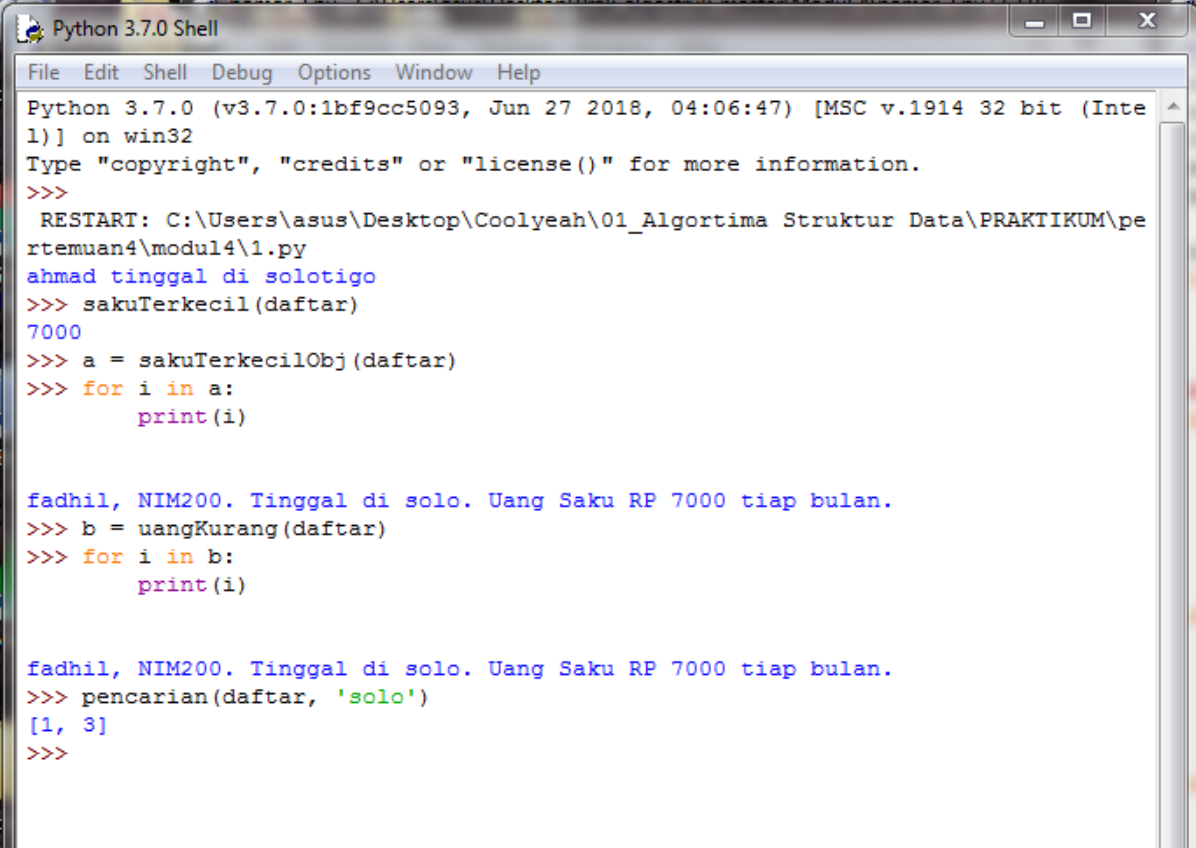


Muhammad Fadhil Bariz Ardanto
L200180200 / Kelas G

Tugas Modul 4

Hasil Kode

1 – 4

A screenshot of a Python 3.7.0 Shell window. The window has a title bar that says "Python 3.7.0 Shell" and standard Windows window controls (minimize, maximize, close). Below the title bar is a menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main area of the window contains the following text:

```
Python 3.7.0 (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: C:\Users\asus\Desktop\Coolyeh\01_Algortima Struktur Data\PRAKTIKUM\pertemuan4\modul4\1.py
ahmad tinggal di solotigo
>>> sakuTerkecil(daftar)
7000
>>> a = sakuTerkecilObj(daftar)
>>> for i in a:
    print(i)

fadhil, NIM200. Tinggal di solo. Uang Saku RP 7000 tiap bulan.
>>> b = uangKurang(daftar)
>>> for i in b:
    print(i)

fadhil, NIM200. Tinggal di solo. Uang Saku RP 7000 tiap bulan.
>>> pencarian(daftar, 'solo')
[1, 3]
>>>
```

Kode

```
1.py - C:\Users\asus\Desktop\Coolyeah\01_Algoritma Struktur Data\PRAKTIKUM\pertemuan4\modul4\1.py (3.7.0)
File Edit Format Run Options Window Help

class mhsTif(object):
    """docstring for mhsTif"""
    def __init__(self, nama, nim, kota, saku):
        self.nama = nama
        self.nim = nim
        self.kota = kota
        self.saku = saku
    def __str__(self):
        s = self.nama + ', NIM' + str(self.nim) \
            + '. Tinggal di ' + self.kota \
            + '. Uang Saku RP ' + str(self.saku) \
            + ' tiap bulan. '
        return s
c0 = mhsTif('ika', 100, 'sukoharjo', 290000)
c1 = mhsTif('ikal', 180, 'solo', 299000)
c2 = mhsTif('ahmad', 230, 'solotigo', 2900800)
c3 = mhsTif('radhil', 200, 'solo', 7000)
daftar = [c0, c1, c2, c3]
daftar1 = [1,2,3,4,5,6,7,8,9]
target = 'solotigo'
for i in daftar:
    if i.kota == target:
        print(i.nama + ' tinggal di ' +target )

#No.1
def pencarian(list, target):
    hasil = []
    for i in list:
        if i.kota == target:
            hasil.append(list.index(i))
    print(hasil)

#No.2
def sakuTerkecil(list):
    terkecil = 9999999
    for i in list:
        if i.saku < terkecil:
            terkecil = i.saku
    return terkecil

#No.3
def sakuTerkecilObj(obj):
    temp = [obj[0]]
    for i in obj:
        if i.saku < temp[0].saku:
            temp = [i]
        elif i.saku == temp[0].saku:
            temp.append(i)
    return temp

#No.4
def uangKurang(o):
    temp = []
    for i in o:
        if i.saku < 250000:
            temp.append(i)
    return temp
```

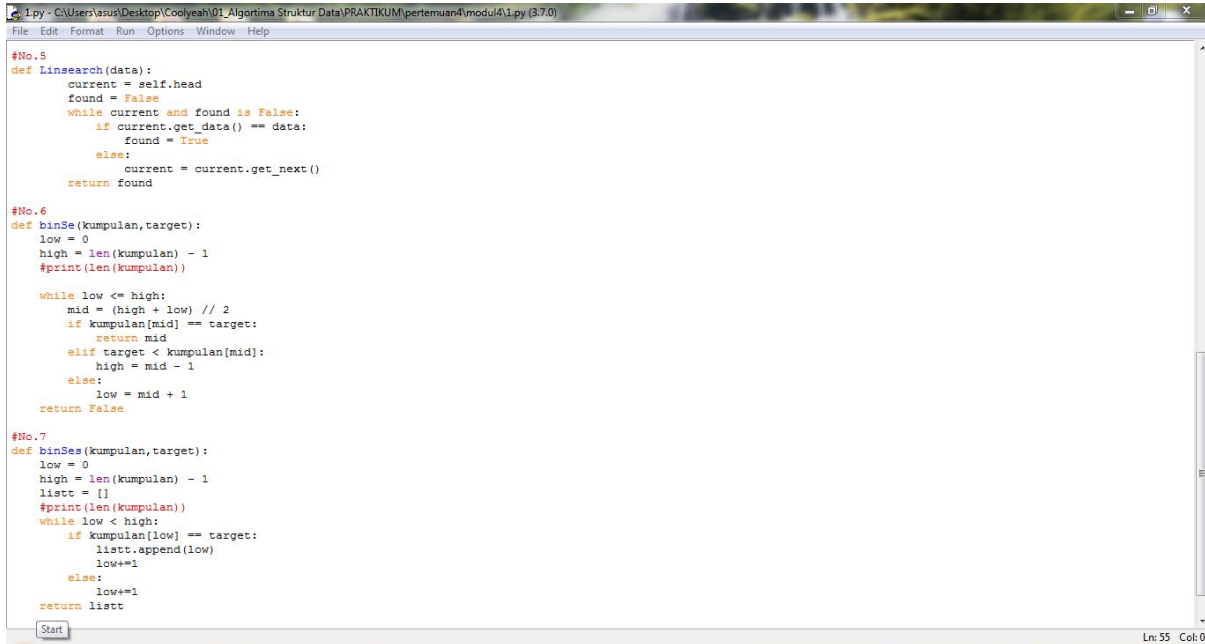
Ln: 47 Col: 5

No. 5-7

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help

Python 3.7.0 (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: C:\Users\asus\Desktop\Coolyeah\01_Algoritma Struktur Data\PRAKTIKUM\pertemuan4\modul4\1.py
ahmad tinggal di solotigo
>>> binSe(daftar1,0)
False
>>> binSe(daftar1,9)
8
>>> binSes(daftar1,9)
[]
>>> binSes(daftar2,9)
[8, 9]
>>> |
```

Kode



```
#No.5
def Linsearch(data):
    current = self.head
    found = False
    while current and found is False:
        if current.get_data() == data:
            found = True
        else:
            current = current.get_next()
    return found

#No.6
def binSe(kumpulan, target):
    low = 0
    high = len(kumpulan) - 1
    #print(len(kumpulan))

    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

#No.7
def binSeS(kumpulan, target):
    low = 0
    high = len(kumpulan) - 1
    listt = []
    #print(len(kumpulan))
    while low < high:
        if kumpulan[low] == target:
            listt.append(low)
            low+=1
        else:
            low+=1
    return listt
```

No.8

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 jumlah 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