Nama : Herlangga Yusuf Syailendra
NIM : L200180186

Modul 4 Output (jupyter notebook)

jupyter **Linier Search** Last Checkpoint: 17/03/2020 (autosaved)

File Edit View Insert Cell Kernel Help

```
In [3]: def IndexedLinearSearch(list,search):
            buffer=[]
            for L in range (len(list)):
                if list[L] == search:
                    buffer.append(L)
            return buffer

In [4]: class Manusia(object):
            keadaan='lapar'
            def __init__(self,nama):
                self.nama=nama
            def ucapkanSalam(self):
                print("Salam, Namaku ",self.nama)
            def makan(self,s):
                print("saya baru saja makan ",s)
                self.keadaan='kenyang'
            def olahraga(self,k):
                print('saya barusaja latihan ',k)
                self.keadaan='lapar'
            def mengalikanDenganDua(self,n):
                return n*2

        class SiswaSMA(Manusia):
            def __init__(self,nama,NIM,kota,us):
                self.nama       = nama
                self.kotaTinggal= kota
                self.uangSaku   = us
            def __str__(self):
                return "{}, Tinggal di {}.Uang saku Rp {} tiap bulannya".format(self.nama,self.kotaTinggal,self.uangSaku)
            def ambiluangSaku   (self):
                return self.uangSaku
            def ambilNama       (self):
                return self.nama
            def makan           (self,a):
                print ("Saya baru saja makan %s sambil belajar"% a)
                self.keadaan = 'kenyang'

        class Mahasiswa(Manusia):
            def __init__(self,nama,NIM,kota,us):
                self.nama       = nama
                self.NIM        = NIM
                self.kotaTinggal= kota
                self.uangSaku   = us
                self.listKul    = []
            def __str__(self):
                return "{}, NIM {}. Tinggal di {}. Uang saku Rp {} tiap bulannya".format(self.nama,self.NIM,self.kotaTinggal,self.u
```

jupyter **Linier Search** Last Checkpoint: 17/03/2020 (autosaved)

```
                self.NIM        = NIM
                self.kotaTinggal= kota
                self.uangSaku   = us
                self.listKul    = []
            def __str__(self):
                return "{}, NIM {}. Tinggal di {}. Uang saku Rp {} tiap bulannya".format(self.nama,self.NIM,self.kotaTinggal,self.u
            def ambilNIM        (self):
                return self.nama
            def ambiluangSaku   (self):
                return self.uangSaku
            def ambilNama       (self):
                return self.nama
            def makan           (self,a):
                print ("Saya baru saja makan %s sambil belajar"% a)
                self.keadaan = 'kenyang'
            #-------------------------------------------------------------------#
            def ambilkotaTinggal(self):
                return self.kotaTinggal
            def perbaruiKotaTinggal(self,a):
                self.kotaTinggal = a
            def tambahUangSaku  (self,b):
                self.uangSaku   += b
            def listkuliah(self):
                return self.listKul
            def ambilkuliah(self,k):
                self.listKul.append(k)
            def hapusKuliah(self,k):
                if k in self.listKul : self.ListKul.remove(k)

        class MhsTIF(Mahasiswa):
            def katakanPy(self):
                print('Python is cool.')

In [12]: c0 = MhsTIF("Ika"     ,10 ,"Sukoharjo"  ,240000)
         c1 = MhsTIF("Budi"    ,51 ,"Sragen"     ,235000)
         c2 = MhsTIF("Ahmad"   ,2  ,"Surakarta"  ,250000)
         c3 = MhsTIF("Chandra",10 ,"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"   ,265000)
         c8 = MhsTIF("Janto"   ,23 ,"Klaten"     ,245000)
         c9 = MhsTIF("Hasan"   ,64 ,"Karanganyar",270000)
         c10 = MhsTIF("Khalid" ,29 ,"Porwodadi"  ,265000)

         Daftar=[c0,c1,c2,c3,c4,c5,c6,c7,c8,c9,c10]
         IndexedLinearSearch([i.ambilKotaTinggal() for i in Daftar],"Klaten")
```

jupyter **Linier Search** Last Checkpoint: 17/03/2020 (autosaved)

```
         c9 = MhsTIF("Hasan"   ,64 ,"Karanganyar",270000)
         c10 = MhsTIF("Khalid" ,29 ,"Porwodadi"  ,265000)

         Daftar=[c0,c1,c2,c3,c4,c5,c6,c7,c8,c9,c10]
         IndexedLinearSearch([i.ambilKotaTinggal() for i in Daftar],"Klaten")
```

Out[12]: [6, 8]

```
In [13]: IndexedLinearSearch([i.ambilKotaTinggal() for i in Daftar],"Surabaya")
```

Out[13]: []

```
In [0]: def Minimum(daftar):
            Minimum = float("inf")
            buffer  = []
            for i in range(len(daftar)):
                if Minimum>daftar[i].ambiluangSaku():
                    Minimum=daftar[i].ambiluangSaku()
                    buffer=[]
                if Minimum==daftar[i].ambiluangSaku():
                    buffer.append(daftar[i])
            return buffer
        k=Minimum(Daftar)
        print(*k,sep="\n")
```

```
Budi, NIM 51. Tinggal di Sragen. Uang saku Rp 235000 tiap bulannya
Chandra, NIM 10. Tinggal di Surakarta. Uang saku Rp 235000 tiap bulannya
```

```
In [10]: def UangKurangdari250k(daftar):
             buffer  = []
             for i in range(len(daftar)):
                 if 250000>daftar[i].ambiluangSaku():
                     buffer.append(daftar[i])
             return buffer
         k=UangKurangdari250k(Daftar)
         print(*k,sep="\n")
```

```
Ika, NIM 10. Tinggal di Sukoharjo. Uang saku Rp 240000 tiap bulannya
Budi, NIM 51. Tinggal di Sragen. Uang saku Rp 235000 tiap bulannya
Chandra, NIM 10. Tinggal di Surakarta. Uang saku Rp 235000 tiap bulannya
Eka, NIM 4. Tinggal di Boyolali. Uang saku Rp 240000 tiap bulannya
Deni, NIM 13. Tinggal di Klaten. Uang saku Rp 245000 tiap bulannya
Galuh, NIM 5. Tinggal di Wonogiri. Uang saku Rp 265000 tiap bulannya
Janto, NIM 23. Tinggal di Klaten. Uang saku Rp 245000 tiap bulannya
```

jupyter **Linier Search** Last Checkpoint: 17/03/2020 (autosaved)

```
         IndexedLinearSearch([i.ambilKotaTinggal() for i in Daftar],"Klaten")
```

Out[12]: [6, 8]

```
In [13]: IndexedLinearSearch([i.ambilKotaTinggal() for i in Daftar],"Surabaya")
```

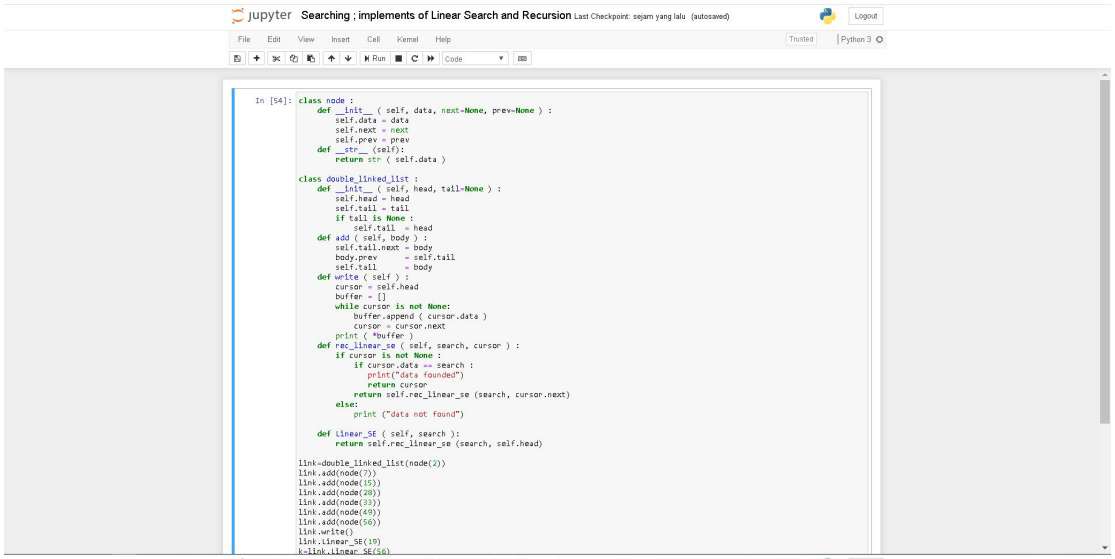
Out[13]: []

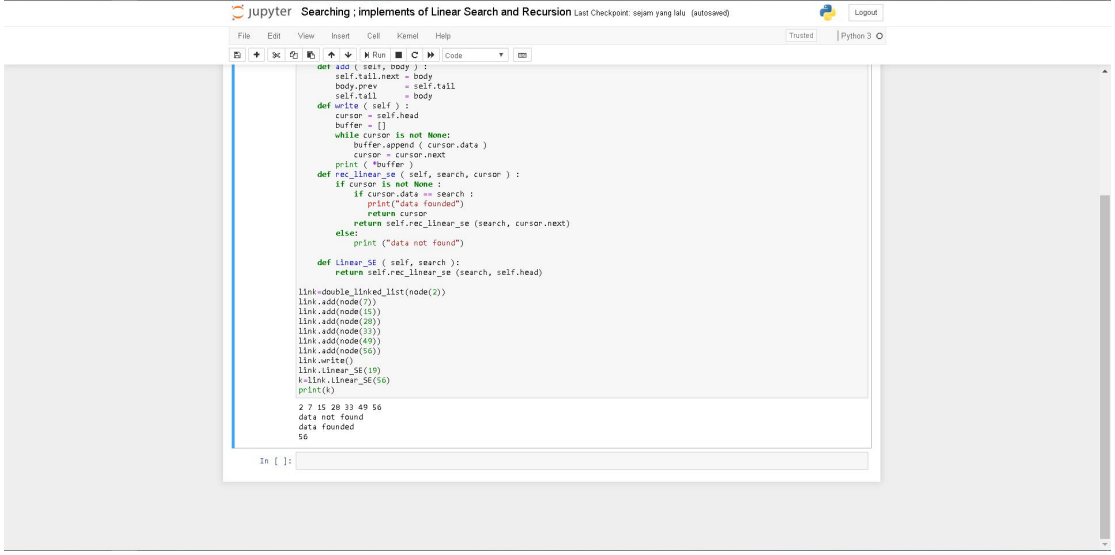
```
In [0]: def Minimum(daftar):
            Minimum = float("inf")
            buffer  = []
            for i in range(len(daftar)):
                if Minimum>daftar[i].ambiluangSaku():
                    Minimum=daftar[i].ambiluangSaku()
                    buffer=[]
                if Minimum==daftar[i].ambiluangSaku():
                    buffer.append(daftar[i])
            return buffer
        k=Minimum(Daftar)
        print(*k,sep="\n")
```

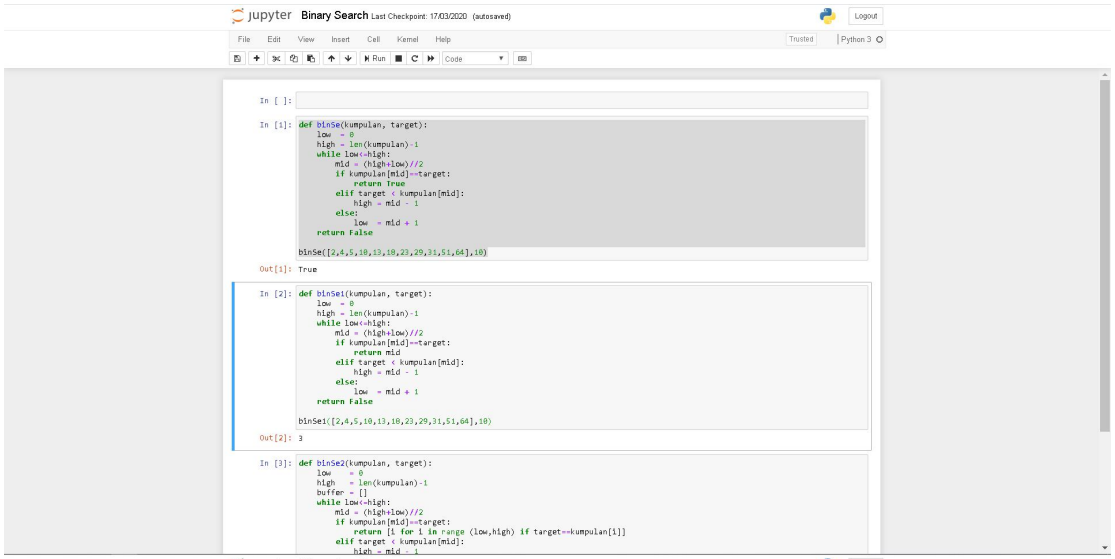
```
Budi, NIM 51. Tinggal di Sragen. Uang saku Rp 235000 tiap bulannya
Chandra, NIM 10. Tinggal di Surakarta. Uang saku Rp 235000 tiap bulannya
```

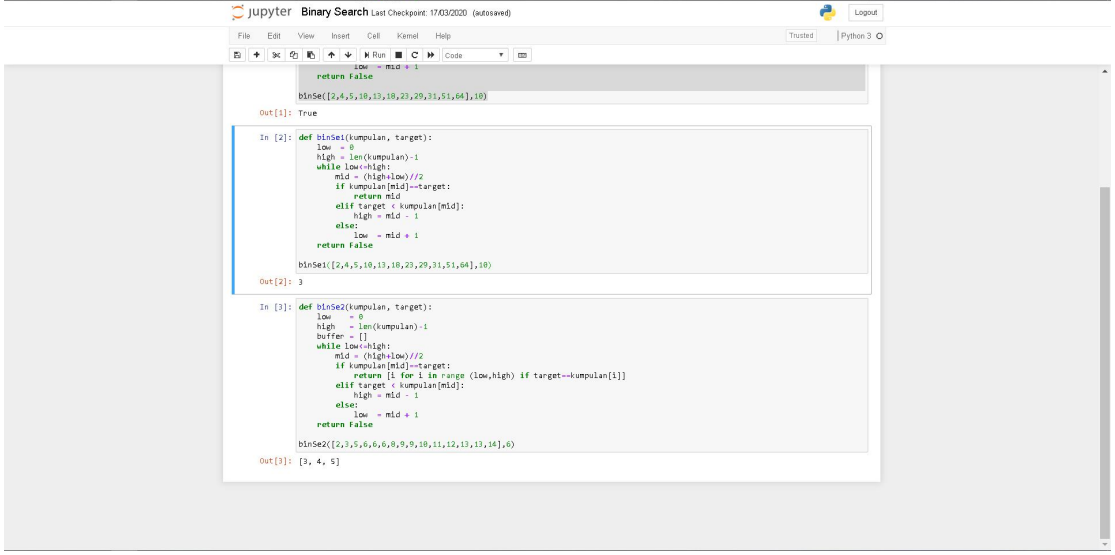
```
In [10]: def UangKurangdari250k(daftar):
             buffer  = []
             for i in range(len(daftar)):
                 if 250000>daftar[i].ambiluangSaku():
                     buffer.append(daftar[i])
             return buffer
         k=UangKurangdari250k(Daftar)
         print(*k,sep="\n")
```

```
Ika, NIM 10. Tinggal di Sukoharjo. Uang saku Rp 240000 tiap bulannya
Budi, NIM 51. Tinggal di Sragen. Uang saku Rp 235000 tiap bulannya
Chandra, NIM 10. Tinggal di Surakarta. Uang saku Rp 235000 tiap bulannya
Eka, NIM 4. Tinggal di Boyolali. Uang saku Rp 240000 tiap bulannya
Deni, NIM 13. Tinggal di Klaten. Uang saku Rp 245000 tiap bulannya
Galuh, NIM 5. Tinggal di Wonogiri. Uang saku Rp 265000 tiap bulannya
Janto, NIM 23. Tinggal di Klaten. Uang saku Rp 245000 tiap bulannya
```


jupyter Searching ; implements of Linear Search and Recursion Last Checkpoint: sejam yang lalu (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
In [54]:
class node :
def __init__ (self, data, next=None, prev=None) :
self.data = data
self.next = next
self.prev = prev
def __str__ (self):
return str (self.data)
class double_linked_list :
def __init__ (self, head, tail=None) :
self.head = head
self.tail = tail
if tail is None :
self.tail = head
def add (self, body) :
self.tail.next = body
body.prev = self.tail
self.tail = body
def write (self) :
cursor = self.head
buffer = []
while cursor is not None:
buffer.append (cursor.data)
cursor = cursor.next
print (*buffer)
def rec_linear_se (self, search, cursor) :
if cursor is not None :
if cursor.data == search :
print("data founded")
return cursor
return self.rec_linear_se (search, cursor.next)
else:
print ("data not found")
def Linear_SE (self, search):
return self.rec_linear_se (search, self.head)
link=double_linked_list(node(2))
link.add(node(7))
link.add(node(15))
link.add(node(20))
link.add(node(33))
link.add(node(49))
link.add(node(56))
link.write()
link.Linear_SE(19)
k=link.Linear_SE(56)


jupyter Searching ; implements of Linear Search and Recursion Last Checkpoint: sejam yang lalu (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
def add (self, body) :
self.tail.next = body
body.prev = self.tail
self.tail = body
def write (self) :
cursor = self.head
buffer = []
while cursor is not None:
buffer.append (cursor.data)
cursor = cursor.next
print (*buffer)
def rec_linear_se (self, search, cursor) :
if cursor is not None :
if cursor.data == search :
print("data founded")
return cursor
return self.rec_linear_se (search, cursor.next)
else:
print ("data not found")
def Linear_SE (self, search):
return self.rec_linear_se (search, self.head)
link=double_linked_list(node(2))
link.add(node(7))
link.add(node(15))
link.add(node(20))
link.add(node(33))
link.add(node(49))
link.add(node(56))
link.write()
link.Linear_SE(19)
k=link.Linear_SE(56)
print(k)
2 7 15 20 33 49 56
data not found
data founded
56
In [ ]:


jupyter Binary Search Last Checkpoint: 17.03.2020 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
In [ ]:
In [1]:
def binSe(kumpulan, target):
low = 0
high = len(kumpulan)-1
while low<=high:
mid = (high+low)//2
if kumpulan[mid]==target:
return True
elif target < kumpulan[mid]:
high = mid - 1
else:
low = mid + 1
return False
binSe([2,4,5,10,13,18,23,29,31,51,64],10)
Out[1]: True
In [2]:
def binSe1(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
binSe1([2,4,5,10,13,18,23,29,31,51,64],10)
Out[2]: 3
In [3]:
def binSe2(kumpulan, target):
low = 0
high = len(kumpulan)-1
buffer = []
while low<=high:
mid = (high+low)//2
if kumpulan[mid]==target:
return [i for i in range (low,high) if target==kumpulan[i]]
elif target < kumpulan[mid]:
high = mid - 1


jupyter Binary Search Last Checkpoint: 17.03.2020 (autosaved)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
low = mid + 1
return False
binSe([2,4,5,10,13,18,23,29,31,51,64],10)
Out[1]: True
In [2]:
def binSe1(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
binSe1([2,4,5,10,13,18,23,29,31,51,64],10)
Out[2]: 3
In [3]:
def binSe2(kumpulan, target):
low = 0
high = len(kumpulan)-1
buffer = []
while low<=high:
mid = (high+low)//2
if kumpulan[mid]==target:
return [i for i in range (low,high) if target==kumpulan[i]]
elif target < kumpulan[mid]:
high = mid - 1
else:
low = mid + 1
return False
binSe2([2,3,5,6,6,6,8,9,9,10,11,12,13,13,14],6)
Out[3]: [3, 4, 5]