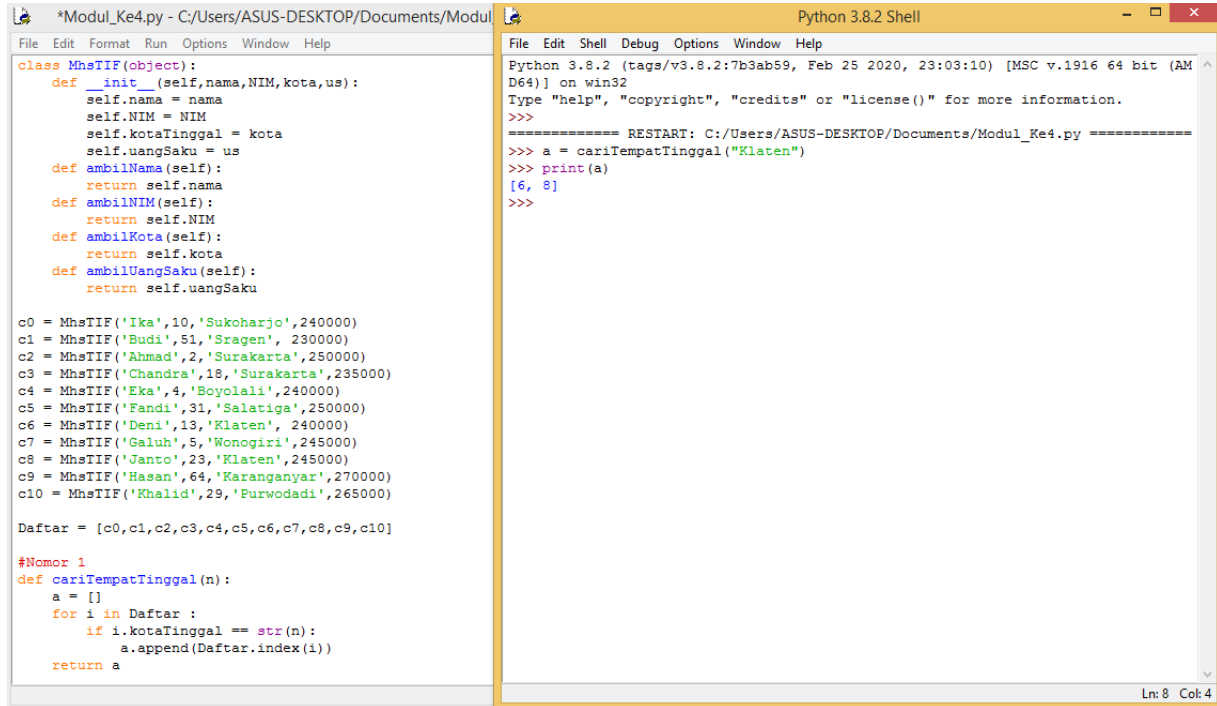


Nama : Anggit Astriani

NIM : L200180111

MODUL 4

Nomor 1



```
*Modul_Ke4.py - C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py
Python 3.8.2 Shell

class MhsTIF(object):
    def __init__(self, nama, NIM, kota, us):
        self.nama = nama
        self.NIM = NIM
        self.kotaTinggal = kota
        self.uangSaku = us
    def ambilNama(self):
        return self.nama
    def ambilNIM(self):
        return self.NIM
    def ambilKota(self):
        return self.kota
    def ambilUangSaku(self):
        return self.uangSaku

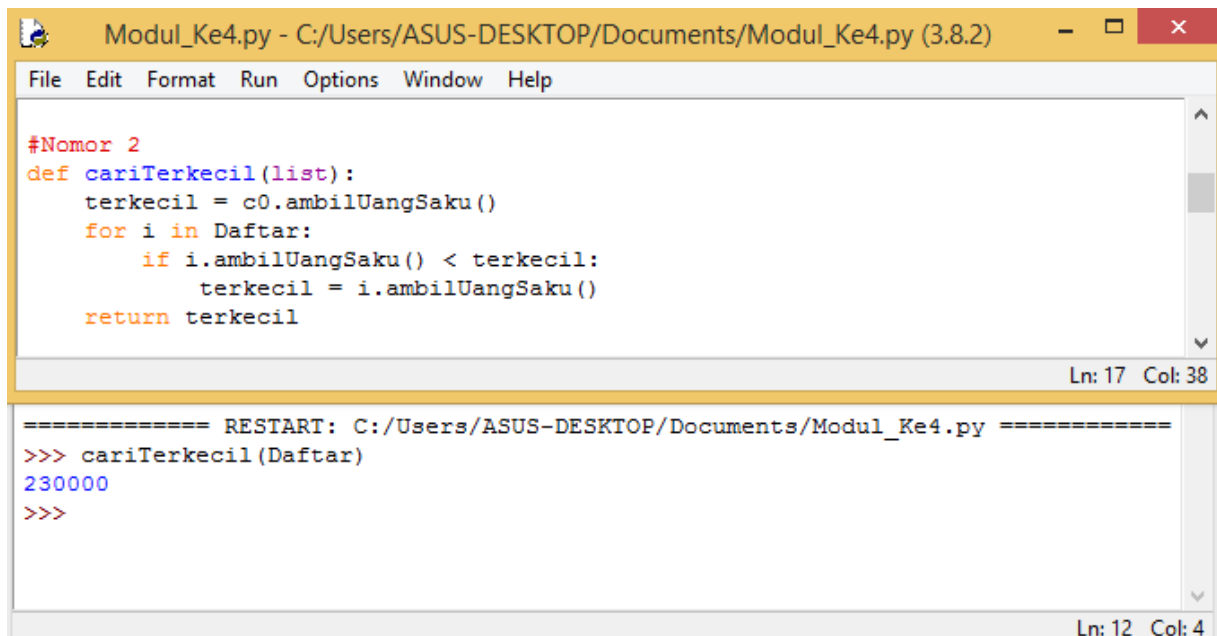
c0 = MhsTIF('Ika', 10, 'Sukoharjo', 240000)
c1 = MhsTIF('Budi', 51, 'Sragen', 230000)
c2 = MhsTIF('Ahmad', 2, 'Surakarta', 250000)
c3 = MhsTIF('Chandra', 18, 'Surakarta', 235000)
c4 = MhsTIF('Eka', 4, 'Boyolali', 240000)
c5 = MhsTIF('Fandi', 31, 'Salatiga', 250000)
c6 = MhsTIF('Deni', 13, 'Klaten', 240000)
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]

#Nomor 1
def cariTempatTinggal(n):
    a = []
    for i in Daftar:
        if i.kotaTinggal == str(n):
            a.append(Daftar.index(i))
    return a

>>> a = cariTempatTinggal("Klaten")
>>> print(a)
[6, 8]
>>>
```

Nomor 2

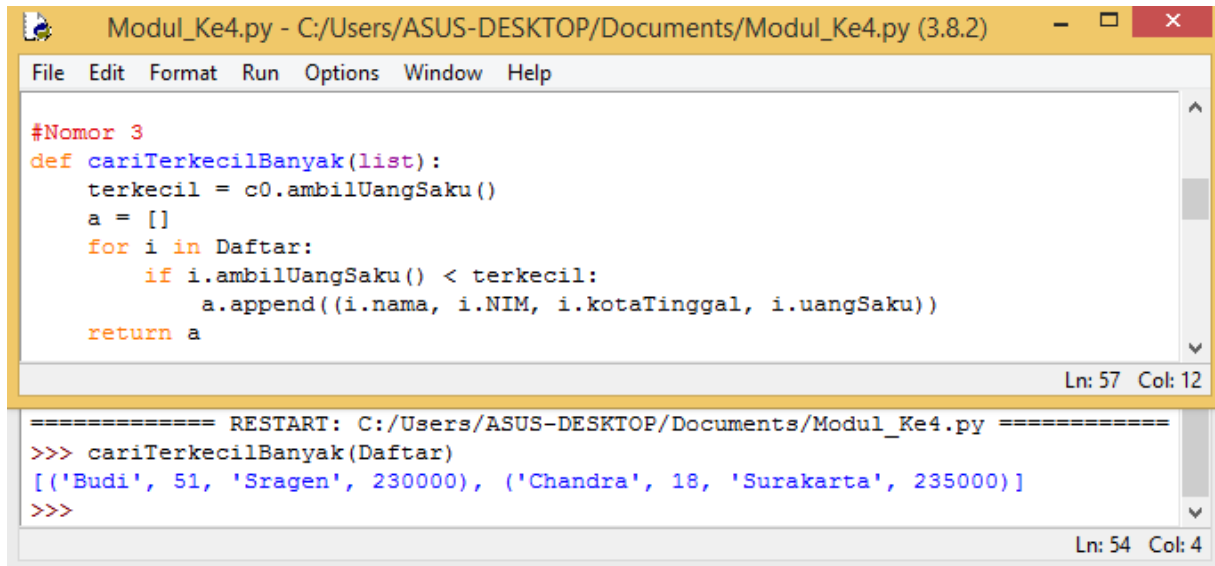


```
Modul_Ke4.py - C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py (3.8.2)

#Nomor 2
def cariTerkecil(list):
    terkecil = c0.ambilUangSaku()
    for i in Daftar:
        if i.ambilUangSaku() < terkecil:
            terkecil = i.ambilUangSaku()
    return terkecil

>>> cariTerkecil(Daftar)
230000
>>>
```

Nomor 3



The screenshot shows a Python IDE window titled "Modul_Ke4.py - C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py (3.8.2)". The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code editor contains the following Python code for "Nomor 3":

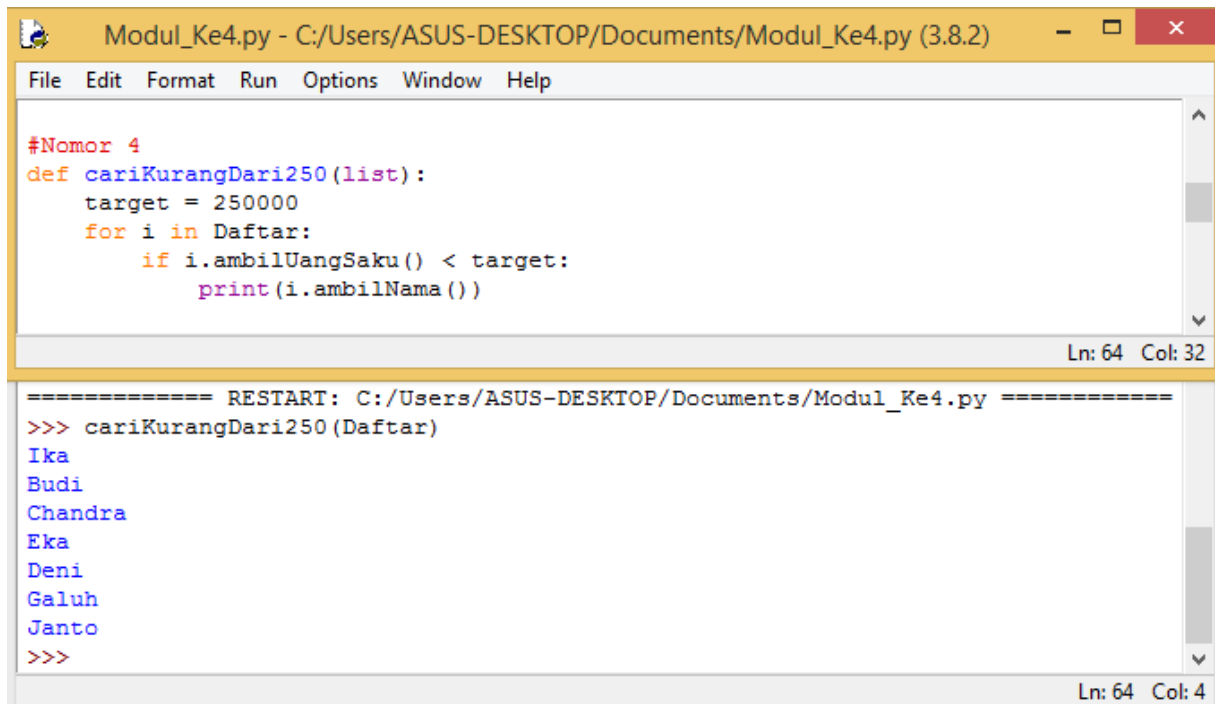
```
#Nomor 3
def cariTerkecilBanyak(list):
    terkecil = c0.ambilUangSaku()
    a = []
    for i in Daftar:
        if i.ambilUangSaku() < terkecil:
            a.append((i.nama, i.NIM, i.kotaTinggal, i.uangSaku))
    return a
```

The status bar at the bottom right of the editor shows "Ln: 57 Col: 12". Below the code editor is a console window showing the execution output:

```
===== RESTART: C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py =====
>>> cariTerkecilBanyak(Daftar)
[('Budi', 51, 'Sragen', 230000), ('Chandra', 18, 'Surakarta', 235000)]
>>>
```

The console window status bar shows "Ln: 54 Col: 4".

Nomor 4



The screenshot shows a Python IDE window titled "Modul_Ke4.py - C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py (3.8.2)". The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code editor contains the following Python code for "Nomor 4":

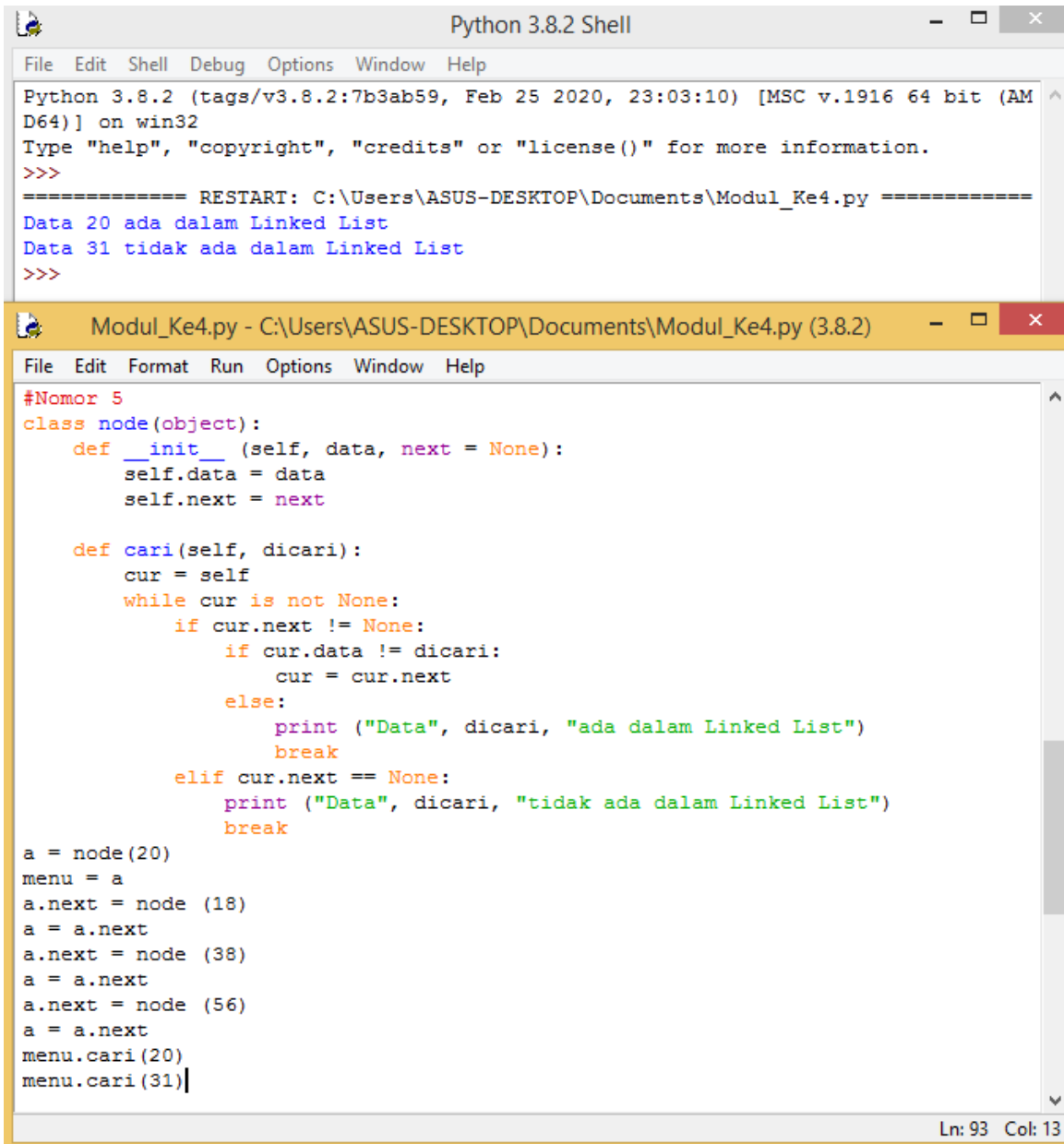
```
#Nomor 4
def cariKurangDari250(list):
    target = 250000
    for i in Daftar:
        if i.ambilUangSaku() < target:
            print(i.ambilNama())
```

The status bar at the bottom right of the editor shows "Ln: 64 Col: 32". Below the code editor is a console window showing the execution output:

```
===== RESTART: C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py =====
>>> cariKurangDari250(Daftar)
Ika
Budi
Chandra
Eka
Deni
Galuh
Janto
>>>
```

The console window status bar shows "Ln: 64 Col: 4".

Nomor 5



The image shows two overlapping windows from a Windows operating system. The top window is titled "Python 3.8.2 Shell" and displays the output of a Python script. The bottom window is a code editor titled "Modul_Ke4.py - C:\Users\ASUS-DESKTOP\Documents\Modul_Ke4.py (3.8.2)" and shows the source code of the script.

Python 3.8.2 Shell Output:

```
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\ASUS-DESKTOP\Documents\Modul_Ke4.py =====
Data 20 ada dalam Linked List
Data 31 tidak ada dalam Linked List
>>>
```

Modul_Ke4.py Source Code:

```
#Nomor 5
class node(object):
    def __init__(self, data, next = None):
        self.data = data
        self.next = next

    def cari(self, dicari):
        cur = self
        while cur is not None:
            if cur.next != None:
                if cur.data != dicari:
                    cur = cur.next
                else:
                    print ("Data", dicari, "ada dalam Linked List")
                    break
            elif cur.next == None:
                print ("Data", dicari, "tidak ada dalam Linked List")
                break

a = node(20)
menu = a
a.next = node (18)
a = a.next
a.next = node (38)
a = a.next
a.next = node (56)
a = a.next
menu.cari(20)
menu.cari(31)
```

Ln: 93 Col: 13

Nomor 6

```
Modul_Ke4.py - C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py (3.8.2)
File Edit Format Run Options Window Help

#Nomor 6
List = [2,3,4,5,8,10,12,43,67,15]

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

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

|
Ln: 135 Col: 0

===== RESTART: C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py =====
>>> binSe(List,10)
'Ada di index ke-5'
>>>
```

Nomor 7

```
Modul_Ke4.py - C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py (3.8.2)
File Edit Format Run Options Window Help

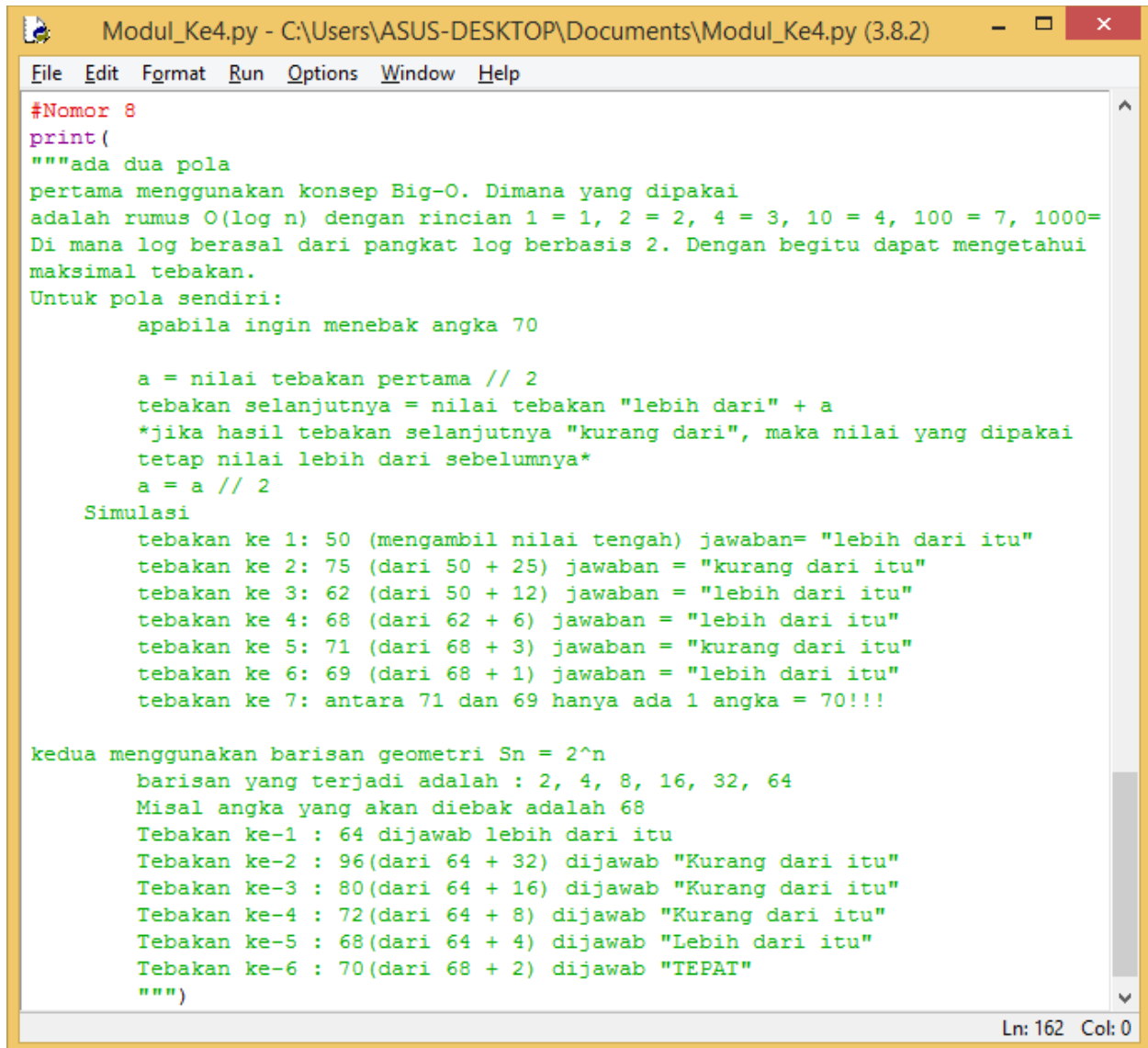
#Nomor 7
Listt = [2, 3, 5, 6, 6, 6, 8, 9, 9, 10, 11, 12, 13, 13, 14]
def binSeLagi(kumpulan, target):
    low = 0
    high = len(kumpulan) -1
    a = []

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

Ln: 150 Col: 0

===== RESTART: C:/Users/ASUS-DESKTOP/Documents/Modul_Ke4.py =====
>>> binSeLagi(Listt,6)
[3, 4, 5]
>>>
Ln: 30 Col: 4
```

Nomor 8



```
#Nomor 8
print(
    """ada dua pola
    pertama menggunakan konsep Big-O. Dimana yang dipakai
    adalah rumus  $O(\log n)$  dengan rincian  $1 = 1$ ,  $2 = 2$ ,  $4 = 3$ ,  $10 = 4$ ,  $100 = 7$ ,  $1000 = 10$ .
    Di mana log berasal dari pangkat log berbasis 2. Dengan begitu dapat mengetahui
    maksimal tebakan.
    Untuk pola sendiri:
        apabila ingin menebak angka 70

        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) jawaban = "lebih dari itu"
        tebakan ke 5: 71 (dari 68 + 3) jawaban = "kurang dari itu"
        tebakan ke 6: 69 (dari 68 + 1) jawaban = "lebih dari itu"
        tebakan ke 7: antara 71 dan 69 hanya ada 1 angka = 70!!!

    kedua 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 itu
        Tebakan ke-2 : 96(dari 64 + 32) dijawab "Kurang dari itu"
        Tebakan ke-3 : 80(dari 64 + 16) dijawab "Kurang dari itu"
        Tebakan ke-4 : 72(dari 64 + 8) dijawab "Kurang dari itu"
        Tebakan ke-5 : 68(dari 64 + 4) dijawab "Lebih dari itu"
        Tebakan ke-6 : 70(dari 68 + 2) dijawab "TEPAT"
    """)
```

Ln: 162 Col: 0