = kota
        self.uangSaku = uangsaku

c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)
c1 = MhsTIF("Budi", 51, "Sragen", 230000)
c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)
c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)
c4 = MhsTIF("Eka", 4, "Boyolali", 240000)
c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)
c6 = MhsTIF("Deni", 13, "Klaten", 245000)
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 29, "Purwodadi", 265000)

Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]

### 4. Fungsi untuk mengembalikan semua objek mahasiswa yang uang sakunya
###      kurang dari 250000

def cariUangSakuKurang250k(list):
    temp = []
    for i in list:
        if i.uangSaku < 250000:
            temp.append(i)
    return temp

#cara pemanggilan
a = cariUangSakuKurang250k(Daftar)
for i in a:
    print(i.nama)
```

Berikut adalah program yang saya buat:

##Membuat kelas MhsTIF

class MhsTIF(object):

def __init__(self, nama, nim, kota, uangsaku):

self.nama = nama

self.nim = nim

self.kotaTinggal = kota

self.uangSaku = uangsaku

c0 = MhsTIF("Ika", 10, "Sukoharjo", 240000)

c1 = MhsTIF("Budi", 51, "Sragen", 230000)

c2 = MhsTIF("Ahmad", 2, "Surakarta", 250000)

c3 = MhsTIF("Chandra", 18, "Surakarta", 235000)

c4 = MhsTIF("Eka", 4, "Boyolali", 240000)

c5 = MhsTIF("Fandi", 31, "Salatiga", 250000)

c6 = MhsTIF("Deni", 13, "Klaten", 245000)

```
c7 = MhsTIF("Galuh", 5, "Wonogiri", 245000)
c8 = MhsTIF("Janto", 23, "Klaten", 245000)
c9 = MhsTIF("Hasan", 64, "Karanganyar", 270000)
c10 = MhsTIF("Khalid", 29, "Purwodadi", 265000)
```

```
Daftar = [c0, c1, c2, c3, c4, c5, c6, c7, c8, c9, c10]
```

4. Fungsi untuk mengembalikan semua objek mahasiswa yang uang sakunya

kurang dari 250000

```
def cariUangSakuKurang250k(list):
```

```
    temp = []
```

```
    for i in list:
```

```
        if i.uangSaku < 250000:
```

```
            temp.append(i)
```

```
    return temp
```

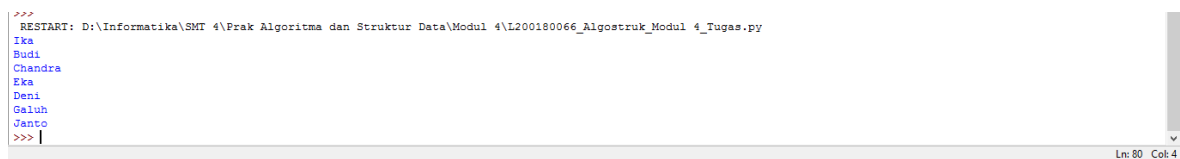
```
##cara pemanggilan
```

```
a = cariUangSakuKurang250k(Daftar)
```

```
for i in a:
```

```
    print(i.nama)
```

Berikut adalah screenshot dari hasil ketika program diatas dijalankan:



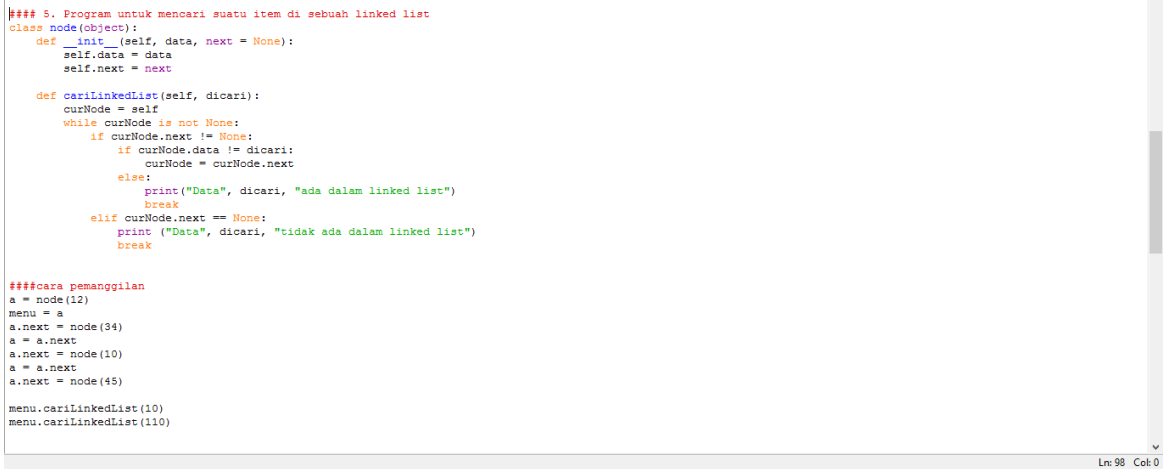
```
RESTART: D:\Informatika\SMT 4\Prak Algoritma dan Struktur Data\Modul 4\L200180066_Algostruk_Modul 4_Tugas.py
Ika
Budi
Chandra
Eka
Deni
Galuh
Janto
>>> |
```

Ln: 80 Col: 4

5. Buatlah suatu program untuk mencari suatu item di sebuah linked list

Jawab:

Berikut adalah screenshot dari program:

A screenshot of a code editor showing a Python program. The program defines a 'node' class with an '__init__' method and a 'cariLinkedList' method. It then creates a linked list 'menu' with nodes containing values 12, 34, 10, and 45. Finally, it calls 'menu.cariLinkedList(10)' and 'menu.cariLinkedList(110)' to demonstrate the search functionality. The output of the program is visible in the bottom right corner of the editor window.

```
#### 5. Program untuk mencari suatu item di sebuah linked list
class node(object):
    def __init__(self, data, next = None):
        self.data = data
        self.next = next

    def cariLinkedList(self, dicari):
        curNode = self
        while curNode is not None:
            if curNode.next != None:
                if curNode.data != dicari:
                    curNode = curNode.next
            else:
                print("Data", dicari, "ada dalam linked list")
                break
        elif curNode.next == None:
            print ("Data", dicari, "tidak ada dalam linked list")
            break

####cara pemanggilan
a = node(12)
menu = a
a.next = node(34)
a = a.next
a.next = node(10)
a = a.next
a.next = node(45)

menu.cariLinkedList(10)
menu.cariLinkedList(110)
```

Berikut adalah program yang saya buat:

5. Program untuk mencari suatu item di sebuah linked list

class node(object):

def __init__(self, data, next = None):

self.data = data

self.next = next

def cariLinkedList(self, dicari):

curNode = self

while curNode is not None:

if curNode.next != None:

if curNode.data != dicari:

curNode = curNode.next

else:

print("Data", dicari, "ada dalam linked list")

break

elif curNode.next == None:

print ("Data", dicari, "tidak ada dalam linked list")

break

Berikut adalah screenshot dari hasil ketika program diatas dijalankan:

```
'''
RESTART: D:\Informatika\SMT 4\Prak Algoritma dan Struktur Data\Modul 4\L200180066_Algostruk_Modul 4_Tugas.py
>>> a = node(12)
>>> menu = a
>>> a.next = node(34)
>>> a = a.next
>>> a.next = node(10)
>>> a = a.next
>>> a.next = node(45)
>>> menu.carilinkedList(10)
Data 10 ada dalam linked list
>>> menu.carilinkedList(110)
Data 110 tidak ada dalam linked list
>>> |
```

6. Binary search. Ubahlah fungsi binSe di halaman 43 agar mengembalikan index lokasi elemen yang ditemukan. Kalau tidak ketemu, akan mengembalikan False

Jawab:

Berikut adalah screenshot dari program:

```
## 6. Fungsi binSe mengembalikan index lokasi elemen yang ditentukan,
## kalau tidak ketemu mengembalikan False

def binSe(kumpulan, target):
    low = 0
    high = len(kumpulan)-1
    while low <= high:
        mid = (high+low)//2
        if kumpulan[mid] == target:
            return mid
        elif target < kumpulan[mid]:
            high = mid-1
        else:
            low = mid+1
    return False

kumpulan = [2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]
print(binSe(kumpulan, 5))|
```

Berikut adalah program yang saya buat:

```
## 6. Fungsi binSe mengembalikan index lokasi elemen yang ditentukan,
## kalau tidak ketemu mengembalikan False
```

```
def binSe(kumpulan, target):
```

```
    low = 0
```

```
    high = len(kumpulan)-1
```

```
    while low <= high:
```

```
        mid = (high+low)//2
```

```
        if kumpulan[mid] == target:
```

```
            return mid
```

```
        elif target < kumpulan[mid]:
```

```
            high = mid-1
```

```
        else:
```

```
            low = mid+1
```

```
    return False
```

Berikut adalah screenshot dari hasil ketika program diatas dijalankan:

```
...
RESTART: D:\Informatika\SMT 4\Prak Algoritma dan Struktur Data\Modul 4\L200180066_Algostruk_Modul 4_Tugas.py
>>> kumpulan = [2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]
>>> print(binSe(kumpulan, 5))
2
>>> print(binSe(kumpulan, 2))
0
>>> print(binSe(kumpulan, 18))
5
>>> print(binSe(kumpulan, 50))
False
>>> print(binSe(kumpulan, 66))
False
>>> |
```

Ln: 108 Col: 4

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.

Jawab:

Berikut adalah screenshot dari program:

```
#### 7. Fungsi binSe mengembalikan semua index lokasi elemen yg ditemukan

def binSeAll(kumpulan, target):
    temp = []
    low = 0
    high = len(kumpulan)-1
    while low <= high :
        mid = (high+low)//2
        if kumpulan[mid] == target:
            midKiri = mid-1
            while kumpulan[midKiri] == target:
                temp.append(midKiri)
                midKiri = midKiri-1
            temp.append(mid)
            midKanan = mid+1
            while kumpulan[midKanan] == target:
                temp.append(midKanan)
                midKanan = midKanan+1
            return temp
        elif target < kumpulan[mid]:
            high = mid-1
        else:
            low = mid+1
    return False

kumpulan = [2, 3, 5, 6, 6, 6, 8, 9, 9, 10, 11, 12, 13, 13, 14]
print(binSeAll(kumpulan, 6))
```

Berikut adalah program yang saya buat:

7. Fungsi binSe mengembalikan semua index lokasi elemen yg ditemukan

```
def binSeAll(kumpulan, target):
```

```
    temp = []
```

```
    low = 0
```

```
    high = len(kumpulan)-1
```

```
    while low <= high :
```

```
        mid = (high+low)//2
```

```
        if kumpulan[mid] == target:
```

```
            midKiri = mid-1
```

```
            while kumpulan[midKiri] == target:
```

```
                temp.append(midKiri)
```

```
                midKiri = midKiri-1
```

```

temp.append(mid)
midKanan = mid+1
while kumpulan[midKanan] == target:
    temp.append(midKanan)
    midKanan = midKanan+1
return temp
elif target < kumpulan[mid]:
    high = mid-1
else:
    low = mid+1
return False

```

Berikut adalah screenshot dari hasil ketika program diatas dijalankan:

```

RESTART: D:\Informatika\SMT 4\Prak Algoritma dan Struktur Data\Modul 4\L200180066_Algostruk_Modul 4_Tugas.py
>>> kumpulan = [2, 3, 5, 6, 6, 6, 8, 9, 9, 10, 11, 12, 13, 13, 14]
>>> print(binSeAll(kumpulan, 6))
[3, 4, 5]
>>> print(binSeAll(kumpulan, 9))
[7, 8]
>>> print(binSeAll(kumpulan, 13))
[12, 13]
>>> print(binSeAll(kumpulan, 2))
[0]
>>> |

```

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

Jawab:

Ada dua kemungkinan pola bisa digunakan.

Misalkan, angka yang akan ditebak adalah 70.

- Pola pertama

a = nilai tebakan pertama // 2

tebakan selanjutnya = nilai tebakan “lebih dari” + a

*jika hasil tebakan selanjutnya “kurang dari”, maka nilai yang dipakai tetap nilai lebih dari sebelumnya

a = a // 2

Simulasi

Tebakan ke-1 : 50 (mengambil nilai tengah) => Jawaban = “Lebih dari itu”

Tebakan ke-2 : 75 (dari 50 + 25) => Jawaban = “Kurang dari itu”

Tebakan ke-3 : 62 (dari 50 + 12) => Jawaban = “Lebih dari itu”

Tebakan ke-4 : 68 (dari $62 + 6$) \Rightarrow Jawaban = “Lebih dari itu”

Tebakan ke-5 : 71 (dari $68 + 3$) \Rightarrow Jawaban = “Kurang dari itu”

Tebakan ke-6 : 69 (dari $68 + 1$) \Rightarrow Jawaban = “Lebih dari itu”

Tebakan ke-7 : antara 71 dan 69 hanya ada 1 angka = 70

- Pola kedua

Menggunakan barisan geometri $S_n = 2^n$

Barisan yang terjadi adalah 2, 4, 8, 16, 32, 64

Misal angka yang akan ditebak adalah 70

Tebakan ke-1 : 64 \Rightarrow Jawaban = “Lebih dari itu”

Tebakan ke-2 : 96 (dari $64 + 32$) \Rightarrow Jawaban = “Kurang dari itu”

Tebakan ke-3 : 80 (dari $64 + 16$) \Rightarrow Jawaban = “Kurang dari itu”

Tebakan ke-4 : 72 (dari $64 + 8$) \Rightarrow Jawaban = “Kurang dari itu”

Tebakan ke-5 : 68 (dari $64 + 4$) \Rightarrow Jawaban = “Lebih dari itu”

Tebakan ke-6 : 70 (dari $68 + 2$) \Rightarrow Jawaban = “TEPAT”