|  |  |  |
| --- | --- | --- |
| **Nama:**  **Dimas Alip Priyono**  **NIM: 064102400032** | C:\Users\RPL-SI 02\Pictures\288px-Trisakti_Logo.svg.png | **MODUL 3**  **Nama Dosen:**  **Binti solihah, S.T, M.KOM** |
| **Hari/Tanggal:**  **Hari, Tanggal Bulan 2022** | **Praktikum Algoritma & Pemrograman** | **Nama Asisten Labratorium:**   1. **Yustianas Rombon - 064002300015** 2. **Vira Aditya Kurniawan - 065002300012** |

**Struktur Kendali (Control Structure)**

1. **Teori Singkat**

Ekspresi Boolean

Eskpresi 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 | != |
| Lebih Besar Sama Dengan | >= |
| Lebih Kecil Sama Dengan | <= |

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 kostruksi percabangan dalam Bahasa pemrograman Python sendiri dapat dibuat dengan memanggil keyword *if/elif/else.* Berikut tabelnya

|  |  |
| --- | --- |
| **Keterangan** | **Keyword** |
| Terdapat 1 pilihan keputusan | if |
| Terdapat 2 pilihan keputusan | if/else |
| Terdapat lebih dari 2 pilihan keputusan | if/elif/else |

Blok program berisi sekumpulan ekpresi dan statement untuk dikerjakan oleh komputer. Dalam Bahasa pemrograman Python blok program sendiri dapat diidentifikasikan 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

|  |
| --- |
|  |

Output

|  |
| --- |
|  |

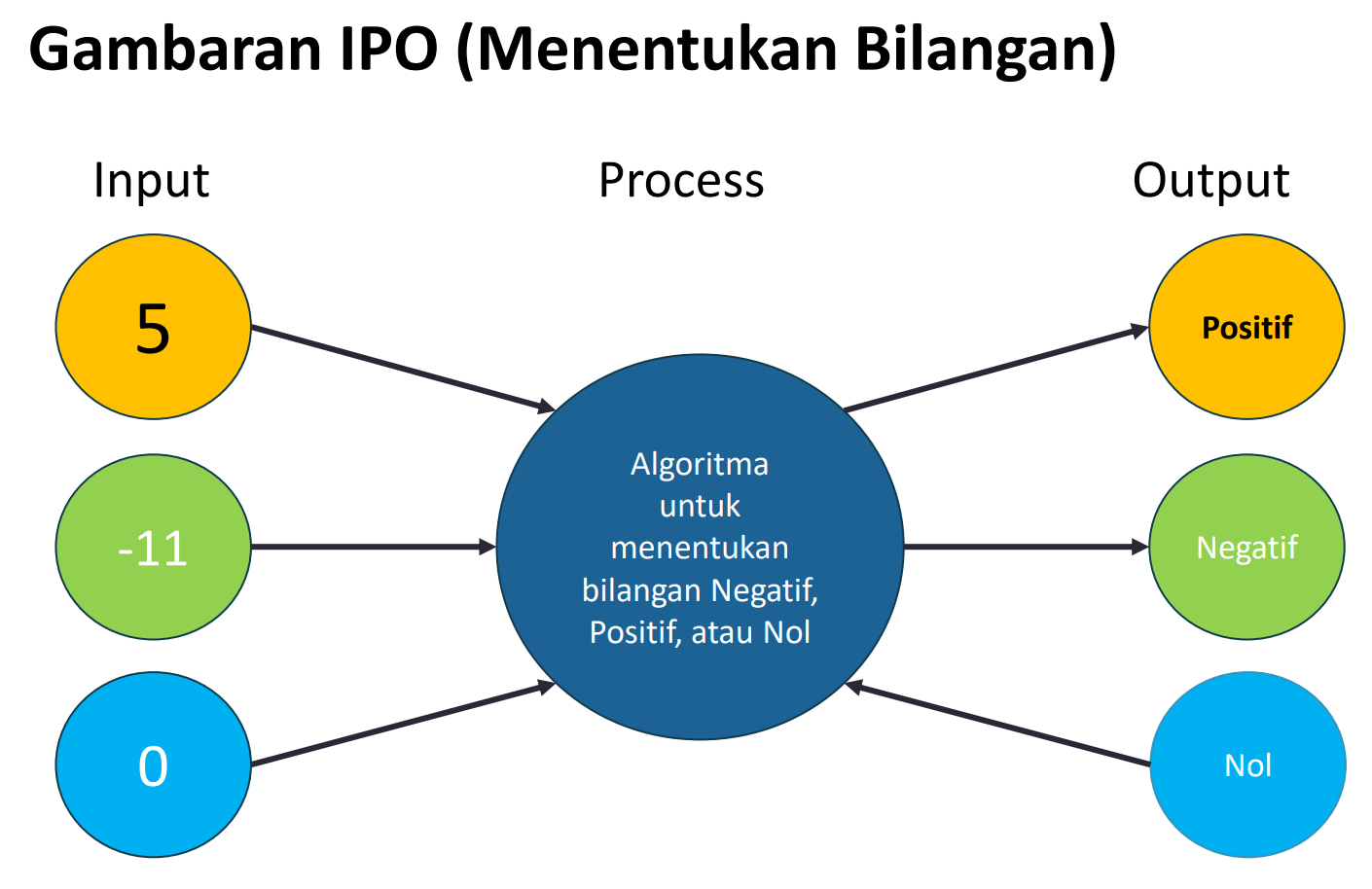
**IPO (Input Process Output)**

Konsep Dasar Input, Process, dan Output (IPO)

▪Konsep input, process, dan output adalah prinsip dasar dalam pemrograman dan pengembangan algoritma.

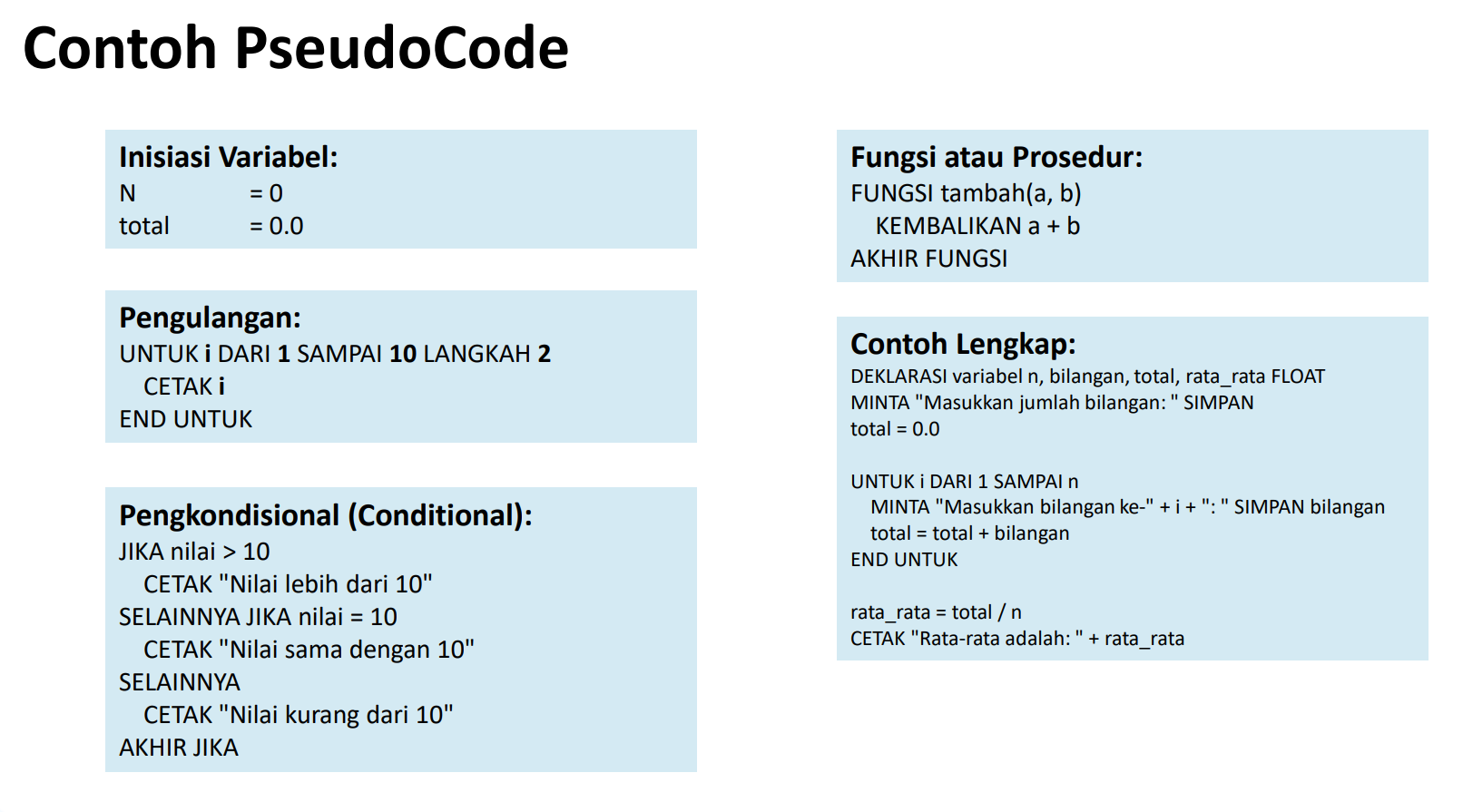
▪Setiap algoritma melibatkan tiga tahap utama: mengambil data masukan (input), melakukan operasi atau pengolahan data (process), dan menghasilkan hasil akhir (output).

▪Konsep ini menggambarkan bagaimana algoritma beroperasi untuk memproses informasi.



**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.



1. **Alat dan Bahan**

Hardware : Laptop/PC

Software : Spyder (Anaconda Python)

1. **Elemen Kompetensi**
   1. 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

|  |
| --- |
| Mulai  Input sisi\_1, sisi\_2, dan sisi\_3 sebagai float    Jika sisi\_1 + sisi\_2 > sisi\_3 dan sisi\_2 + sisi\_3 > sisi\_1 dan sisi\_1 + sisi\_3 > sisi\_2  Jika sisi\_1 == sisi\_2 == sisi\_3  Cetak "Segitiga ini adalah Segitiga SAMA SISI"    Jika tidak, jika sisi\_1 == sisi\_2 atau sisi\_2 == sisi\_3 atau sisi\_1 == sisi\_3  Cetak "Segitiga ini adalah Segitiga SAMA KAKI"    Jika tidak, jika sisi\_1^2 + sisi\_2^2 == sisi\_3^2 atau sisi\_2^2 + sisi\_3^2 == sisi\_1^2 atau sisi\_1^2 + sisi\_3^2 == sisi\_2^2  Cetak "Segitiga ini adalah Segitiga SIKU-SIKU"    Jika tidak  Cetak "Segitiga ini adalah Segitiga SEMBARANG"    Jika tidak  Cetak "Nilai yang dimasukkan tidak dapat membentuk segitiga"  Selesai |

Input Process Output

|  |
| --- |
|  |

Source Code

|  |
| --- |
| sisi\_1 = float(input('Masukan Panjang Sisi Pertama: '))  sisi\_2 = float(input('Masukan Panjang Sisi Kedua: '))  sisi\_3 = float(input('Masukan Panjang Sisi Ketiga:  '))  if sisi\_1 + sisi\_2 > sisi\_3 and sisi\_2 + sisi\_3 > sisi\_1 and sisi\_1 + sisi\_3 > sisi\_2:    if sisi\_1 == sisi\_2 == sisi\_3:      print('Segitiga ini adalah Segitiga SAMA SISI')    elif sisi\_1 == sisi\_2 > sisi\_3:     print('Segitiga ini adalah Segitiga SAMA KAKI')    elif sisi\_1 <= sisi\_2  >= sisi\_3:      print('Segitiga ini adalah Segitiga SIKU SIKU')  else:    print('Segitiga ini adalah Segitiga SEMBARANG') |

Output

|  |
| --- |
|  |

* 1. Latihan Kedua

Buatlah program untuk mencari Akar Persamaan Kuadrat dan Determinan

Pseudocode

|  |
| --- |
| ALGORITMA CariAkarPersamaanKuadrat  INPUT: a, b, c (nilai koefisien dari persamaan kuadrat)    // Cek apakah nilai a adalah 0  IF a = 0 THEN  PRINT "Nilai a tidak boleh 0 pada persamaan kuadrat."  ELSE  // Hitung determinan  determinan = b^2 - 4\*a\*c  PRINT "Determinannya adalah: " + determinan  // Tentukan jenis akar berdasarkan nilai determinan  IF determinan > 0 THEN  // Jika determinan positif, ada dua akar nyata yang berbeda  akar1 = (-b + sqrt(determinan)) / (2 \* a)  akar2 = (-b - sqrt(determinan)) / (2 \* a)  PRINT "Persamaan memiliki dua akar nyata yang berbeda: " + akar1 + " dan " + akar2  ELSE IF determinan = 0 THEN  // Jika determinan nol, ada satu akar nyata (akar kembar)  akar1 = -b / (2 \* a)  PRINT "Persamaan memiliki satu akar nyata (akar kembar): " + akar1  ELSE  // Jika determinan negatif, ada dua akar kompleks  real\_part = -b / (2 \* a)  imag\_part = sqrt(abs(determinan)) / (2 \* a)  PRINT "Persamaan memiliki dua akar kompleks: " + real\_part + " + " + imag\_part + "i dan " + real\_part + " - " + imag\_part + "i"  END IF  END IF  END ALGORITMA |

Input Output Process

|  |
| --- |
|  |

Source Code

|  |
| --- |
| import math  def cari\_akar\_persamaan\_kuadrat(a, b, c):      determinan = b\*\*2 - 4\*a\*c      print(f"Determinannya adalah: {determinan}")        if determinan > 0:          akar1 = (-b + math.sqrt(determinan)) / (2 \* a)          akar2 = (-b - math.sqrt(determinan)) / (2 \* a)          print(f"Persamaan memiliki dua akar nyata yang berbeda: {akar1} dan {akar2}")      elif determinan == 0:          akar1 = akar2 = -b / (2 \* a)          print(f"Persamaan memiliki satu akar nyata (akar kembar): {akar1}")      else:          real\_part = -b / (2 \* a)          imag\_part = math.sqrt(abs(determinan)) / (2 \* a)          print(f"Persamaan memiliki dua akar kompleks: {real\_part} + {imag\_part}i dan {real\_part} - {imag\_part}i")  a = float(input("Masukkan nilai a: "))  b = float(input("Masukkan nilai b: "))  c = float(input("Masukkan nilai c: "))  if a == 0:      print("Nilai a tidak boleh 0 pada persamaan kuadrat.")  else:      cari\_akar\_persamaan\_kuadrat(a, b, c) |

Output

|  |
| --- |
|  |

1. **File Praktikum**

Github Repository:

|  |
| --- |
| print(“Taruh link github repository praktikum kalian disini yaa…”) |

1. **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. Jika kondisi if bernilai **false,** maka blok else akan dijalankan. Blok else tidak memiliki kondisi khusus dan akan **selalu dieksekusi** jika if tidak terpenuhi. Ini berfungsi sebagai **jalur alternatif** ketika if gagal, sehingga program dapat tetap berjalan sesuai alur logika yang ditentukan.
2. Meminta Input: Pengguna memasukkan koefisien a, b, dan c.

Menghitung Determinan: D=b2−4acD = b^2 - 4acD=b2−4ac

Menentukan Jenis Akar:

Jika D > 0: Dua akar real.

Jika D = 0: Satu akar real kembar.

Jika D < 0: Dua akar imajiner.

Menghitung dan Menampilkan Akar menggunakan rumus kuadrat:

1. **Kesimpulan**
   1. Dalam pengerjaan program dengan bahasa pemrograman Python, kita harus benar-benar teliti dalam menginputkan suatu fungsi untuk menampilkan suatu keluaran pada layar dengan sesuai.
   2. Kita dapat mengetahui tentang bentuk segitiga seperti segitiga sama sisi, segitiga sama kaki, segitiga sembarang. Dan kita dapat mengetahui tentang akar persamaan kuadrat seperti D > 0, D = 0, D < 0
2. **Cek List (✔)**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Elemen Kompetensi** | **Penyelesaian** | |
| **Selesai** | **Tidak Selesai** |
| **1.** | Latihan Pertama | **Checkmark with solid fill** |  |
| **2.** | Latihan Kedua | **Checkmark with solid fill** |  |

1. **Formulir Umpan Balik**

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Elemen Kompetensi** | **Waktu Pengerjaan** | **Kriteria** |
| **1.** | Latihan Pertama | 8 Menit | menarik |
| **2.** | Latihan Kedua | 6 Menit | menarik |

Keterangan:

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