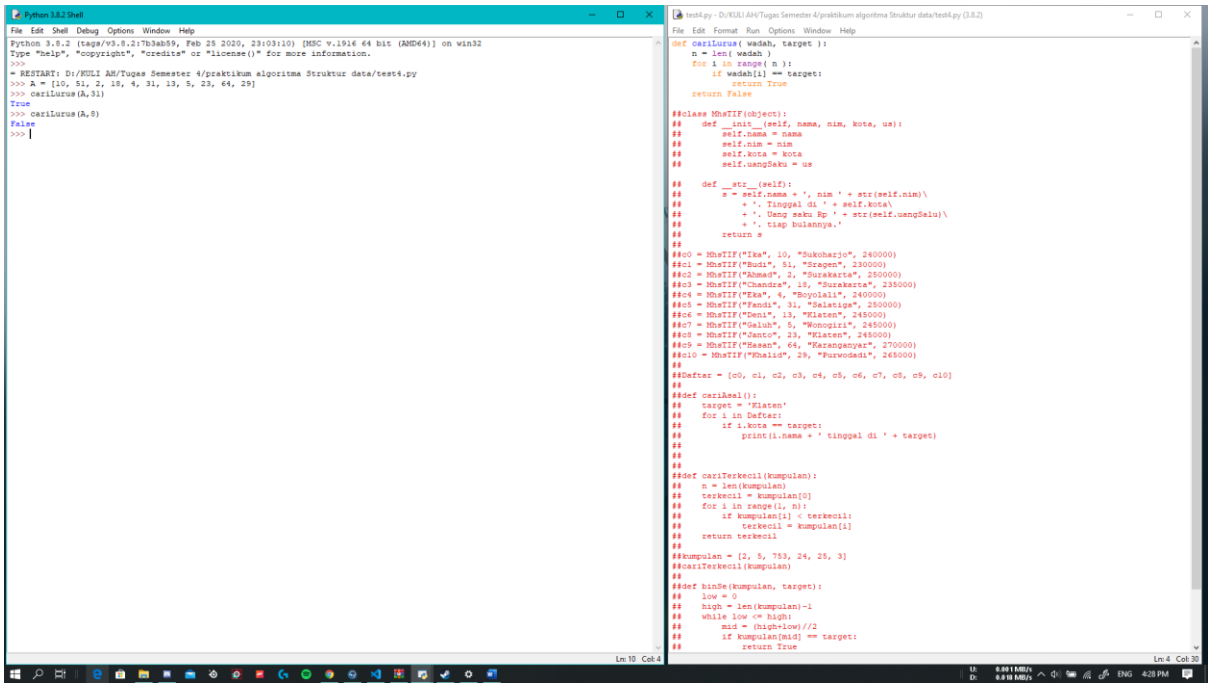
Nama : Yudha Gana Prasetyo Wibowo

Nim : L200180150

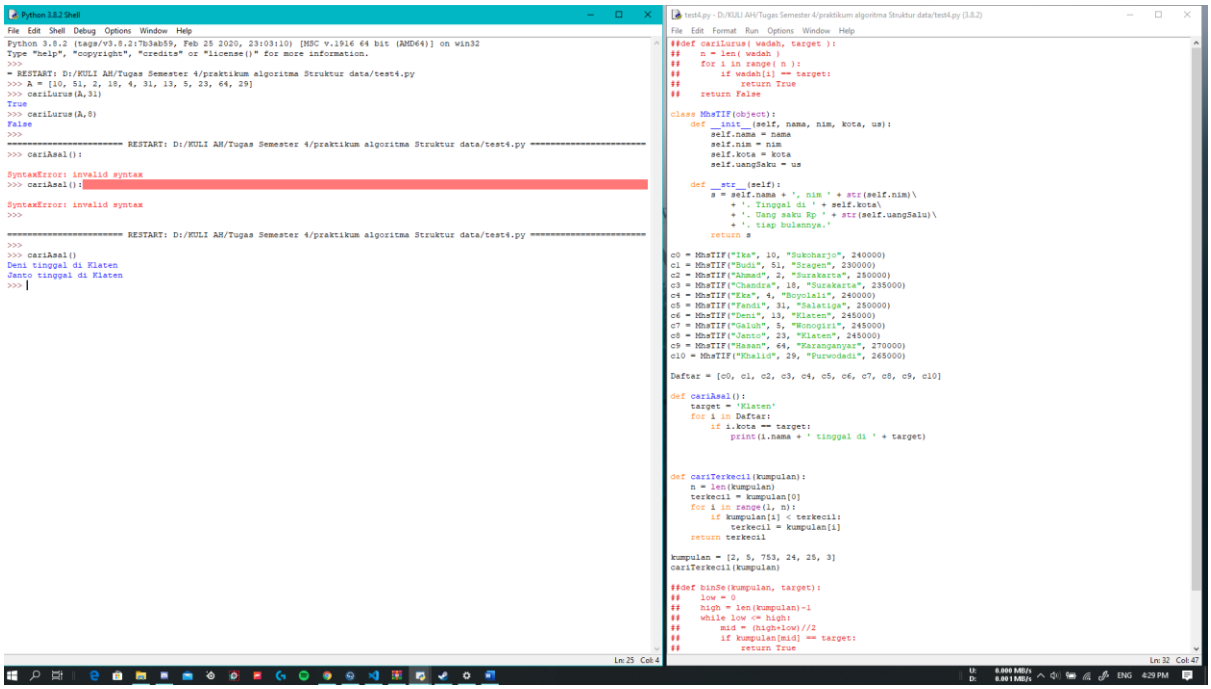
Kelas : F

Laporan Kegiatan Praktikum Modul 4 “PENCARIAN”

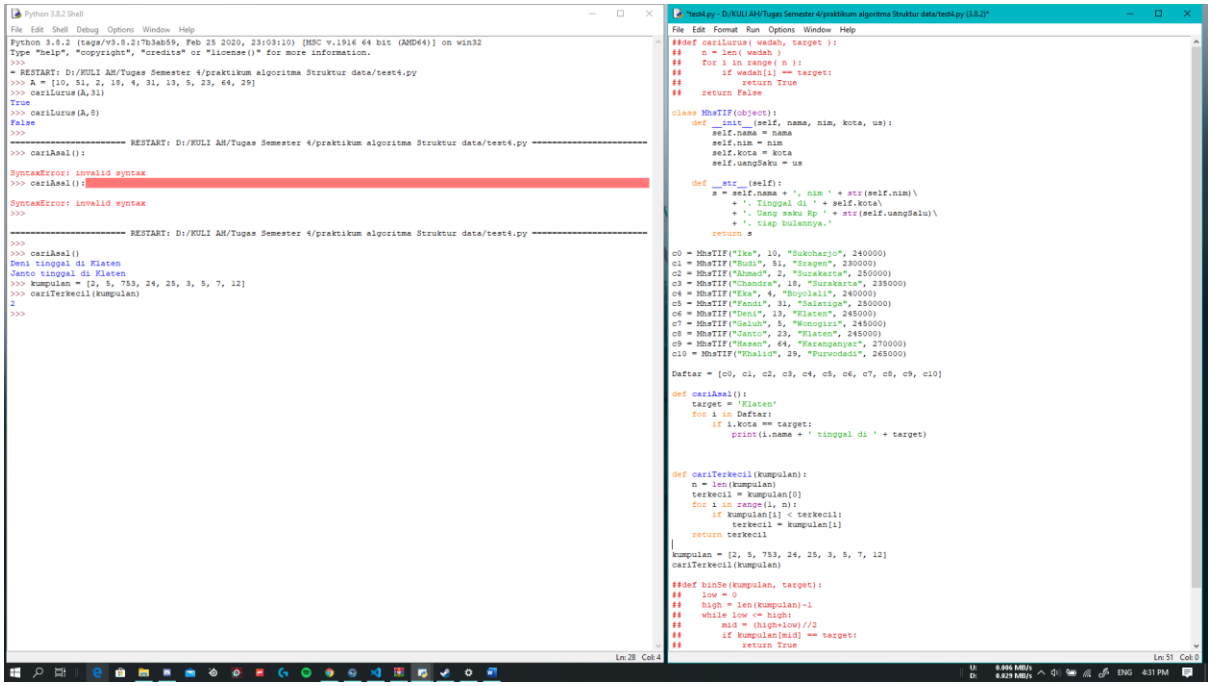
Linear Search



Pencarian Lurus Objek Buatan Sendiri



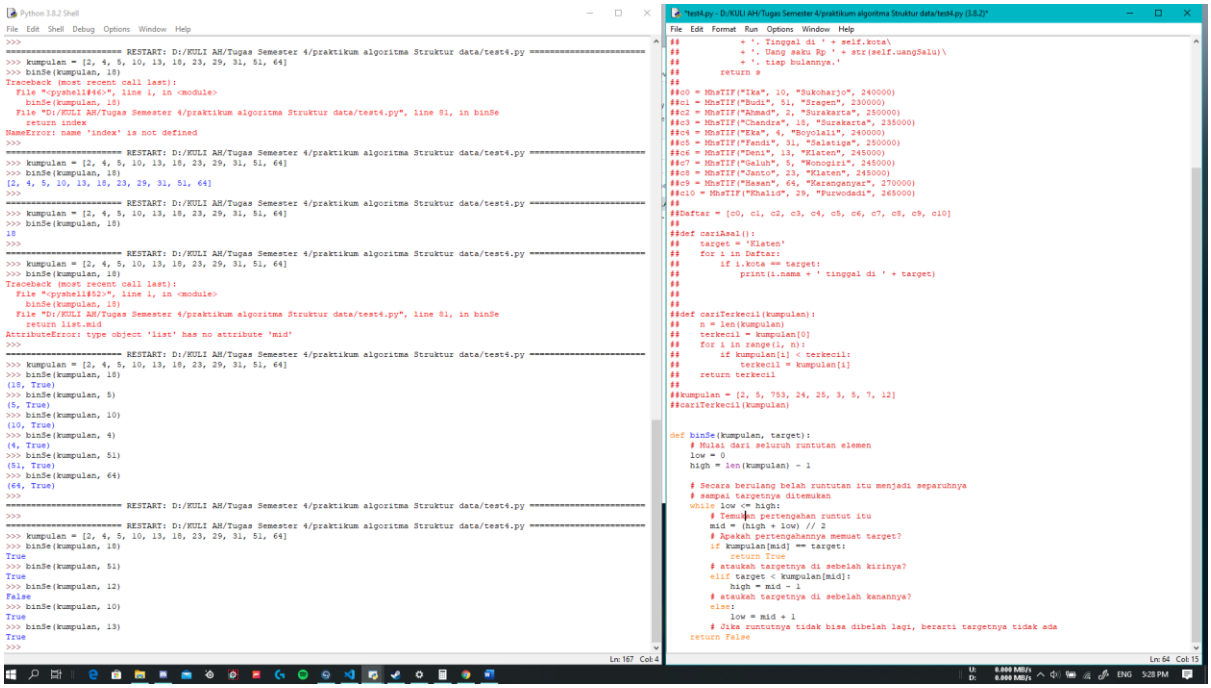
Pencarian Lurus di Linked List



```
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: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py
>>> A = [10, 51, 2, 18, 4, 31, 13, 5, 23, 64, 29]
>>> cariLurus(A,51)
True
>>> cariLurus(A,8)
False
>>>
========== RESTART: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py ==========
>>> cariAsal():

SyntaxError: invalid syntax
>>> cariAsal():

SyntaxError: invalid syntax
>>>

========== RESTART: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py ==========
>>>
>>> cariAsal()
Deni tinggal di Klaten
Janto tinggal di Klaten
>>> kumpulan = [2, 5, 753, 24, 25, 3, 5, 7, 12]
>>> cariTerkecil(kumpulan)
2
>>>
```

```
##def cariLurus( wadah, target ):
##    n = len( wadah )
##    for i in range( n ):
##        if wadah[i] == target:
##            return True
##    return False

class MhsTIF(object):
    def __init__(self, nama, nim, kota, us):
        self.nama = nama
        self.nim = nim
        self.kota = kota
        self.uangSaku = us

    def __str__(self):
        s = self.nama + ', nim ' + str(self.nim)\
            + '. Tinggal di ' + self.kota\
            + '. Uang saku Rp ' + str(self.uangSaku)\
            + ', tiap bulannya.'
        return s

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]

def cariAsal():
    target = 'Klaten'
    for i in Daftar:
        if i.kota == target:
            print(i.nama + ' tinggal di ' + target)


def cariTerkecil(kumpulan):
    n = len(kumpulan)
    terkecil = kumpulan[0]
    for i in range(1, n):
        if kumpulan[i] < terkecil:
            terkecil = kumpulan[i]
    return terkecil

kumpulan = [2, 5, 753, 24, 25, 3, 5, 7, 12]
cariTerkecil(kumpulan)

##def binSe(kumpulan, target):
##    low = 0
##    high = len(kumpulan)-1
##    while low <= high:
##        mid = (high+low)//2
##        if kumpulan[mid] == target:
##            return True
```

Binary Search



```
>>>
========== RESTART: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py ==========
>>> kumpulan = [2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]
>>> binSe(kumpulan, 18)
Traceback (most recent call last):
  File "<pyshell#46>", line 1, in <module>
    binSe(kumpulan, 18)
  File "D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py", line 81, in binSe
    return index
NameError: name 'index' is not defined
>>>
========== RESTART: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py ==========
>>> kumpulan = [2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]
>>> binSe(kumpulan, 18)
[2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]
>>>
========== RESTART: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py ==========
>>> kumpulan = [2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]
>>> binSe(kumpulan, 18)
18
>>>
========== RESTART: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py ==========
>>> kumpulan = [2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]
>>> binSe(kumpulan, 18)
Traceback (most recent call last):
  File "<pyshell#52>", line 1, in <module>
    binSe(kumpulan, 18)
  File "D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py", line 81, in binSe
    return list.mid
AttributeError: type object 'list' has no attribute 'mid'
>>>
========== RESTART: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py ==========
>>> kumpulan = [2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]
>>> binSe(kumpulan, 18)
(18, True)
>>> binSe(kumpulan, 5)
(5, True)
>>> binSe(kumpulan, 10)
(10, True)
>>> binSe(kumpulan, 4)
(4, True)
>>> binSe(kumpulan, 51)
(51, True)
>>> binSe(kumpulan, 64)
(64, True)
>>>
========== RESTART: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py ==========
>>>
========== RESTART: D:/KULI AH/Tugas Semester 4/praktikum algoritma Struktur data/test4.py ==========
>>> kumpulan = [2, 4, 5, 10, 13, 18, 23, 29, 31, 51, 64]
>>> binSe(kumpulan, 18)
True
>>> binSe(kumpulan, 51)
True
>>> binSe(kumpulan, 12)
False
>>> binSe(kumpulan, 10)
True
>>> binSe(kumpulan, 13)
True
>>>
```

```
##            + '. Tinggal di ' + self.kota\
##            + '. Uang saku Rp ' + str(self.uangSaku)\
##            + '. tiap bulannya.'
##        return s
##
##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]
##
##def cariAsal():
##    target = 'Klaten'
##    for i in Daftar:
##        if i.kota == target:
##            print(i.nama + ' tinggal di ' + target)
##
##
##def cariTerkecil(kumpulan):
##    n = len(kumpulan)
##    terkecil = kumpulan[0]
##    for i in range(1, n):
##        if kumpulan[i] < terkecil:
##            terkecil = kumpulan[i]
##    return terkecil
##
##kumpulan = [2, 5, 753, 24, 25, 3, 5, 7, 12]
##cariTerkecil(kumpulan)


def binSe(kumpulan, target):
    # Mulai dari seluruh runtutan elemen
    low = 0
    high = len(kumpulan) - 1

    # Secara berulang belah runtutan itu menjadi separuhnya
    # sampai targetnya ditemukan
    while low <= high:
        # Temukan pertengahan runtut itu
        mid = (high + low) // 2
        # Apakah pertengahannya memuat target?
        if kumpulan[mid] == target:
            return True
        # ataukah targetnya di sebelah kirinya?
        elif target < kumpulan[mid]:
            high = mid - 1
        # ataukah targetnya di sebelah kanannya?
        else:
            low = mid + 1
    # Jika runtutnya tidak bisa dibelah lagi, berarti targetnya tidak ada
    return False
```