

Nama : Arindita Prihastama

NIM : L200180058

Kelas : C

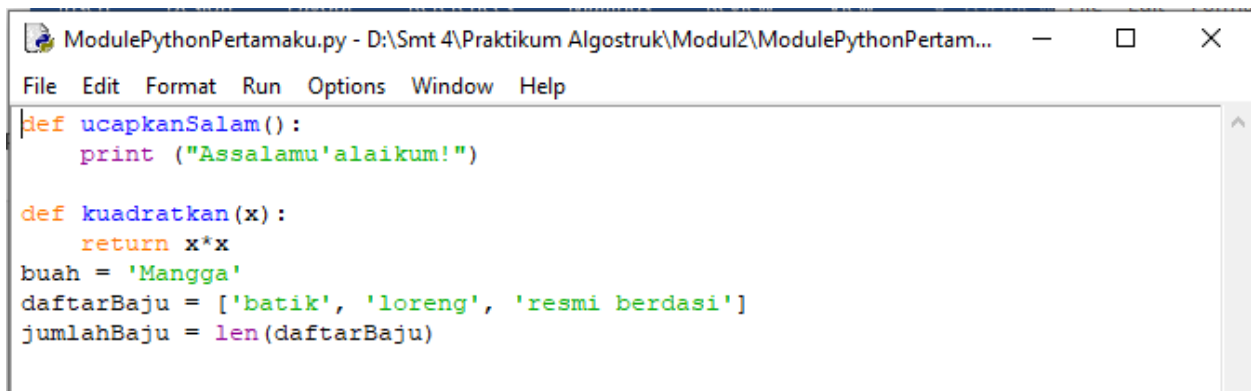
MODUL 2

Mengenal OOP pada Python

2.1 Module

Module dipahami sebagai kumpulan prosedur dan nilai yang tersimpan dalam satu atau beberapa file, yang bisa diakses dengan meng-import-nya.

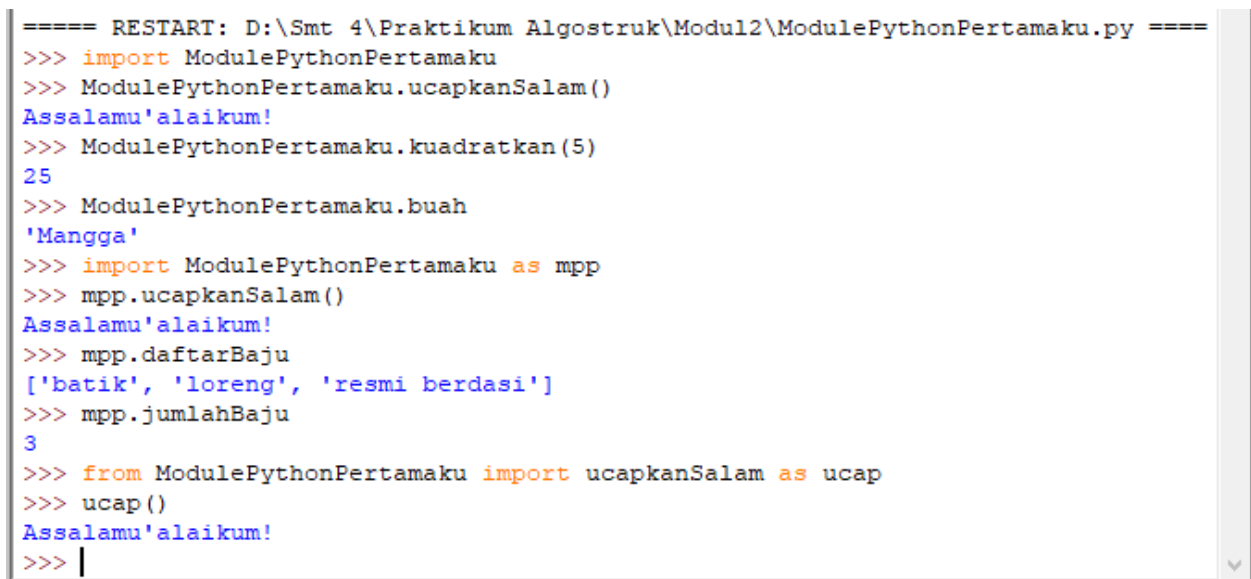
Latihan 2.1



```
ModulePythonPertamaku.py - D:\Smt 4\Praktikum Algostruk\Modul2\ModulePythonPertam...
File Edit Format Run Options Window Help
def ucapkanSalam():
    print ("Assalamu'alaikum!")

def kuadratkan(x):
    return x*x
buah = 'Mangga'
daftarBaju = ['batik', 'loreng', 'resmi berdasi']
jumlahBaju = len(daftarBaju)
```

Saat dijalankan di python shell :



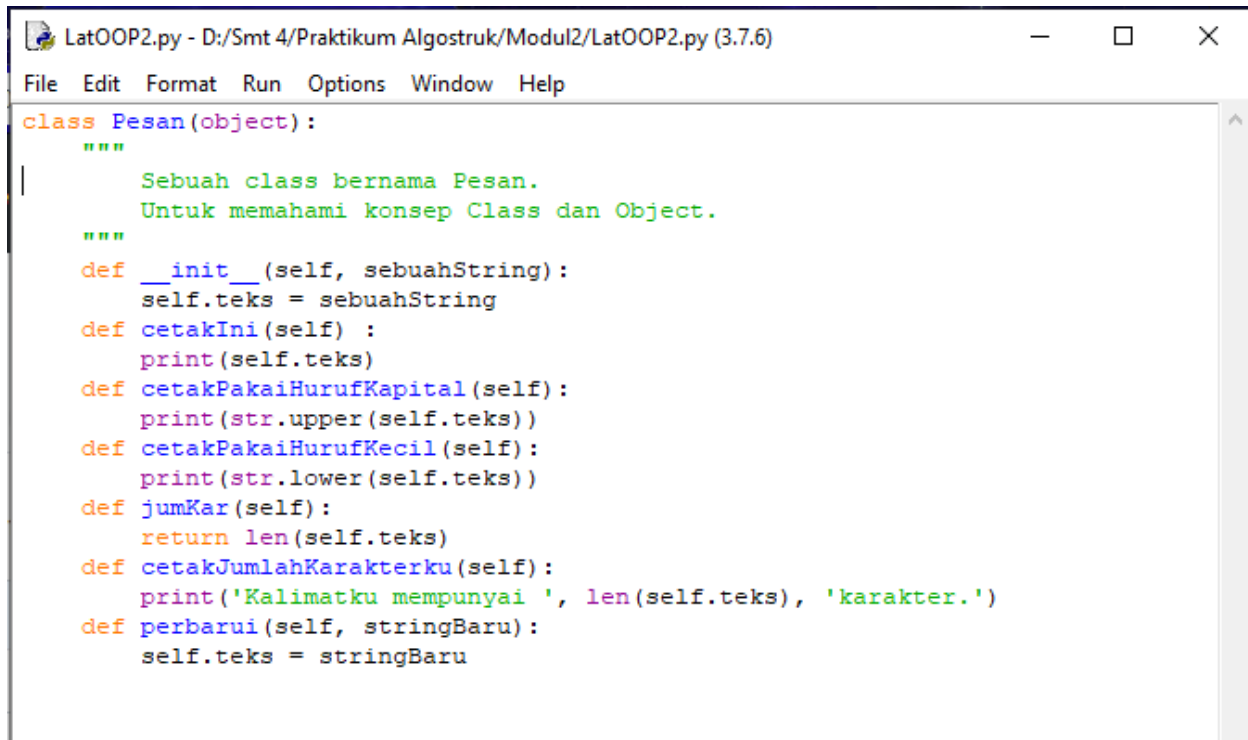
```
===== RESTART: D:\Smt 4\Praktikum Algostruk\Modul2\ModulePythonPertamaku.py =====
>>> import ModulePythonPertamaku
>>> ModulePythonPertamaku.ucapkanSalam()
Assalamu'alaikum!
>>> ModulePythonPertamaku.kuadratkan(5)
25
>>> ModulePythonPertamaku.buah
'Mangga'
>>> import ModulePythonPertamaku as mpp
>>> mpp.ucapkanSalam()
Assalamu'alaikum!
>>> mpp.daftarBaju
['batik', 'loreng', 'resmi berdasi']
>>> mpp.jumlahBaju
3
>>> from ModulePythonPertamaku import ucapkanSalam as ucap
>>> ucap()
Assalamu'alaikum!
>>> |
```

2.2 Class dan Object

Class adalah sebuah konsep atau cetak biru mengenai 'sesuatu' (umumnya kata benda).

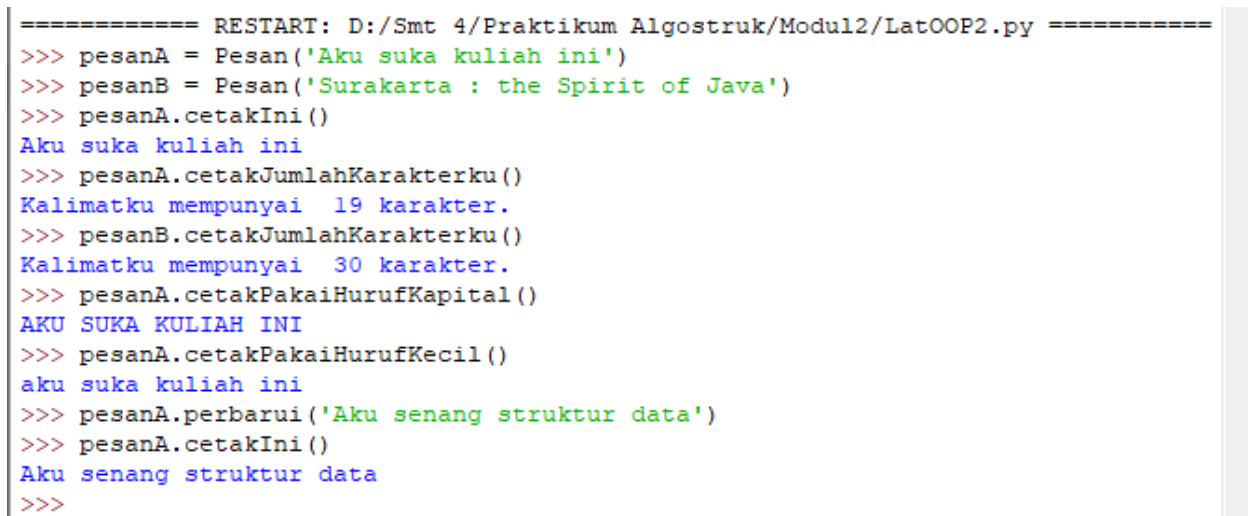
Object adalah 'sebuah class yang mewujud'.

Latihan 2.2



```
class Pesan(object):
    """
    Sebuah class bernama Pesan.
    Untuk memahami konsep Class dan Object.
    """
    def __init__(self, sebuahString):
        self.teks = sebuahString
    def cetakIni(self) :
        print(self.teks)
    def cetakPakaiHurufKapital(self):
        print(str.upper(self.teks))
    def cetakPakaiHurufKecil(self):
        print(str.lower(self.teks))
    def jumKar(self):
        return len(self.teks)
    def cetakJumlahKarakterku(self):
        print('Kalimatku mempunyai ', len(self.teks), 'karakter.')
    def perbarui(self, stringBaru):
        self.teks = stringBaru
```

Saat dijalankan di python shell :



```
===== RESTART: D:/Smt 4/Praktikum Algostruk/Modul2/LatOOP2.py =====
>>> pesanA = Pesan('Aku suka kuliah ini')
>>> pesanB = Pesan('Surakarta : the Spirit of Java')
>>> pesanA.cetakIni()
Aku suka kuliah ini
>>> pesanA.cetakJumlahKarakterku()
Kalimatku mempunyai 19 karakter.
>>> pesanB.cetakJumlahKarakterku()
Kalimatku mempunyai 30 karakter.
>>> pesanA.cetakPakaiHurufKapital()
AKU SUKA KULIAH INI
>>> pesanA.cetakPakaiHurufKecil()
aku suka kuliah ini
>>> pesanA.perbarui('Aku senang struktur data')
>>> pesanA.cetakIni()
Aku senang struktur data
>>>
```

Latihan 2.3

```
LatOOP3.py - D:/Smt 4/Praktikum Algostruk/Modul2/LatOOP3.py (3.7.6)
File Edit Format Run Options Window Help

class Manusia(object):
    """ Class 'Manusia' dengan inisiasi 'nama' """
    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 baru saja latihan ", k)
        self.keadaan = 'lapar'
    def mengalikanDenganDua(self, n):
        return n*2
```

Saat di run di python shell :

```
===== RESTART: D:/Smt 4/Praktikum Algostruk/Modul2/LatOOP3.py =====
>>> p1 = Manusia('Fatimah')
>>> p1.ucapkanSalam()
Salam, namaku Fatimah
>>> p2 = Manusia('Budi')
>>> p2.ucapkanSalam()
Salam, namaku Budi
>>> ak = Manusia('Abdul Karim')
>>> ak.ucapkanSalam()
Salam, namaku Abdul Karim
>>> ak.keadaan
'lapar'
>>> ak.makan('nasi goreng')
Saya baru saja makan nasi goreng
>>> ak.keadaan
'kenyang'
>>> ak.olahraga('renang')
Saya baru saja latihan renang
>>> ak.keadaan
'lapar'
>>> ak.makan('bakso')
Saya baru saja makan bakso
>>> ak.keadaan
'kenyang'
>>> ak.mengalikanDenganDua(8)
16
>>> |
```

Keadaan dapat berubah dari 'lapar' menjadi 'kenyang' karena saat kita memanggil method makan() terdapat perintah yang membuat keadaan menjadi 'kenyang'. Hal tersebut juga terjadi

pada saat kita memanggil method olahraga() yang didalamnya juga terdapat perintah yang membuat keadaan menjadi 'lapar' kembali .

2.2.1 Pewarisan

Pewarisan atau inheritance adalah pembuatan suatu class berdasarkan class lain.

Latihan 2.4

```
class Mahasiswa(Manusia):
    """ Class Mahasiswa yang dibangun dari class Manusia"""
    def __init__(self, nama, NIM, kota, us):
        """Metode inisiasi ini menutupi metode inisiasi di class Manusia"""
        self.nama = nama
        self.NIM = NIM
        self.kotaTinggal = kota
        self.uangSaku = us
    def __str__(self):
        s = self.nama + ', NIM ' + str(self.NIM) \
            + ', Tinggal di ' + self.kotaTinggal \
            + ', Uang Saku Rp. ' + str(self.uangSaku) \
            + ' tiap bulannya.'
        return s
    def ambilNama(self):
        return self.nama
    def ambilNIM(self):
        return self.NIM
    def ambilUangSaku(self):
        return self.uangSaku
    def makan(self, s):
        """Metode ini menutupi metode 'makan'-nya class Manusia.
        Mahasiswa kalau makan sambil belajar."""
        print("Saya baru saja makan", s, "sambil belajar.")
        self.keadaan = 'kenyang'
```

Class Mahasiswa saya buat di file yang sama dengan file yang memuat class Manusia.

Saat dijalankan di python shell :

```

===== RESTART: D:/Smt 4/Praktikum Algostruk/Modul2/LatOOP3.py =====
>>> m1 = Mahasiswa('Jamil', 234, 'Surakarta', 250000)
>>> m2 = Mahasiswa('Andi', 365, 'Magelang', 275000)
>>> m3 = Mahasiswa('Sri', 676, 'Yogyakarta', 240000)
>>> m1.ambilNama()
'Jamil'
>>> m2.ambilNIM()
365
>>> m3.ucapkanSalam()
Salam, namaku Sri
>>> m3.keadaan
'lapar'
>>> m3.makan('gado-gado')
Saya baru saja makan gado-gado sambil belajar.
>>> m3.keadaan
'kenyang'
>>> print(m3)
Sri, NIM 676, Tinggal di Yogyakarta, Uang Saku Rp. 240000 tiap bulannya.
>>> |

```

Latihan 2.5

```

class MhsTIF(Mahasiswa):
    """ Class MhsTIF yang dibangun dari class Mahasiswa"""
    def katakanPy(self):
        print('Python is cool...')

```

Karena saya sudah mencoba untuk membuat class MhsTIF di file yang berbeda namun ketika dijalankan error, saya membuat class ini di file yang sama dengan Class Manusia dan Mahasiswa.

Saat dijalankan di python shell :

```

===== RESTART: D:/Smt 4/Praktikum Algostruk/Modul2/LatOOP3.py =====
>>> m4 = MhsTIF('Badu', 334, 'Sragen', 230000)
>>> m4.katakanPy()
Python is cool...
>>> print(m4)
Badu, NIM 334, Tinggal di Sragen, Uang Saku Rp. 230000 tiap bulannya.
>>> m4.keadaan
'lapar'
>>> m4.makan('pecel')
Saya baru saja makan pecel sambil belajar.
>>> m4.keadaan
'kenyang'
>>> m4.ucapkanSalam()
Salam, namaku Badu
>>> |

```

2.3 Object dan List

Seperti halnya object-object yang berasal dari class lain seperti `int`, `str`, `float`, object juga bisa dikumpulkan di dalam suatu list.

Latihan 2.6

Masih melanjutkan Latihan 2.5

```
===== RESTART: D:/Smt 4/Praktikum Algostruk/Modul2/LatOOP3.py =====
>>> m1 = Mahasiswa('Jamil', 234, 'Surakarta', 250000)
>>> m2 = Mahasiswa('Andi', 365, 'Magelang', 275000)
>>> m3 = Mahasiswa('Sri', 676, 'Yogyakarta', 240000)
>>> m4 = MhsTIF('Badu', 334, 'Sragen', 230000)
>>> daftar = [m1, m2, m3, m4]
>>> for i in daftar:
>>>     print(i.NIM)

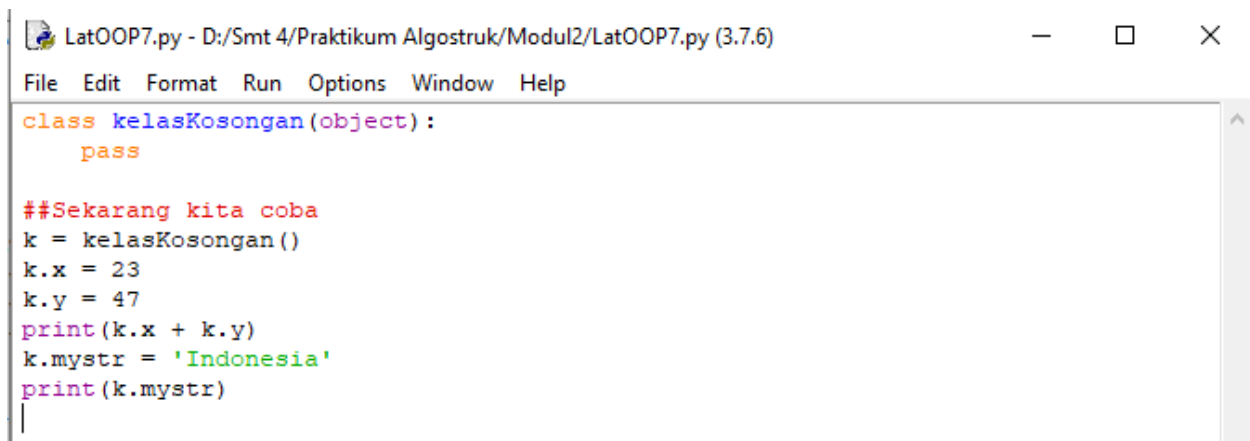
234
365
676
334
>>> for i in daftar:
>>>     print i

SyntaxError: Missing parentheses in call to 'print'. Did you mean print(i)?
>>> for i in daftar:
>>>     print(i)

Jamil, NIM 234, Tinggal di Surakarta, Uang Saku Rp. 250000 tiap bulannya.
Andi, NIM 365, Tinggal di Magelang, Uang Saku Rp. 275000 tiap bulannya.
Sri, NIM 676, Tinggal di Yogyakarta, Uang Saku Rp. 240000 tiap bulannya.
Badu, NIM 334, Tinggal di Sragen, Uang Saku Rp. 230000 tiap bulannya.
>>> daftar[2].ambilNama()
'Sri'
>>> |
```

2.4 Class sebagai *namespace*

Latihan 2.7

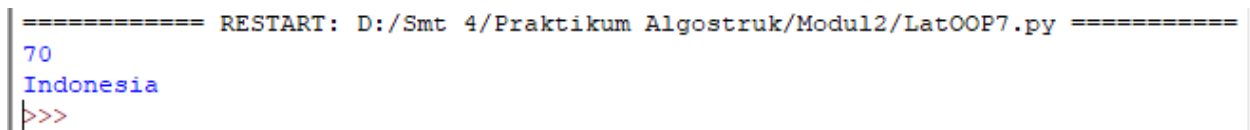


```
LatOOP7.py - D:/Smt 4/Praktikum Algostruk/Modul2/LatOOP7.py (3.7.6)
File Edit Format Run Options Window Help

class kelasKosongan(object):
    pass

##Sekarang kita coba
k = kelasKosongan()
k.x = 23
k.y = 47
print(k.x + k.y)
k.mystr = 'Indonesia'
print(k.mystr)
```

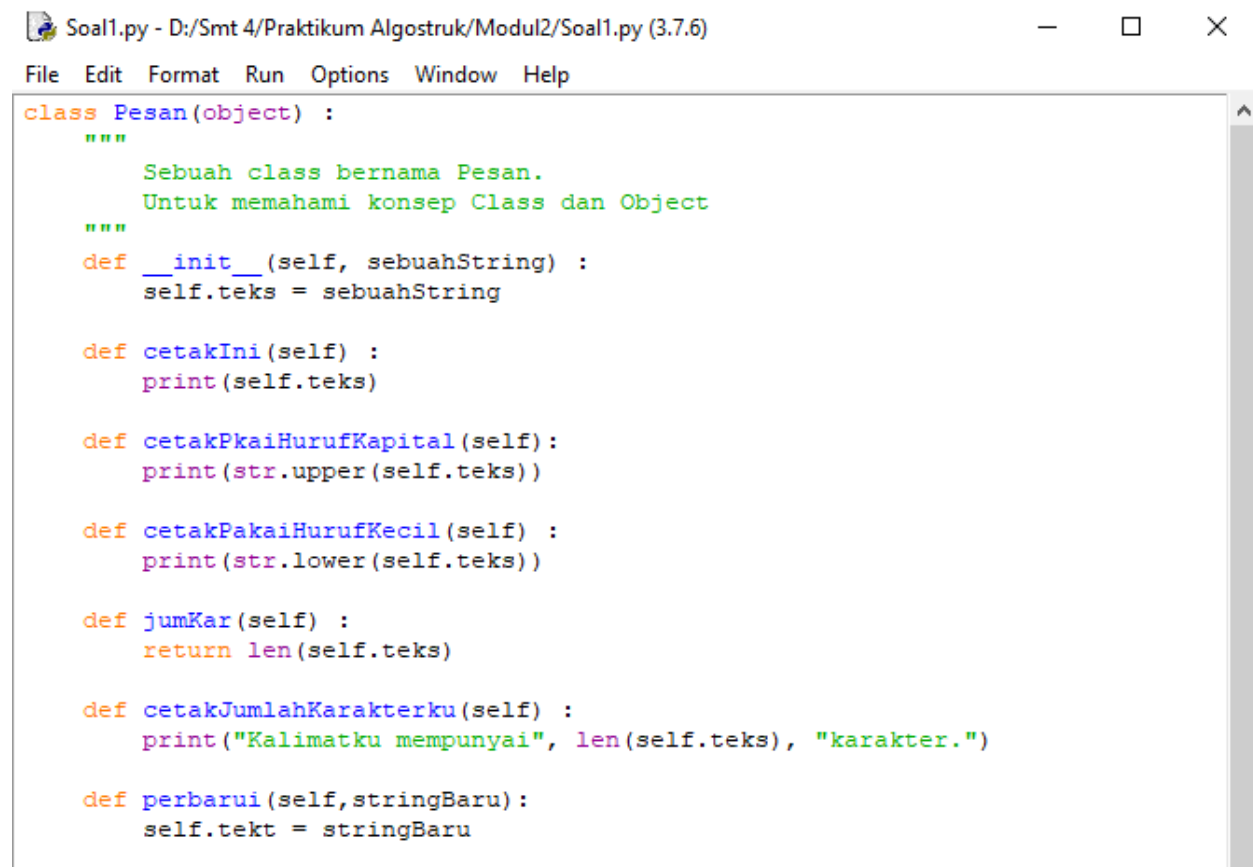
Saat dijalankan di python shell, ouputnya :



```
===== RESTART: D:/Smt 4/Praktikum Algostruk/Modul2/LatOOP7.py =====
70
Indonesia
>>>
```

2.6 Soal-soal untuk Mahasiswa

1. Dari latihan 2.2 kita telah membuat class `Pesan` yang berisi beberapa metode,



```
Soal1.py - D:/Smt 4/Praktikum Algostruk/Modul2/Soal1.py (3.7.6)
File Edit Format Run Options Window Help

class Pesan(object) :
    """
        Sebuah class bernama Pesan.
        Untuk memahami konsep Class dan Object
    """
    def __init__(self, sebuahString) :
        self.teks = sebuahString

    def cetakIni(self) :
        print(self.teks)

    def cetakPakaiHurufKapital(self):
        print(str.upper(self.teks))

    def cetakPakaiHurufKecil(self) :
        print(str.lower(self.teks))

    def jumKar(self) :
        return len(self.teks)

    def cetakJumlahKarakterku(self) :
        print("Kalimatku mempunyai", len(self.teks), "karakter.")

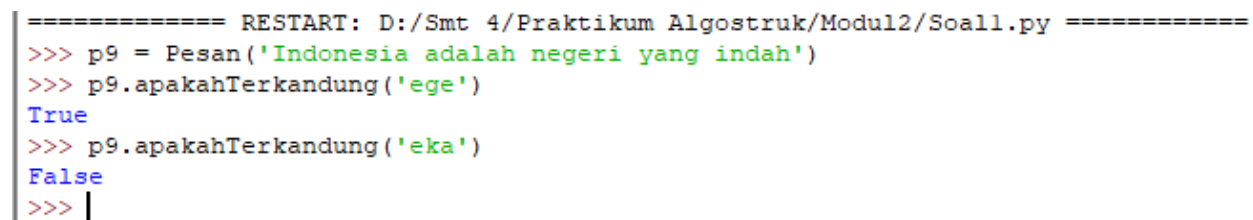
    def perbarui(self,stringBaru):
        self.tekt = stringBaru
```

ditambahkan :

- a. Metode untuk memeriksa apakah suatu string terkandung di object `Pesan` itu.

```
#(a)
def apakahTerkandung(self, kandung) :
    if kandung in self.teks :
        print(True)
    else :
        print(False)
```

Seperti ini hasilnya :



```
===== RESTART: D:/Smt 4/Praktikum Algostruk/Modul2/Soal1.py =====
>>> p9 = Pesan('Indonesia adalah negeri yang indah')
>>> p9.apakahTerkandung('ege')
True
>>> p9.apakahTerkandung('eka')
False
>>> |
```


- b. Metode untuk menghitung jumlah konsonan

```
#(b)
def hitungKonsonan(self) :
    vokal = "AIUEOaiueo"
    numvoc = 0
    teks_len = len(self.teks)
    for i in self.teks :
        if i in vokal :
            numvoc = numvoc + 1
    return teks_len - numvoc
```

Seperti ini hasilnya :

```
>>> p10 = Pesan('Surakarta')
>>> p10.hitungKonsonan()
5
>>> |
```

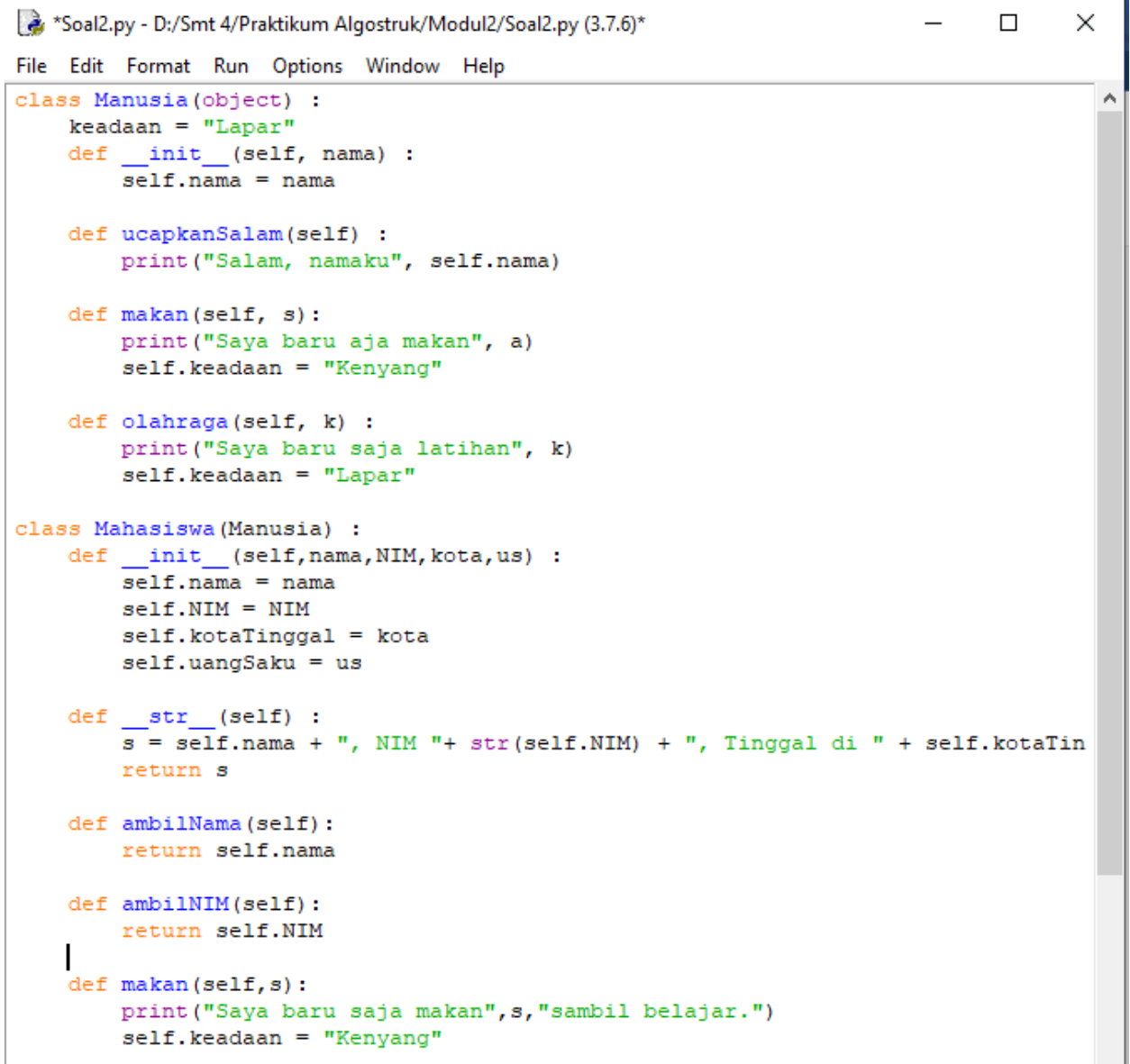
- c. Metode untuk menghitung jumlah huruf vocal

```
#(c)
def hitungVokal(self) :
    vokal = "AIUEOaiueo"
    numvoc = 0
    teks_len = len(self.teks)
    for i in self.teks :
        if i in vokal :
            numvoc = numvoc + 1
    return numvoc
```

Seperti ini hasilnya :

```
>>> p10.hitungVokal()
4
>>> |
```

2. Dari latihan 2.4,



```
*Soal2.py - D:/Smt 4/Praktikum Algostruk/Modul2/Soal2.py (3.7.6)*
File Edit Format Run Options Window Help

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 aja makan", a)
        self.keadaan = "Kenyang"

    def olahraga(self, k) :
        print("Saya baru saja latihan", k)
        self.keadaan = "Lapar"

class Mahasiswa(Manusia) :
    def __init__(self,nama,NIM,kota,us) :
        self.nama = nama
        self.NIM = NIM
        self.kotaTinggal = kota
        self.uangSaku = us

    def __str__(self) :
        s = self.nama + ", NIM " + str(self.NIM) + ", Tinggal di " + self.kotaTin
        return s

    def ambilNama(self):
        return self.nama

    def ambilNIM(self):
        return self.NIM

    def makan(self,s):
        print("Saya baru saja makan",s,"sambil belajar.")
        self.keadaan = "Kenyang"
```

ditambahkan :

- a. Metode untuk mengambil kota tempat tinggal si mahasiswa

```
#(a)
def ambilKotaTinggal(self) :
    return self.kotaTinggal
```

Seperti ini hasilnya :

```
>>> m9 = Mahasiswa('Jihan', 331, 'Surabaya', 300000)
>>> m9.ambilKotaTinggal()
'Surabaya'
>>> |
```

- b. Metode untuk memperbarui kota tinggal

```
#(b)
def perbaruiKotaTinggal(self, kotaBaru) :
    self.kotaTinggal = kotaBaru
```

Seperti ini hasilnya :

```
>>> m9.perbaruiKotaTinggal('Sleman')
>>> m9.ambilKotaTinggal()
'Sleman'
>>> |
```

- c. Metode untuk menambah uang saku

```
#(c)
def ambilUangSaku(self):
    return int(self.uangSaku)

def tambahUangSaku(self, tambahUS) :
    self.uangSaku = int(self.uangSaku)+tambahUS
```

Seperti ini hasilnya :

```
>>> m7 = Mahasiswa('Johan', 332, 'Sleman', 270000)
>>> m7.ambilUangSaku()
270000
>>> m7.tambahUangSaku(50000)
320000
>>> |
```

3. Membuat sebuah program untuk memasukkan data mahasiswa baru lewat Python Shell secara interaktif, menggunakan input().

```
a = input("Masukkan Nama      : ")
b = input("Masukkan NIM       : ")
c = input("Masukkan Kota      : ")
d = input("Masukkan Uang Saku : ")

m1 = Mahasiswa(a,b,c,d)
print(m1)
|
```

Seperti ini hasilnya :

```
Masukkan Nama      : Arindita
Masukkan NIM       : L200180058
Masukkan Kota      : Ngawi
Masukkan Uang Saku : 500000
Arindita, NIM L200180058, Tinggal di Ngawi, Uang saku Rp 500000 tiap bulannya.
>>> |
```

4. Membuat state baru di class Mahasiswa bernama listKuliah yang berupa list berisi daftar matakuliah yang diambil. Juga membuat metode ambilKuliah() yang akan menambah daftar matakuliah ini.

```
listKuliah=[]
def ambilKuliah(self, makul) :
    self.listKuliah.append(makul)
```

Seperti ini hasilnya :

```
>>> m9 = Mahasiswa('Jihan', 331, 'Surabaya', 300000)
>>> m9.listKuliah
[]
>>> m9.ambilKuliah('Matematika Diskrit')
>>> m9.listKuliah
['Matematika Diskrit']
>>> m9.ambilKuliah('Algoritma dan Struktur Data')
>>> m9.listKuliah
['Matematika Diskrit', 'Algoritma dan Struktur Data']
>>> |
```

5. Membuat metode untuk menghapus sebuah matakuliah dari listKuliah

```
def hapusKuliah(self, makul) :
    self.listKuliah.remove(makul)
```

Seperti ini hasilnya :

```
>>> m9 = Mahasiswa('Jihan', 331, 'Surabaya', 300000)
>>> m9.listKuliah
[]
>>> m9.ambilKuliah('Matematika Diskrit')
>>> m9.listKuliah
['Matematika Diskrit']
>>> m9.ambilKuliah('Algoritma dan Struktur Data')
>>> m9.listKuliah
['Matematika Diskrit', 'Algoritma dan Struktur Data']
>>> m9.hapusKuliah('Matematika Diskrit')
>>> m9.listKuliah
['Algoritma dan Struktur Data']
>>> |
```

6. Dari class Manusia, membuat class SiswaSMA yang memuat metode-metode baru.

```

class SiswaSMA(Manusia):
    def __init__(self, nama, NISN, uangSaku, alamat):
        self.nama = nama
        self.nisn = NISN
        self.uangSaku = uangSaku
        self.alamat = alamat

    def __str__(self):
        return 'Nama      : ' + str(self.nama) + "\n" + 'NISN      : ' +
            str(self.nisn) + "\n" + 'Alamat    : ' + str(self.alamat) + "\n" +
            |'Uang Saku : ' + str(self.uangSaku)

    def ambilNama(self):
        return self.nama

    def ambilNisn(self):
        return self.nisn

    def ambilUangSaku(self):
        return self.uangSaku

    def ambilKotaTinggal(self):
        return self.kotaTinggal

    def perbaruiKotaTinggal(self, stringbaru):
        self.kotaTinggal = stringbaru

    def tambahUangSaku(self, tambah):
        self.uangSaku += tambah

```

Seperti ini hasilnya :

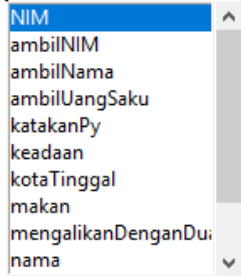
```

>>> s1 = SiswaSMA('Melvin', 201, 20000, 'Jakarta')
>>> s1.ambilNama()
'Melvin'
>>> s1.ambilUangSaku()
20000
>>> s1.ambilAlamat()
'Jakarta'
>>> s1.perbaruiAlamat('Surakarta')
>>> s1.ambilAlamat()
'Surakarta'
>>> s1.ambilNisn()
201
>>> s1.tambahUangSaku(5000)
>>> s1.ambilUangSaku()
25000
>>> |

```

7. Memberi keterangan pada setiap metode dan state yang tampak.

```
>>>
===== RESTART: D:\Smt 4\Praktikum Algostruk\Modul2\LatOOP3.py =====
>>> m8 = MhsTIF('Regan', 127, 'Madiun', 600000)
>>> m8.
```



m8.NIM → class Mahasiswa
m8.ambilNIM() → class Mahasiswa
m8.ambilNama() → class Mahasiswa
m8.ambilUangSaku() → class Mahasiswa
m8.katakanPy() → class MhsTIF
m8.keadaan → class Manusia
m8.kotaTinggal → class Mahasiswa
m8.makan() → class Manusia
m8.mengalikanDenganDua() → class Manusia
m8.nama → class Manusia dan class Mahasiswa

Kesimpulan :

Metode dan state yang tampak di object itu berasal dari semua class, dari class Manusia, class Mahasiswa, ataupun class MhsTIF. Ini merupakan penerapan konsep pewarisan. Class MhsTIF mewarisi sifat class Manusia dan class Mahasiswa. Karena class MhsTIF adalah anak kelas dari class Mahasiswa dan class Mahasiswa adalah anak kelas class Manusia.