


<p>Nama: Aini Rihhadatul Aisy</p> <p>NIM: 064102400024</p>	 <p>Praktikum Algoritma & Pemrograman</p>	<p>MODUL 11</p> <p>Nama Dosen: Binti solihah, S.T, M.KOM</p>
<p>Hari/Tanggal: Jumat, 06 Desember 2024</p>		<p>Nama Asisten Labratorium:</p> <ol style="list-style-type: none"> 1. Yustianas Rombon - 064002300015 2. Vira Aditya Kurniawan - 065002300012

Object Oriented Programming pada Python

1. Teori Singkat

Object Oriented Programming atau Pemrograman Berorientasi Objek merupakan paradigma pemrograman berdasarkan konsep "objek", yang dapat berisi data, dalam bentuk field atau dikenal juga sebagai atribut serta kode, dalam bentuk fungsi/prosedur atau dikenal juga sebagai method. Python telah menjadi bahasa berorientasi objek sejak bahasa Python sendiri dibuat.

Class

Class adalah prototype, atau blueprint, atau rancangan yang mendefinisikan variable dan method-method pada seluruh objek tertentu. Class berfungsi untuk menampung isi dari program yang akan di jalankan, di dalamnya berisi atribut / type data dan method untuk menjalankan suatu program. Dalam Python sendiri class didefinisikan dengan keyword class dan diikuti oleh penamaan kelas tersebut "*class nama_kelas*". Pemanggilan kelas sendiri sama seperti pemanggilan sebuah fungsi/method dalam sebuah program yaitu memanggil nama class tersebut beserta parameter classnya. Biasanya class berisi banyak method/fungsi yang merupakan turunan sifat dari kelas tersebut.

Class sendiri memiliki banyak bentuk dalam setiap Bahasa pemrograman yang berbeda, seperti abstrak class, data class dan lain sebagainya. Class juga dapat memiliki keterkaitan dengan class lainnya yang dapat disebut sebagai class turunan atau inheritance, Inheritance merupakan

sebuah hubungan Parent Class (Kelas Induk) dengan Child Class (Kelas Anak) yang dimana memiliki pewarisan sifat dan pewarisan variabel turunan yang sama.

Contoh Program OOP sederhana pada Python

```
class Employee:
    'Common base class for all employees'
    empCount = 0

    def __init__(self, name, salary):
        self.name = name
        self.salary = salary
        Employee.empCount += 1

    def displayCount(self):
        print ("Total Employee %d" % Employee.empCount)

    def displayEmployee(self):
        print ("Name : ", self.name, ", Salary: ", self.salary)

# Deklarasi Objek Pertama dari Employee Class
emp1 = Employee("Zara", 2000)
# Deklarasi Objek Kedua dari Employee Class
emp2 = Employee("Manni", 5000)
emp1.displayEmployee()
emp2.displayEmployee()
print ("Total Employee %d" % Employee.empCount)
```

Output





```
Name:  Zara , Salary:  2000
Name:  Manni , Salary:  5000
Total Employee 2
```

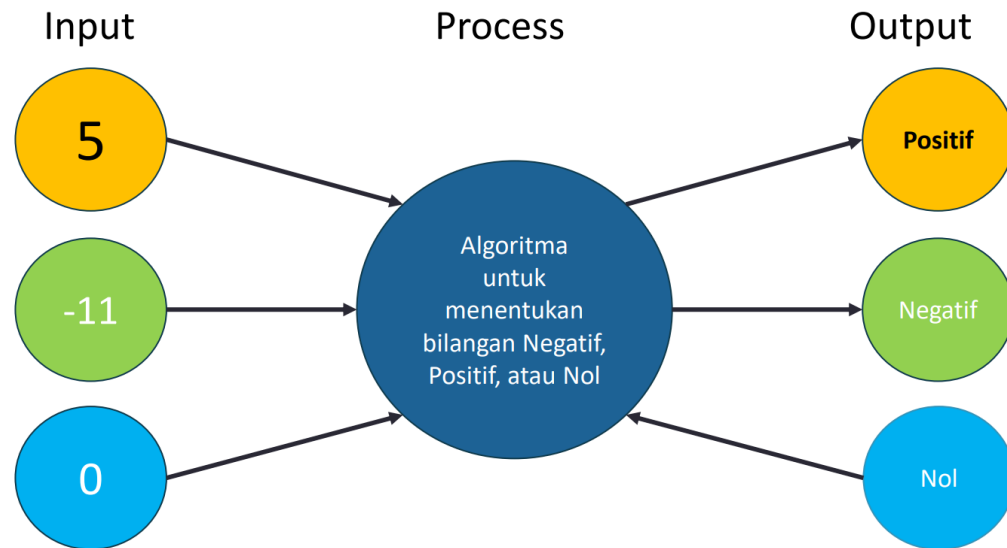
IPO (Input Process Output)

Konsep Dasar Input, Process, dan Output (IPO)

- Konsep input, process, dan output adalah prinsip dasar dalam pemrograman dan pengembangan algoritma.
- Setiap algoritma melibatkan tiga tahap utama: mengambil data masukan (input), melakukan operasi atau pengolahan data (process), dan menghasilkan hasil akhir (output).
- Konsep ini menggambarkan bagaimana algoritma beroperasi untuk memproses informasi.



Gambaran IPO (Menentukan Bilangan)



Notasi Algoritma Flowchart

1. Flowchart adalah representasi visual atau diagram alir yang digunakan untuk menggambarkan langkahlangkah dan urutan proses suatu algoritma atau program.
2. Flowchart menyajikan langkah-langkah dalam bentuk simbol-simbol grafis yang saling terhubung, membantu dalam memvisualisasikan bagaimana informasi mengalir dan bagaimana proses dilakukan.
3. Dalam kaitannya dengan notasi deskriptif, notasi algoritma yang menggunakan flowchart dapat lebih cepat dibaca dan dilihat alur dan hubungannya.

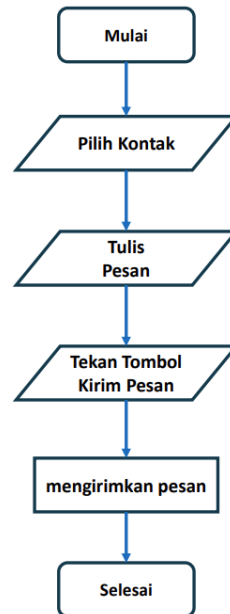
Simbol-simbol pada Flowchart

1. Setiap elemen flowchart dihubungkan oleh garis aliran bertanda panah
2. Garis aliran dimulai dari atas symbol dan keluar dari bagian bawah, kecuali symbol keputusan yang alirannya keluar dari bawah atau samping
3. Aliran bergerak dari atas ke bawah
4. Proses awal dan akhir menggunakan symbol terminal.



Contoh sederhana
Penggunaan *flowchart*
untuk menunjukan algoritma

Kasus/Aliran:
Mengirim pesan WhatsApp



2. Alat dan Bahan

Hardware : Laptop/PC

Software : Spyder (Anaconda Python)

3. Elemen Kompetensi

a. Latihan pertama

Buatlah sebuah program yang mengimplementasikan sebuah class yang memiliki nama class mahasiswa dan memiliki method yang dapat digunakan untuk menampilkan biodata mahasiswa yang diinputkan oleh user.

IPO (Input Process Output)

Input:

- Nama
- NIM
- Jurusan

Proses:

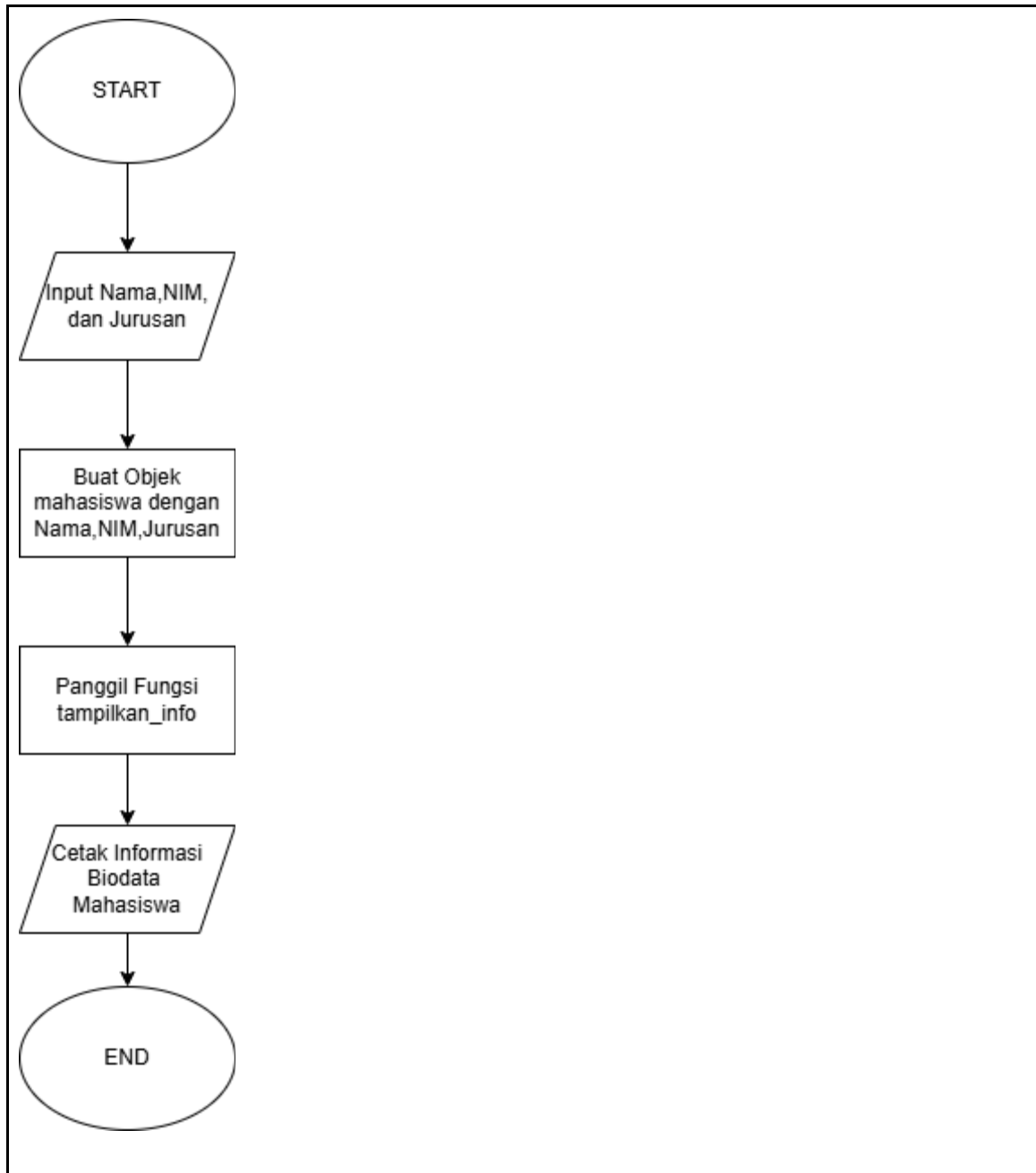
- Program meminta pengguna untuk memasukkan nama, nim, dan jurusan.
- Data yang dimasukkan disimpan sebagai atribut objek Mahasiswa.
- Objek Mahasiswa akan mencetak informasi yang berisi nama, nim, dan jurusan mahasiswa menggunakan metode tampilkan_info.

Output:

- Mencetak informasi mahasiswa



Flowchart



Source Code



```
[ ] class Mahasiswa:
    def __init__(self, nama, nim, jurusan):
        """Inisialisasi atribut Mahasiswa."""
        self.nama = nama
        self.nim = nim
        self.jurusan = jurusan

    def tampilkan_info(self):
        """Menampilkan informasi mahasiswa."""
        print("==== Biodata Mahasiswa =====")
        print(f>Nama      : {self.nama}")
        print(f"NIM       : {self.nim}")
        print(f"Jurusan   : {self.jurusan}")

nama = input("Masukkan Nama Anda: ")
nim = input("Masukkan NIM Anda: ")
jurusan = input("Masukkan Jurusan Anda: ")

mahasiswa_1 = Mahasiswa(nama=nama, nim=nim, jurusan=jurusan)
mahasiswa_1.tampilkan_info()
```

Output

```
➡ Masukkan Nama Anda: Aini Rihhadatul Aisy
Masukkan NIM Anda: 064102400024
Masukkan Jurusan Anda: Informatika
==== Biodata Mahasiswa =====
Nama      : Aini Rihhadatul Aisy
NIM       : 064102400024
Jurusan   : Informatika
```

b. Latihan Kedua

Buatlah sebuah kelas yang menerapkan method getter dan setter dimana menggunakan implementasi program percabangan serta perulangan seperti pada Latihan sebelumnya.



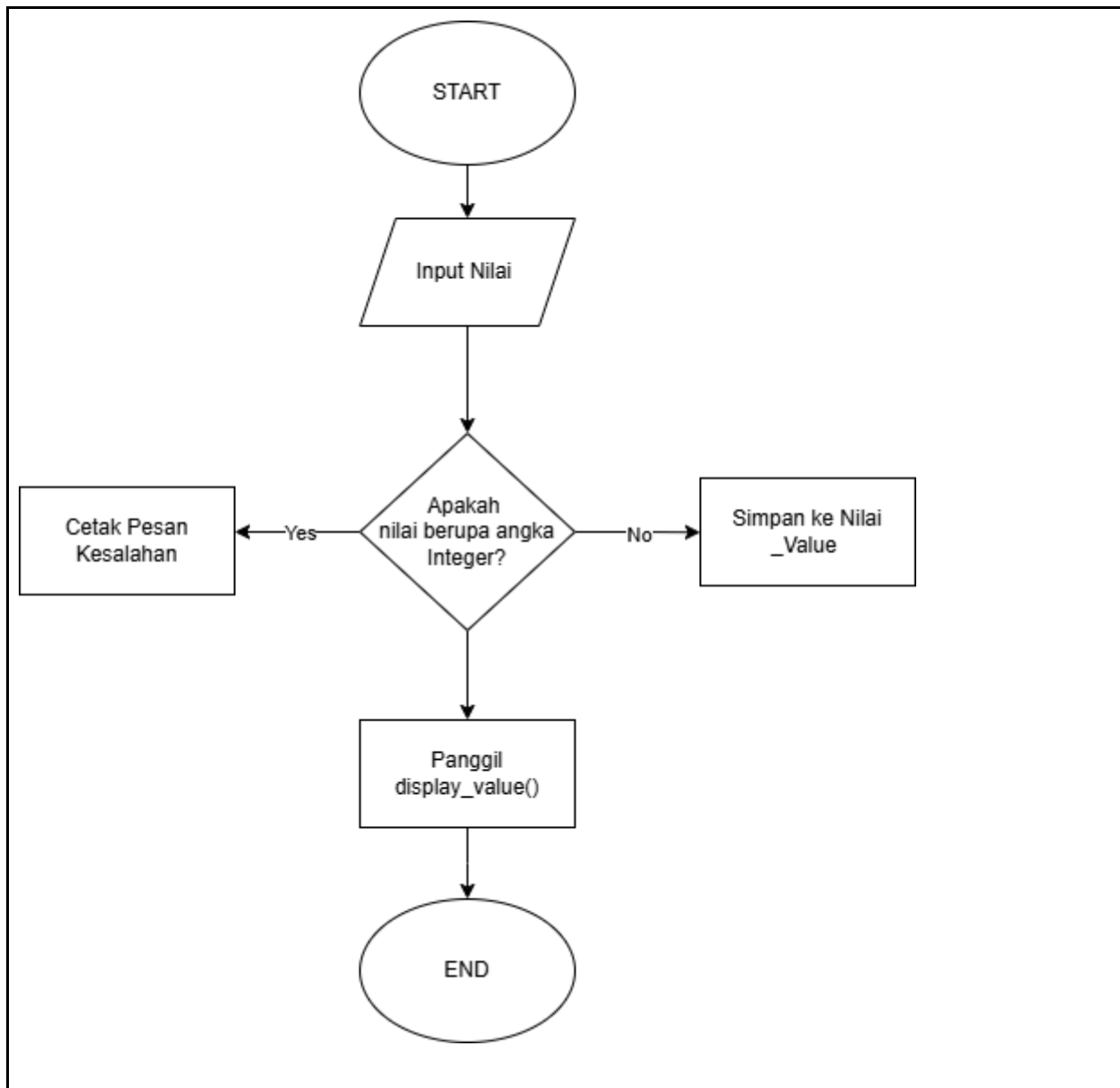
Program menerima deklarasi nilai inputan dari user dan menampungnya dalam sebuah kelas dan variabel didalam kelas tersebut dapat dimanipulasi serta dirubah sesuai keinginan dan perubahan yang diberikan oleh user melalui inputan user itu sendiri serta dapat ditampilkan menggunakan method getter dan setter.

IPO (Input Process Output)

- **Input:** Nilai integer dari pengguna untuk change_value().
- **Process:**
 1. change_value():
 - Meminta input dari pengguna.
 - Memverifikasi apakah inputan adalah integer.
 - Jika valid, setter value menyimpan nilai baru.
 - Jika tidak valid, mencetak pesan kesalahan.
 2. display_value():
 - Mencetak nilai saat ini dari atribut _value.
- **Output:**
 - Nilai _value disimpan atau pesan kesalahan jika input tidak valid.
 - Mencetak nilai saat ini setelah manipulasi.

Flowchart





Source Code



```
class SimpleInputHandler:
    def __init__(self):
        self._value = None
    @property
    def value(self):
        return self._value
    @value.setter
    def value(self, new_value):
        if isinstance(new_value, int):
            self._value = new_value
        else:
            print("Nilai inputan harus berupa integer.")
    def change_value(self):
        while True:
            try:
                new_input = int(input("Masukkan nilai baru: "))
                self.value = new_input
                break
            except ValueError:
                print("Masukkan nilai integer yang valid.")
    def display_value(self):
        print(f"Nilai saat ini: {self._value}")

input_handler = SimpleInputHandler()
input_handler.change_value()
input_handler.display_value()
input_handler.change_value()
input_handler.display_value()
```



Output

```
Masukkan nilai baru: 34  
Nilai saat ini: 34  
Masukkan nilai baru: 23  
Nilai saat ini: 23
```

4. File Praktikum

Github Repository:

5. Soal Latihan

Soal:

1. Jelaskan apa itu method getter dan setter pada sebuah class dan apa kegunaan serta fungsi method getter dan setter dalam sebuah class?
2. Deskripsikan serta narasikan jalannya alur source code program yang sebelumnya telah kalian buat pada Elemen Kompetensi Latihan Kedua!

Jawaban:

1. Method getter adalah fungsi yang digunakan untuk mengambil nilai dari atribut private dalam sebuah class, sedangkan method setter digunakan untuk menetapkan atau mengubah nilai dari atribut tersebut. Keduanya berguna untuk menjaga prinsip enkapsulasi dalam pemrograman berorientasi objek, memungkinkan kontrol penuh terhadap akses dan manipulasi atribut dalam class. Getter memastikan nilai atribut dapat diakses dengan aman, sementara setter dapat digunakan untuk memvalidasi atau memproses data sebelum menyimpannya ke atribut.
2. Program dimulai dengan mendefinisikan sebuah class bernama SimpleInputHandler, yang memiliki atribut private `_value`. Class ini dilengkapi dengan getter untuk membaca nilai `_value` dan setter untuk memvalidasi serta menetapkan nilai baru ke atribut tersebut.



Program meminta pengguna untuk memasukkan nilai melalui metode `change_value()`. Input pengguna akan diperiksa apakah berupa integer; jika valid, nilai tersebut disimpan ke atribut `_value`, tetapi jika tidak valid, pesan kesalahan akan ditampilkan. Program kemudian mencetak nilai saat ini menggunakan metode `display_value()`. Pengguna diberi kesempatan untuk mengubah nilai kembali, dan prosesnya diulang sampai input yang diinginkan disimpan dan ditampilkan.

6. Kesimpulan

- a. Dalam pengerjaan program dengan bahasa pemrograman Python, kita harus benar-benar teliti dalam menginputkan suatu fungsi untuk menampilkan suatu keluaran pada layar dengan sesuai.
- b. Kita dapat mengetahui bahwa dalam pemrograman berorientasi objek, penggunaan getter dan setter penting untuk menjaga keamanan dan validitas data pada atribut `private`. Implementasi class seperti pada latihan kedua membantu memahami cara kerja enkapsulasi, memastikan manipulasi data dilakukan dengan kontrol penuh, dan menjadikan program lebih terstruktur.

7. Cek List (✓)

No	Elemen Kompetensi	Penyelesaian	
		Selesai	Tidak Selesai
1.	Latihan Pertama	✓	
2.	Latihan Kedua	✓	

8. Formulir Umpan Balik

No	Elemen Kompetensi	Waktu Pengerjaan	Kriteria
1.	Latihan Pertama	20 Menit	Menarik
2.	Latihan Kedua	20 Menit	Menarik

Keterangan:

1. Menarik
2. Baik
3. Cukup
4. Kurang