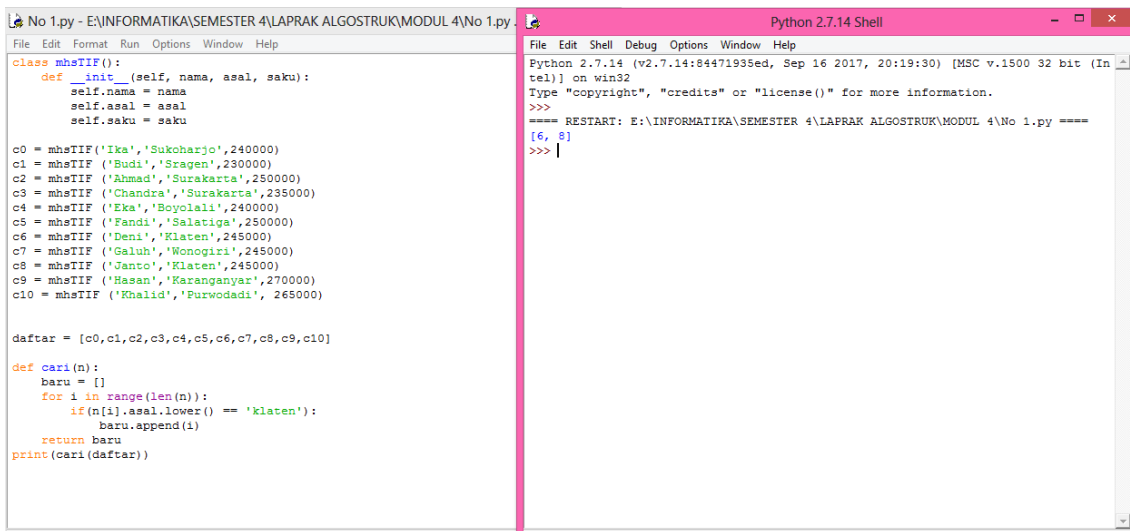


Nama : Sukma Nindi Listyarini
Kelas : D
NIM : L200170147

Modul 4

Laporan Praktikum - Algoritma dan Struktur Data

1. Membuat fungsi pencarian dengan mengembalikan semua index lokasi elemen yang dicari.



```
class mhsTIF():
    def __init__(self, nama, asal, saku):
        self.nama = nama
        self.asal = asal
        self.saku = saku

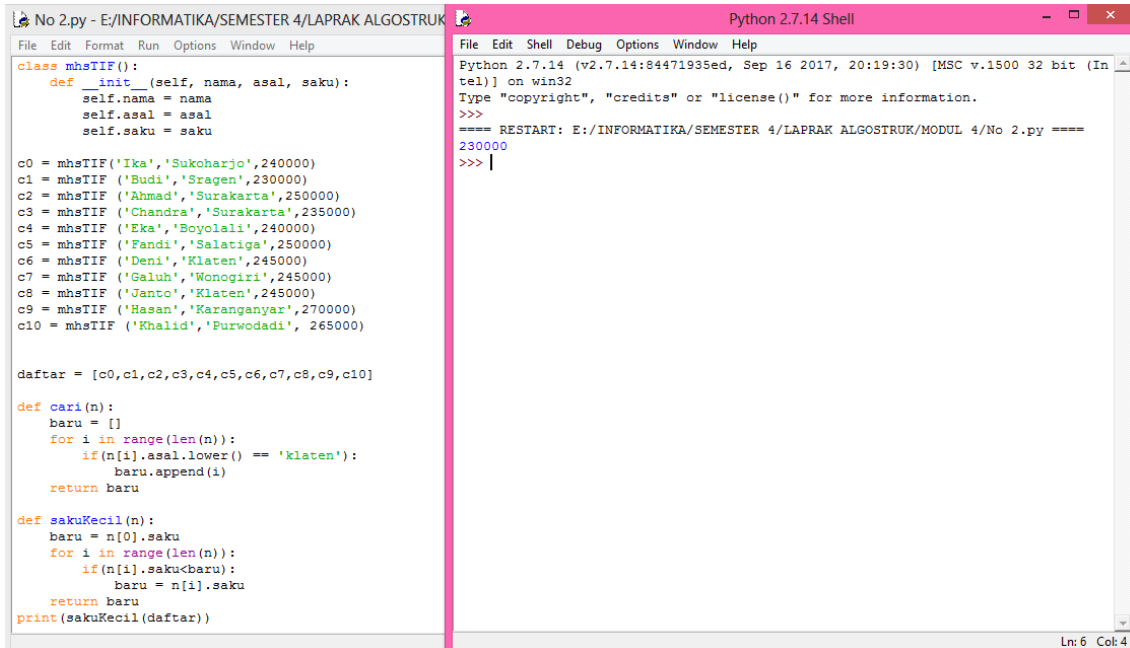
c0 = mhsTIF('Ika','Sukoharjo',240000)
c1 = mhsTIF('Budi','Sragen',230000)
c2 = mhsTIF('Ahmad','Surakarta',250000)
c3 = mhsTIF('Chandra','Surakarta',235000)
c4 = mhsTIF('Eka','Boyolali',240000)
c5 = mhsTIF('Fandi','Salatiga',250000)
c6 = mhsTIF('Deni','Klaten',245000)
c7 = mhsTIF('Galuh','Wonogiri',245000)
c8 = mhsTIF('Janto','Klaten',245000)
c9 = mhsTIF('Hasan','Karanganyar',270000)
c10 = mhsTIF('Khalid','Purwodadi', 265000)

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

def cari(n):
    baru = []
    for i in range(len(n)):
        if n[i].asal.lower() == 'klaten':
            baru.append(i)
    return baru
print(cari(daftar))
```

```
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: E:\INFORMATIKA\SEMESTER 4\LAPRAK ALGOSTRUK\MODUL 4\No 1.py ====
[6, 8]
>>>
```

2. Membuat fungsi untuk menemukan uang saku terkecil yang ada diantara daftar



```
class mhsTIF():
    def __init__(self, nama, asal, saku):
        self.nama = nama
        self.asal = asal
        self.saku = saku

c0 = mhsTIF('Ika','Sukoharjo',240000)
c1 = mhsTIF('Budi','Sragen',230000)
c2 = mhsTIF('Ahmad','Surakarta',250000)
c3 = mhsTIF('Chandra','Surakarta',235000)
c4 = mhsTIF('Eka','Boyolali',240000)
c5 = mhsTIF('Fandi','Salatiga',250000)
c6 = mhsTIF('Deni','Klaten',245000)
c7 = mhsTIF('Galuh','Wonogiri',245000)
c8 = mhsTIF('Janto','Klaten',245000)
c9 = mhsTIF('Hasan','Karanganyar',270000)
c10 = mhsTIF('Khalid','Purwodadi', 265000)

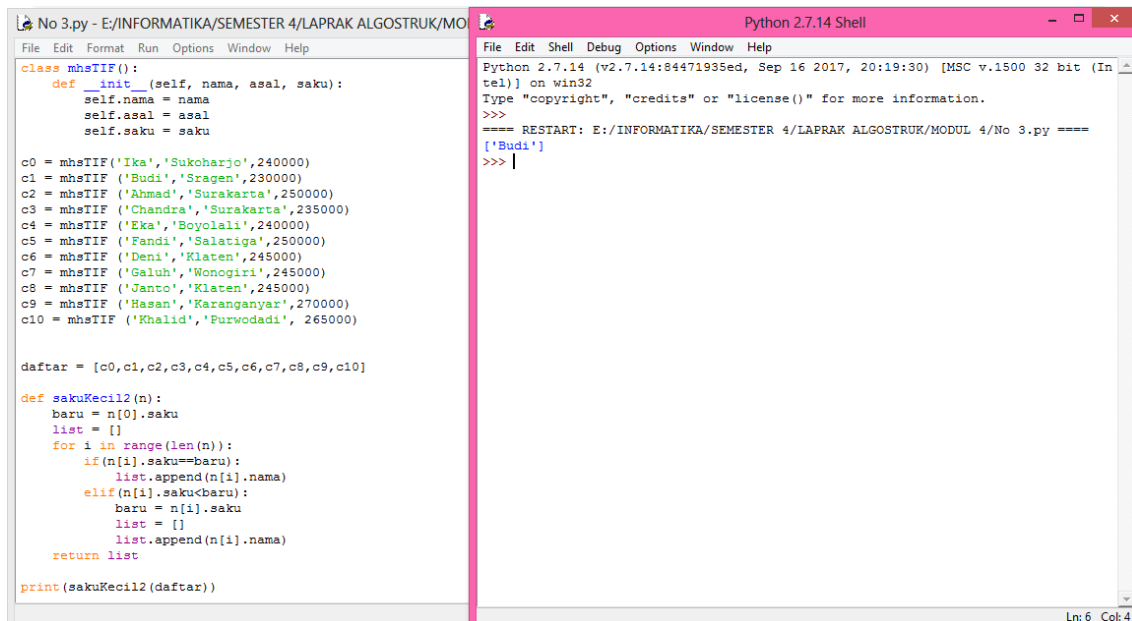
daftar = [c0,c1,c2,c3,c4,c5,c6,c7,c8,c9,c10]

def cari(n):
    baru = []
    for i in range(len(n)):
        if n[i].asal.lower() == 'klaten':
            baru.append(i)
    return baru

def sakuKecil(n):
    baru = n[0].saku
    for i in range(len(n)):
        if n[i].saku < baru:
            baru = n[i].saku
    return baru
print(sakuKecil(daftar))
```

```
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: E:\INFORMATIKA\SEMESTER 4\LAPRAK ALGOSTRUK\MODUL 4\No 2.py ====
230000
>>>
```

3. Mengubah program agar dapat mengembalikan object mahasiswa yang mempunyai uang saku terkecil.



The screenshot shows a Python IDE with two windows. The left window, titled 'No 3.py', contains the following code:

```
class mhsTIF():
    def __init__(self, nama, asal, saku):
        self.nama = nama
        self.asal = asal
        self.saku = saku

c0 = mhsTIF('Ika', 'Sukoharjo', 240000)
c1 = mhsTIF('Budi', 'Sragen', 230000)
c2 = mhsTIF('Ahmad', 'Surakarta', 250000)
c3 = mhsTIF('Chandra', 'Surakarta', 235000)
c4 = mhsTIF('Eka', 'Boyolali', 240000)
c5 = mhsTIF('Fandi', 'Salatiga', 250000)
c6 = mhsTIF('Deni', 'Klaten', 245000)
c7 = mhsTIF('Galuh', 'Wonogiri', 245000)
c8 = mhsTIF('Janto', 'Klaten', 245000)
c9 = mhsTIF('Hasan', 'Karanganyar', 270000)
c10 = mhsTIF('Khalid', 'Purwodadi', 265000)

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

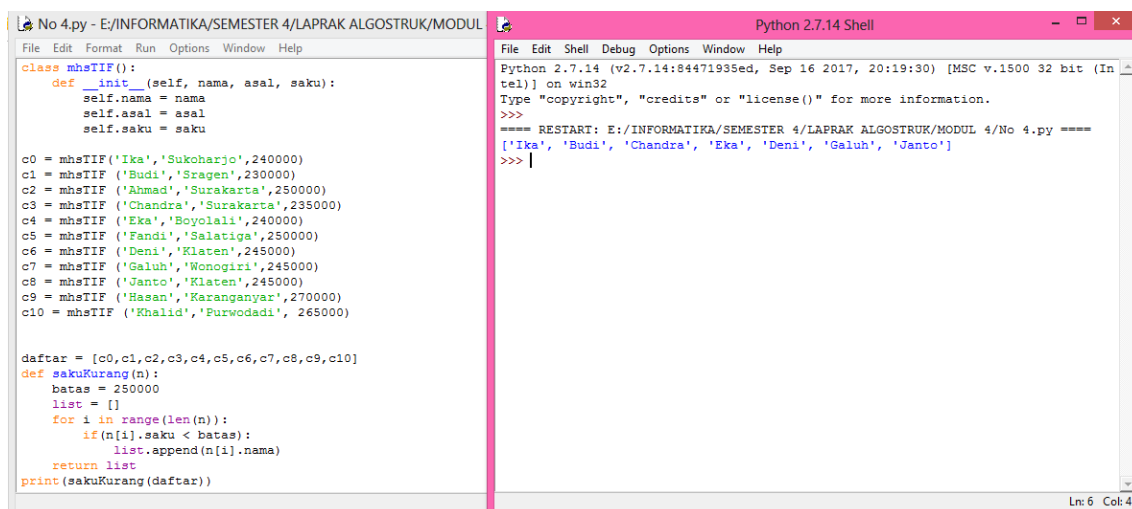
def sakuKecil2(n):
    baru = n[0].saku
    list = []
    for i in range(len(n)):
        if n[i].saku < baru:
            list.append(n[i].nama)
        elif n[i].saku == baru:
            baru = n[i].saku
            list = []
            list.append(n[i].nama)
    return list

print(sakuKecil2(daftar))
```

The right window, titled 'Python 2.7.14 Shell', shows the execution output:

```
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: E:/INFORMATIKA/SEMESTER 4/LAPRAK ALGOSTRUK/MODUL 4/No 3.py ====
['Budi']
>>>
```

4. Membuat suatu fungsi untuk mengembalikan semua objek mahasiswa yang uang sakunya kurang dari 25000.



The screenshot shows a Python IDE with two windows. The left window, titled 'No 4.py', contains the following code:

```
class mhsTIF():
    def __init__(self, nama, asal, saku):
        self.nama = nama
        self.asal = asal
        self.saku = saku

c0 = mhsTIF('Ika', 'Sukoharjo', 240000)
c1 = mhsTIF('Budi', 'Sragen', 230000)
c2 = mhsTIF('Ahmad', 'Surakarta', 250000)
c3 = mhsTIF('Chandra', 'Surakarta', 235000)
c4 = mhsTIF('Eka', 'Boyolali', 240000)
c5 = mhsTIF('Fandi', 'Salatiga', 250000)
c6 = mhsTIF('Deni', 'Klaten', 245000)
c7 = mhsTIF('Galuh', 'Wonogiri', 245000)
c8 = mhsTIF('Janto', 'Klaten', 245000)
c9 = mhsTIF('Hasan', 'Karanganyar', 270000)
c10 = mhsTIF('Khalid', 'Purwodadi', 265000)

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

def sakuKurang(n):
    batas = 250000
    list = []
    for i in range(len(n)):
        if n[i].saku < batas:
            list.append(n[i].nama)
    return list

print(sakuKurang(daftar))
```

The right window, titled 'Python 2.7.14 Shell', shows the execution output:

```
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: E:/INFORMATIKA/SEMESTER 4/LAPRAK ALGOSTRUK/MODUL 4/No 4.py ====
['Ika', 'Budi', 'Chandra', 'Eka', 'Deni', 'Galuh', 'Janto']
>>>
```

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

The image shows two windows from a Python 2.7.14 IDE. The left window, titled 'No 5.py', contains the source code for a linked list. It defines a `Node` class with `data` and `next` attributes, and a `LinkedList` class with methods `pushAw`, `search`, and `display`. The `search` method iterates through the list to find a specific value. The right window, titled 'Python 2.7.14 Shell', shows the execution output. It displays the restart path and the results of the `search` method for values 21 and 29.

```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None
class LinkedList:
    def __init__(self):
        self.head = None
    def pushAw(self, new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        self.head = new_node
        return self.head
    def search(self, x):
        current = self.head
        while current != None:
            if current.data == x:
                return "True"
            current = current.next
        return "False"
    def display(self):
        current = self.head
        while current is not None:
            print(current.data, end == ' ')
            current = current.next

l1ist = LinkedList()
l1ist.pushAw(21)
l1ist.pushAw(22)
l1ist.pushAw(12)
l1ist.pushAw(14)
l1ist.pushAw(2)
l1ist.pushAw(19)

print(l1ist.search(21))
print(l1ist.search(29))
```

Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: E:\INFORMATIKA\SEMESTER 4\LAPRAK ALGOSTRUK\MODUL 4\No 5.py ====
True
False
>>>

6. Mengubah fungsi *binse* di halaman 43 agar dapat mengembalikan index lokasi elemen yang ditentukan.

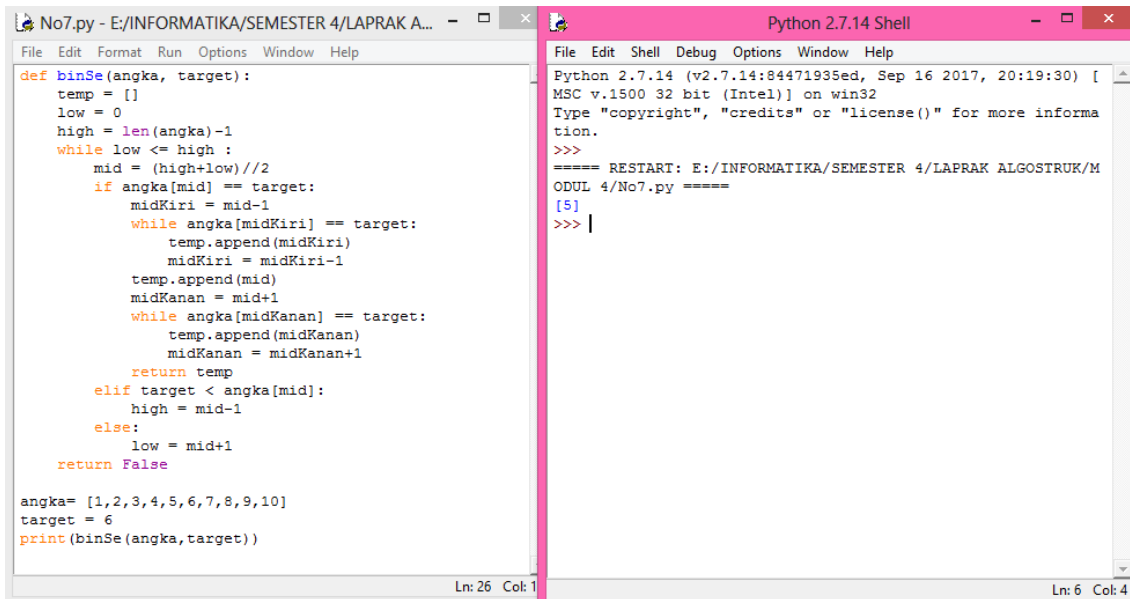
The image shows two windows from a Python 2.7.14 IDE. The left window, titled 'No 6.py', contains the source code for a binary search function `binSe`. It takes a list and a search value as input and returns the index of the element if found, or a message if not found. The right window, titled 'Python 2.7.14 Shell', shows the execution output. It displays the restart path and the results of the `binSe` function for a search value of 27.

```
def binSe(list, cari):
    low = 0
    high = len(list) - 1
    while low <= high:
        mid = (low+high)//2
        if list[mid] == cari:
            return "Elemen terletak di Index : "+str(mid)
        elif cari < list[mid]:
            high = mid - 1
        else:
            low = mid + 1
    return "Elemen tidak ditemukan"

list = [2, 4, 6, 9, 12, 27, 39, 46, 59, 77]
cari = 27
print(binSe(list, cari))
list = [2, 4, 6, 9, 12, 27, 39, 46, 59, 77]
cari = 115
print(binSe(list, cari))
```

Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: E:\INFORMATIKA\SEMESTER 4\LAPRAK ALGOSTRUK\MODUL 4\No 6.py ====
Elemen terletak di Index : 5
Elemen tidak ditemukan
>>>

7. Mengubah fungsi *binse* agar dapat mengembalikan semua index lokasi elemen yang telah ditemukan.



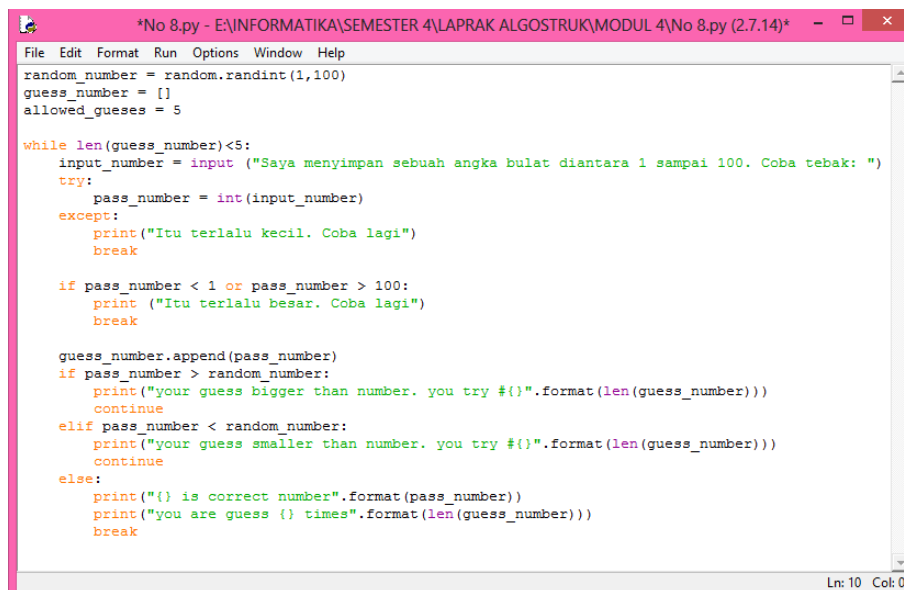
The screenshot shows a Python IDE with two windows. The left window, titled 'No7.py', contains a function `binSe(angka, target)` that performs a binary search on a sorted list `angka` and returns a list of indices where the `target` is found. The right window, titled 'Python 2.7.14 Shell', shows the execution of the script, which prints the output `[5]`.

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

angka= [1,2,3,4,5,6,7,8,9,10]
target = 6
print(binSe(angka,target))
```

Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: E:/INFORMATIKA/SEMESTER 4/LAPRAK ALGOSTRUK/MODUL 4/No7.py =====
[5]
>>> |

8. Menjelaskan pola tebak angka yang ada di modul 1.



The screenshot shows a Python IDE with a single window titled '*No 8.py - E:\INFORMATIKA\SEMESTER 4\LAPRAK ALGOSTRUK\MODUL 4\No 8.py (2.7.14)*'. The code implements a number guessing game where a random number is generated, and the user is given up to 5 attempts to guess it. The program provides feedback on whether the guess is too small, too large, or correct.

```
random_number = random.randint(1,100)
guess_number = []
allowed_guesses = 5

while len(guess_number)<5:
    input_number = input("Saya menyimpan sebuah angka bulat diantara 1 sampai 100. Coba tebak: ")
    try:
        pass_number = int(input_number)
    except:
        print("Itu terlalu kecil. Coba lagi")
        break

    if pass_number < 1 or pass_number > 100:
        print("Itu terlalu besar. Coba lagi")
        break

    guess_number.append(pass_number)
    if pass_number > random_number:
        print("your guess bigger than number. you try {}".format(len(guess_number)))
        continue
    elif pass_number < random_number:
        print("your guess smaller than number. you try {}".format(len(guess_number)))
        continue
    else:
        print("{} is correct number".format(pass_number))
        print("you are guess {} times".format(len(guess_number)))
        break
```

Ada fungsi built in yang diperlukan untuk mengacak angka dari angka 1 sampai 10. di baris ketiga kita import fungsi random ke dalam kode kita. Selanjutnya di baris kelima, kita generate angka random dari angka 1 sampai 10. Di baris keenam variabel list yang akan kita masukkan counter berapa kali pemain sudah menebak angkanya. Dan di baris ketujuh didefinisikan berapa kali pemain boleh menebak angka.

Selanjutnya dari baris 9 hingga 31 adalah blok untuk melakukan perulangan sekaligus pengecekan apakah tebakan sudah benar, pemberian clue dan counter berapa kali pemain sudah menebak angka. Di baris 11 hingga 15 adalah blok pengecekan apakah input yang dimasukkan adalah sebuah angka atau bukan. Di baris 17 hingga 19 adalah kondisi untuk mengecek apakah inputan angka yang dimasukkan berada pada range angka tebakan (1 sampai 10).

Pada baris ke 21, counter tebakan dari pemain dihitung selanjutnya pada baris 22 hingga 31 adalah pengecekan kondisi apakah tebakan benar atau salah, jika salah maka diberikan clue apakah tebakan lebih besar atau lebih kecil dari angka yang dimaksud. Pada baris 22 sampai 24 diberikan kondisi jika angka yang ditebak lebih besar dari angka yang dimaksud, maka akan di tampilkan notifikasi jika tebakan salah dan juga ditampilkan berapa kali pemain sudah menebak.

Sedangkan pada baris 25 sampai 27 diberikan kondisi jika angka yang ditebak lebih kecil dari angka yang dimaksud, maka akan di tampilkan notifikasi jika tebakan salah dan juga ditampilkan berapa kali pemain sudah menebak. Pada baris ke 28 hingga 31 akan ditampilkan notifikasi jika tebakan benar, serta juga ditampilkan berapa kali pemain menebak dan juga dilakukan perhentian perulangan (baris ke 31).