

Nama : Anang Prasetyo

NIM : L200180063

Kelas : C

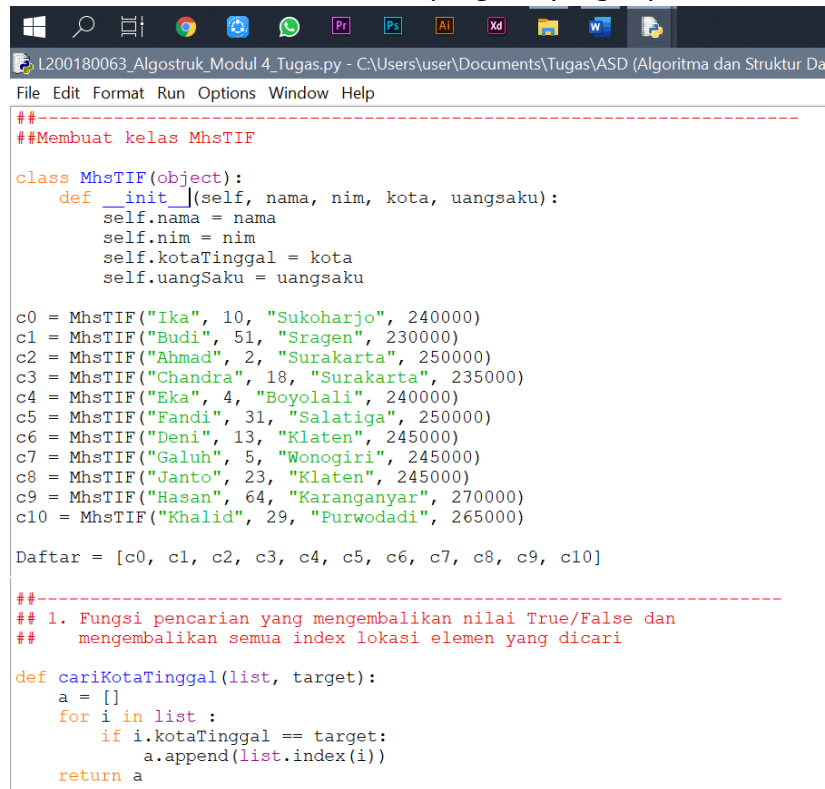
Modul 4 - Pencarian

Tugas

Soal-soal untuk mahasiswa

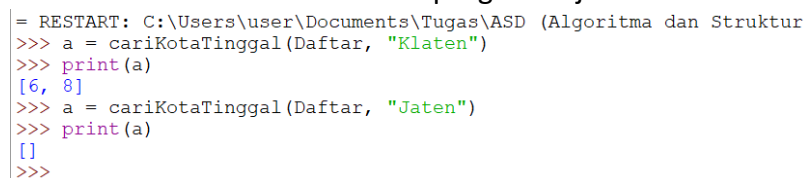
1. Membuat suatu fungsi pencarian yang mengembalikan nilai True/False dan mengembalikan semua index lokasi elemen yang dicari pada kelas MhsTIF

Berikut adalah screenshoot dari 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 nilai True/False dan  
##      mengembalikan semua index lokasi elemen yang dicari  
  
def cariKotaTinggal(list, target):  
    a = []  
    for i in list :  
        if i.kotaTinggal == target:  
            a.append(list.index(i))  
    return a
```

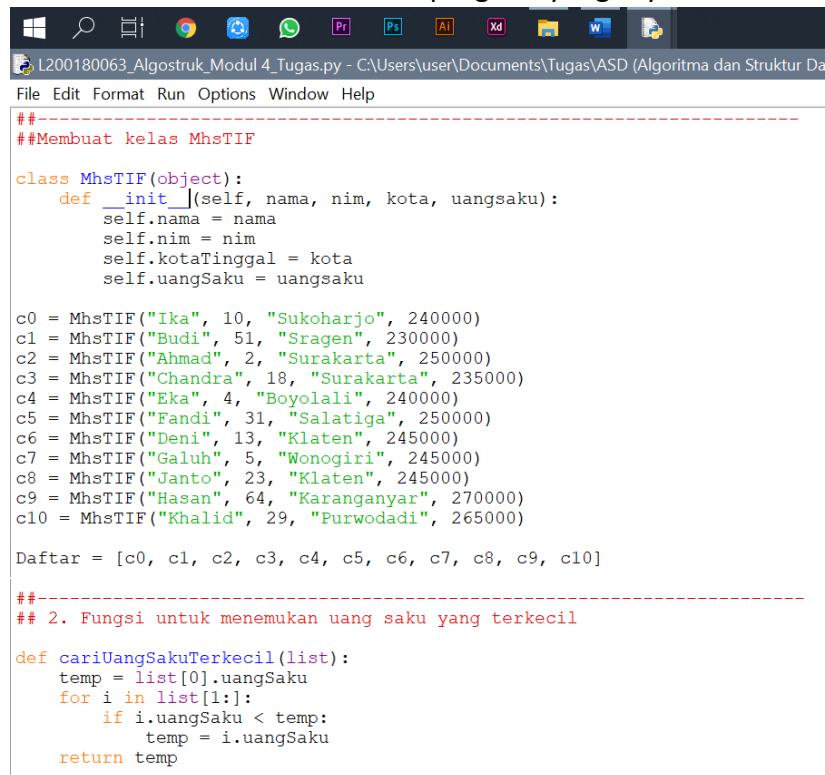
Berikut adalah screenshoot saat program dijalankan:



```
= RESTART: C:\Users\user\Documents\Tugas\ASD (Algoritma dan Struktur  
>>> a = cariKotaTinggal(Daftar, "Klaten")  
>>> print(a)  
[6, 8]  
>>> a = cariKotaTinggal(Daftar, "Jaten")  
>>> print(a)  
[]  
>>>
```

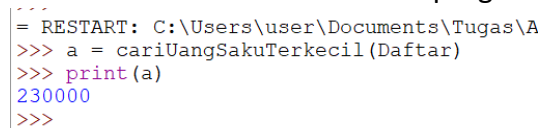
2. Membuat suatu fungsi untuk menemukan uang saku yang terkecil dari kelas MhsTIF

Berikut adalah screenshoot dari 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
```

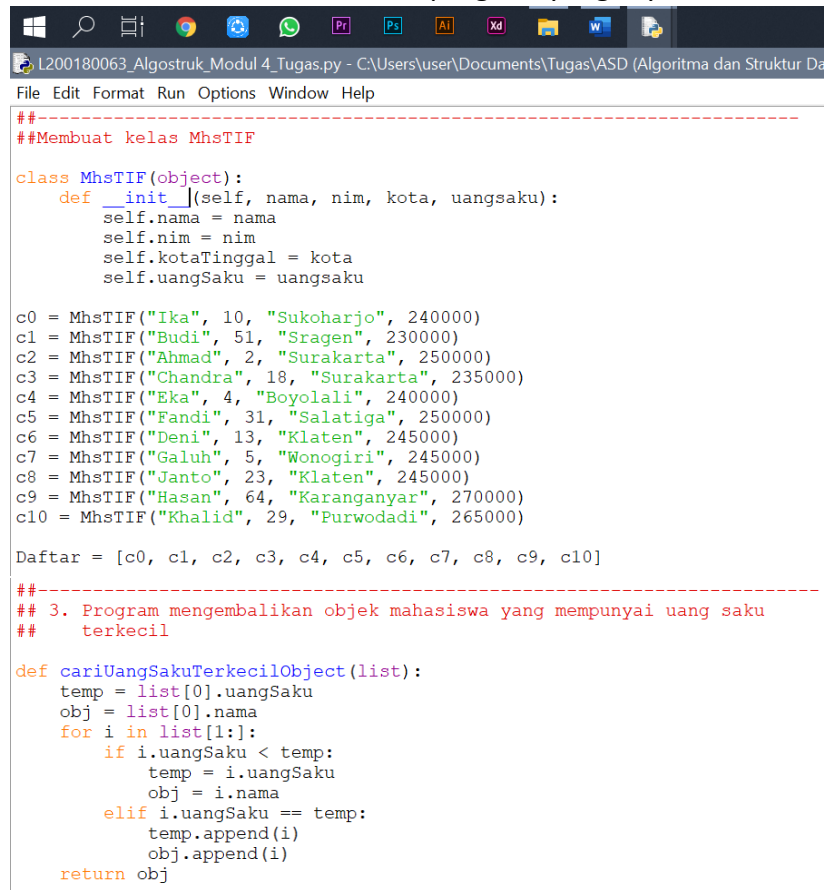
Berikut adalah screenshoot saat program dijalankan:



```
>>> RESTART: C:\Users\user\Documents\Tugas\A  
>>> a = cariUangSakuTerkecil(Daftar)  
>>> print(a)  
230000  
>>>
```

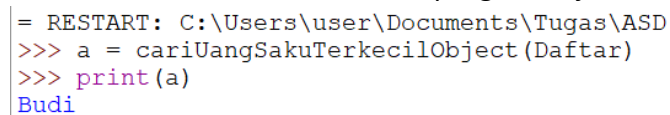
3. Mengubah program no 2 agar mengembalikan objek mahasiswa yang mempunyai uang saku terkecil

Berikut adalah screenshoot dari 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
```

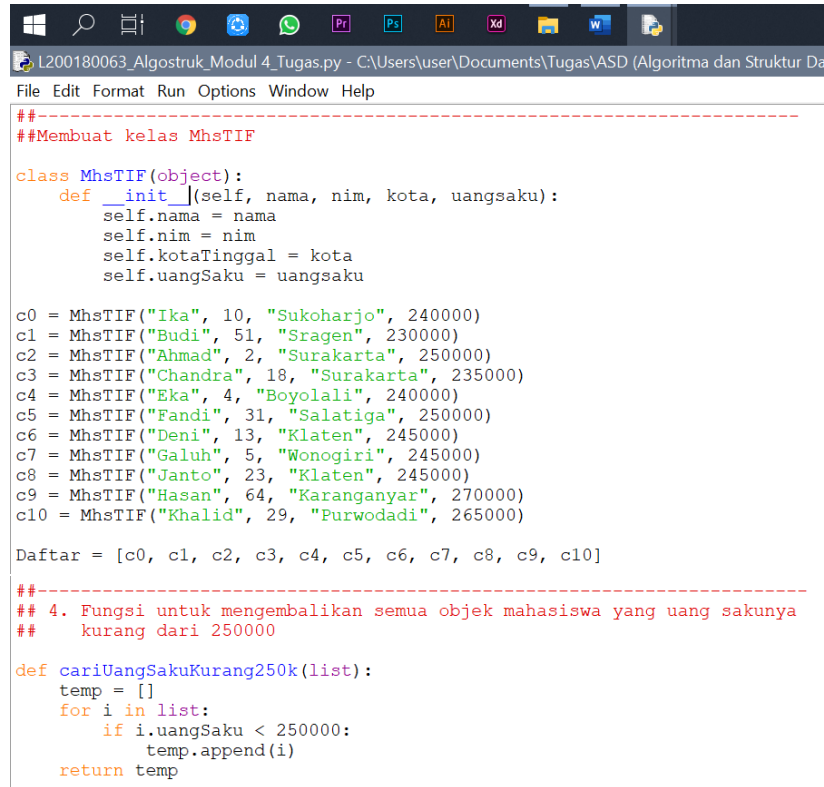
Berikut adalah screenshoot saat program dijalankan:



```
= RESTART: C:\Users\user\Documents\Tugas\ASD  
>>> a = cariUangSakuTerkecilObject(Daftar)  
>>> print(a)  
Budi
```

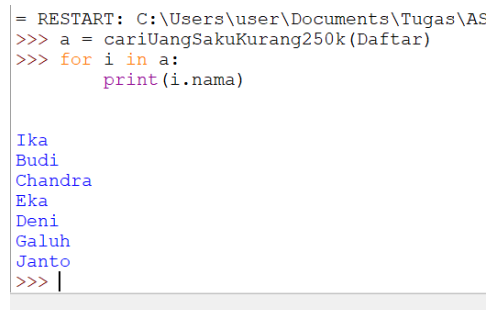
4. Membuat suatu fungsi untuk mengembalikan semua objek mahasiswa yang uang sakunya kurang dari 250000

Berikut adalah screenshot dari 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
```

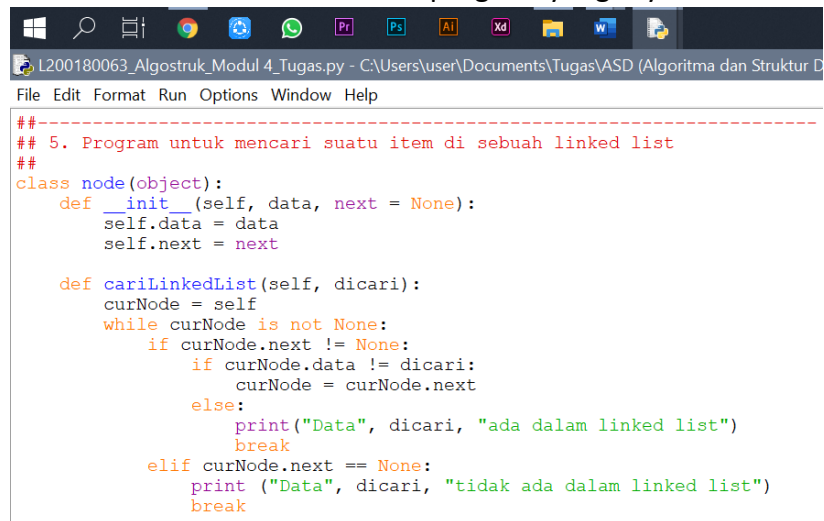
Berikut adalah screenshot saat program dijalankan:



```
= RESTART: C:\Users\user\Documents\Tugas\AS  
>>> a = cariUangSakuKurang250k(Daftar)  
>>> for i in a:  
    print(i.nama)  
  
Ika  
Budi  
Chandra  
Eka  
Deni  
Galuh  
Janto  
>>> |
```

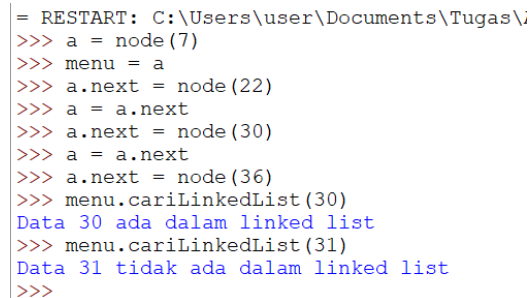
5. Membuat suatu program untuk mencari suatu item di sebuah linked list

Berikut adalah screenshoot dari 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
```

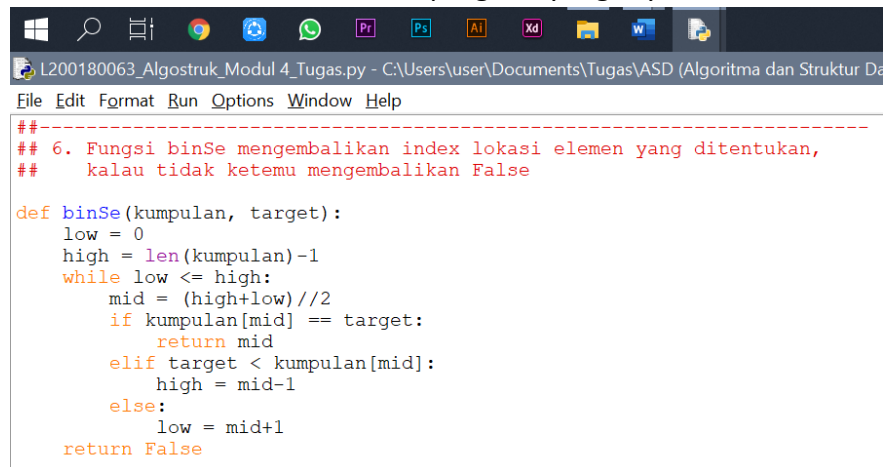
Berikut adalah screenshoot saat program dijalankan:



```
= RESTART: C:\Users\user\Documents\Tugas\  
>>> a = node(7)  
>>> menu = a  
>>> a.next = node(22)  
>>> a = a.next  
>>> a.next = node(30)  
>>> a = a.next  
>>> a.next = node(36)  
>>> menu.cariLinkedList(30)  
Data 30 ada dalam linked list  
>>> menu.cariLinkedList(31)  
Data 31 tidak ada dalam linked list  
>>>
```

6. Mengubah fungsi binSe pada halaman 43 agar mengembalikan index lokasi elemen yang ditentukan, kalau tidak ketemu mengembalikan False

Berikut adalah screenshoot dari program yang saya buat:



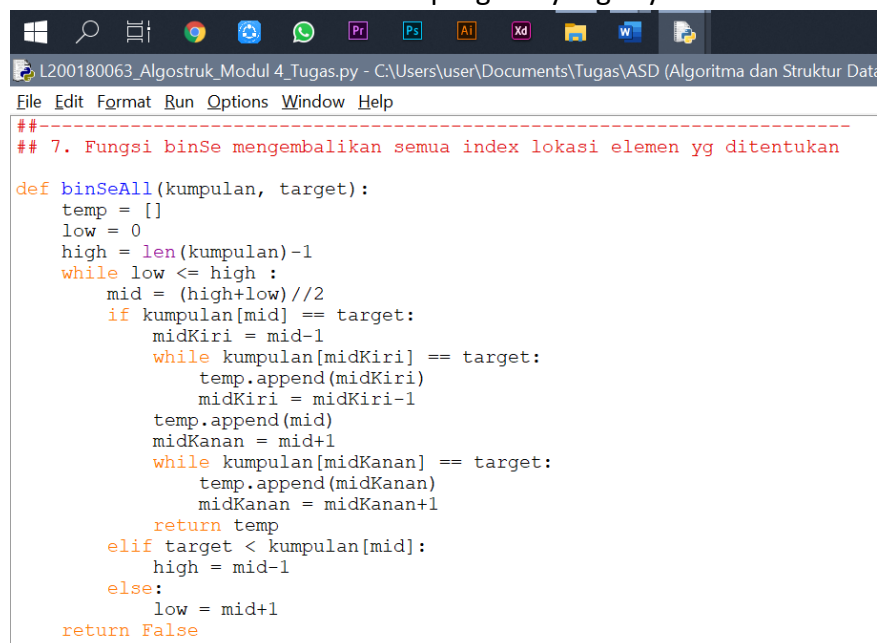
```
##-----  
## 6. Fungsi binSe mengembalikan index lokasi elemen yang ditentukan,  
##    kalau tidak ketemu mengembalikan False  
  
def binSe(kumpulan, target):  
    low = 0  
    high = len(kumpulan)-1  
    while low <= high:  
        mid = (high+low)//2  
        if kumpulan[mid] == target:  
            return mid  
        elif target < kumpulan[mid]:  
            high = mid-1  
        else:  
            low = mid+1  
    return False
```

Berikut adalah screenshoot saat program dijalankan:

```
= RESTART: C:\Users\user\Documents\Tugas\ASD (Algoritma dan  
>>> kumpulan = [2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]  
>>> print(binSe(kumpulan, 31))  
8  
>>> print(binSe(kumpulan, 9))  
False  
>>>
```

7. Mengubah fungsi binSe agar mengembalikan semua index lokasi elemen yang ditentukan

Berikut adalah screenshoot dari program yang saya buat:



```
##-----  
## 7. Fungsi binSe mengembalikan semua index lokasi elemen yg ditentukan  
  
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 saat program dijalankan:

```
= RESTART: C:\Users\user\Documents\Tugas\ASD (Algoritma dan Struktur)
>>> 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, 13))
[12, 13]
>>> print(binSeAll(kumpulan, 7))
False
>>>
```

8. Pada permainan tebak angka yang sudah saya 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 ?

Ada dua kemungkinan pola bisa digunakan.

Misalkan, angka yang akan ditebak adalah 70.

- Pola pertama $a = \text{nilai tebakan pertama} // 2$ 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

- 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 => Jawaban = “Lebih dari itu”
Tebakan ke-2 : 96 (dari 64 + 32) => Jawaban = “Kurang dari itu”
Tebakan ke-3 : 80 (dari 64 + 16) => Jawaban = “Kurang dari itu”
Tebakan ke-4 : 72 (dari 64 + 8) => Jawaban = “Kurang dari itu”
Tebakan ke-5 : 68 (dari 64 + 4) => Jawaban = “Lebih dari itu”
Tebakan ke-6 : 70 (dari 68 + 2) => Jawaban = “TEPAT”