



NAMA : KEVIN AVICENNA WIDIARTO
NIM : L200200183
Modul : 4

Praktikum Algoritma Struktur Data

MODUL 4

No 1

```
from mhs import MhsTIF, Daftar

def cariTempatTinggal(x):
    b = []
    for i in Daftar :
        if i.kotaTinggal == str(x):
            b.append(Daftar.index(i))
    return b
```

IDLE Shell 3.10.3

File Edit Shell Debug Options Window Help

```
>>> cariTempatTinggal("Klaten")
[6, 8]
>>> cariTempatTinggal("Jakarta")
[]
>>>
```

No 2

```
from mhs import MhsTIF, Daftar, c0

def CariNilaiTerkecil(list):
    k = c0.ambilUangSaku()
    for x in Daftar:
        if x.ambilUangSaku() < k:
            k = x.ambilUangSaku()
    return k
```

IDLE Shell 3.10.3

File Edit Shell Debug Options Window Help

```
>>> CariNilaiTerkecil(Daftar)
230000
```

No 3

```
from mhs import MhsTIF, Daftar, c0

def CariNilaiTerkecilBanyak(list):
    k = c0.ambilUangSaku()
    a = []
    for x in Daftar:
        if x.ambilUangSaku() < k:
            a.append((x.nama, x.NIM, x.kotaTinggal, x.uangSaku))
    return a
```

IDLE Shell 3.10.3

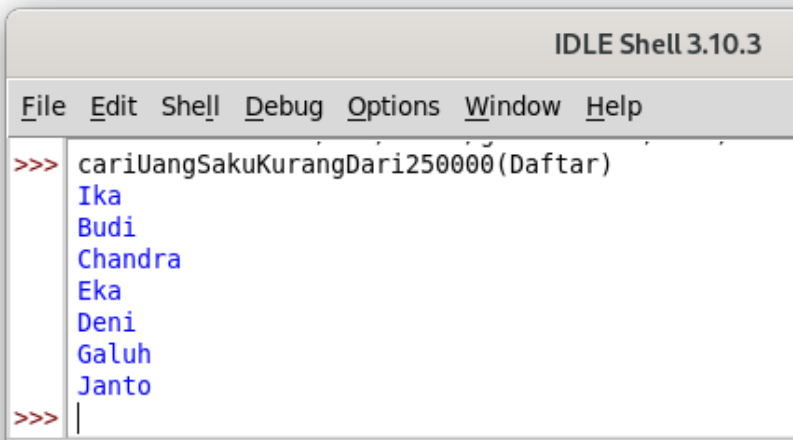
File Edit Shell Debug Options Window Help

```
>>> CariNilaiTerkecilBanyak(Daftar)
[('Budi', 51, 'Sragen', 230000), ('Chandra', 18, 'Surakarta', 235000)]
>>>
```

No 4

```
from mhs import MhsTIF, Daftar

def cariUangSakuKurangDari250000(list):
    nilai = 250000
    for x in Daftar:
        if x.ambilUangSaku() < nilai:
            print(x.ambilNama())
```



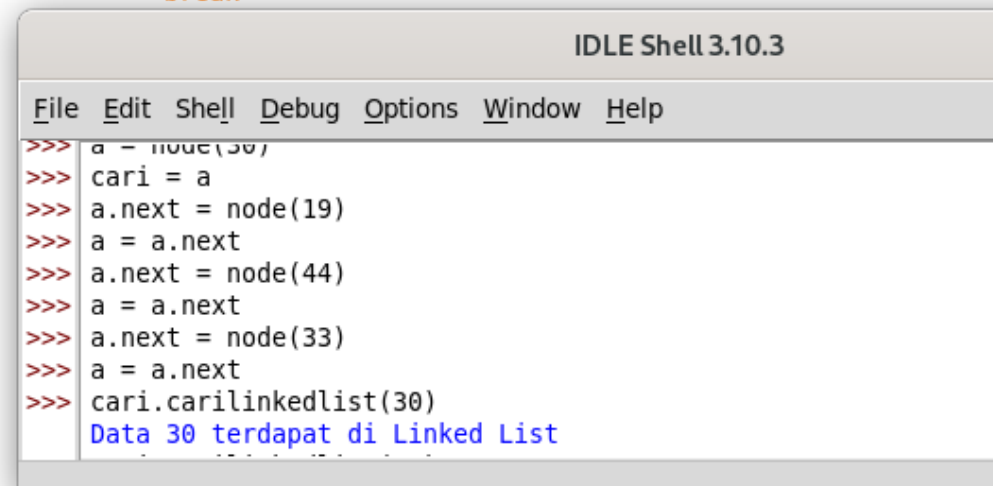
The screenshot shows the IDLE Shell 3.10.3 interface. The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The shell prompt is >>>. The code cariUangSakuKurangDari250000(Daftar) has been executed, resulting in the following output: Ika, Budi, Chandra, Eka, Deni, Galuh, and Janto, each on a new line.

```
IDLE Shell 3.10.3
File Edit Shell Debug Options Window Help
>>> cariUangSakuKurangDari250000(Daftar)
Ika
Budi
Chandra
Eka
Deni
Galuh
Janto
>>> |
```

NO 5

```
class node(object):
    def __init__(self, data, next = None):
        self.data = data
        self.next = next

def carilinkedlist(self, dicari):
    cur = self
    while cur is not None:
        if cur.next != None:
            if cur.data != dicari:
                cur = cur.next
            else:
                print("Data", dicari, "terdapat di Linked List")
                break
        elif cur.next == None:
            print("Data", dicari, "terdapat di Linked List")
            break
```



The screenshot shows the IDLE Shell 3.10.3 interface. The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The shell prompt is >>>. The code creates a linked list with nodes containing values 30, 19, 44, 33, and 19. Then, the carilinkedlist method is called with the value 30 as an argument. The output is "Data 30 terdapat di Linked List".

```
IDLE Shell 3.10.3
File Edit Shell Debug Options Window Help
>>> a = node(30)
>>> cari = a
>>> a.next = node(19)
>>> a = a.next
>>> a.next = node(44)
>>> a = a.next
>>> a.next = node(33)
>>> a = a.next
>>> cari.carilinkedlist(30)
Data 30 terdapat di Linked List
```

No 6

```
List = [28,53,50,40,10,43,45,64,16]
```

```
def binSel(kumpulan,target):
    low = 0
    high = len(kumpulan) - 1

    while low <= high:
        mid = (high + low) // 2
        if kumpulan[mid] == target:
            return "Terdapat di index ke-" + str(mid)
            break
        elif target < kumpulan[mid]:
            high = mid - 1
        else:
            low = mid + 1
    return False
```

IDLE Shell 3.10.3

File Edit Shell Debug Options Window Help

```
>>> binSel(List,10)
'Terdapat di index ke-4'
>>> binSel(List,22)
False
```

No 7

```
List = [2, 3, 4, 4, 5, 6, 6, 6, 8, 9, 9, 10,11,11,11,11, 12, 12,12,13,13, 13,14, 14,15]
```

```
def binSe2(kumpulan, target):
    low = 0
    high = len(kumpulan) -1
    a = []

    while low <= high:
        if kumpulan[low] == target:
            a.append(low)
            low += 1
        else:
            low += 1
    return a
```

IDLE Shell 3.10.3

-

File Edit Shell Debug Options Window Help

```
>>> binSe2(List,11)
[12, 13, 14, 15]
>>>
```

```

print(
    """meggunakan dua pola
    pertama menggunakan konsep Big-O yang dipakai
    rumus  $O(\log n)$  dengan  $1 = 1$ ,  $2 = 2$ ,  $4 = 3$ ,  $10 = 4$ ,  $100 = 7$ ,  $1000=10$ .
    log berasal dari pangkat log 2.
    Untuk pola sendiri:
        jika ingin menebak angka 70
        a = tebakan pertama // 2
        tebakan selanjutnya = nilai tebakan "lebih dari" + a
        *jika hasil tebakan selanjutnya "kurang dari", maka nilai yang digunakan
        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 nilai itu"
        tebakan ke 3: 62 (dari 50 + 12) jawaban = "lebih dari nilai itu"
        tebakan ke 4: 68 (dari 62 + 6) jawaban = "lebih dari nilai itu"
        tebakan ke 5: 71 (dari 68 + 3) jawaban = "kurang dari nilai itu"
        tebakan ke 6: 69 (dari 68 + 1) jawaban = "lebih dari nilai itu"
        tebakan ke 7: antara 71 dan 69 hanya ada 1 angka = 70!!!

    menggunakan barisan geometri  $S_n = 2^n$ 
        barisan yang terjadi adalah : 2, 4, 8, 16, 32, 64
        Misal angka yang akan diebak adalah 68
        Tebakan ke-1 : 64 dijawab "lebih dari niali itu"
        Tebakan ke-2 : 96(dari 64 + 32) dijawab "Kurang dari nilai itu"
        Tebakan ke-3 : 80(dari 64 + 16) dijawab "Kurang dari nilai itu"
        Tebakan ke-4 : 72(dari 64 + 8) dijawab "Kurang dari nilai itu"
        Tebakan ke-5 : 68(dari 64 + 4) dijawab "Lebih dari nilai itu"
        Tebakan ke-6 : 70(dari 68 + 2) dijawab "TEPAT"
    """)

```