

<p>Nama: Steven Raja P.M NIM: 064102400018</p>	 <p>Praktikum Algoritma & Pemrograman</p>	<h2>MODUL 3</h2> <p>Nama Dosen: Binti solihah, S.T, M.KOM</p>
<p>Hari/Tanggal: Jumat, 4 Oktober 2024</p>		<p>Nama Asisten Laboratorium:</p> <ul style="list-style-type: none">1. Yustianas Rombon - 0640023000152. Vira Aditya Kurniawan - 065002300012

Struktur Kendali (Control Structure)

1. Teori Singkat

Ekspresi Boolean

Ekspresi Boolean merupakan ekspresi yang mengembalikan nilai True atau False, menggunakan operator relasional/operator perbandingan, dan juga operator logika. Selain itu Ekspresi Boolean juga dapat menggunakan operator keanggotaan (*membership operator*) dan juga operator identitas dalam beberapa kasus.

Operator Perbandingan

Operator Perbandingan adalah operator yang melakukan perbandingan antara dua buah nilai. Operator ini juga dikenal dengan operator relasional dan sering digunakan untuk membuat sebuah logika atau kondisi. Berikut ini adalah daftar Operator Aritmatika dalam Python:

Operator	Simbol
Lebih Besar	>
Lebih Kecil	<
Sama Dengan	==



Tidak Sama Dengan	\neq
Lebih Besar Sama Dengan	\geq
Lebih Kecil Sama Dengan	\leq

Operator Logika

Operator Logika merupakan sebuah operator yang digunakan untuk membuat logika dalam program yang kita buat. Operator logika juga sering disebut juga sebagai Operator Aljabar Boolean, biasanya operator logika ini digunakan untuk membuat operasi percabangan pada program. Operator Logika diantaranya seperti logika AND, OR, dan NOT.

Operator logika terdiri dari:

Operator	Simbol
Logika AND	and
Logika OR	or
Logika Negasi/Kebalikan	not

Konstruksi Percabangan & Blok Program

Konstruksi Percabangan adalah sebuah program yang ketika dijalankan akan menimbulkan percabangan kedalam sub cabangnya yang berisi sebuah blok program sesuai dengan kondisi dan logika yang diminta. Umumnya konstruksi percabangan dalam Bahasa pemrograman Python sendiri dapat dibuat dengan memanggil keyword *if/elif/else*. Berikut tabelnya

Keterangan	Keyword
Terdapat 1 pilihan keputusan	<i>if</i>
Terdapat 2 pilihan keputusan	<i>if/else</i>
Terdapat lebih dari 2 pilihan keputusan	<i>if/elif/else</i>

Blok program berisi sekumpulan ekspresi dan statement untuk dikerjakan oleh komputer. Dalam Bahasa pemrograman Python blok program sendiri dapat diidentifikasi dengan tanda *colon* (“:”) setelah pendeklarasian konstruksi *if/elif/else, for, while* ataupun ketika melakukan definisi fungsi.

Blok program yang terdapat pada kondisi *if* sendiri akan dijalankan jika kondisi yang diminta bernilai *true*.



Blok program yang terdapat pada kondisi kondisi *elif* sendiri yang merupakan kepanjangan dari else if yang berarti jika tidak sesuai dengan kondisi sebelumnya maka akan disesuaikan dengan kondisi lainnya yang dapat bernilai *true*.

Blok program yang terdapat pada kondisi *else* akan dijalankan ketika nilai dari kondisi sebelumnya yaitu *if/elif* bernilai *false*.

Berikut ini adalah contoh sederhana program konstruksi percabangan yang menggunakan operator perbandingan:

Source Code

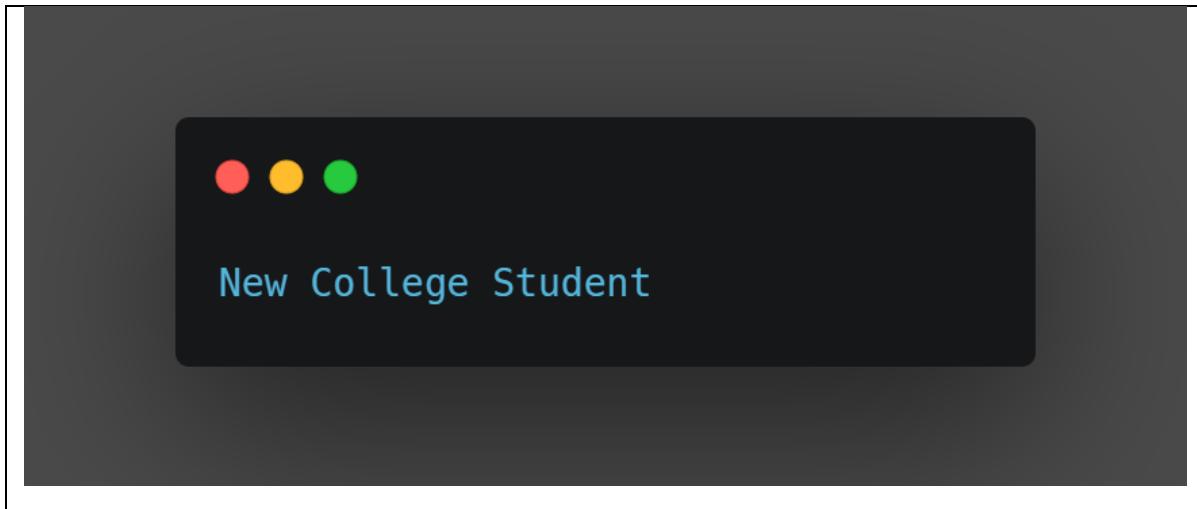
```
● ● ●

credits = 45

if credits >= 120:
    print('Senior')
elif credits >= 90:
    print('Junior')
else:
    print('New College Student')
```

Output





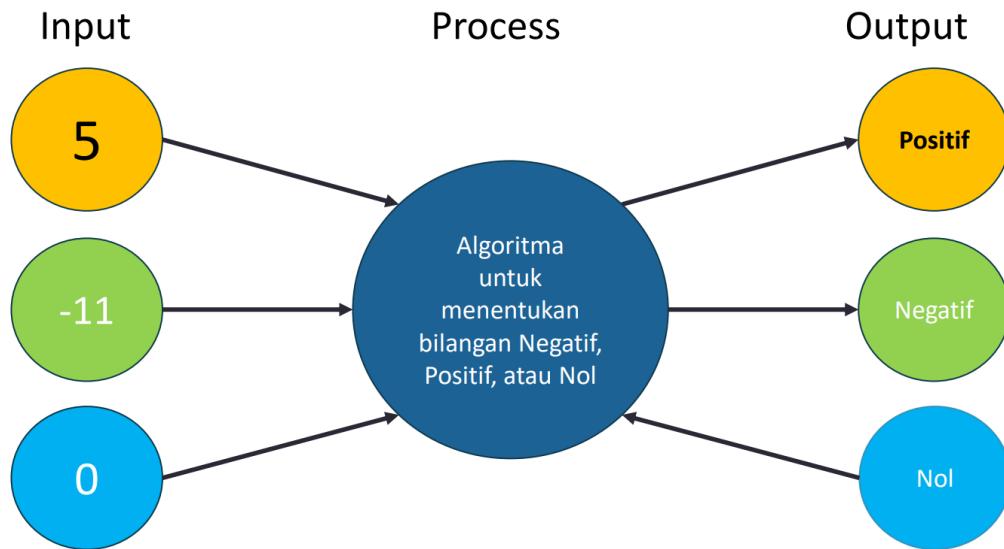
IPO (Input Process Output)

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)



Pseudocode

Pseudocode adalah suatu bentuk deskripsi informal yang mirip dengan bahasa manusia dan digunakan untuk menggambarkan algoritma atau proses secara naratif. Ini tidak terikat pada bahasa pemrograman tertentu, tetapi memberikan panduan tentang langkah-langkah yang harus diambil dalam suatu algoritma dengan bahasa yang lebih mudah dimengerti.



Contoh PseudoCode

Inisiasi Variabel:

```
N      = 0
total = 0.0
```

Pengulangan:

```
UNTUK i DARI 1 SAMPAI 10 LANGKAH 2
    CETAK i
END UNTUK
```

Pengkondisional (Conditional):

```
JIKA nilai > 10
    CETAK "Nilai lebih dari 10"
SELAINNYA JIKA nilai = 10
    CETAK "Nilai sama dengan 10"
SELAINNYA
    CETAK "Nilai kurang dari 10"
AKHIR JIKA
```

Fungsi atau Prosedur:

```
FUNGSI tambah(a, b)
    KEMBALIKAN a + b
AKHIR FUNGSI
```

Contoh Lengkap:

```
DEKLARASI variabel n, bilangan, total, rata_rata FLOAT
MINTA "Masukkan jumlah bilangan: " SIMPAN
total = 0.0

UNTUK i DARI 1 SAMPAI n
    MINTA "Masukkan bilangan ke-" + i + ": " SIMPAN bilangan
    total = total + bilangan
END UNTUK

rata_rata = total / n
CETAK "Rata-rata adalah: " + rata_rata
```

2. Alat dan Bahan

Hardware : Laptop/PC

Software : Spyder (Anaconda Python)

3. Elemen Kompetensi

a. Latihan pertama

Sebuah segitiga dibangun dari tiga garis lurus. Berdasarkan panjang dari sisi-sisinya, segitiga dapat dibedakan menjadi tiga jenis. Ada segitiga sama sisi, segitiga sama kaki, segitiga siku-siku, atau segitiga sembarang. Buatlah sebuah program yang menerima tiga bilangan yang merupakan panjang dari sisi-sisi sebuah segitiga. Berdasarkan panjang yang diberikan, program anda akan mencetak jenis segitiganya (sama sisi, sama kaki, atau sembarang). Hati-hati: Tidak semua kombinasi tiga bilangan dapat membentuk segitiga. Contoh: 1, 2, 3 tidak mungkin membentuk segitiga.



Pseudocode

Membuat 3 Variabel a b dan c dalam float

Lalu buat penjelasan yang menjelaskan bahwa itu bukan segitiga contoh:

```
sisi_a + sisi_b <= sisi_c or sisi_a + sisi_c <= sisi_b or sisi_b +  
sisi_c <= sisi_a
```

dan buat rumus segitiga yang menentukan apa itu segitiga sama sisi,sama kaki, atau sembarang contoh:

```
sisi_a == sisi_b == sisi_c:  
    Segitiga sama sisi  
    sisi_a == sisi_b or sisi_a == sisi_c or sisi_b == sisi_c:  
        Segitiga sama kaki  
    else:  
        print("Segitiga sembarang")
```

Lalu print hasil nya dan masukan angka

Input Process Output

```
|  
|  
| sisi_a = float(input("Masukkan panjang sisi a: "))  
| sisi_b = float(input("Masukkan panjang sisi b: "))  
| sisi_c = float(input("Masukkan panjang sisi c: "))  
  
| if sisi_a + sisi_b <= sisi_c or sisi_a + sisi_c <= sisi_b or sisi_b + sisi_c <= sisi_a:  
| ... print("Bukan segitiga")  
| else:  
| ... if sisi_a == sisi_b == sisi_c:  
| ... ... print("Segitiga sama sisi")  
| ... elif sisi_a == sisi_b or sisi_a == sisi_c or sisi_b == sisi_c:  
| ... ... print("Segitiga sama kaki")  
| ... else:  
| ... ... print("Segitiga sembarang")  
  
|  
|  
| Masukkan panjang sisi a: 2  
| Masukkan panjang sisi b: 2  
| Masukkan panjang sisi c: 1  
| Segitiga sama kaki
```



Source Code

```
sisi_a = float(input("Masukkan panjang sisi a: "))

sisi_b = float(input("Masukkan panjang sisi b: "))

sisi_c = float(input("Masukkan panjang sisi c: "))

if sisi_a + sisi_b <= sisi_c or sisi_a + sisi_c <= sisi_b or sisi_b + sisi_c <= sisi_a:
    print("Bukan segitiga")

else:
    if sisi_a == sisi_b == sisi_c:
        print("Segitiga sama sisi")

    elif sisi_a == sisi_b or sisi_a == sisi_c or sisi_b == sisi_c:
        print("Segitiga sama kaki")

    else:
        print("Segitiga sembarang")
```



Output

```
→ Masukkan panjang sisi a: 5  
Masukkan panjang sisi b: 7  
Masukkan panjang sisi c: 3  
Segitiga sembarang
```

- b. Latihan Kedua
Buatlah program untuk mencari Akar Persamaan Kuadrat dan Determinan

Pseudocode



Masukan variabel a, b , dan c

```
a = float(input("Masukkan koefisien a: "))
```

```
b = float(input("Masukkan koefisien b: "))
```

```
c = float(input("Masukkan koefisien c: "))
```

Masukan Rumus

```
akar = akar_persamaan_kuadrat(a, b, c)
```

```
det = hitung_determinan(a, b, c)
```

Print hasilnya dan masukan angka agar mengetahui hasilnya

Input Output Process



```
a = float(input("Masukkan koefisien a: "))
b = float(input("Masukkan koefisien b: "))
c = float(input("Masukkan koefisien c: "))

akar = akar_persamaan_kuadrat(a, b, c)
det = hitung_determinan(a, b, c)

print(f"Determinant: {det}")

if akar is None:
    print("Persamaan memiliki akar kompleks.")
elif len(akar) == 1:
    print(f"Akar tunggal: {akar[0]}")
else:
    print(f"Akar-akar: {akar[0]} dan {akar[1]}")
```

→ Masukkan koefisien a: 3
Masukkan koefisien b: 5
Masukkan koefisien c: 7
Determinant: -59.0
Persamaan memiliki akar kompleks.

[Source Code](#)



```
import math

def hitung_determinan(a, b, c):
    return b**2 - 4*a*c

def akar_persamaan_kuadrat(a, b, c):
    D = hitung_determinan(a, b, c)

    if D > 0:
        x1 = (-b + math.sqrt(D)) / (2*a)
        x2 = (-b - math.sqrt(D)) / (2*a)
        return (x1, x2)
    elif D == 0:
        x = -b / (2*a)
        return (x,)
    else:
        return None

a = float(input("Masukkan koefisien a: "))
b = float(input("Masukkan koefisien b: "))
c = float(input("Masukkan koefisien c: "))

akar = akar_persamaan_kuadrat(a, b, c)
det = hitung_determinan(a, b, c)

print(f"Determinant: {det}")

if akar is None:
    print("Persamaan memiliki akar kompleks.")
elif len(akar) == 1:
    print(f"Akar tunggal: {akar[0]}")
else:
    print(f"Akar-akar: {akar[0]} dan {akar[1]}")
```



Output

```
→ Masukkan koefisien a: 3
    Masukkan koefisien b: 5
    Masukkan koefisien c: 7
    Determinant: -59.0
    Persamaan memiliki akar kompleks.
```

4. File Praktikum

Github Repository:

```
print("Taruhan link github repository praktikum kalian disini yaa...")
```

5. Soal Latihan

Soal:

1. Dalam sebuah kasus program, terdapat sebuah kondisi percabangan *if/else*. Jika program yang dijalankan pada kondisi *if* tidak sesuai dengan kondisinya, maka itu akan menghasilkan status nilai *false* pada percabangan *if* tersebut, dan program tersebut akan masuk ke kondisi *else*, apakah status yang diberikan kondisi *else* tersebut? Jelaskan dan berikan alasannya serta deskripsikan kelanjutan dari program tersebut!
2. Deskripsikan serta narasikan jalannya alur source code program yang sebelumnya telah kalian buat pada Elemen Kompetensi Latihan Kedua!

Jawaban:

1. (Isi Dengan Jawaban Kalian)



2. (Isi Dengan Jawaban Kalian)

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... (Tolong Isi lebih dari dua baris!)

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	25 Menit	Menarik

Keterangan:

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

