BUKU AJAR

BAHASA PYTHON :

ALGORITMA, KONSEP DASAR DAN IMPLEMENTASINYA

Disusun oleh :
Bonifacius Vicky Indriyono, S.Kom, M.Kom
Syahril Wahyu Ramadhan, S.Par., M.M

UNIVERSITAS STRADA INDONESIA
2025

Practiced using :

python™

ISBN 978-623-74724-7-8

BUKU AJAR

Bahasa Python : Algoritma, Konsep Dasar dan Implementasinya

DISUSUN OLEH

Bonifacius Vicky Indriyono, S.Kom., M.Kom
Syahril Wahyu Ramadhan, S.Par. M.M

Penerbit :

SICH PRESS

BUKU AJAR
Bahasa Python : Algoritma, Konsep Dasar dan Implementasinya

Penulis	**:** Bonifacius Vicky Indriyono Syahril Wahyu Ramadhan
ISBN	**:** 978-623-94724-9-8
Korektor	**:** Team
Penyunting	**:** Team
Desain Sampul	**:** Syahril Wahyu Ramadhan
Tata Letak	: Syahril Wahyu Ramadhan
Penerbit	: SICH PRESS
Redaksi	: Jl. Manila 37 kota Kediri Jawa Timur Indonesia
Website	: press.thesich.org
Email	**:** sichstrada@gmail.com
Kontak	085748959055
Cetakan	: Pertama, 2025

Kata Pengantar

Puji dan syukur penyusun panjatkan kehadirat Tuhan Yang Maha Esa, karena atas rahmat dan karuniaNya semata, maka penyusunan buku ajar dengan judul "Bahasa Python :Algoritma, Konsep Dasar dan Implementasinya" dapat terselesaikan tepat pada waktunya. Penyusunan buku ajar ini ditujukan agar mahasiswa lebih memahami tentang konsep dari algoritma termasuk notasi-notasi dalam algoritma dan sebagai bahan acuan dalam mempelajari teori serta konsep pemrograman menggunakan bahasa Phyton.

Dalam buku ini, setiap bab penyajian akan diberikan contoh-contoh kasus penerapan algoritma dan terutama contoh kasus yang dapat diselesaikan dengan bahasa Python. Beberapa bahasan penting yang tersaji dalam buku ini antara lain : konsep dasar algoritma, notasi algoritma, pengenalan bahasa Python, variabel, tipe data dan konstanta, fungsi masukan dan keluaran, pemrograman teknik runtutan, fungsi percabangan, perualangan, larik, fungsi dan prosedur serta beberapa pembahasan lainnya.

Akhirnya kami mengucapkan terima kasih yang sebesar-besarnya kepada semua pihak yang membantu penyelesaian Buku Ajar ini terutama keluarga penyusun yang telah banyk memberikan dukungan agar buku ini terselesaikan. Besar harapan kami buku ini dapat bermafaat bagi kelancaran proses belajar mengajar dan berguna bagi mahasiswa dalam mempelajari Algoritma dan Pemrograman menggunakan Phyton.

Kediri, Juni 2025

Penyusun

DAFTAR ISI

BAB I
KONSEP DASAR ALGORITMA

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu:
   1. Mengenal dan memahami lebih dalam tentang Algoritma.
   2. Mampu membuat contoh penyelesaian masalah dengan menggunakan konsep Algoritma.

b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami dasar-dasar komputer serta pengoperasiannya.

c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya

d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk mengenali dan memahami Algoritma pemrograman untuk menyelesaikan menyelesaikan suatu masalah sehari-hari.

B. Pokok Bahasan

Program komputer adalah sebuah rangkaian dan tahapan yang terstruktur serta sistematis untuk menghasilkan sebuah solusi dari sebuah permasalahan yang dijalankan menggunakan komputer. Banyak permasalahan rumit yang mungkin kita sebagai manusia tidak mampu untuk menyelesaikannya atau mampu menyelesaikan namun masih membutuhkan waktu yang cukup lama. Hal ini bisa disebabkan langkah dalam menyelesaikan permasalahan tersebut hars dilakukan berulang kali. Sebagai contoh sederhana ketika kita ingin menjumlahkan 15 bilangan puluhan dari sebuah barisan seperti berikut 23; 44; 35; 77; 64;

16; 24; 56; 45; 70; 55; 39; 39; 83; 96. Untuk menyelesaikan masalah tersebut, hal yang bisa dilakukan adalah dengan menjumlahkan dua bilangan puluhan yang berada dalam barisan terlebih dahulu, hasil penjumlahan dari kedua bilangan puluhan tersebut akan dijumlahkan dengan satu bilangan puluhan berikut, dan seterusnya hingga bilangan puluhan akhir yang berada di barisan.

Proses untuk menjumlahkan bilangan-bilangan puluhan tersebut merupakan proses *looping* (perulangan). Proses yang dilakukan seperti yang digambarkan dalam contoh serhana ini membutuhkan waktu yang cukup lama, misalkan penjumlahan dua buah bilangan puluhan membutuhkan waktu 8 detik, maka untuk mendapatkan jumlah dari 10 bilangan puluhan tersebut membutuhkan waktu 75 detik atau 1 menit 15 detik. Waktu 1 menit 15 detik memang tampak singkat, namun waktu tersebut hanya untuk menyelesaikan permasalahan penjumlahan 15 bilangan puluhan dalam barisan, bagaimana dengan penjumlahan 1000 bilangan ratusan dalam barisan? Atau 100 bilangan ribuan dalam barisan?

Maka dari permasalahan diatas, dapat diambil sebuah kesimpulan bahwa untuk menyelesaikan sebuah persoalan komputasi secara lebih efektif dari sisi waktu, peran program komputer sangatlah diperlukan. Hal ini sudah menjadi sesuatu yang lazim dan terus berkembang pada masa sekarang.

1. Sejarah Algoritma

Istilah Algoritma sendiri mempunyai sejarah yang cukup aneh. Orang hanya menemukan kata algorism yang berarti proses menghitung dengan angka arab. Kita dikatakan *algorist* jika kita menghitung menggunakan angka arab. Para ahli dibidang bahasa berusaha untuk menemukan asal kata Algoritma ini, namun hasilnya kurang memuaskan. Dan pada akhirnya para ahli matematika menemukan asal kata tersebut. Nama penulis buku arab yang terkenal yaitu Abu Ja‟far Muhammad Ibnu Musa Al-

Khuwarizmi. Oleh orang barat Al-Khuwarizmi dibaca menjadi *Algorism*. Al-Khuwarizmi menulis sebuah buku yang berjudul Kitab Al-Jabar Wal- Muqabala (*The book of restoration and reduction*) yang artinya Buku pemugaran dan pengurangan. Perubahan kata dari algorism menjadi algorithm muncul karena kata algorism sering dikelirukan dengan arithmetic, sehingga akhiran *–sm* berubah menjadi *–thm*. Perhitungan dengan menggunakan angka Arab menjadi hal yang biasa, maka lama kelamaan kata algorithm sering dipakai sebagai sebuah metode perhitungan (komputasi) secara umum, sehingga makna aslinya hilang. Dalam bahasa Indonesia, kata algorithm menjadi Algoritma.

2. Pengertian Algoritma

Algoritma didefinisikan sebagai urutan sejumlah langkah yang disusun secara logis dan sistematis untuk menyelesaikan suatu masalah. Dalam dunia pemrograman yang sederhana, Algoritma merupakan langkah yang pertama dan ditulis sebelum melakukan penulisan program. Suatu masalah yang dapat diselesaikan dengan pemrograman komputer adalah masalah-masalah yang berhubungan dengan perhitungan matematik.

Algoritma merupakan jantung dari ilmu komputer atau ilmu informatika. Banyak sekali bidang ilmu komputer yang mengarah ke dalam terminologi Algoritma itu sendiri. Namun tidak boleh beranggapan bahwa Algoritma selalu identik dengan ilmu komputer saja. Dalam kehidupan keseharian kita sebagai manusia, juga memiliki proses Algoritma. Contoh pada langkah atau cara orang berkirim surat. Banyak tahapan yang harus dilakukan oleh seseorang jika ingin berkirim surat ke orang lain. Ketika langkah- langkah atau cara pembuatan tersebut tidak logis atau masuk akal, maka surat yang akan kita kirim tersebut salah dan batal dikirim. Kita akan mencoba menulis dan mengirimkan surat ke seseorang yang jauh tempatnya. Maka langkah pertama kita akan menyiapkan kertas dan pena terlebih dahulu. Lalu kita mulai menulis isi surat. Secara umum, benda

yang membantu mengerjakan proses kita dalam membuat surat disebut dengan pemroses (*processor*). Pemroses tersebut bisa berupa manusia, robot, komputer atau alat-alat elektronik lainnya. Pemroses tersebut akan melakukan suatu lengkah proses dengan mengeksekusi Algoritma tersebut dengan teratur dan baik sesuai dari tujuan.

Untuk dapat lebih mudah memahami arti dari Algoritma, dicontohkan ada sebuah permasalahan penukaran isi dari dua gelas yaitu gelas X dan gelas Y. Diberikan dua buah gelas X yang isinya larutan berwarna biru dan gelas Y berisi larutan berwarna kuning (Gambar 1.1). Pertukarkan isi gelas tersebut sehingga menghasilkan gelas X yang isinya semula larutan berwarna biru menjadi isinya adalah larutan gelas Y yang isinya kuning.

Ilustrasi permasalahan ini dapat dilihat pada dibawah ini,

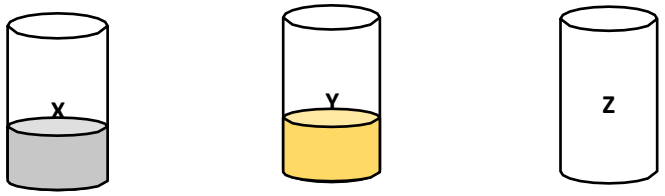
Gambar 1.1. Penukaran isi gelas isinya X dan gelas Y

Untuk menyelesaikan permasalahan ini adalah sebagai berikut. Untuk mempertukarkan isi gelas dengan benar, maka kita memerlukan sebuah gelas tambahan yang kita beri nama gelas Z yang isinya kosong sebagai tempat penampungan sementara. Berikut alur Algoritmanya:

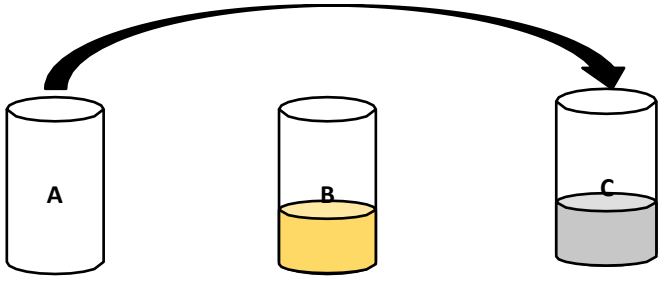
1. Siapkan gelas kosong (gelas Z)
2. Pindahkan air dari gelas X ke gelas Z.
3. Pindahkan air dari gelas Y ke gelas X.
4. Pindahkan dari gelas Z ke gelas Y.

Untuk proses pertukaran dapat di ilustrasikan dalam urutan pada gambar dibawah ini:

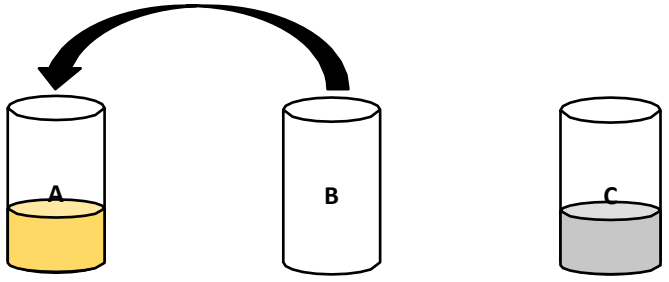
1. Keadaan awal sebelum pertukaran dengan tambahan gelas kosong (gelas Z):



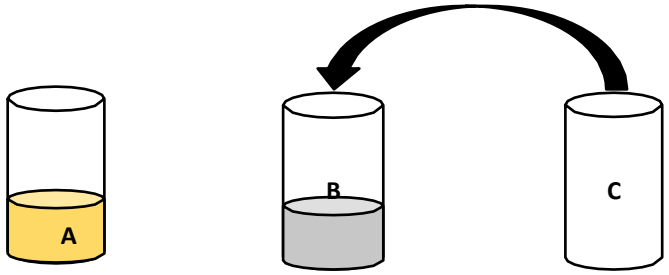
2. Pindahkan air dari gelas A ke gelas C.



3. Pindahkan air dari gelas B ke gelas A



4. Pindahkan air dari gelas C ke gelas B



Dari contoh Algoritma pertukaran isi dua buah gelas tersebut, kita bisa melihat bahwa proses penyelesaian permasalahan penukaran isi dua buah gelas sangatlah sederhana. Namun mesin komputer tentu tidak akan bisa menerjemahkan seperti halnya pemikiran kita sebagai manusia. Disini digunakan urutan langkah yang logis atau masuk akal sehingga isi dari kedua gelas X dan gelas Y sudah berpindah media, dari X ke Y dan Y ke X. Inilah yang dinamakan "Algoritma", urutan penyelesaian sebuah permasalahan dengan logis dan terurut sehingga menghasilkan sesuatu langkah yang benar atau sesui.
Berikut ini adalah contoh lain dari penggunaan Algoritma dalam kehidupan keseharian kita:

- Algoritma untuk menghitung luas Lingkaran:
  1. Mulai
  2. Menentukan nilai tetap phi.
  3. Memasukkan nilai jari-jari (R) dan menghitung luas lingkaran dengan rumus Phi*r*r
  4. Menampilkan luas lingkaran.
  5. Selesai.
- Algoritma Berangkat bekerja:
  1. Mulai
  2. Bangun tidur
  3. Siapkan baju kerja dan mulai mandi
  4. Ganti baju dan makan pagi

5. Berangkat ke kantor
6. Siapkan berkas kerja
7. Mulai bekerja
8. Selesai

- Algoritma Mengirim *E-mail*:
  1. Mulai
  2. Masuk ke layanan situs email
  3. Tulis tujuan email
  4. Ketik isi email dan lampiran jika ada
  5. Klik tombol kirim email
  6. Selesai

3. Ciri, Sifat, Struktur Dasar dan Cara Penulisan Algoritma

Pada dasarnya Algoritma itu menerima masukan (inputan), kemudian di lakukan proses sesuai urutan Langkah-langkah, dan kemudian menghasilkan keluaran. Ketika sebuah Algoritma dijalankan oleh manusia atau komputer maka langkah-langkah tersebut dikerjakan mulai dari awal sampai akhirnya berhenti dan menghasilkan solusi dari permasalahan. Jika Algoritma yang dibuat itu benar, maka keluaran yang dihasilkan pasti benar, namun sebaliknya jika algoritma yang dibuat itu salah, maka kelauran yang akan dihasilkan salah. Jadi ada dua hal penting dalam Algoritma yaitu Algoritma harus dibuat dengan Benar, dan Algoritma harus berhenti serta setelah berhenti harus memberikan solusi yang benar.

Tidak semua urutan Langkah-langkah penyelesaian masalah yang logis atau masuk akal dapat disebut sebagai Algoritma. Menurut *Donald E. Knuth di dalam Art of Komputer Programing [KNU73]*, Algoritma mempunyai lima ciri penting yang meliputi:

1. *Finiteness* (keterbatasan), Algoritma harus berakhir setelah mengerjakan sejumlah tahapan proses. Sebagai contoh pada Algoritma pertukaran isi gelas, Algoritma berhenti setelah larutan dituangkan dari gelas Z ke gelas Y.

2. *Definiteness* (kepastian), setiap langkah harus didefinisikan secara tepat dan tidak berarti ganda. Contoh, ada pernyataan "bagilah t dengan beberapa bilangan pembagi". Hal ini menimbulkan makna ganda karena makna dari "beberapa" tidak jelas. Dan akan menjadi jelas jika kita ubah menjadi "bagilah t dengan bilangan 2".
3. *Input* (masukan), Algoritma memiliki nol atau lebih data masukan (input). Yang dimaksud dengan masukan adalah parameter nilai yang diberikan kepada Algoritma untuk diproses. Pada Algoritma menghitung luas persegi Panjang adalah nilai P dan L.
4. *Output* (keluaran), Algoritma mempunyai nol atau lebih hasil keluaran (output). Keluaran bisa berupa besaran atau pesan yang berhubungan erat dengan masukan.
5. *Effectiveness* (efektivitas), Algoritma harus sangkil (efektif). Setiap langkah-langkah Algoritma harus sederhana sehingga dapat dikerjakan dalam waktu yang wajar atau masuk akal. Langkah yang masih global perlu dilakukan proses perincian sehingga menjadi langkah- langkah yang dapat diproses.

4. Sifat Algoritma

Beberapa sifat utama yang harus dimiliki oleh sebuah algoritma:

1. Tidak menggunakan simbol atau sintaks dari suatu bahasa pemrograman tertentu.
2. Tidak tergantung pada suatu bahasa pemrograman tertentu.
3. Notasinya dapat digunakan untuk seluruh bahasa manapun.
4. Algoritma dapat digunakan untuk merepresentasikan suatu urutan kejadian secara logis dan dapat diterapkan di semua kejadian sehari-hari.

5. Bahasa Pemrograman

Bahasa secara umum memiliki pengertian sebagai sarana komunikasi, dalam hal ini komunikasi anatara manusia dengan komputer. Komputer adalah mesin yang menjalankan instruksi-

instruksi di dalam Algoritma. Algoritma tersebut di masukkan ke dalan komputer kemudian komputer akan menbaca langkah-langkah di dalam Algoritma, kemudian mengerjakan operasi sesuai dengan instruksi tersebut. Instruksi-instruksi harus di tulis dengan bahasa yang bisa dipahami oleh komputer yaitu Bahasa komputer. Hal ini bertujuan tidak lain agar komputer mengerti dengan instruksi atau perintah yang di masukkan ke dalam komputer melalui Algoritma yang sudah dibuat.

Bahasa Pemrograman atau juga sering di sebut bahasa komputer adalah merupakan sebuah instruksi untuk memerintah komputer agar bisa menjalankan fungsi tertentu, namun hanya perintah standar saja. Bahasa pemrograman merupakan sebuah kumpulan dari aturan sintaks dan semantik yang tugasnya untuk mendefinisikan program komputer.

Seseorang yang bisa memahami bahasa pemrograman dapat menentukan mana data yang akan di simpan / diteruskan, data mana yang akan di olah, dan langkah apa saja yang harus di ambil dalam berbagai situasi. Algoritma yang ditulis dala Bahasa komputer di namakan program**.** Orang yang menulis program komputer dinamakan pemrogram (*programer*) dan kegiatan mulai dari mendesain sampai menulis proram dinamakan pemrograman. Teks program dalam Bahasa pemrograman dinamakan kode program (*source code*), dan proses penuliasan program dinamakan *coding*.

Belajar pemrograman tidak sama dengan belajar Bahasa pemrograman. Belajar pemrograman berarti mempelajari metodologi pemecahan suatu masalah, kemudian menuliskan Algoritma pemecahan masalahnya dalam notasi tertentu [LIE96]. Mempelajari Bahasa pemrograman adalah belajar memakai suatu Bahasa komputer, aturan tata bahasanya, perintah-perintahnya, cara meng-*compile*nya, dan memanfaatkan perintah-perintah tersebut untuk membuat program yang ditulis hanya dalam Bahasa itu saja [LIE96].

Bahasa pemrograman memiliki beberapa karekteristik, yaitu:

1. Memiliki tata Bahasa dengan aturan khusus dalam pendeklarasiannya.
2. Memiliki *Interrupt Library* untuk menerjemahkan perintah yang dinputkan.
3. Menggunakan *interpreter* atau *compiler* untuk menerjemahkan sintaks pemrogaraman kedalam Bahasa mesin.

6. Fungsi Bahasa Pemrograman

Secara umum, bahasa pemrograman berfungsi untuk memberikan sejumlah perintah kepada komputer supaya dapat mengolah data sesuai dengan perintah langkah-langkah penyelesaian yang telah dibuat oleh pembuat program.

Selain itu ada beberapa fungsi bahasa pemrograman yang perlu untuk diketahui bersama, diantaranya adalah sebagai berikut.

1) Media Komunikasi antara *Developer* dengan Komputer atau Mesin.

Pada umumnya, fungsi bahasa adalah sebagai sarana komunikasi dengan pihak lain di dalam konteks ini pihak lain tersebut adalah mesin / komputer. Begitu juga dengan bahasa pemrograman. Namun, bahasa yang hanya dimengerti oleh mesin atau komputer berbeda dengan bahasa yang digunakan manusia dalam bentuk kode-kode. Seperti contohnya mesin atau komputer hanya dapat mengerti penggunaan bilangan biner (angka 1 dan 0) untuk dapat menerjemahkan instruksi yang diberikan. Contoh Bilangan biner: 01111001.

2) Media dalam Mengembangkan Suatu Sistem

Pada jaman sekarang ini, telah banyak dikenal berbagai sistem atau aplikasi, seperti aplikasi Coreldraw, *Ms. Office*, *Kalkulator*, Winamp, *Whatsapp* dan lainnya. Aplikasi tersebut dikembangkan dengan menggunakan berbagai jenis bahasa pemrograman yang sesuai dengan penggunaanya.

Secara umum, bahasa pemrograman dapat di kelompokkan menjadi 3 level, yaitu tingkat rendah (*Low-Level*), tingkat menengah (*Medium-level*), dan tingkat paling tinggi (*High-*

Level). Berikut ini penjabaran mengenai pengelompokan tingkat bahasa pemrograman.

a. Bahasa Pemrograman Tingkat Rendah (*Low-level*)
   Bahasa pemrograman tingkat rendah berisi perintah-perintah yang ditujukan kepada komputer dengan menggunakan kode-kode biner (angka 1 dan 0). Bahasa pemrograman pada level ini lebih ditujukan pada mesin yang langsung diterjemahkan oleh komputer tanpa harus melalui proses kompilasi. Pada tingkat ini komputer belum mengenal bahasa manusia. Tingkatan rendah ini masih memiliki banyak kekurangan sehingga diciptakannya bahasa tingkatan medium. Contoh dari bahasa tingkat rendah ini adalah bahasa *assembly* atau bahasa rakitan.
b. Bahasa Pemrograman Tingkat Menengah (Medium-level)
   Bahasa pemrograman tingkat menengah berisi perintah yang diberikan berupa kode mnemonic. Namun, tingkatan medium ini masih terdapat berbagai kekurangan, salah satunya bahasa tersebut harus diterjemahkan terlebih dahulu ke dalam bahasa mesin karena komputer hanya mengerti penggunaan bahasa mesin. Assembler sendiri merupakan sebutan untuk penerjemah bahasa *assembly* kedalam bahasa mesin. Bahasa pemrograman ini dikatakan tingkat menengah karena jenis bahasa ini tidak dapat masuk ke dalam jenis bahasa tingkat tinggi atau tingkat rendah. Contoh dari bahasa tingkat menengah ini adalah bahasa yaitu Bahasa C yang diciptakan oleh Dennis Ritchie.
c. Bahasa Pemrograman Tingkat Tinggi (High-level)
   Untuk mengatasi banyak kekurangan yang dimiliki bahasa tingkat menengah, maka diciptakanlah bahasa pemrograman tingkat tinggi. Bahasa pemrograman tingkat tinggi ini mempunyai ciri yang mudah dimengerti oleh siapa saja, karena memiliki kedekatan terhadap bahasa sehari-hari. Bahasa pemrograman tingkat tinggi ini berisi instruksi-instruksi dengan menggunakan bahasa alamiah yang

dimengerti oleh manusia, seperti bahasa Inggris atau Matematika. Contohnya seperti penggunaan if dan lain sebagainya. Contoh bahasa tingkat tinggi Pascal, Java, Python, C, C++ dan masih banyak lainnya.

7. Bahasa Pemrograman Menurut Generasi

Sejarah dari Bahasa Pemrograman muncul dari generasi ke generasi berikutnya. Generasi pemrograman digolongkan menjadi 5 generasi yaitu:

1) Generasi Pertama adalah bahasa yang pertama kali dibuat yaitu Bahasa Mesin / *Machine Language*.
2) Generasi Kedua adalah penerus dari generasi pertama yaitu *Assembly Language* / Bahasa Rakitan, *Assembler*.
3) Generasi Ketiga adalah bahasa pemrograman era sekarang yang mempunyai level tinggi seperti C dan pascal.
4) Generasi Keempat adalah bahasa pemrograman seperti *SQL Database*
5) Generasi Kelima adalah generasi kelima dari bahasa pemrograman yaitu *programming languagebased object oriented* (Pemrograman Berorientasi Objek) dan *web development*.

Contoh Bahasa Pemrogaraman:

Gambar 1.2. Macam-macam Bahasa Pemrograman

C. Kesimpulan

1. Algoritma adalah langkah-langkah yang disusun secara logis dan sistematis untuk menyelesaikan suatu masalah.
2. Algoritma mempunyai lima ciri penting yang meliputi: *Finiteness* (keterbatasan), *Definiteness* (kepastian), *Input* (masukan), *Output* (keluaran), dan *Effectiveness* (efektivitas).
3. Pengelompokan bahasa pemrograman berdasarkan tingkatanya.
   a. Bahasa Pemrograman Tingkat Rendah (*Low-level*)
   b. Bahasa Pemrograman Tingkat Menengah (*Medium-level*)
   c. Bahasa Pemrograman Tingkat Tinggi (*High-level*)

D. Latihan

1. Deskripsikan pengertian dari Algoritma menurut pemikiran Anda.
2. Tulislah beberapa contoh Algoritma yang lain dalam kehidupan sehari-hari dengan langkah-langkah Algoritmanya.
3. Tuliskan Algoritma untuk membeli baju pada sebuah *marketplace* digital.
4. Silahkan Analisa Algoritma di bawah ini, temukan letak kesalahnnya:

ALGORITMA menyalakan televisi

1. Colokkan kabel televisi
2. Nyalakan televisi
3. Cari saluran televisi
4. Siaran muncul
5. Lihat siaran

5. Terdapat lima ember berkapasitas 5,5liter dengan satu ember berisi air putih dan sisisanya kosong. Di sampingnya, terdapat sebuah ember berkapasitas 1liter dengan berisi setengahnya.

Tuliskan Algoritma untuk mendapatkan keenam ember semuanya berisi air dengan volume yang sama.

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.
- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London-England. The MIT Press. 2009.

BAB II
NOTASI PENULISAN ALGORITMA

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran
   Setelah mempelajari materi dalam bab ini, Mahasiswa mampu menemukan penyelesaian alur pemrograman dalam bentuk deskriptif, *flowchart* dan *pseudocode*
b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami tentang pengertian Algoritma dan pemrograman.
c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya
d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk menggunakan elemen-elemen *flowchart* sesuai fungsinya serta mendiskusikan *flowchart* alur program pada contoh kasus membuat *pseudocode* alur programnya

B. Pokok Bahasan

1. Pengertian Notasi Algoritma
   Notasi Algoritma adalah hal dasar yang harus diketahui oleh setiap orang yang membangun suatu program/sistem, karena dalam notasi Algoritma itulah terdapat kerangka-kerangka suatu program yang akan dibangun. Penjelasan tahapan dalam Algoritma tidak mengacu pada sintaks pemrograman apapun dan tidak tergantung pada spesifikasi komputer yang mengeksekusinya. Tidak ada aturan yang baku dalam menuliskan Algoritma, yang terpenting adalah mudah dibaca dan dipahami. Meski demikian untuk menghindari kekeliruan dan salah pemahaman sehingga notasi Algoritma perlu

diperhatikan. Secara umum terdapat tiga bentuk dalam menuliskan Algoritma yaitu sebagai berikut:

1. *Deskriptif*
2. *Pseudecode*
3. *Flowchart*

1) Kalimat Deskriptif

Notasi Algoritma dengan menggunakan kalimat deskriptif disebut juga notasi alami. Notasi Algoritma deskriptif dilakukan dengan cara menuliskan intruksi-intruksi yang musti dilaksanakan dalam bentuk untaian kalimat deskriptif dengan menggunakan bahasa yang jelas. Notasi deskriptif ini disarankan untuk Algoritma yang pendek karena apabila untuk Algoritma yang panjang notasi deskriptif kurang efektif. Secara garis besar notasi deskriptif tersusun atas tiga bagian utama, yaitu:

1. Bagian judul (*header*)
   Merupakan bagian yang terdiri atas nama Algoritma dan penjelasan atau spesifikasi Algoritma tersebut.
2. Bagian deklarasi (kamus)
   Merupakan bagian untuk mendefinisikan semua nama yang digunakan pada Algoritma dapat berupa variabel, konstanta, tipe ataupun fungsi
3. Bagian deskripsi
   merupakan bagian inti pada struktur Algoritma yang berisi uraian langkah-langkah penyelesaian masalah.
   Setiap bagian diatas diberikan dengan komentar untuk memperjelas maksud teks yang dituliskan.

Berikut ini adalah beberapa contoh Algoritma yang penulisannya ditulis secara deskripsi.

1. *Algoritma Meminjam Buku di Perpustakaan*:

- Menuju ke lokasi perpustakaan
- Siapkan kartu identitas (KTP/SIM/Kartu pelajar)
- Cari rak buku yang diinginkan

- Cari buku yang akan dipinjam
- Menanyakan ketersediaan buku ke petugas.
- Serahkan buku yang dipinjam
- Petugas memasukkan data buku yang dipinjam.

2. Algoritma Mengambil Uang di ATM:

- Mulai
- Memasukan kartu dimesin ATM
- Pilih bahasa
- Masukan *password*
- Pilih menu Tarik tunai
- Pilih jenis rekening
- Masukan jumlah uang yang ingin diambil
- Jika saldo mencukupi, mesin ATM akan mengeluarkan uang sesuai dengan jumlahnya, Jika tidak maka kembali ke nomor 6
- Ambil uangnya
- Ambil kartu ATM
- Selesai

3. Algoritma Mengirim Whatsapp

- Buka Menu Whatsapp di HP
- Cari kontak tujuan
- Ketikan pesan
- Tekan tombol kirim
- WA akan terkirim

2) Flowchart

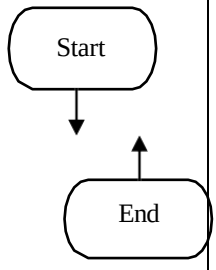
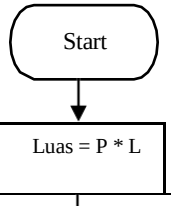
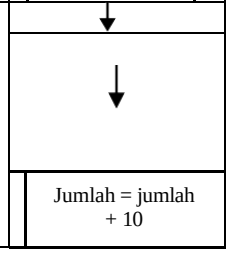
Flowchart bisa diartikan sebagai diagram alir atau bagan yang menggambarkan langkah-langkah dan struktur suatu Algoritma atau program. Ilustrasi ini dinyatakan dalam simbol, setiap simbol mempunyai makna tertentu untuk proses tertentu. Simbol-simbol *flowchart* yang umumnya digunakan adalah simbol-simbol *flowchart* standart yang dikeluarkan oleh ANSI dan ISO.

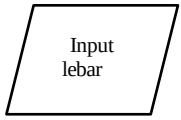
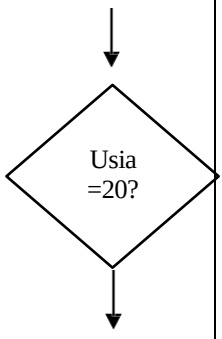
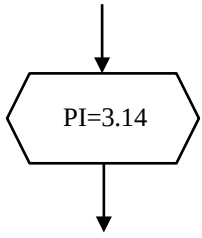
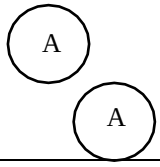
Aturan umum dalam *flowchart*:

- Semua simbol dari *flowchart* dihubungkan oleh garis aliran (*arrows*) yang mengindikasikan arah aliran bukan garis biasa.
- Garis aliran memasuki bagian atas simbol dan keluar dari bagian bawah, kecuali untuk simbol keputusan (decision), yang memiliki garis aliran yang keluar dari bawah atau samping.
- Aliran proses bergerak dari atas ke bawah.
- Awal dan akhir pada *flowchart* disimbolkan dengan terminal.

Berikut ini adalah tabel dari simbol-simbol dalam *flowchart* yang sering digunakan.

Tabel 1.1 Simbol-simbol dalam *Flowchart.*

Simbol	Nama	Fungsi	Contoh
	Terminator	Menandakan Start (awal)/ End (akhir) program.	
	Flow line	Menunjukkan arah aliran proses pada program	
	Proses	Menunjukkan proses yang dilakukan komputer (perhitungan/	

		pengolahan)	
	Input/ output data	Menandakan proses input/ output data	
	Decision atau kondisi	Menandakan pernyataan pilihan, berisi suatu kondisi yang selalu menghasilkan 2 nilai keluaran yaitu benar atau salah	
	Preparation	Proses deklarasi pemberian nilai-nilai awal pada variable yang digunakan.	
	On Page Connector	Penghubung Flow chart pada satu halaman	

Berikut ini adalah keuntungan dari metode penulisan Algoritma dengan *flowchart*:

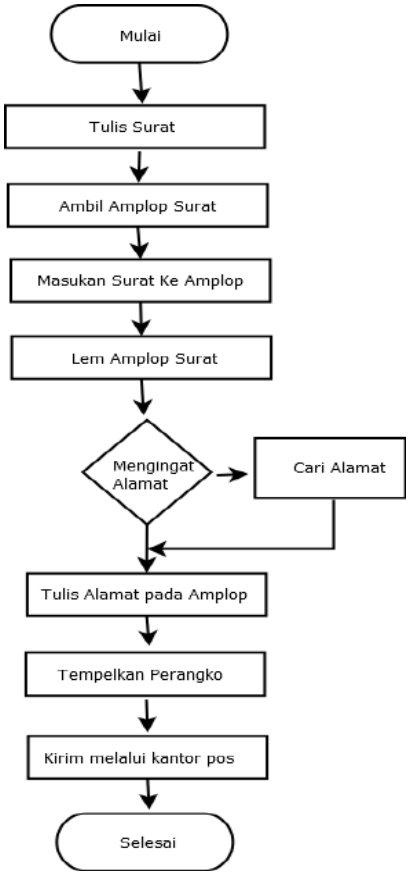
a. Memberikan kemudahan kepada setiap orang yang membaca untuk memahami Algoritma.
b. Sangat cocok untuk digunakan pada studi kasus Algoritma yang kerumitannya pada level tingkat rendah.
c. Aliran proses programnya sangat rinci dengan pemodelan secara visual dengan symbol.

Namun, tidak hanya kemudahannya saja, *flowchart* juga memunyai kekurangan diantaranya:

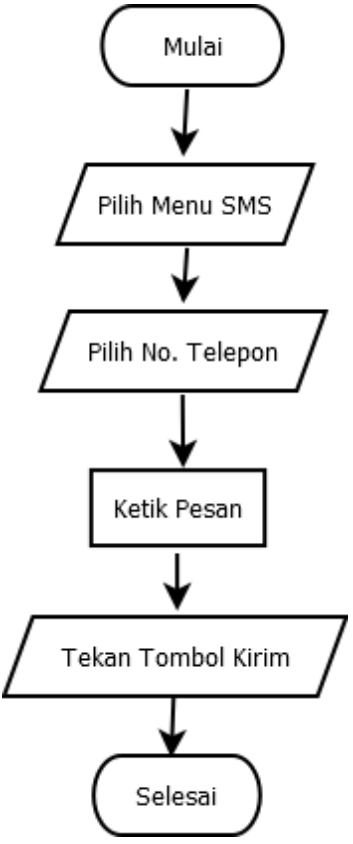
a. Tidak cocok untuk merepresentasikan alur program yang komplek.
b. Membutuhkan tools khusus untuk merancang Algoritmanya.
c. Penjelasan dalam alur proses tidak detail karena keterbatasan ruang.

Contoh Algoritma yang di tulis dengan *flowchart*:

Algoritma mengirimkan surat:



Algoritma mengirimkan SMS:



3) *Pseudo Code*

Pseudecode merupakan cara penulisan Algoritma yang menyerupai bahasa pemrograman tingkat tinggi. Pada umumnya notasi *pseudecode* menggunakan bahasa yang mudah dimengerti secara umum dan juga lebih ringkas dari pada Algoritma.

Pseudecode berisi deskripsi dari Algoritma pemrograman komputer yang menggunakan struktur sederhana dari beberapa bahasa pemrograman tetapi bahasa tersebut hanya ditujukan agar bisa terbaca dan dimengerti manusia, sehingga *pseudecode* tidak dipahami oleh komputer. Agar notasi *pseudecode* dapat dimengerti oleh komputer maka musti diterjemahkan ke dalam sintaks bahasa pemrograman tertentu. Pada notasi pseudecode tidak aturan tertentu yang resmi. Disarankan untuk menggunakan kata kunci yang umum digunakan seperti *if, then, else, while, do, for, repeat* dan lainya.

Penulisan Pseudo code memiliki beberapa pedoman sebagai berikut:

a. Memiliki bagian Header / Kepala yang menunjukkan judul dari Algoritma, Komentar, dan Deklarasi.
b. Memiliki badan Algoritma
c. Memiliki bagian akhir Algoritma yang menandakan bagian akhir dari suatu Algoritma
d. Untuk menuliskan komentar diawali dengan karakter "{"dan diakhiri dengan karakter "}".

Berikut ini adalah contoh dari pseudocode:

```
program hitung_luas_segitiga

deklarasi
var luas, alas, tinggi : integer;

algoritma:
alas <-- 5;
tinggi <-- 10;

luas <-- (alas * tinggi)/2;

write(luas);
```

Judul

Deklarasi

Deskripsi / Algoritma

```
program hitung_luas_segi_panjang

deklarasi
    var panjang,lebar,luas:integer;

algoritma:
    read(panjang);
    read(lebar);

luas <– panjang * lebar;

print(luas);
```

C. Kesimpulan

1. Notasi Algoritma adalah hal dasar yang musti diketahui oleh setiap orang yang membangun suatu program, karena dalam notasi Algoritma itulah terdapat kerangka-kerangka suatu program yang akan dibangun.
2. Secara umum terdapat tiga cara dalam menuliskan Algoritma yaitu sebagai berikut:
    a. Deskriptif
    b. *Pseudecode*
    c. *Flowchart*

D. LATIHAN

1. Jelaskan Keuntungan dan kerugian Algoritma dengan *flowchart*.
2. Buatlah Algoritma mengirim mesan lewat email dengan kalimat deskriptif dan *flowchart*.
3. Buatlah Algoritma untuk permasalahan menghitung luas setiga dengan menggunakan *flowchart*.
4. Buatlah Algoritma untuk mencari nilai terbesar dari 2 buah bilangan menggunakan *pseudocode*.
5. Tentukan *pseudocode* luas persegi panjang yang mana lebar dan banjangnya menggunakan inputan.

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.
- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan-python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB III
PENGANTAR PYTHON

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran

   Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu menginstal dan menggunakan tools yang digunakan dalam pemrograman python.

b. Kompetensi yang Diharapkan;

   Mahasiswa telah memahami tentang notasi Algoritma.

c. Keterkaitan materi dengan materi yang lain;

   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya

d. Pentingnya mempelajari isi bab;

   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk menginstal dan cara menggunakana python untuk membuat program.

B. Pokok Bahasan

Dalam pembuatan program komputer dibutuhkan sebuah bahasa yang dapat di tranformasi ke dalam bahasa mesin.

Mengapa bahasa mesin? Karena program yang akan kita jalankan menggunakan komputer, dan komputer tidak memahami bahasa yang digunakan oleh manusia. Komputer adalah sebuah mesin yang dirancang oleh manusia, sehingga bahasa yang digunakan oleh komputer adalah bahasa mesin yang disebut dengan assembly dengan bilangan biner 0 dan 1. Oleh karena itu, agar perintah yang diberikan oleh manusia kepada komputer dapat dijalankan, maka bahasa yang digunakan oleh manusia sebagai perintah harus ditransformasi ke dalam bahasa mesin yang disebut bahasa pemrograman.

Tidak banyak orang yang mampu membuat kode program sebagai perintah manusia yang dapat dijalankan oleh komputer. Hanya orangorang yang Mempelajari pemrograman komputer yang mampu membuat kode. program.

Python adalah bahasa pemograman interpreatif yang dianggap mudah dipelajari serta berfokus pada keterbacaan kode. Dengan kata lain python diklaim sebagai bahasa pemograman yang memiliki kode-kode pemograman yang sangat jelas dan mudah dipahami, lengkap, dan mudah. Python secara umum berbentuk pemograman berorientasi objek, pemograman imperetatif, dan pemograman fungsional. Python juga dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai *platform* sistem operasi. Dan juga python memiliki lisensi yang dapat diperoleh dan dipergunakan secara bebas oleh siapapun, bahkan untuk para *developer* yang menggunakan bahasa ini untuk kepentingan komersial.

1. Sejarah Python

   Bahasa pemrograman Python dirilis pertama kali oleh Guido van Rossum di tahun 1991, yang sudah dikembangkan sejak tahun 1989. Awal pemilihan nama Python tidak secara langsung berasal dari nama ular piton, tapi sebuah acara humor di BBC pada era 1980an dengan judul "Monty Python"s Flying Circus". Monty Python adalah kelompok lawak yang membawakan acara tersebut. Kebetulan Guido van Rossum adalah penggemar dari acara ini. Pada tahun 1994, Python 1.0 dirilis, yang diikuti dengan Python 2.0 pada tahun 2000. Python 3.0 keluar pada tahun 2008.
2. Keunggulan Python

   Python memiliki keunggulan yang bisa membuat penggunanya senang dan banyak memilih menggunakan bahasa Pemrograman ini dibandingkan yang lainnya. Keunggulana yang dimiliki adalah:

   a. Python memiliki konsep atas desainnya yang sederhana dan

bagus, yang terfokus dengan kemudahan penggunaannya. Kode dalam Python dirancang supaya lebih mudah untuk membacanya, mempelajarinya, bisa digunakan ulang, juga dirawat.

b. Python bisa mendukung berbagai pemograman yang berorientasi kepada objek serta pemograman secara fungsional.
c. Python bisa meningkatkan segala produktivias juga menghemat waktu untuk para programmer. Supaya dalam mendapatkan hasil program bisa sama, dengan kode Python yang jauh sangat sedikit dibandingkan menggunakan kode yang tertulis untuk menggunakan bahasa-bahasa pada pemograman lain seperti C, C++, C# maupun Java.

Program yang tertulis di penggunakan Python ini bisa dijalankan pada semua jenis sistem operasi (Windows, Unix, Mac OS X, dan lain sebagainya) dan juga bisa digunakan untuk perangkat-perangkat di *mobile*.

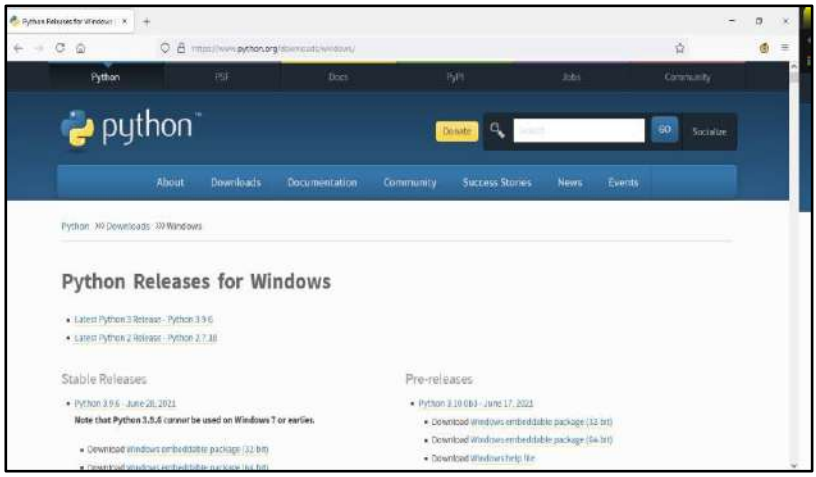
Python mempunyai berbagai dukungan pustaka dan dikembangkan dan dirancang oleh pihak ketiga, contohnya para pustaka dengan kegunaan pengembangan web, serta pengembangan sebuah aplikasi secara visual dengan basis GUI, melakukan pengembangan dalam permainan komputer atau game, juga banyak yang lainnya. Dengan menggunakan mekanisme tertenu, adanya kode Python bisa dnegan mudah diintegrasikan dengan aplikasi yang tertulis di dalam bahasa pemograman yang lainnya.

Bahasa pemrograman Python sifatnya gratis atau bisa didapatkan secara bebas atau free juga open source, meski digunakan dalam kepentingan secara komersial. Kelebihan di bahasa Pemrograman Python inilah yang menjadikan Python menjadi salah satu bahasa Pemrograman yang banyak dipilih oleh para pengguna. Namun jangan senang dulu bagi anda yang ingin memulai belajar tentang bahasa Pemrograman Python, namun sebelum itu anda perlu mengetahui bagaimana cara mudah membuat sebuah script dengan bahasan di bawah ini.

3. Instalasi Python di Windows

   Instalasi Python sangat gampang sekali, seperti halnya kita menginstal software lain pada umumnya di windows. Hanya ada konfigurasi yang harus kita lakukan di tengah-tengan proses install. Berikut ini adalah Langkah-langkah instalasi Python:

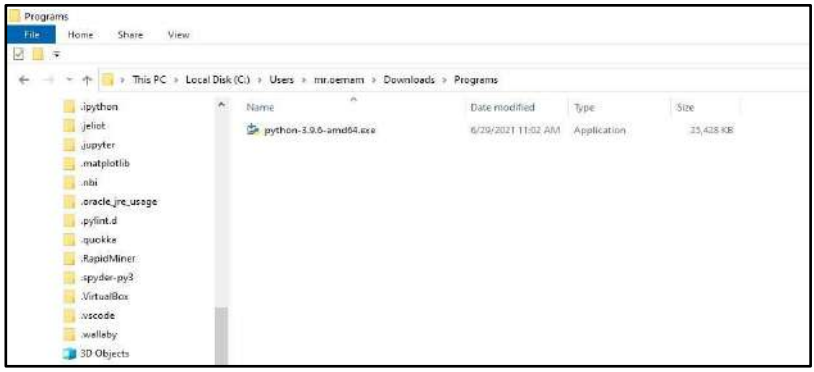
   1. Buka situs *https://www.python.org/downloads/windows/* untuk mendownload softwarenya.



   2. Kita pilih versi yang kita inginkan atau memilih versi terbaru yang ditawarkan oleh penyedia python yaitu versi Python 3.9.6.

   3. Silahkan download file instalasinya.

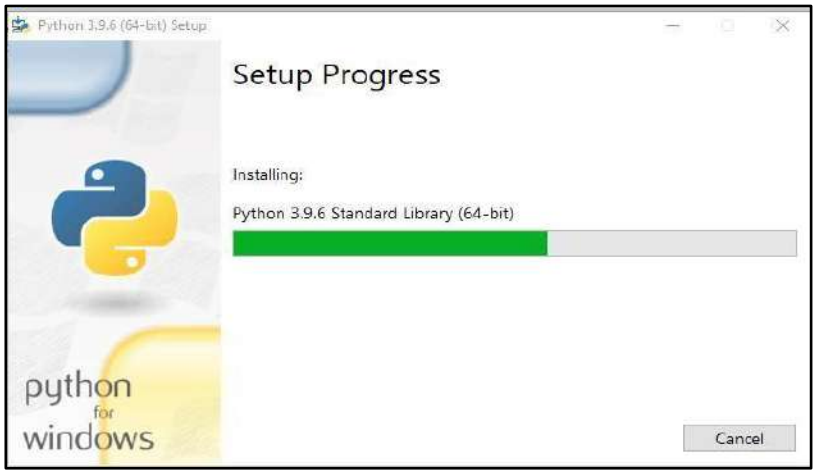
4. Setelah proses *download* selesai, klik ganda file yang sudah selesai di *download* utnuk memulai menginstal



5. Setelah muncul jendela instalasi, tentukan lokasi python akan di install. Biarkan saja defaultnya.

6. Setelah itu tekan Install Now.

7. Tunggu sampai proses instalasi selesai.



8. Pilih close apabila proses instalasi sudah selesai.

9. Proses instalasi selesai

4. Membuka dan Menjalankan Python

Setelah proses instalasi selesai, kita dapat mengakses Python melalui *All Programs* > Python 3.9

Terdapat beberapa aplikasi yang kita gunakan

a. Python IDLE Shell
b. IDLE adalah kependekan dari Python's Integrated Development and Learning Environment yang merupakan IDE standar Python.

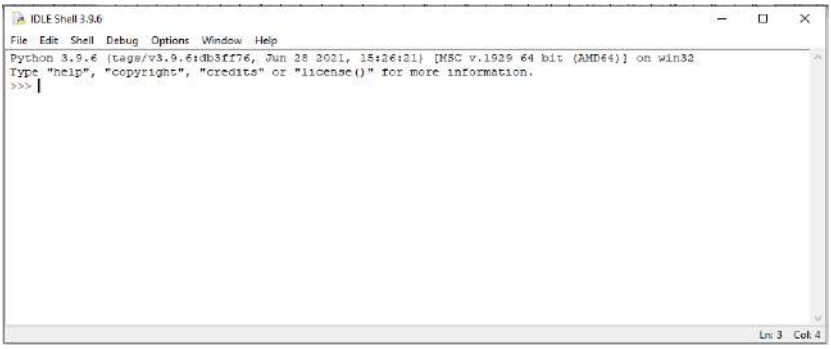
c. Python 3.9

Python 3.9 untuk membuka python melalui command prompt.

d. Python 3.6 Manual
Panduan manual Python.

e. Python Module 3.6 Doc

Panduan mengenai module pada Python.



C. Kesimpulan

1. Bahasa pemrograman Python dirilis pertama kali oleh Guido van Rossum di tahun 1991.
2. Python merupakan aplikasi yang dapat digunakan oleh pemrogram secara gratis (open source), dan dapat di unduh di web resmi python *https://www.python.org/*.
3. Python memiliki konsep atas desainnya yang sederhana dan bagus, yang terfokus dengan kemudahan penggunaannya sehingga banyak di gunakan oleh programmer.

D. Latihan

1. Download Python di *https://www.python.org/* dengan meilih versi terbaru.
2. Instal Python pada laptop atau computer masing-masing.

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.
- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB IV
TIPE DATA, KONSTANTA, VARIABEL DAN NILAI

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu mengaplikasikan konsep tipe data, variable, konstanta dan operator dalam pemrograman.
b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami pengoperasian bahasa pemrograman python.
c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya
d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk membuat variabel menggunakan tipe data yang cocok sesuai dengan konteks permasalah yang ingin di selesaikan serta dapat membedakan operator yang ada di python.

B. Pokok Bahasan

1. Pengertian Tipe Data

Dalam dunia pemrograman kita akan bertemu dengan istilah tipe data yang berfungsi untuk melakukan identifikasi suatu nilai ke alamat memory komputer. Biasanya tipe data data ditampung kedalam variabel agar nilai dari tipe data tersebut dapat diolah untuk keperluan tertentu. Tipe Data adalah sebuah pengelompokan data untuk memberitahu *compiler* atau *interpreter* bagaimana pembuat program ingin mengolah data tersebut. Sederhananya, tipe data kita dapat diartikan sebagai suatu cara memberitahu komputer kita untuk mengelompokkan

atau menggolongkan data berdasarkan apa yang dimengerti oleh komputer.

Sebagai contoh, misalkan kita memiliki sebuah data berupa angka. Agar bisa dipahami oleh *interpreter* bahasa Python, data ini harus disimpan ke dalam sebuah variabel. Kemudian variabel ini akan diproses sesuai dengan tipe data angka, misalnya bisa ditambah, dikalikan, dibagi, dst. Jika suatu variabel tersebut berisikan teks (*string*), maka operasi penambahan atau pengurangan tidak akan bisa dilakukan. Setiap jenis tipe data akan memiliki sifat dan fitur masing- masing.

Dalam bahasa pemrograman python, terdapat 9 tipe data dasar:

1. String
2. Integer
3. Float
4. Complex Number
5. Boolean
6. List
7. Tuple
8. Set
9. Dictionary

2. Tipe Data String

String adalah sekumpulan karakter atau huruf. Di python, tipe data string ditulis (str). Objek string dibatasi oleh tanda kutip satu (‘) atau kutip dua (“), jadi semua karakter di dalam tanda kutip, baik itu berupa huruf, angka, atau karakter unik, merupakan bagian dari string tersebut.

String adalah tipe data yang anggotanya berurut dan memiliki indeks. Indeks dimulai dari angka 0 bila dimulai dari depan dan -1 bila diindeks dari belakang.

Untuk jelasnya bisa diperhatikan contoh berikut.

```
D: > string.py
1    print ("Ini Adalah Contoh String")
2    print ('Ini Adalah Contoh String')
```

Hasil Outputnya:

```
Ini Adalah Contoh String
Ini Adalah Contoh String
```

Pada contoh di atas, kita menampilkan string "Ini Adalah Contoh String" menggunakan *built-in function* dari python yaitu print (). Kita bisa lihat bahwa, penggunaan tanda kutip satu ataupun kutip dua sebagai batasan dari string tetap akan menghasilkan output yang sama, jadi bisa pilih salah satunya saja.

3. Tipe Data Integer

Integer merupakan tipe data untuk objek numerik yaitu berupa bilangan bulat positif dan bilangan negatif, contoh, misalnya 0, 1, 2, 3,4,5 dan seterusnya, serta -1, -2, -3,
-4, -5 dan seterusnya. Untuk mengeluarkan output berupa integer kita juga bisa menggunakan function print () maupun langsung menuliskan bilangannya langsung tanpa menggunakan tanda kutip.

```
D: > integer.py
1    print (12345)
2    print (50)
```

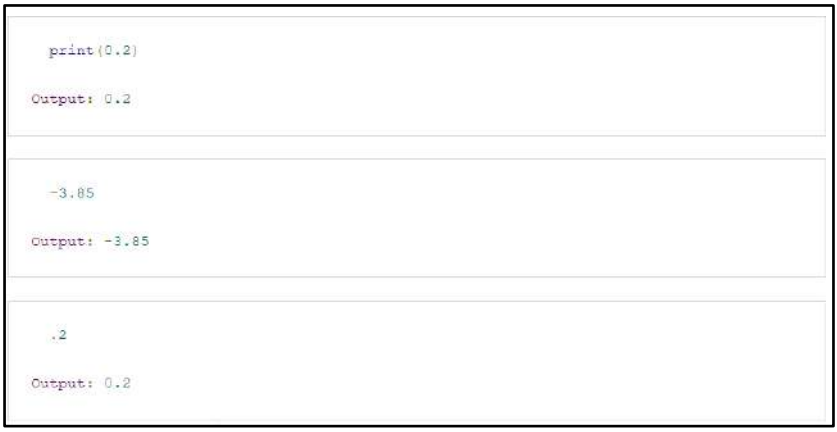
Hasil Outputnya:

```
12345
50
```

4. Tipe Data Float

Float merupakan tipe data untuk objek numerik berupa bilangan desimal, baik positif atau negatif, misalnya 1.4, 3.5, -2.876, -74.2, dan lainnya.

Sama seperti penulisan kode untuk menampilkan integer, kita tidak perlu membatasi bilangan float dengan tanda kutip.



5. Pengertian Variabel

Variabel adalah lokasi memori yang dicadangkan untuk menyimpan nilai-nilai. Ini berarti bahwa ketika kita membuat sebuah variabel Kita memesan beberapa ruang di memori. Variabel menyimpan data yang dilakukan selama program dieksekusi, yang nantinya isi dari variabel tersebut dapat diubah oleh operasi - operasi tertentu pada program yang menggunakan variabel.

Variabel dapat menyimpan berbagai macam tipe data. Di dalam pemrograman Python, variabel mempunyai sifat yang dinamis, artinya variabel Python tidak perlu didekralasikan tipe data tertentu dan variabel Python dapat diubah saat program dijalankan. Bisa diibaratkan bahwa variabel ini seperti lemari penyimpanan yang bisa diisi dengan berbagai data, dan isi lemari bisa ditukar jika diperlukan.

Dalam Penamaan Variable ada beberapa Aturan Penulisan yang harus kita ketahui.

- Nama variabel harus diawali menggunakan huruf(A-z) dan garis bawah (_).
- Nama Variabel hanya boleh mengandung huruf(A-z), angka (0-9) dan garis bawah (_).
- Karakter pada nama variabel bersifat sensitif (casesensitif). Artinya huruf besar dan kecil dibedakan.
- Nama variabel tidak boleh menggunakan kata kunci yang sudah ada dalam python seperti *if*, *while*, *for*, dan sebagainya.

Untuk mulai membuat variabel di Python caranya cukup mudah, Kita hanya menuliskan variabel lalu kemudian mengisinya dengan suatu nilai dengan cara menambahkan tanda sama dengan = diikuti dengan nilai yang ingin dimasukan.
Dibawah ini adalah contoh penggunaan variabel dalam bahasa pemrograman Python

```
#proses memasukan data ke dalam variabel
nama = "John Doe"
#proses mencetak variabel
print(nama)

#nilai dan tipe data dalam variabel  dapat diubah
umur = 20               #nilai awal
print(umur)             #mencetak nilai umur
type(umur)              #mengecek tipe data umur
umur = "dua puluh satu" #nilai setelah diubah
print(umur)             #mencetak nilai umur
type(umur)              #mengecek tipe data umur
```

6. Pengenalan Operator

Operator di dalam python adalah simbol khusus yang berfungsi untuk menjalankan suatu operasi tertentu, baik operasi aritmatika maupun operasi logika. Sedangkan nilai yang dioperasikan oleh operator dinamakan sebagai operan. Berikut ini adalah salah satu contoh dari operator aritmatika pada python:

```
>>> 10 + 5
15
```

Pada kode program di atas, tanda (+) adalah sebuah operator. Sedangkan angka 10 dan 5 keduanya merupakan operan.
Dari operasi tersebut, didapatkanlah sebuah hasil akhir berupa nilai integer yaitu 15.

Terdapat 7 jenis operator pada python:

- Operator Aritmatika (*Arithmetic Operators*)
- Operator Perbandingan (*Comparison (Relational)*
- *Operators*) Operator Penugasan (*Assignment Operators*)
- Operator Logika (*Logical Operators*)
- Operator Identitas (*Identity Operators*)
- Operator Keanggotaan (*Membership Operators*)
- Operator Bitwise (*Bitwise Operators*)

1) Operator Aritmatika

Fungsi operator aritmatika mencakup penjumlahan, pengurangan, pembagian, perkalian, dan lain-lain. Beberapa simbol dari operator aritmatika sudah kita kenal semenjak kita belajar di bangku sekolah dasar, seperti penjumlahan yang disimbolkan dengan tanda plus (+), pengurangan minus (-), perkalian (x, namun karena di python x merupakan huruf maka diganti *), dan sebagainya.

Operator	Penjelasan	Penerapan
+	Penjumlahan, menambahkan dua buah operator	5 + 2 = 7
-	Pengurangan, mengurangkan operan disebelah kiri operator dengan operan di sebelah kanan operator	5 – 2 = 3
*	Perkalian, menghasilkan operan di sebalah kiri dengan operan di sebelah kanan operator	5 * 2 = 0
/	Pembagian, membagi operan di sebelah kiri dengan operan disebelah kanan operator	5 / 2 = 2.5
//	Pembagian bulat, prosesnya sama seperti pembagian. Hanya saja angka dibelakang koma dihilangkan	5 / 2 = 2
%	Modulus, mendapatkan sisa pembagian dari operan di sebelah kiri operator ketika dibagi oleh operan di sebelah kanan	5 % 2 = 1
**	Pemangkatan, memangkatkan operan disebelah kiri operator dengan operan di sebelah kanan operator	5 **2 = 25

Contoh penggunaan dari operator aritmatika di bahasa pemrograman Python:

```
#masukkan nilai variabel nilai_pertama dan nilai_kedua dan isi nilainya
nilai_pertama = 10
nilai_kedua = 5

#pemjumlahan
hasil = nilai_pertama + nilai_kedua
print("hasil penjumlahan = "+str(hasil))

#pengurangan
hasil = nilai_pertama - nilai_kedua
print("hasil pengurangan = "+str(hasil))

#perkalian
hasil = nilai_pertama * nilai_kedua
print("hasil perkalian = "+str(hasil))

#pembagian
hasil = nilai_pertama / nilai_kedua
print("hasil pembagian = "+str(hasil))

#perpangkatan
hasil = nilai_pertama ** nilai_kedua
print("hasil perpangkatan = "+str(hasil))

#modulus (sisa bagi)
hasil = nilai_pertama % nilai_kedua
print("hasil modulus = "+str(hasil))
```

Hasil Outputnya:

```
[Running] python -u "d:\operator_aritmatika.py"
hasil penjumlahan = 15
hasil pengurangan = 5
hasil perkalian = 50
hasil pembagian = 2.0
hasil perpangkatan = 100000
hasil modulus = 0
```

2) Operator Perbandingan

Digunakan untuk membandingkan nilai di sebelah kiri dan kanan. Kemudian akan ditetapkan hubungan antara keduanya. Maka dari itu operator ini juga disebut operator relasi. Hasil dari operator perbandingan adalah nilai tipe data boolean (True dan False).

Operator	Penjelasan	Penerapan	Hasil
==	Menyatakan sama dengan, bernilai BENAR jika nilai sebelah kiri dan kanan sama	2 == 2	TRUE
!=	Menyatakan tidak sama dengan, bernilai BENAR jika nilai sebelah kiri dan kanan berbeda	1 != 2	TRUE
<	Menyatakan lebih kecil, bernilai BENAR jika nilai sebelah kiri lebih kecil daripada nilai sebelah kanan	1 < 2	TRUE
>	Menyatakan lebih besar, bernilai BENAR jika nilai sebelah kiri lebih besar daripada nilai sebelah kanan	1 > 2	TRUE
<=	Menyatakan lebih kecil atau sama dengan, bernilai BENAR jika nilai sebelah kiri lebih kecil atau sama dengan nilai sebelah kanan	1 <= 2	TRUE
>=	Menyatakan lebih besar atau sama dengan, bernilai BENAR jika nilai sebelah kiri lebih besar atau sama dengan nilai sebelah kanan	2 >= 1	TRUE

Contoh Penggunaan Operator perbandingan di bahasa pemrograman Python:

```
# masukkan nilai variabel nilai_pertama dan nilai_kedua
nilai_pertama = 1
nilai_kedua = 2

#Bandingkan antara nilai dari variabel nilai_pertama dan nilai_kedua

print("nilai_pertama > nilai_kedua = " +str(nilai_pertama > nilai_kedua))

print("nilai_pertama < nilai_kedua = " +str(nilai_pertama < nilai_kedua))

print("nilai_pertama >= nilai_kedua = " +str(nilai_pertama >= nilai_kedua))

print("nilai_pertama <= nilai_kedua = " +str(nilai_pertama <= nilai_kedua))

print("nilai_pertama == nilai_kedua = " +str(nilai_pertama == nilai_kedua))

print("nilai_pertama != nilai_kedua = " +str(nilai_pertama != nilai_kedua))
```

Hasil Outputnya:

```
[Running] python -u "d:\operator_perbandingan.py"
nilai_pertama > nilai_kedua = False
nilai_pertama < nilai_kedua = True
nilai_pertama >= nilai_kedua = False
nilai_pertama <= nilai_kedua = True
nilai_pertama == nilai_kedua = False
nilai_pertama != nilai_kedua = True
```

3) Operator Penugasan

Merupakan operator yang digunakan atau ditugaskan untuk menyimpan nilai ke dalam variabel. Simbol dari operator jenis ini pada dasarnya diwakili dengan tanda =. Sering gunakan saat kita membuat sebuah variabel.

Selain itu simbol = pada operator ini juga bisa dikombinasikan dengan operator jenis lain seperti operator aritmatika dan *bitwise*, hal ini disebut operator penugasan gabungan. Ketika operator gabungan ini digunakan, maka fungsinya tidak hanya akan menyimpan nilai tapi juga akan memanipulasi nilainya sesuai dengan jenis operator gabungannya.

Operator	Penjelasan	Penerapan	Sama Dengan
=	Memberikan operan di sebelah kanan untuk disimpan oleh variabel di sebelah kiri	x = 2	-
+=	Menjumlahkan operan sebelah kanan ke operan sebelah kiri/nilai variabel itu sendiri dan menyimpan hasilnya kembali ke variabel tersebut.	x += 2	x = x + 2
-=	Mengurangi operan sebelah kanan ke operan sebelah kiri/nilai variabel itu sendiri dan menyimpan hasilnya kembali ke variabel tersebut.	x -= 2	x = x - 2
*=	Mengalikan operan sebelah kanan ke operan sebelah kiri/nilai variabel itu sendiri dan menyimpan hasilnya kembali ke variabel tersebut.	x *= 2	x = x * 2
/=	Membagi operan sebelah kiri operator dengan operan sebelah kanan operator kemudian hasilnya diisikan ke operan sebelah kiri	x /= 2	x = x / 2
//=	Membagi operan sebelah kanan ke operan sebelah kiri/nilai variabel itu sendiri dan menyimpan hasilnya kembali ke variabel tersebut. (Hasil	x // = 2	x = x // 2

	pembagiannya dibulatkan ke bawah)		
%=	Membagi operan sebelah kanan ke operan sebelah kiri/nilai variabel itu sendiri dan menyimpan hasil sisa baginya kembali ke variabel tersebut	x % = 2	x = x % 2
**=	operan sebelah kanan dipangkatkan berdasarkan operan sebelah kiri/nilai variabel itu sendiri dan menyimpan hasil sisa baginya kembali ke variabel tersebut	x ** = 2	x = x ** 2

Contoh Penggunaan Operator penugasan di Bahasa pemrograman Python:

```
#operator (=) untuk mendefinisikan nilai variabel
x = 20
#lakukan operasi penugasan dengan nilai integer 2 dan cetak menggunakan fungsi print()
#pemjumlahan
x += 2
print("hasil penjumlahan = "+str(x))  #20 + 2 = 22

#pengurangan
x -= 2
print("hasil pengurangan = "+str(x)) #20 - 2 = 17

#perkalian
x *= 2
print("hasil perkalian = "+str(x)) #20 * 2 = 40

#pembagian
x /= 2
print("hasil pembagian = "+str(x)) #20 /2 = 20

#perpangkatan
x **= 2
print("hasil perpangkatan = "+str(x)) #20 ** 2 = 400

#modulus (sisa bagi)
x %= 2
print("hasil modulus = "+str(x))  #100 % 2 = 0
```

Simpan di folder dengan nama penugasan dan jalankan python dengan sintaks python operator_penugasan.py

Hasil Outputnya:

```
[Running] python -u "d:\operator_penugasan.py"
hasil penjumlahan = 22
hasil pengurangan = 20
hasil perkalian = 40
hasil pembagian = 20.0
hasil perpangkatan = 400.0
hasil modulus = 0.0
```

4) Operator Logika

Operator	Penjelasan	Penerapan	Hasil
and	Jika kedua operan bernilai True, maka kondisi akan bernilai True. Selain kondisi tadi maka akan bernilai False	TRUE and TRUE	TRUE
or	Jika salah satu atau kedua operan bernilai True maka kondisi akan bernilai True. Jika keduanya False maka kondisi akan bernilai False	TRUE or FALSE	TRUE
not	Membalikan nilai kebenaran pada operan missal jika asalnya true akan menjadi false dan begitupun sebaliknya	not FALSE	TRUE

Contoh penggunaan operator logika di bahasa pemrograman python:

```
#operator aritmatika
nilai_pertama = True
nilai_kedua = False
nilai_ketiga = True

#Logika and
print("nilai_kesatu and nilai_kedua = " +str(nilai_kesatu and nilai_kedua))
print("nilai_kesatu and nilai_ketiga = " +str(nilai_kesatu and nilai_ketiga))

#Logika or
print("nilai_kesatu or nilai_kedua = " +str(nilai_kesatu or nilai_kedua))
print("nilai_kesatu or nilai_ketiga = " +str(nilai_kesatu or nilai_ketiga))

#Logika not
print("not nilai_kesatu = " + str(not nilai_kesatu))
print("not nilai_kedua = " + str(not nilai_kedua))
```

Hasil Outputnya:

```
[Running] python -u "d:\operator_logika.py"
nilai_pertama and nilai_kedua = False
nilai_pertama and nilai_ketiga = True
nilai_pertama or nilai_kedua = True
nilai_pertama or nilai_ketiga = True
not nilai_pertama = False
not nilai_kedua = True
```

5) Operator Identitas (*Identity Operators*)

Operator identitas lebih digunakan untuk membandingkan dan memastikan apakah variabel x menunjuk lokasi memori yang sama dengan variabel y. Jadi lebih ke mencocokan isi antar variabel.

Jumlah anggotanya sama dengan operator keanggotaan, hanya berisi 2. Bila di operator keanggotaan ada in dan not in. Maka di operator identitas ada is dan not is.

Operator	Penjelasan	Penerapan	Hasil
is	Memeriksa apakah nilai di sebelah kiri operan memiliki identitas memori yang sama	x = 10 y = 10 x is y	TRUE

	dengan nilai di sebelah kanan operan. Jika sama maka kondisi bernilai True		
Is not	Memeriksa apakah nilai di sebelah kiri operan memiliki identitas memori yang berbeda dengan nilai di sebelah kanan operan. Jika berbeda maka kondisi bernilai True	x = 10 y = 5 x is not y	TRUE

Contoh penggunaan operator logika di bahasa pemrograman python:

```
#operator_identitas
nilai_pertama = 20
nilai_kedua = 5


# membandingkan nilai_pertama dengan nilai_kedua
print(nilai_pertama is nilai_kedua)

# membandingkan nilai_pertama dengan nilai 20
print(nilai_pertama is 20)
```

Hasil outputnya:

```
[Running] python -u "d:\operator_identitas.py"
d:\operator_identitas.py:8: SyntaxWarning: "is" with a literal. Did you mean "=="?
False
True
  print(nilai_1 is 20)
```

6) Operator Keanggotaan (*Membership Operators*)

Operator ini diperuntukkan untuk mencari keanggotaan dalam suatu tipe data urutan seperti String, List, dan Tuple. Tak mengherankan jika operator ini sering diterapkan dalam perulangan. Karena hanya memastikan suatu nilai ADA atau TIDAK ADA di dalam daftar, maka operator ini hanya berisi 2 anggota saja.

Operator	Penjelasan	Penerapan	Hasil
in	Memeriksa apakah nilai yang dicari berada pada list atau struktur data python lainnya. Jika nilai tersebut ada maka kondisi akan bernilai True	3 in [1, 2, 3, 4]	TRUE
not in	Memeriksa apakah nilai yang dicari tidak ada pada list atau struktur data python lainnya. Jika nilai tersebut tidak ada maka kondisi akan bernilai True	5 not in [1, 2, 3, 4]	TRUE

Contoh penggunaan operator keanggotaan di bahasa pemrograman python:

```
nama = "umam"

hoby = ["membaca", "ngoding", "jalan-jalan"]

nilai = [1, 2, 3, 4]

# cari huruf 'u' di variabel nama
print('u' in nama)

#cari string 'bermain' di list hoby
print("ngoding" in hoby)

# cari angka '3' di variabel nilai
print(3 in nilai)

# melihat angka '5' apakah tidak ada di variabel nilai
print(5 not in nilai)
```

Hasil Outputnya:

```
[Running] python -u "d:\operator_keanggotaan.py"
True
True
True
True
```

7) Operator Bitwise (*Bitwise Operators*)

Untuk menggunakan operator bitwise minimal kalian perlu tahu pemahaman tentang apa itu bilangan biner. Karena operator ini diperuntukkan untuk hal tersebut. Nah sekarang apa itu bilangan biner? Bilangan biner merupakan bilangan yang tersusun atas dua angka 0 dan 1. Operator bitwise merupakan operator logika yang dikhususkan untuk bilangan biner.

Operator	Penjelasan	Penerapan
&	Operator biner AND, memeriksa apakah operan di sebelah kiri dan operan sebelah kanan mempunyai angka biner 1 di setiap bit. Jika keduanya bernilai 1 maka bit hasil operasi akan bernilai 1	x & y
|	Operator biner OR, memeriksa apakah operan di sebelah kiri dan operan sebelah kanan mempunyai angka biner 1 di setiap bit. Jika salah satunya bernilai 1 maka bit hasil operasi akan bernilai 1	x | y
^	Operator biner XOR, memeriksa apakah operan disebelah kiri dan operan sebelah kanan mempunyai angka biner 1 di setiap bit. Jika keduanya bernilai 1 maka bit hasil operasi akan bernilai 0	x ^ y
~	Operator biner Negative, membalik nilai bit. Misal dari 1 menjadi 0, dari 0 menjadi 1	x ~ y
<<	Operator penggeser biner ke kiri, deret bit akan digeser ke kiri sebanyak n kali	x << y
>>	Operator penggeser biner ke kanan, deret bit akan digeser ke kanan 1 x	x >> y

Contoh penggunaan operator *bitwise* di bahasa pemrograman python:

```
x = 4
y = 6

print('x berisi angka',x ,'desimal atau',bin(x),'biner')
print('y berisi angka',y ,'desimal atau',bin(y),'biner')

# opearator &
print('x & y  :',x & y)
# opearator |
print('x | y  :',x | y)
# opearator ^
print('x ^ y  :',x ^ y)
# opearator ~
print('~x     :',~x)
# opearator <<
print('x << 1 :',x << 1)
# opearator >>
print('x >> 1 :',x >> 1)
```

Hasil Outputnya:

```
[Running] python -u "d:\operator_bitwes.py"
x berisi angka 4 desimal atau 0b100 biner
y berisi angka 6 desimal atau 0b110 biner
x & y  : 4
x | y  : 6
x ^ y  : 2
~x     : -5
x << 1 : 8
x >> 1 : 2
```

Pada contoh di atas, saya mendefinisikan 2 variabel yaitu x dan y. Kemudian memberikan nilai awal 4 dan 6. Jika di konversi ke dalam bentuk biner, keduanya berisi angka berikut:

x = 4 (desimal) = 100 (biner) y

= 6 (desimal) = 110 (biner)

Di baris 3 dan 4 saya menggunakan fungsi bawaan python, yakni bin (), ini bisa dipakai untuk menampilkan versi biner dari sebuah angka desimal. Awalan 0b merupakan penanda bahwa ini adalah

angka biner. Artinya, angka 0b100 adalah 100 dalam bilangan biner. Operator *bitwise* pertama adalah operasi & (And) terhadap kedua variabel. Operasi *bitwise* "and" ini akan memproses bit per bit dari kedua variabel, jika kedua bit sama-sama 1, maka hasilnya juga 1, selain kondisi tersebut, nilai akhirnya adalah 0. Berikut perhitungan *bitwise* "and":

```
x      = 100
y      = 110
         ----------
x & y = 100 = 4 (desimal)
```

Operator bitwise | (Or), hasilnya akan bernilai 0 jika kedua bit bernilai 0, selain itu nilai bit akan di set menjadi 1. **Berikut cara** perhitungan bitwise "or":

```
x      = 100
y      = 110
         ----------
x | y = 110 = 6 (desimal)
```

Operator *bitwise* ^ (Xor), hasilnya akan bernilai 1 jika salah satu dari kedua variabel bernilai 1 (namun tidak keduanya). Atau dengan kata lain jika kedua bit berlainan, hasilnya 1 tapi kalau sama-sama 0 atau sama-sama 1, hasilnya 0.

```
x      = 100
y      = 110
         ----------
x ^ y = 010 = 2 (desimal)
```

C. Kesimpulan

1. Tipe data adalah suatu cara memberitahu komputer kita untuk mengelompokkan atau menggolongkan data berdasarkan apa yang dimengerti oleh komputer.
2. Dalam bahasa pemrograman Python, terdapat 9 tipe data dasar yaitu *String, Integer, Float, Complex Number, Boolean, List, Tuple, Set dan Dictionary.*

D. Latihan

1. Apakah yang dimaksud dengan tipe data?
2. Sebutkan dan jelaskan penggolongan tipe data?
3. Buatlah sebuah kode program yang menggunakan tipe data string dan mumerik?
4. Buatlah sebuah kode program yang menggunakan tipe data Boolean?
5. Buatlah satu program python sederhana konversi waktu (detik) dari jam dan menit, dan konversi hari dari tahun dan bulan?

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula.*Yoyakarta:PT Elex Komputindo.

- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB V
PERINTAH MASUKAN DAN KELUARAN

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran
   Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu mengimplementasikan fungsi perintah masukan dan keluaran pada Bahasa pemrograman python.
b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami dalam pengunnakan tools python dan penggunakaan tipe data pada python.
c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya
d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk mengaplikasikan fungsi perintah masukan dan keluaran sesuai dengan fungsinya dalam script program di python.

B. Pokok Bahasan

Pada umumnya sebuah aplikasi berinteraksi dengan user supaya dapat bekerja sesuai dengan keinginan user. Untuk bisa melakukan interaksi, sebuah aplikasi perlu dilengkapi dengan metode input dari pengguna. Python menyediakan suatu fungsi untuk melakukan input dari user yaitu fungsi input (). Fungsi input () ini pada python berfungsi untuk mendapatkan sebuah nilai masukan manual dari user. Selain untuk mendapatkan inputan dari user, python juga menyediakan fungsi print () untuk menampilkan hasil dari inputan. Kedua buah fungsi tersebut merupakan fungsi bawaan dari Bahasa pemrograman python atau sering juga disebut built-in function.

1. Inputan

   Input adalah proses memasukkan sebuah data secara manual dari user. Program yang kita buat akan berinteraksi dengan user. Dimana program yang kita buat tersebut akan meminta sebuah data yang diperlukan oleh user utnuk menjalankan perintah dalam program tersebut. Contoh, misalkan kita membuat sebuah program menghitung luas persegi panjang. Pada program tersebut user dimita untuk mengimputkan data panjang dan lebar oleh program.

   Untuk menggunakan sebuah fungsi input sangatlah mudah. Kita hanya memerlukan atau menyiapkan sebuah variabel untuk menyimpan nilai yang di dapat dari fungsi tersebut.

   Berikut ini adalah contoh penggunaan fungsi input () di python:

```
D: > input.py > ...
  1    nama = input('Silahkan Masukkan Nama Anda: ')
```

Hasil Outputnya setelah di running:

```
Silahkan Masukkan Nama Anda : Wiro Sableng
```

pada kode program diatas, ketika dijalankan, kita akan diminta untuk menginputkan sesuai dengan teks yang terlihat di dalam panel output. Dalam hal ini kita menginputkan sebuah nama yang kemudian diakhiri dengan menekan tombol enter. Nama yang di inputkan oleh kita akan dismpan kedalam variabel nama.

Apapun yang kita inputkan atau user inputkan akan di anggap sebagai string meskipun kita menginputkan angka. Misalkan kita menginlutkan usia dengan mengetik 30 dan disimpan dalam variabel, maka python menganggapnya angka tersebut adalah string “30”. Oleh karena itu, operasi aritmatika

terhadap variabel itu akanmenghasilkan pesan error. Contoh lebih detailnya sebagai berikut:

```
D: > inputan_umur.py > ...
1   #inputan umur
2   umur = input('Silahkan Masukkan Umur Anda: ')
3
4   #variabel umur
5   umur = umur + 10
6
7   #output
8   print(umur)
```

Hasil Outputnya:

```
Silahkan Masukkan Umur Anda: 30
Traceback (most recent call last):
  File "d:/inputan_umur.py", line 5, in <module>
    umur = umur + 10
TypeError: can only concatenate str (not "int") to str
```

Ketika kita jalankan program diatas, akan muncul sebuah pesan eror. Hal ini dikarenakan pada code program diatas nilai dalam variabel umur tidak bisa dilakukan penjumlahan dengan angka 10 menggunakan operator (+). Ini karena variabel umur menyimpan string bukan number. Untuk mengatasi hal tersebut, nilai didalam variabel umur itu perlu di ubha menjadi number (bilangan bulat atau pecahan). Berikut ini adalah fungsi yang bis akita gunakan:

- **Int ()** : digunakan untuk proses mengubah nilai string menjadi integer (bilangan bulat)
- **Float ()** : digunakan untuk proses mengubah nilai string menjadi integer (builangan pecahan)

untuk lebih jelasnya berikut ini contoh dari mengubah string ke integer:

```
D: > string_ineteger.py > ...
1   #inputan umur
2   umur = input('Silahkan Masukkan Umur Anda: ')
3
4   #variabel umur
5   umur = int(umur) + 10
6
7   #output
8   print(umur)
```

Hasil Outpunya:

```
Silahkan Masukkan Umur Anda: 10
20
```

2. Output

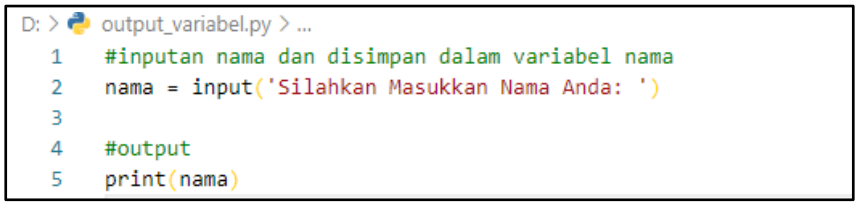
Dalam Bahasa Python, untuk menampilkan suatu text, number atau variabel lainnya yaitu dengan menggunakan perintah fungsi print (). Hal yang perlu kita pahami dalam penggunaan output di python.

Dalam menampilkan sebuah string kita harus menggunakan pengapit tanda petik print („Algoritma Pemrogaram‟) hasilnya python akan menampilkan string Algoritma Pemrograman. Jika tidak kita apit dengan tanda petik, maka akan di baca sebagai variabel oleh python dan akan mencari di dalam list variabel untuk menemukan isi dari variabel tersebut. Jika isi variabel tersebut tidak ditemukan makan akan muncul pesan error. Untuk memunculkan isi dari sebuah variabel yaitu dengan cara memasukkan nama variabel di antara tanda kurung seperti contoh print (nama) python akan memunculkan isi dari variabel nama tersebut ke layar. Contoh dari penggunaan fungsi output sebagai berikut:

```
D: > output.py
1   #fungsi output
2   print('Selamat Belajar Python')
```

Hasil Outputnya:

Jika kita ingin mencetak atau memunculkan nilai dari suatu inputan user yang kemudian di simpan pada sebuah variabel.



Hasil Outputnya

Nama yang di munculkan di layar adalah dari hasil inputan pengguna.

C. Kesimpulan

1. Python menyediakan suatu fungsi untuk melakukan input dari user yaitu fungsi input ().
2. Input adalah proses memasukkan sebuah data secara manual dari user.
3. Untuk menampilkan suatu text, number atau variabel lainnya yaitu dengan menggunakan perintah fungsi print ().

D. Latihan

Selesaikan permasalahan berikut ini dengan menggunakan Python

1. Arahkan user untuk menginput Nama, tanggal lahir, dan hobi mereka?
2. Tampilkan kata “Selamat sudah mengisi formulir kami. Berikut data yang sudah anda input”?
3. Tampilkan data yang sudah diinputkan user beserta keterangannya dengan format: Nama: {inputan_nama}, Tanggal Lahir: dan seterusnya.?

4. Tampilkan text: “Terima kasih, sukses selalu.”?
5. Terkhir simpanlah file tersebut.?

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.
- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB VI
TEKNIK RUNTUTAN

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran

Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu menganalisis konsep Algoritma runtunan dan mengaplikasikannya dalam program aplikasi.

b. Kompetensi yang Diharapkan;

Mahasiswa telah memahami tentang dasar Algoritma, notasi Algoritma serta pengunaan tools python dan penggunakaan tipe data pada python.

c. Keterkaitan materi dengan materi yang lain;

Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya

d. Pentingnya mempelajari isi bab;

Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk memahami pengertian dari Algoritma runtunan dan mengaplikasikannya untuk meyelesaikan suatu masalah sehari-hari.

B. Pokok Bahasan

Pengertian Algoritma Teknik Runtutan

Algoritma runtunan merupakan Algoritma yang paling sederhana. Algoritma runtunan adalah sekumpulan perintah atau pernyataan yang dikerjakan komputer berdasarkan dengan urutan perintahnya. Secara sederhnanya adalah proses yang dilakukan secara berurutan dari Langkah ke-1 sampai Langkah terakhir. Tiap barin dilakukan satu persatu tanpa ada loncatan atau perulangan. Aequence Algoritma sendiri terdiri dari satu atau lebih intruksi, yang berarti bahwa :

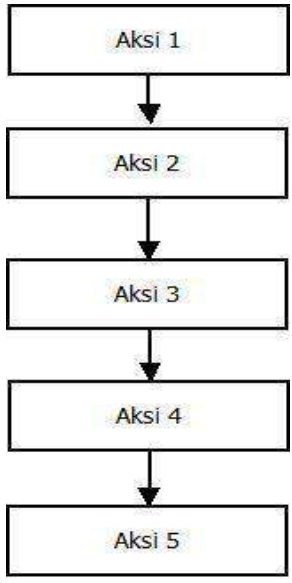
1. Tiap intruksi dikerjakan satu persatu.
2. Tiap intruksi dilaksanakan sekali, dan tidak ada intruksi

yang diulang.

3. Urutan intruksi yang dilaksanakan sama dan sesuai dengan intruksi yang dibuat
4. Akhir dari intruksi terakhir merupakan akhir Algoritma

jadi *sequence* (urutan) dalam Algoritma sangat penting, karena kita bisa lebih efektif dalam menjalankan suatu Algoritma dengan urut dan sesuai dengan apa yang diinginkan karena jika suatu Algoritma tidak ada sequence (urutan) maka Algoritma tersebut akan kacau dan bisa saja mengeluarkan output yang tidak sesuai.

Berikut ini adalah runtunan yang terdiri dari 5 buah instruksi. Setiap instruksi di jalankan satu persatu sesuai urutannya.



Dari *flowchart* diatas mula-mula pemroses akan melaksanakan aksi 1 terlebih dahulu, kemudian Aksi 2 akan dikerjakan setelah Aksi 1 selesai dikerjakan. Selanjutnya Aksi 3 dikerjakan jika Aksi 2 telah selesai dikerjakan. Selanjutnya Aksi 4 dikerjakan jika Aksi 3 telah selesai dikerjakan Selanjutnya Aksi 5 dikerjakan jika Aksi 4 telah selesai dikerjakan. Setelah Aksi 5 selesai dilaksanakan maka Algoritma berhenti.

Di dalam Bab 1 sudah dicontohkan pertukaran dua buah gelas isi, itu adalah salah satu contoh dari Algoritma runtunan, dimana setiap langkah penyelesaiannya di lakukan dengan satu persatu. Maka dari itu masalah pertukaran adalah persoalan fundamental dalam pemrograman.

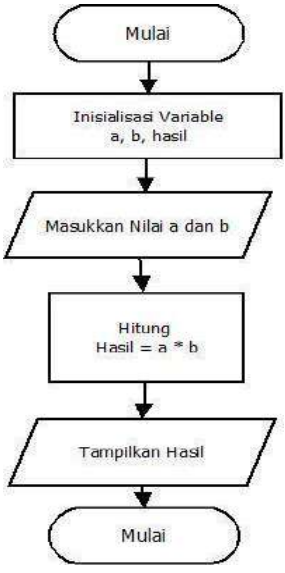
Berikut ini beberapa contoh penerapan Algoritma sekuensial kedalam bentuk bahasa natural, *flowchart* maupun *pseudocode*:

2. Algoritma Perkalian Dua Buah bilang Bilangan

1) Bahasa Natural / Deskripsi

- Mulai
- Inisialisasi variabel x, y dan hasil
- Masukan nilai x dan y
- Hitung hasil=x * y;
- Tampilkan nilai hasil
- Selesai

2) Flowchart



3) Pseudocode

```
algoritma
perkalian_dua_bilangan
deklarasi
     var x: y: hasil:
integer; deskripsi
x <- 4;
y <- 2;
hasil <- x + y;
```

Contoh kode program di bahasa python

```
# Program Perkalian Dua Buah Bilangan

# meminta inputan kepada user
bil1 = input('Masukkan bilangan pertama: ')
bil2 = input('Masukkan bilangan kedua: ')

# Menjumlahkan bilangan
jumlah = float(bil1) + float(bil2)

# Menampilkan hasil
print('Jumlah {0} + {1} adalah {2} = '.format(bil1, bil2, jumlah))
```

Hasil Outputnya:

```
========================== RESTART: D:\perkalian.py ==========================
Masukkan bilangan pertama: 4
Masukkan bilangan kedua: 2
Jumlah 4 + 2 adalah 6.0 =
>>>
```

C. Kesimpulan

1. Algoritma runtunan adalah sekumpulan perintah atau pernyataan yang dikerjakan komputer berdasarkan dengan urutan perintahnya.
2. Sequence Algoritma sendiri terdiri dari satu atau lebih intruksi, yang berarti bahwa:
   - Tiap intruksi dikerjakan satu persatu.

- Tiap intruksi dilaksanakan sekali, dan tidak ada intruksi yang diulang.
- Urutan intruksi yang dilaksanakan sama dan sesuai dengan intruksi yang dibuat
- Akhir dari intruksi terakhir merupakan akhir Algoritma

D. Latihan

1. Jelaskan definisi Algoritma teknik runtutan menurut pemahaman Anda dan kemudian berikan cohtoh dari Algoritma teknik runtutan dalam kehidupan kita sehari-hari?
2. Rancanglah flowchart dan kode program untuk mencari luas persegi empat?
3. Rancanglah kode program untuk menghitung luas bujur sangkar?
4. Buatlah kode program untuk menghitung luas trapisium?
5. Buatlah kode program menghitung harga barang pada suatu took dengan ketentuan diskon 20% dari setiap harga barang?

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.

- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB VII
FUNGSI PERCABANGAN

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran
   Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu menganalisis konsep Algoritma fungsi percabangan dan mengaplikasikannya dalam program aplikasi.
b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami tentang dasar Algoritma, notasi Algoritma serta pengunaan tools python dan penggunakaan tipe data pada python.
c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya
d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk memahami pengertian dari Algoritma fungsi percabangan dan mengaplikasikannya untuk meyelesaikan suatu masalah sehari-hari.

B. Pokok Bahasan

1. Pengertian Algoritma Fungsi Percabangan

Program yang hanya berisi teknik runtutan instruksi biasanya terdapat hanya pada persoalan yang sederhana. Pada persoalan yang kompleks akan melibatkan nnalisa berbagai kemungkinan yang terdapat didalamnya. Untuk setiap masing-masing kasus terdapat persyaratan yang harus dipenuhi dan aksi apa yang harus di lakukan ketika syarat itu terpenuhi. Dengan adanya analisi kasus seperti itu makan instruksi tidak lagi di lakukan secara teknik runtutan, tetapi berdasarkan syarat syarat yang terpenuhi. Istilah lain untuk pemilihan adalah pencabangan atau *control flow*.

contoh. Misalkan, kita hendak menentukan apakah suatu bilangan termasuk bilangan genap atau bilangan ganjil. Nah, Algoritmanya dapat kita jelaskan sebagai berikut:

1. Mulai
2. Masukkan suatu bilangan, misalkan bilangan Y)
3. Jika bilangan Y habis dibagi dua, maka lanjut ke perintah keempat. Jika tidak lanjut ke perintah kelima.
4. Tuliskan "Y adalah bilangan genap". Lanjut ke perintah keenam.
5. Tuliskan "Y adalah bilangan ganjil"
6. Selesai

Pada Algoritma di atas, kita bisa lihat bahwa ada dua kemungkinan perintah yang akan dikerjakan setelah perintah ketiga dikerjakan. Perintah pertama, jika bilangan Y habis dibagi dua maka selanjutnya perintah keempat yang dikerjakan, kemudian lompat ke perintah keenam dan perintah kelima tidak akan dikerjakan. Perintah kedua, jika bilangan Y tidak habis dibagi dua maka melompat ke perintah kelima dan perintah keempat tidak akan dikerjakan. Kedua perintah tersebut sama-sama berakhir pada perintah keenam, yang menyatakan bahwa proses Algoritma telah selesai.

Tidak seperti bahasa pemrograman lainnya, Python hanya mengenal satu fungsi fungsi percabangan (kondisi) saja. Tidak ada *switch* atau *case* dalam python, tetapi hanya fungsi if saja. Pada bahasa pemrograman Python dikenal dengan beberapa fungsi yang menggambarkan kondisi fungsi percabangan. Fungsi tersebut diantaranya adalah

1. if
2. if, else
3. if, elif, else
4. if bersarang

2. IF

Statmen if biasanya digunakan untuk melakukan penyelesaian dimana jika kondisi bernilai benar (true) maka program akan mengeksekusikan statmen yang berada dibawahnya. Dalam bahasa pemograman Pada python anda dapat membuat statement if dengan menulis if, diikuti *conditional expression* dan tanda titik dua (:).

Contoh bentuk umum perintah if:

```
If (kondisi):
    Aksi
```

```
1   nilai = 100
2
3   if nilai == 100:
4       print('Lulus')
```

Pada contoh code program if di atas (nilai == 100) adalah merupakan conditional expression dimana pada conditional expression tersebut menggunakan operator pembanding (==) untuk mengetahui jika kedua nilai itu sama. Dalam menulis aksi pada if anda harus melakukan pengindentasian atau penulisan code yang agak menjorok masuk ke dalam agar supaya bisa dijalankan jika kondisi bernilai benar. Bagian yang tidak diintensai akan berjalan tidak melihat kondisi benar atau salah.

```
1   nilai = 75
2
3   if nilai == 100:
4       print('Lulus')
5       print ('Pertahankan pertasimu')
6
```

Dari contoh code program diatas outpunya tidak mucul karena nilai diberikan bernilai salah atau tidak sama.

```
1   nilai = 75
2
3   if nilai == 100:
4       print('Lulus')
5   print ('Pertahankan pertasimu')
6
```

Berbeda dengan program sebelumnya, pada code proram ini menghasilkan output aksi (“pertahankan prestasimu”) dikarenakn aksi ini tidak di lakukan indentasi sehingga tidak masuk dalam kondisi if di atasnya.

3. If-Else

Stetmen if-else berfungsi untuk melakukan penyelesaian dimana kondisi bernilai benar (true) maka program akan mengeksekusi statmen yang pertama atau ke satu. Jika nilai kondisinya bernilai salah maka statmen kedua akan dieksekusi. Berikut merupakan bentuk sederhana perintah if-else :

```
If (kondisi):
        Aksi 1
Else
        Aksi 2
```

Berikut contoh kode program yang merupakan penggunaan kondisi if else.

```
D: > ifelse.py > ...
1   total = int(input("Total Belanja : Rp. "))
2   if total > 150000:
3       print("Anda Mendapatkan Diskon 5%")
4   else:
5       print("Tidak ada Diskon")
```

Hasil Outputnya:

```
Total Belanja : Rp. 200000
Anda Mendapatkan Diskon 5%
```

Pada contoh di atas, jika program dijalankan dan apabila nilai yang diinputkan lebih dari 100 ribu maka akan mencetak string "Anda Mendapatkan Bonus Bimolli 1L" sedangkan apabila nilai salah atau kurang dari 100 ribu maka akan mencetak string "Tidak ada Bonus".

4. If-Elif-Else

Perintah if–else-elif berfungsi untuk melakukan penyeleksian kondisi dimana kondisi yang diberikan lebih dari 1 kondisi atau memiliki beberapa kondisi. Pengambilan keputusan (kondisi if elif) merupakan lanjutan/fungsi percabangan logika dari "kondisi if". Dengan *elif* kita bisa membuat kode

program yang akan menyeleksi beberapa kemungkinan yang bisa terjadi. Hampir sama dengan kondisi "*else*", bedanya kondisi "*elif*" bisa banyak dan tidak hanya satu. Bentuk ketiga ini juga memiliki arti bentuk dimana kita memiliki lebih dari dua pilihan kode untuk dieksekusi berdasarkan kondisi tertentu. Perintah *elif* (bentuk singkatan dari *else if*) disisipkan di antara *if* dan *else*. Berikut merupakan bentuk umum perintah *if–else-elif*

If (kondisi 1):

Aksi 1

Elif (kondisi 2)

Aksi 2

Else

Aksi 3

Berikut contoh kode program yang merupakan penggunaan kondisi *if-elif-else*.

D: > ifelif.py > ...

```
1   nilai = 75
2
3   if nilai == 100:
4
5       print('Anda Dapat Nilai Baik')
6
7   elif nilai >= 60:
8
9       print ('Anda Dapat Nilai Cukup')
10
11  else:
12
13      print('Anda Dapat Nilai Buruk')
```

Hasil Outputnya:

Pada code program diatas control flow akan berhenti ketika kondisi bernilai benar, yaitu pada kondisi (nilai >= 60). Sedangkan untuk code program dibwahnya tidak akan di jalankan walaupun kondisinya juga bernilai benar. Sehingga keluaranya adalah (Anda Dapat Nilai Cukup).

5. ***Percabangan Bersarang***

Kondisi bersarang adalah suatu kondisi di dalam kondisi tertentu, Jika terdapat 2 cabang kondisi maka di dalam salah satu cabang kondisi tersebut dapat pula di isi suatu kondisi tertentu. Atau dalam artian gampangnya adalah merupakan *if* dalam *if*, jadi dalam if itu ada *if* lagi. *if* bersarang ini juga sering disebut dengan *nested if*.

Berikut merupakan bentuk umum perintah nested if

```
If (kondisi 1):
        If (kondisi 2):
                Aksi 1:
        Else
                Aksi 2:
else
        Aksi 3
```

Berikut contoh kode program yang merupakan penggunaan kondisi if-elif-else.

D: > nestedif.py > ...

```
1   #code nestedif
2
3   jenis_kelamin = "Pria"
4   umur = 20
5
6   if (jenis_kelamin=="Pria"):
7       if (umur >= 25):
8           print ("Pria boleh menikah")
9       else:
10          print ("Pria tidak boleh menikah")
11  elif(jenis_kelamin=="Wanita"):
12      if (umur >= 20):
13          print ("Wanita boleh menikah")
14      else:
15          print ("Wanita tidak boleh menikah")
16  else:
17      print ("Jenis kelamin tidak terdaftar")
```

Hasil Outputnya:

```
Pria tidak boleh menikah
```

Contoh lain dari penggunaan Nested if di python yaitu menentukan apakah inputan angka dari user bernilai ganjil positif atau negative dan genap positi atau negatif. Berikut ini code programnya:

D: > nestedif3.py > ...

```
1   bil = int(input("Masukkan Angka = "))
2
3   if bil > 0:
4       if bil % 2 == 0:
5           print ("Angka {} bernilai Positif Genap".format(bil))
6       elif bil % 2 != 0:
7           print ("Angka {} bernilai Poitif Ganjil".format(bil))
8   elif bil < 0:
9       if bil % 2 == 0:
10          print ("Angka {} bernilai Negetif Genap".format(bil))
11      elif bil % 2 != 0:
12          print ("Angka {} bernilai Negatif Ganjil".format(bil))
13  else:
14      print("NOL")
```

C. Kesimpulan

1. Pemilihan kondisi dalam python menggunakan kode program *if, if else, if elif else* dan *nested if* ataun *if* bersarang

D. Latihan

1. Buatlah program untuk menghitung total pembelian N buah barang dengan harga dan jumlah tertentu?
2. Buatlah program untuk menentukan kelulusan berdasarkan minimal nilai 80, jika nilai sama dengan atau lebih dari 75 beri keterangan lulus?
3. Buatlah program untuk menghitung banyaknya bilangan genap ataupun bilangan ganjil dari N buah bilangan yang diinput secara acak?
4. Buatlah program dengan kasus (Jika umur yang di inputkan user lebih besar dari 17 maka boleh mencetak KTP, namun sebaliknya jika umur yang di inputkan kurang dari 17 maka boleh mencetak KTP)?
5. Buatlah soal matematika yang proses penyelesaiannya menggunakan bahasa python yang memanfaatkan kondisi *if, if else, if elif else.*

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer*

Pemula.Yoyakarta:PT Elex Komputindo.

- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB VIII
FUNGSI PERULANGAN

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran;
   Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu menganalisis konsep Algoritma perulangan dan mengaplikasikannya dalam program aplikasi.
b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami tentang dasar Algoritma, notasi Algoritma serta pengunaan tools python dan penggunakaan tipe data pada python.
c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya
d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk memahami pengertian dari Algoritma perulangan dan mengaplikasikannya untuk meyelesaikan suatu masalah sehari-hari.

B. Pokok Bahasan

- **Pengertian Algoritma Perulangan**

  Perulangan atau juga sering dikenal dengan *looping* merupakan pernyataan atau perintah yang diberikan kepada komputer agar komputer dapat melakukan sesuatu entah itu memproses data, menampilkan data, atau yang lainnya secara berulang. Dengan menggunakan perulangan, waktu yang dibutuhkan untuk membuat suatu program akan lebih singkat. Contohnya jika ingin membuat program sederhana menampilkan angka 1 sampai 10. Jika jmlah perulangan hanya sedikit, dapat dikerjakan secara manual.

Tapi bagaimana kalau *range*-nya sampai 1000 kali atau lebih. Sangat tidak efisien jika ditulis secara manual satu persatu. Maka dari itu agar kerja kita lebih efisien dibuatlah perulangan. Pada python untuk menggunakan perulangan terdapat 2 cara yaitu :

1. for
2. while

Keduanya memiliki perbedaan pada segi penggunaan, dikatakan jika for lebih digunakan dalam perulangan yang sudah diketahui jumlah perulangannya (*countable*). Sedangkan perulangan while digunakan ketika jumlah perulangannya belum ditentukan (*uncountable*). Baik for dan while keduanya merupakan blok kode, sama seperti *if else*. Jadi dipastikan ada indentasi di dalamnya.

2. Perulangan For

Perulangan for disebut *counted loop* (perulangan yang terhitung). Fungsi perulangan for melakukan pengulangan dengan meng- iterasi elemen dari sebuah list. List merupakan kumpulan karakter, kumpulan string, angka dan kumpulan data yang lainnya. For dalam bahasa Python memiliki ciri khas tersendiri dibandingkan dengan bahasa pemograman yang lainnya. Tidak hanya mengulang bilangan-bilangan ekspresi aritmatika atau memberikan keleluasan dalam mendefinisikan iterasi perulangan dan menghentikan perulangan pada saat kondisi tertentu saja. Statmen for dalam Python bekerja berbagai tipe data sekuensial seperti List, String, dan Tuple. Berikut merupakan bentuk umum dari perintah for:

```
for indek in range (banyak_perulangan):
    perintah
```

Berikut adalah contoh penerapan perulangan *for loop* dalam bahasa Python:

```
D: > loop1.py > ...
1   # perulangan
2
3   ulang = 15
4
5   for i in range(ulang):
6       print ("Perulangan ke-"+str(i))
```

Dalam code program python siatas kita menentukan terlebih dahulu jumlah perulanganya yaitu 15 kali perulangan. Variabel i berfungsi untuk menampung indeks, dan fungsi range () berfungsi untuk membuat list dengan *range* dari 0-15. Fungsi str () berfungsi merubah tipe data ineger ke string.

Hasil Output dari code program diatas:

```
Perulangan ke-0
Perulangan ke-1
Perulangan ke-2
Perulangan ke-3
Perulangan ke-4
Perulangan ke-5
Perulangan ke-6
Perulangan ke-7
Perulangan ke-8
Perulangan ke-9
Perulangan ke-10
Perulangan ke-11
Perulangan ke-12
Perulangan ke-13
Perulangan ke-14
```

Contoh lain menggunakan senarai (list):

```
D: > loop2.py > ...
1   warna = ['Merah','Biru','Kuning','Biru']
2
3   for i in warna:
4       print(i)
```

Di baris 1 terdapat variabel warna sebagai tipe data list. List ini terdiri dari 4 element. Menggunakan perulangan for, kita menampilkan semua isi variabel warna. Di dalam perulangan, variabel i dipakai untuk menyimpan isi dari element yang saat ini sedang di proses. Nama variabel i sendiri boleh bebas, fungsinya hanya sebagai variabel bantu. Hasil Outputnya:

```
Merah
Biru
Kuning
Biru
```

Perulangan for juga bisa dipakai untuk tipe data lain, misalnya tipe data set:

```
D: > loop3.py > ...
1   warna = {'Merah','Biru','Kuning','Biru'}
2   for i in warna:
3       print(i)
```

Perhatikan perubahan tanda kurung di baris 1. Sekarang akan dicoba dengan menggunakan kurung kurawal yang merupakan cara pembuatan tipe data set di dalam Python.
Hasilnya:

```
Merah
Kuning
Biru
```

Nama warna hanya tampil 3 buah. Loh, kenapa bukan 4? ini karena perilaku dari tipe data set Python, dimana jika terdapat data yang berulang, data tersebut tidak akan disimpan. Dalam hal ini, warna „biru" dicontohkan ditulis sebanyak 2 kali. Lebih lanjut tentang tipe data set bisa dipelajari kembali di bab selanjutnya.

Contoh for dengan tipe data string sebagai berikut

```
D: > loop4.py > ...
1   program = 'PYTHON'
2   for huruf in program:
3       print(huruf)
```

Hasil Outputnya:

```
P
Y
T
H
O
N
```

C. Kesimpulan

1. Perulangan atau juga sering dikenal dengan looping merupakan pernyataan atau intruksi yang diberikan kepada komputer agar ia mau melakukan sesuatu entah itu

memproses data, menampilkan data, atau yang lainnya secara berulang.

2. Pada python untuk menggunakan perulangan terdapat 2 cara yaitu for dan while

D. Latihan

1. Jelaskan pengertian struktur Algoritma perulangan menurut pemahaman Anda dan berikan contohnya dalam kehidupa sehari-hari kita?
2. Buatlah program untuk menampilkan nama Anda sebanyak 200?
3. Buatlah program untuk menampilkan angka 1 sampai 500?
4. Buatlah kode program dengan python untuk menampilkan deret bilangan ganjil dari 1 sampai 200 menggunakan while?
5. Buatlah soal matematika yang proses penyelesaiannya menggunakan bahasa python yang memanfaatkan kondisi for dan while?

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.

- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan-python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB IX
LIST DAN TUPLE

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran
   Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu memahami dan mengimplementas tipe data list dan tuple.
b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami tentang tipe data dan variabel.
c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya
d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk mampu menjelaskan tipe data list dan tuple serta membuatnya dalam sebuah Bahasa pemprograman.

B. Pokok Bahasan

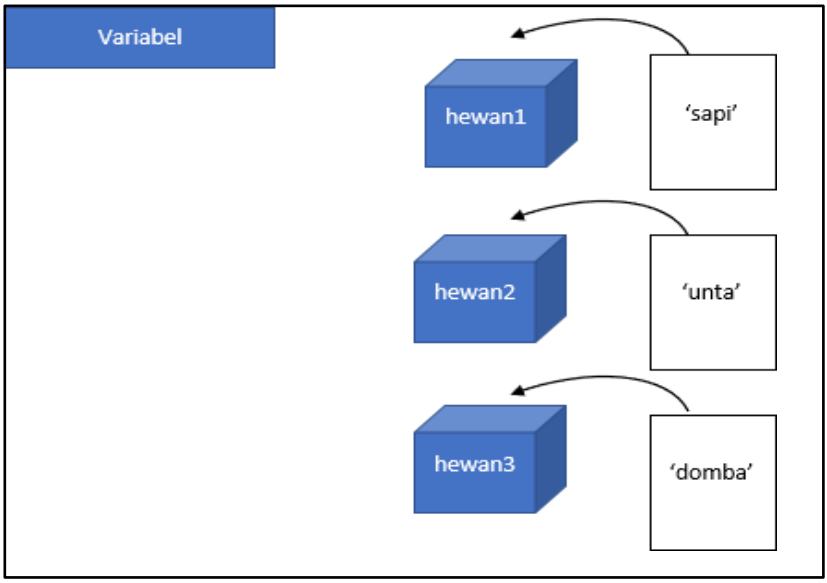
1. Pengantar List

Python memiliki kelebihan dalam kemampuannya mengolah struktur data. Struktur data adalah kumpulan elemen- elemen data (angka atau karakter) yang disusun sedemikian rupa, misalkan memberikan nomor yang spesifik pada elemen-elemen tersebut.

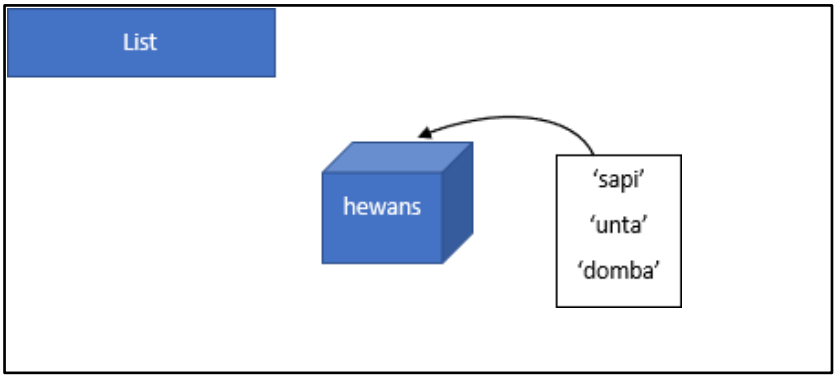
Pada python, struktur data yang paling sederhana disebut *sequence*. Dimana setiap elemennya dalam sebuah *sequence* diberi nomor khusus yang menunjukkan lokasi dari elemen tersebut, atau bisa juga disebut index. Index itu biasanya diurut dari angka 0 dan seterusnya.

Sequence memiliki keterkaitan dengan menggunaan list dan tuples pada pemrograman python. List salah satu tipe data

yang disediakan oleh python dengan tipe data berurut atau *sequence*. Sabagai contoh ilustrasi kita gambar dibawah ini:



Pada gambar diatas, dalam pengolaan data dilakukan secara mandiri atau terpisah, seperti hewan1, hewan2, hewan3. Hal tersebut tentunya tidak efisien, untuk itu solusinya yang lebih baik adalan membuat variabel baru dengan nama hewans untuk mengelola keseluruhan dafatar tersebut. Hasilnya sebagai berikut:



Jadi Tipe data list adalah tipe data koleksi yang bersifat ordered (terurut) dan juga bersifat *changable* (bisa diubah). Tipe data ini bisa kita definisikan dengan tanda kurung siku [] di dalam Python.

Bentuk umum dari list adalah sebagai berikut:

```
[element1, elemen2, elemen3, ...]
```

Setiap nilai di dalam list disebut elemen dan setiap elemenya dipisahkan oleh tanda koma. Berikut ini adalah contoh list pada code program di python.

D: > list.py > ...

```
1   # list kosong
2   list_kosong = []
3
4   # list yang berisi kumpulan string
5   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
6
7   # list yang berisi kumpulan integer
8   list_nilai = [80, 70, 90, 60]
9
10  # list campuran berbagai tipe data
11  list_jawaban = [150, 33.33, 'Presiden Sukarno', False]
```

Pada kode diatas kita bisa melihat bahwa sebuah list di definisikan menggunakan tanda kurung []. List juga tidak hanya berisi tipe data sejenis, pada list juga bisa diisikan berbeda ipt atau campuran. Hal itu dapat dilihat pada contoh di atas pada baris nomor 11, dimana list_jawaban isi datanya ada yang tipe data string, number Boolean.

Untuk menampilkan list kita menggunakan fungsi print (), baik secara keseluruhan maupun secara sebagian saja. Berdasarkan kode diatas, akan dicoba untuk menampilkannya.

```
D: > list.py > ...
1   # list kosong
2   list_kosong = []
3   print('list_kosong:', list_kosong)
4
5   # list yang berisi kumpulan string
6   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
7   print('list_buah:', list_hewan)
8
9   # list yang berisi kumpulan integer
10  list_nilai = [80, 70, 90, 60]
11  print('list_nilai:', list_nilai)
12
13  # list campuran berbagai tipe data
14  list_jawaban = [150, 33.33, 'Presiden Sukarno', False]
15  print('list_jawaban:', list_jawaban)
```

Hasil Outputnya:

```
list_kosong: []
list_buah: ['Sapi', 'Unta', 'Domba', 'Kerbau']
list_nilai: [80, 70, 90, 60]
list_jawaban: [150, 33.33, 'Presiden Sukarno', False]
```

2. Cara Akses Nilai List Dengan Python

Setiap elemen pada list dinomori 0, 1, 2, …dan seterusnya itu disebut dengan indeks. Di atas sudah dijelaskan bahwa indeks di mulai dari angka 0. Kita bisa mendapatkan elemen individual dengan menuli list[index]. Lebih jelasnya sebagai berikut;

```
list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']

Index          0       1       2         3
```

List diatas dimulai dari nilai index 0 dan seterusnya. Untuk mengakses nilainya atau menampilkan isi tertentu dari list tersebut kita gunakan fungsi index. Contoh sebagai berikut:

```
D: > akseslist.py > ...
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3   print(list_hewan [1])
```

Hasil Outputnya:

Contoh kode diatas adalah Kode untuk penampilkan isi nilai dari lisr_hewan dengan indek ke 1 yaitu „Unta". Kita bisa menampilkan sesuai apa yang kita inginkan hanya dengan mengubah nilai indexnya. Jika nilai index pada kode diatas kita ubah menjadi 0 maka hasil outputnya adalah „Sapi".

Kita juga bisa menggunakan indeks negatif untuk menampilkan data dari belakang. Perhatikan contoh berikut ini:

```
D: > akseslist.py > ...
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3   print(list_hewan [-1])
```

Hasil Outputnya:

Hal yang perlu kita perhatikan adalah indeks negatif tidak dimulai dari 0 akan tetapi dimulai dari angka 1.

3. *Slicing List*

Slicing list adalah teknik untuk memotong nilai pada list. Maksudnya adalah kita mengambil beberapa nilai dari anggota list dengan mendefinisikan indeks kiri dan indeks kanan. Perhatikan contoh kode program berikut ini :

```
D: > slicing.py > ...
1
2   # list
3   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
4
5   print(list_hewan[0:1])
6   print(list_hewan[0:2])
7   print(list_hewan[1:3])
8   print(list_hewan[0:-1])
9   print(list_hewan[-1:-3])
10  print(list_hewan[-1:3])
11  print(list_hewan[-3:-1])
```

Hasil Outputnya:

```
['Sapi']
['Sapi', 'Unta']
['Unta', 'Domba']
['Sapi', 'Unta', 'Domba']
[]
[]
['Unta', 'Domba']
```

Keterangan

- parameter indeks sebelah kiri adalah mendefinisikan awal indeks dari nilai yang akan ditampilkan.
- parameter indeks sebelah kanan adalah mendefinisikan batas yang harus ditampilkan.

4. *Slicing* tanpa batas

Kita juga bisa melakukan slicing data tanpa mendefinisikan indeks batas. Coba perhatikan contoh berikut:

```
D: > slicing_tanpabatas.py > ...
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3
4   print(list_hewan[0:])
5   print(list_hewan[1:])
6   print(list_hewan[2:])
7   print(list_hewan[3:])
8   print(list_hewan[:0])
9   print(list_hewan[:1])
10  print(list_hewan[:2])
11  print(list_hewan[:3])
12  print(list_hewan[:4])
```

Hasil Outputnya:

```
['Sapi', 'Unta', 'Domba', 'Kerbau']
['Unta', 'Domba', 'Kerbau']
['Domba', 'Kerbau']
['Kerbau']
[]
['Sapi']
['Sapi', 'Unta']
['Sapi', 'Unta', 'Domba']
['Sapi', 'Unta', 'Domba', 'Kerbau']
```

5. Merubah Nilai List Dengan Python

Kita dapat memperbarui satu atau beberapa nilai di dalam list dengan memberikan potongan di sisi kiri operator penugasan, dan Anda dapat menambahkan nilai ke dalam list dengan metode append (). Sebagai contoh:

```
D: > update_list.py > ...
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3   print ("Isi dari list index ke-1 sebelun diupdate: ",list_hewan[1])
4
5   #update isi nilai list
6   list_hewan[1] = 'Kuda'
7   #menampilkan isi index ke-1 setelah di update
8   print ("Isi dari list index ke-1 setelah diupdate: ",list_hewan[1])
```

Hasil Outputnya:

```
Isi dari list index ke-1 sebelun diupdate:  Unta
Isi dari list index ke-1 setelah diupdate:  Kuda
```

1. **Merubah Nilai dalam Jangkauan**

Di dalam python, kita juga bisa mengubah data dalam range tertentu secara sekaligus. Caranya tidak jauh berbeda dengan apa yang telah kita pelajari pada poin slicing data list.

```
D: > ubahdatarange.py > ...
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3
4   # ubah data dalam range
5   list_hewan[1:3] = ['Srigala', 'Kuda']
6
7   #hasil dari ubah data
8   print(list_hewan)
```

Hasil outputnya:

```
['Sapi', Srigala', 'Kuda', 'Kerbau']
```

Pada code program diatas kita mengubah nilai indek ke-1 dan indek ke-3 menjadi „Srigala " dan „Kuda".

7. **Hapus Nilai List**

Untuk menghapus nilai di dalam list python, kita dapat menggunakan salah satu pernyataan *del* jika Anda tahu persis elemen yang kita hapus. Kita dapat menggunakan metode *remove* () jika Anda tidak tahu persis item mana yang akan dihapus. Sebagai contoh:

```
D: > delete_list.py > ...
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3
4   print (list_hewan)
5   del list_hewan[2]
6   print ("Setelah dihapus nilai pada index 2 : ", list_hewan)
```

Hasil Outputnya:

```
['Sapi', 'Unta', 'Domba', 'Kerbau']
Setelah dihapus nilai pada index 2 :  ['Sapi', 'Unta', 'Kerbau']
```

8. Menambah Nilai List

Setelah kita mengubah data pada list, sekarang kita akan mencoba untuk menambahkan sebuah data baru ke dalam list. Ada 3 cara dalam menambah data nilai pada list:

1. Menambah data di belakang

   Pertama, kita bisa menggunakan fungsi *append* (). Fungsi ini menerima satu parameter, yang mana parameter tersebut akan dimasukkan sebagai nilai baru pada list, dan nilai baru tersebut berada pada akhir item.

   Berikut ini adalah contohnya:

```
D: > tambahdata1.py > ...
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3   print(list_hewan)
4
5   # tambah data di belakang list
6   list_hewan.append('Kuda')
7
8   #menampilkan hasil dari penambahan data
9   print(list_hewan)
```

   Hasil Outputnya:

```
['Sapi', 'Unta', 'Domba', 'Kerbau']
['Sapi', 'Unta', 'Domba', 'Kerbau', 'Kuda']
```

   Kode program diatas adalah menambahkan data di belakang yaitu „Kuda".

2. Menambah data di depan

   Selain fungsi *append* (), kita juga bisa menambahkan item ke dalam list dengan menggunakan fungsi *insert*(). Fungsi insert ini menerima dua buah parameter:

   1) Parameter pertama untuk mendefinisikan posisi indeks dari data yang akan dimasukkan
   2) Parameter kedua untuk mendefinisikan nilai yang akan dimasukkan ke dalam list

Berikut ini contoh untuk memasukkan nilai Kuda ke dalam list_hewan pada indeks ke-0.

```
D: > tambahdata2.py
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3   print(list_hewan)
4
5   # tambah data di awal list
6   list_hewan.insert(0, 'Kuda')
7
8   #menampilkan hasil dari penambahan data
9   print(list_hewan)
```

Hasil Outputnya:

```
['Sapi', 'Unta', 'Domba', 'Kerbau']
['Kuda', 'Sapi', 'Unta', 'Domba', 'Kerbau']
```

3. Menambah data dimanapun

   Tidak hanya terbatas indeks 0, kita juga bisa memasukkan nilai pada indeks berapa pun pada list. Berikut ini contohnya:

```
D: > tambahdata3.py > ...
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3   print(list_hewan)
4
5   # tambah data di index mana pun dalam list
6   list_hewan.insert(2, 'Kuda')
7
8   #menampilkan hasil dari penambahan data
9   print(list_hewan)
```

Hasil Outputnya:

```
['Sapi', 'Unta', 'Domba', 'Kerbau']
['Sapi', 'Unta', 'Kuda', 'Domba', 'Kerbau']
```

9. Menggabungkan dua buah list atau lebih

Berikutnya hal umum yang biasa kita lakukan dengan list adalah: menggabungkan dua buah list (atau lebih) menjadi satu kesatuan.

Bisa jadi kita memiliki 3 list berikut:

a = [1, 2, 3]

b = [„nama“]

c = [True, „alamat“, False]

kita ingin menggabungkan ketiga list tersebut menjadi satu.

```
D: > gabunglist.py > ...
1   #list
2   a = [1, 2, 3]
3   b = ['nama']
4   c = [True, 'alamat', False]
5
6   #menggabungkan list
7   listbaru = a + b +c
8
9   #hasil penggabungan
10  print(listbaru)
```

Hasil Outputnya:

```
[1, 2, 3, 'nama', True, 'alamat', False]
```

10. Mengurutkan List

List juga bisa diurutkan. Untuk mengurutkan list kita menggunakan fungsi *sort*().

Berikut ini contohnya:

```
D: > sortlist.py > ...
1   # list
2   list_hewan = ['Sapi', 'Unta', 'Domba', 'Kerbau']
3   print(list_hewan)
4
5   # urutkan secara ascending
6   list_hewan.sort()
7   print('Hasil mengurutan List')
8   print(list_hewan)
9
10  # membalikkan posisi item list (tidak harus berurut)
11  list_hewan.reverse()
12  print('Membalikkan Urutan List')
13  print(list_hewan)
```

Hasil outputnya:

```
['Sapi', 'Unta', 'Domba', 'Kerbau']
Hasil mengurutan List
['Domba', 'Kerbau', 'Sapi', 'Unta']
Membalikkan Urutan List
['Unta', 'Sapi', 'Kerbau', 'Domba']
```

11. Pengertian Tuple

Secara umum, terdapat 4 tipe data koleksi pada python, yaitu:

- *List*
- *Tuple*
- *Set*
- *Dictionary*

Masing-masing dari 4 tipe data di atas memiliki sifat dan kegunaan masing-masing. Agar kita tahu kapan kita membutuhkan tipe data a dan kapan kita membutuhkan tipe data b, maka kita harus mempelajari semuanya dengan baik. Tuple adalah 1 dari 4 tipe data kolektif pada python yang berguna untuk menyimpan lebih dari satu nilai dalam satu variabel secara sekaligus [1].

Tuple bersifat *ordered* (terurut) dan juga bersifat *unchangable* (tidak bisa diubah). Ordered berarti datanya bisa kita akses menggunakan indeks, dan unchangeable berarti datanya tidak akan pernah bisa diubah setelah pertama kali definisikan.

Tuple sama saja dengan list. Sama-sama digunakan untuk menyimpan data himpunan. Sama-sama bisa menampung berbagai macam tipe data dalam satu himpunan. Hanya saja setelah diberi nilai, tuple tidak bisa diubah lagi. Hal ini berbeda dengan list. Dari segi penulisan, list menggunakan kurung siku [] sedangkan tuple menggunakan kurung biasa ().

3 cara untuk membuat sebuah tuple. Perhatikan contoh code program berikut:

```
D: > tuple.py > ...
1   # cara standar
2   tuple_jenis_kelamin = ('laki-laki', 'perempuan')
3
4   # tanpa kurung
5   tuple_status_perkawinan = 'menikah', 'lajang'
6
7   # menggunakan fungsi tuple()
8   tuple_lulus = tuple(['lulus', 'tidak lulus'])
```

Cara yang pertama adalah cara biasa/standar. Cara yang kedua tanpa tanda kurung, ini mungkin kelihatan agak aneh, tapi yang seperti ini normal di python. Cara yang ketiga adalah dengan menggunakan fungsi tuple () dan melemparkan list sebagai parameternya.

12. Cara Mengakses Nilai Tuple

Sama halnya dengan list, Tuple juga memiliki indeks untuk dapat mengakses nilai di dalamnya. Indeknya juga dimulai dari angka 0.

```
D: > aksestuple.py > ...
1   # membuat tuple
2   nama_hewan = ('Sapi', 'Unta', 'Domba', 'Kerbau')
3
4   # mengakses nilai tuple
5   print(nama_hewan[1])
```

Hasil Outputnya:

```
Unta
```

Pada code program diatas kita menampilkan nilai indeks ke 1 dari tuple nama_hewan yaitu „Unta".

Selain indeks posirtif kita juga bisa melakukan pemanggilan dengan indek negatif seperti contoh beikut:

```
D: > aksestuple.py > ...
1   # membuat tuple
2   nama_hewan = ('Sapi', 'Unta', 'Domba', 'Kerbau')
3
4   # mengakses nilai tuple
5   print(nama_hewan[-2]) #indek ke-(-2)
6   print(nama_hewan[-1])#indek ke-(-1)
```

Hasil outputnya:

```
Domba
Kerbau
```

Selain cara diatas, dalam mengakses tuple, kita bisa juga menggunakan perulangan *for*. Berikut cohtohnya:

```
D: > fortuple.py > ...
1   # membuat tuple
2   nama_hewan = ('Sapi', 'Unta', 'Domba', 'Kerbau')
3
4   for i in nama_hewan:
5       print(i)
```

Hasil Outputnya:

```
Sapi
Unta
Domba
Kerbau
```

13. *Slicing* Tuple

Slicing adalah teknik memotong nilai dari sebuah tuple. Sintaksnya sama saja dengan teknis slicing di list. Tidak berbeda. Untuk melakukan slicing, kita perlu mendefinisikan range indeks dengan pemisah tanda titik dua (:). Perhatikan kode berikut:

```
D: > slicingtuple.py > ...
1   # membuat tuple
2   nama_hewan = ('Sapi', 'Unta', 'Domba', 'Kerbau')
3
4   # dari indeks nomo 0 sampai 1
5   print(nama_hewan[0:1])
6   # dari indeks nomo 0 sampai 2
7   print(nama_hewan[0:2])
8   # dari indeks nomo 1 sampai 3
9   print(nama_hewan[1:3])
10  # dari indeks nomo 0 sampai -1
11  print(nama_hewan[0:-1])
```

Hasil Outputnya:

```
('Sapi',)
('Sapi', 'Unta')
('Unta', 'Domba')
('Sapi', 'Unta', 'Domba')
```

14. Mengubah Data Tuple

Dalam mengubah data di tuple sangat berbeda dibandingkan mengubah data pada list. Pada list kita bisa mengubah data sesuai yang kita inginkan. Namu, pada data tuple kita tidak bisa mengubhanya. Jika datanya kita ubah maka akan menimbulkan pesan eror pada program. Perhatikan contoh tuple berikut:

```
D: > updatetuple.py > ...
1   # membuat tuple
2   nama_hewan = ('Sapi', 'Unta', 'Domba', 'Kerbau')
3   nama_hewan[0] = 'Gajah'
```

Hasil Outputnya:

```
Traceback (most recent call last):
  File "d:/updatetuple.py", line 3, in <module>
    nama_hewan[0] = 'Gajah'
TypeError: 'tuple' object does not support item assignment
```

Hasilnya akan terjadi error. Jadi, jika kita ingin bisa mengubah nilai pada data, disarankan untuk menggunakan list. Tuple dapat dipergunakan untuk kasus-kasus tertentu, dimana nilai datanya bersifat tetap atau tida berubah selama runtime seperti jenis kelamin.

15. Tuple Bersarang

Tuple juga bisa nested, artinya Tuple bisa diisi dengan Tuple. Untuk lebih jelasnya perhatikan contoh berikut ini:

```
# Representasi koordinat dalam ruang 3D
koordinat = ((0, 0, 0), (1, 2, 3), (4, 5, 6))

# Akses koordinat tertentu
koordinat_pertama = koordinat[0]
x, y, z = koordinat_pertama

print("Koordinat x, y, z:", x, y, z)
# Output: Koordinat x, y, z: 0 0 0
```

16. *Sequence Unpacking*

Fitur selanjutnya dari Tuple adalah: *sequence unpacking*. Fitur ini berfungsi untuk mengekstrak isi dari tuple ke dalam variabel-variabel tunggal secara berurutan. Kita hanya perlu menggunakan operator assignment standar (simbol sama dengan =) dan mendefinisikan nama variabel dengan koma

Contohnya adalah seperti ini dengan Asterisk (*):

```
# Tuple
my_tuple = (1, 2, 3, 4, 5)

# Unpacking dengan asterisk
a, *b = my_tuple

print(a)  # Output: 1
print(b)  # Output: [2, 3, 4, 5]
```

Dengan nested tuple:

```
# Nested Tuple
nested_tuple = (1, (2, 3), 4)

# Unpacking nested tuple
a, (b, c), d = nested_tuple

print(a)  # Output: 1
print(b)  # Output: 2
print(c)  # Output: 3
print(d)  # Output: 4
```

Dengan melakukan upacking, isi tuple akan di-copy ke variabel. Lalu dengan variabel kita bisa melakukan apapun, seperti mengubah isinya. Karena variabel bersifat mutable.

Fungsi bawaan python yang berfungsi untuk melakukan operasi pada tuple. Berikut adalah daftarnya:

ungsi	Deskripsi
all()	Mengembalikan True jika semua anggota tuple adalah benar (tidak ada yang kosong)
any()	Mengembalikan True jika salah satu atau semua bernilai benar. Jika tuple kosong, maka akan mengambalikan False.
enumerate()	Mengembalikan objek enumerasi. Objek enumerasi adalah objek yang terdiri dari pasangan indeks dan nilai.
len()	Mengembalikan panjang (jumlah anggota) tuple
max()	Mengembalikan anggota terbesar di tuple
min()	Mengembalikan anggota terkecil di tuple
sorted()	Mengambil anggota tuple dan mengembalikan list baru yang sudah diurutkan
sum()	Mengembalikan jumlah dari semua anggota tuple
tuple()	Mengubah sequence (list, string, set, dictionary) menjadi tuple

C. Kesimpulan

1. List salah satu tipe data yang disediakan oleh python dengan tipe data berurut atau sequence
2. Tuple sama seperti string dan list, merupakan anggota tipe data berurut, cara membuatnya dengan menggunakan kurung biasa atau tanpa tanda kurung

D. Latihan

1. Jelaskan apakah yang membedakan antara list dan tuple?
2. Buatlah kode program sederhana untuk nilai mahasiswa menggunakan list?
3. Buatlah suatu list bernama buku dengan 3 string ini sebagai elemennya: “Ekonomi”, “Matematika”, “IPS“.
   - Buatlah elemen “Biologi” dari buku menjadi huruf besar dan kemudian cetak list?
   - Hapus elemen “disease” dari things dan cetak list?
4. Buatlah suatu list bernama tahun_list, dimulai dengan tahun kelahiran anda, dan seterusnya sampai tahun saat

anda berumur 5 tahun. Sebagai contoh, jika anda lahir pada 1980. List tahun_list = [1989, 1990, 1991, 1992, 1993, 1994].

- Pada taun berapakah dalam tahun_list anda berumur 4 tahun?
- Pada tahun berapakah dalam tahun_list anda paling tua?

5. Buatlah sebuah tuple bernama mahasiwa dengan 5 string ini sebagai elemennya: "Anton", "Kiki", "Desy", "Fania", "Rendra".
   - Buatlah elemen "Fajar" dari mahasiswa menjadi "Fajar" dan kemudian cetak tuple?
   - Hapus elemen "Anton" dari mahasiswa dan cetak tuple?

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.
- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.

- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.
- https://www.programiz.com/python-programming/while- loop – diakses tanggal 17 Mei 2021

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB X
LARIK

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran
   Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu menjelaskan dan mengimplementasikan tipe data larik.
b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami dan menggunakan tipe data list dan tuple pada Bahasa pemrograman python.
c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya
d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk mampu menjelaskan tipe data LARIK serta membuatnya dalam sebuah bahasa pemprograman python.

B. Pokok Bahasan

1. Pengertian Larik Pada Python

Larik adalah sekumpulan variabel yang memiliki tipe data yang sama dan dinyatakan dengan nama yang sama. Larik merupakan konsep yang penting dalam pemrogaman karena larik memungkinkan untuk menyimpan data maupun referensi objek dalam jumlah banyak dan terindeks.

Bagian yang menyusun larik di sebut elemen larik. Sehingga dengan menggunakan larik, sejumlah variabel dapat menggunakan nama yang sama.

Sebagai contoh, kita bisa menggunakan larik untuk menampung data siswa yang lebih dari satu.

```
siswa = ["Ali", "Andi", "Ani"]
```

Dengan menggunakan larik pada Python maka kita dapat menghemat banyak waktu. Seperti yang disebutkan sebelumnya, larik membantu kita mengurangi ukuran keseluruhan kode yang kita buat, sementara Python membantu kita menghilangkan sintaks yang bermasalah, tidak seperti bahasa lain. Misalnya: Jika kita ingin menyimpan bilangan bulat dari 1–100, kita tidak akan dapat mengingat 100 nama variabel secara eksplisit, oleh karena itu, kita dapat menyimpannya dengan mudah menggunakan larik.

2. Cara Mengakses Larik di Python

Anda dapat mengakses salah satu nilai di dalam kumpulan larik dengan mengacu kepada indeks dari setiap data. Di dalam larik, indeks di mulai dari nol. Jadi untuk mengakses nilai pertama dimulai dari nol.

Contohnya pada saat Anda ingin menampilkan Budi maka bisa menggunakan kode di bawah ini:

```
D: > array.py > ...
1   siswa = ["Budi", "Charlie", "Erlangga"]
2   print(siswa[0])
```

Output:

```
Budi
```

3. Fungsi Larik di Python

Beberapa Fungsi yang Berkaitan dengan larik di Python sebagai barikut:

1. *Index*: Fungsi ini digunakan untuk mengembalikan indeks kemunculan pertama yang nilainya disebutkan dalam argument. Ini digunakan jika ada kesalahan.

2. *Append*: Fungsi ini digunakan untuk menambahkan nilai di akhir daftar. Jika data siswa di atas belum cukup, Anda bisa menambahkan nilai di dalamnya dengan menggunakan keyword *append* ().
   Contoh:

```
x = [1,2,3,4,5]
y = (6,7,8)
x.append(y)
x
```

```
print(f"Index 5 pada list x adalah: {x[5:]}")
Index 5 pada list x adalah: [(6, 7, 8)]
```

3. *Remove*: Fungsi ini digunakan untuk menghapus nilai pertama dari daftar. Anda juga bisa menghapus nilai di dalam larik dengan menggunakan *keyword pop*() atau *remove*(). Perbedaan keduanya terletak pada target yang ingin dihapus. Keyword pop() digunakan dengan berdasarkan indeks, sedangkan remove digunakan dengan berdasarkan value.
   Contoh :

```
job = ["Data Scientist", "Data Analyst", "Data Engineer", "Data Scientist",
"Business Intelligence"]
job.remove("Data Analyst")
print(job)
```

Outputnya akan menjadi,

```
["Data Scientist", "Data Analyst", "Data Engineer", "Data Scientist"]
```

4. *Pop*: Fungsi ini digunakan untuk menghapus item pada posisi tertentu dalam daftar, dan mengembalikannya. Jika tidak ada indeks yang ditentukan, a.pop () menghapus dan mengembalikan item terakhir dalam daftar.

Contoh :

```
buah = ['apel', 'pisang', 'ceri']

x = buah.pop(1)

print(x)
```

Output:

```
pisang
```

5. *Count*: Fungsi ini digunakan untuk mengembalikan jumlah item dengan nilai yang ditentukan.

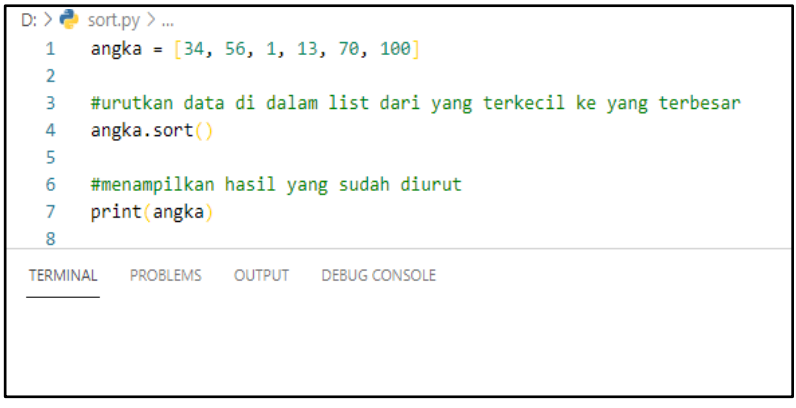
```
angka = [1, 4, 2, 9, 7, 8, 9, 3, 1]

x = angka.count(9)

print(x)
```

Output:

```
2
```

6. *Sort*: Fungsi ini digunakan untuk mengurutkan item dari daftar.

Contoh:



7. *Reverse*: Fungsi ini digunakan untuk mengembalikan urutan daftar.
   Contoh :

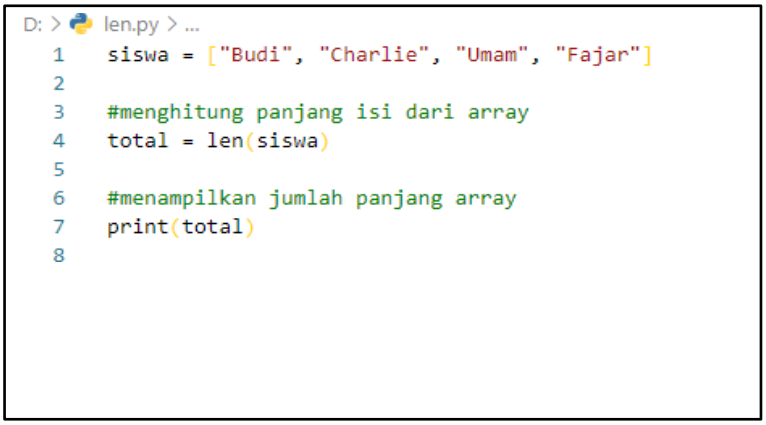
```
D: > reserve.py > ...
1   angka = [34, 56, 1, 13, 70, 100]
2
3   #membalikkan  urutan list
4   angka.reverse()
5
6   #menampilkan hasil urutan yang di balik
7   print(angka)
8
```

8. *Extend*: Fungsi ini digunakan untuk memperpanjang daftar dengan menambahkan semua item dalam daftar yang diberikan.
   Contoh:

```
D: > extend.py > ...
1   siswa = ["Budi", "Charlie"]
2
3   #menambahkan string
4   siswa.extend("Umam")
5
6   #menampilkan hasil
7   print(siswa)
8
```

9. Len: Fungsi ini digunakan untuk digunakan untuk mengembalikan nilai berupa jumlah item di daftar.

Cohtoh ;



C. Kesimpulan

1. Larik adalah sekumpulan variabel yang memiliki tipe data yang sama dan dinyatakan dengan nama yang sama.
2. Beberapa Fungsi yang Berkaitan dengan larik di Python yaitu Index. Append, Remove dan masih banyak lainnya.

D. Latihan

1. Apa yang dimaksud dengan larik?
2. Bagaimana cara mendeklarasikan sebuah larik di python?
3. Bagaimana cara mengakses elemen sebuah larik di python?
4. Buatlah dan jelaskan langkah-langkahnya untuk membuat sebuah larik di python?
5. Jelaskan beberapa istilah/methode/fungsi yang berkaitan dengan larik di python berikut ini serta buatkan contoh kode programnya: index, append, insert, remove.

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.

- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.
- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB XI
FUNGSI

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran

   Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu menganalisis konsep fungsi serta mengimplementasikan dalam program aplikasi.

b. Kompetensi yang Diharapkan;

   Mahasiswa telah memahami penggunaan tipe data, variabel, Algoritma runtutan, Algoritma fungsi percabangan dan Algoritma fungsi perulangan dalam bahasa pemrograman python.

c. Keterkaitan materi dengan materi yang lain;

   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya

d. Pentingnya mempelajari isi bab;

   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk memahami konsep dan menggunakan jenis-jenis fungsi sesuai dengan kegunaannya serta mampu menggabungkan konsep fungsi percabangan, perulangan, dan fungsi dalam sebuah contoh kasus.

B. Pokok Bahasan

1. Pengertian Fungsi

Function (fungsi) merupakan salah satu operator dalam python untuk mempermudah penulisan yang menggunakan kode yang berulang-ulang dengan nilai yang berbeda-beda. Sehigga nilai yang berbeda tersebut dapat digunakan oleh fungsi lain didalam program tersebut.

Tujuan dari fungsi yaitu untuk mengefisienkan penulisan suatu program dan mempermudah melakukan pengembangan pada suatu program. Hal ini dikareanan dengan fungsi kita

mendefinisikan sekali dan dapat digunakan berkali-kali pada program tersebut sesuai dengan kebutuhan. Dengan begitu kita tidak perlu munuliskan kode secara berulang-ulang yang mengakibatkan kode program terlalu panjang.

Berikut ini adalah keuntungan dalam menggunakan fungsi:

a. Program besar dapat di pisah-pisah menjadi program-program kecil melalui Function.
b. Kemudahan dalam mencari kesalahan-kesalahan karena alur logika jelas dan kesalahan dapat dilokalisasi dalam suatu modul tertentu.
c. Memperbaiki atau memodifikasi program dapat dilakukan pada suatu modul tertentu saja tanpa menggangu keseluruhan program.
d. Dapat digunakan kembali (*Reusability*) oleh program atau fungsi lain.
e. Meminimalkan penulisan perintah yang sama.

Bentuk Umum dari Fungsi sebagai:

```
def <nama fungsi> (argumen1, argumen2, ..., argumenN):
```

Sebuah fungsi diawali dengan statemen def kemudian diikuti oleh sebuah nama_fungsi nya. Sebuah fungsi dapat memiliki daftar argumen (parameter) ataupun tidak. Tanda titik dua (:) menandakan awal pendefinisian tubuh dari fungsi yang terdiri dari statemen-statemen. Berikut merupakan contoh penggunaan def dalam baha pemrograman python untuk mencetak “Selamat Datang Di Python”:

```
D: > fungsi1.py > ...
1   #membuat Fungsi
2   def cetak():
3       print("Selamat Datang Di Python")
4
5   #memanggil Fungsi
6   cetak()
```

Pada kode program diatas untuk mencetak “Selamat Datang Di Python” kita hanya cukup memanggil nama fungsinya saja *cetak()* sehingga code blok fungsi yang sudah dibuat dijalankan dan output atau keluaran dari code program diatas adalah sebagai berikut:

```
Selamat Datang Di Python
```

Berikut ini adalah contoh perbedaan dari program tanpa menggunakan fungsi dengan menggunakan fungsi:

```
D: > notfungsi.py
1   print("Halo!!!")
2   print("Selamat Pagi Budi")
3   print("Apa Kabarmu Hari Ini")
4
5   print("Halo!!!")
6   print("Selamat Pagi Adi")
7   print("Apa Kabarmu Hari Ini")
8
9   print("Halo!!!")
10  print("Selamat Pagi Irma")
11  print("Apa Kabarmu Hari Ini")
```

Hasil outputnya:

```
Halo!!!
Selamat Pagi Budi
Apa Kabarmu Hari Ini
Halo!!!
Selamat Pagi Adi
Apa Kabarmu Hari Ini
Halo!!!
Selamat Pagi Irma
Apa Kabarmu Hari Ini
```

Pada kode proragm diatas, kode yang dituliskan terlalu banyak dan ditulis secara berulang-ulang. Dengan menggunakan fungsi kita bisa lebih mengefisienkan penuliasan kode proram diatas. Berikut ini kode program diatas kita ubah menjadi lebih efisien dengan menggunakan fungsi.

Hasil Outpunya:

D: > fungsi2.py > ...

```python
#membuat Fungsi
def salampagi(nama):
    print("Halo!!!")
    print("Selamat Pagi", nama)
    print("Apa Kabarmu Hari Ini")

#memanggil Fungsi
salampagi("Budi")
salampagi("Adi")
salampagi("Irma")
```

```
Halo!!!
Selamat Pagi Budi
Apa Kabarmu Hari Ini
Halo!!!
Selamat Pagi Adi
Apa Kabarmu Hari Ini
Halo!!!
Selamat Pagi Irma
Apa Kabarmu Hari Ini
```

Kode program diatas menggunakan fungsi dengan nama fungsinya salampagi(nama). Dan fungsi tersebut diatas dibuat dengan cara menambhakan kata kunci def di awal deklarasi.

2. Cara Memanggil Fungsi di Python

Sebuah fungsi diawali dengan statemen def kemudian diikuti oleh sebuah nama_fungsi nya. Sebuah fungsi dapat memiliki daftar argumen (parameter) ataupun tidak. Tanda titik dua (:) menandakan awal pendefinisian tubuh dari fungsi yang terdiri dari statemen-statemen.

a) Fungsi Dengan Argumen (parameter)

Kita dapat memberikan sebuah nilai pada fungsi ketika kita memangginya, nilai itulah yang disebut argument dan dapat digunakan berulang-ulang dalam sebuah fungsi. Untuk memberika argument pada sebuah fungsi, fungsi tersebut harus mempunyai variable untuk dapat menerimanya dan variable tersebut disebut parameter.

Kita dapat lihat pada contoh code program diatas pada fungsi salam pagi () dilengkapi dengan argument nama dan nama itulah yang disebut variabel. Parameter nama tersbut dimaksudkan agar ketika kita gunakan, kita bisa mengatur nama orang yang akan muncul pada output.

Jadi ketika kita menulis salampagi(“Budi”) maka yang muncu outpunya adalah seperti berikut

```
Halo!!!
Selamat Pagi Budi
Apa Kabarmu Hari Ini
Halo!!!
```

b) Fungsi Dengan Lebih Dari Satu Argumen (parameter)

Kita dapat membuat sebuha fungsi dengan lebih dari satu argument atau parameter dengan cara memisahkannya setiap parameter dengan tanda koma (,). Jumlah Argument dapat di sesuaikan dengan kebutuhan kita. Sebagai contoh kita ingin membuat fungsi untuk menghitung luas persegi panjang.

D: > fungsi2parameter.py > ...

```
1  #membuat Fungsi
2  def menghitung_luas_persegipanjang(panjang, lebar):
3      luas = (panjang * lebar)
4      print("Luas Persgi Panjang = ", luas)
5
6  #memanggil Fungsi
7  menghitung_luas_persegipanjang(10,5)
8  menghitung_luas_persegipanjang(3,7)
```

Hasil Outputnya adalah:

```
Luas Persgi Panjang =  50
Luas Persgi Panjang =  21
```

Dibaris nomor 2, kita definisikan sebuah fungsi *menghitung_luas_persegipanjang (panjang, lebar)* dengan 2 buah parameter yaitu panjang dan lebar. Ketika fungsi dijalankan cukup masukkan nilai sebagai parameter

panjang dan lebar. Anka 10 adalah nilai untuk argument panjang dan 5 adalah untuk lebar.

c) Fungsi Dengan Memanfaatkan Keyword

Pada contoh kode fungsi menghitung luas persegi panjang argumen pada fungsi di sisikan berdasarkan urutannya. Jadi jika dalam pembuatan fungsinya dituliskan *menghitung_luas_persegi Panjang (panjang, lebar)* maka nilai pertama yang digunakan untuk mengisi argumen tersebut adalah nilai panjang dan di ikuti nilai kedua yaitu lebar dan ini tidak bisa di tukar. Namun pada Bahasa python kita diberikan izin utnu membuat fungsi yang nialinya tidak harus berurutan sesuai dengan yang dituliskan. Dengan menambahkan *keyword* pada setiap argumennya. Contohnya sebagai berikut:

D: > keywordfungsi.py > ...

```
1  #membuat Fungsi
2  def menghitung_luas_persegipanjang(panjang, lebar):
3      luas = (panjang * lebar)
4      print("Luas Persgi Panjang = ", luas)
5
6  #memanggil Fungsi
7  menghitung_luas_persegipanjang(panjang=10,lebar=5)
8  menghitung_luas_persegipanjang(lebar=5,panjang=10)
9
```

Hasil Outpunya:

```
Luas Persgi Panjang =  50
Luas Persgi Panjang =  50
```

Pada kode program diatas posisi penulisan nilai argumen diberikan keyword argumen pada fungsinya. Dan hasil keduanya adalah sama tidak membedakan hasil.

d) Fungsi Dengan Jumlah Argumen yang tak diatur

Kita bisa membuat fungsi yang jumlah argumennya fleksibel. Yang artinya fungsi itu bisa diisi dengan satu, dua tiga atau lebih argumen oleh para pengguna dengan

cara menambahkan (*) sebelum nama parameter dalam definisi fungsi. Berikut ini contohnya:

```
D: > arg.py > ...
1   #membuat Fungsi
2   def salampagi(*nama):
3       print("Halo!!!")
4       print("Selamat Pagi")
5       print("Apa Kabarmu Hari Ini")
6       for orang in nama:
7           print (orang)
8
9   #memanggil Fungsi
10  salampagi("Budi", "Adi", "Irma")
```

Hasil Outputnya:

```
Halo!!!
Selamat Pagi
Apa Kabarmu Hari Ini
Budi
Adi
Irma
```

Kita bisa menggunakan satu atau lebih argumen secara langsung saat menulis fungsi salam pagi. ketika kita menambahkan tanda (*) maka kita bisa memberikan argumen berapapun sesuai keinginan kita. Untuk menampilkan nilai dalam argumen, kita bisa memanfaatkan perulangan For.

3. Fungsi Menggunakan R*eturn*

Fungsi Return di gunakan ketiak kita ingin membuat sebuah fungsi yang hanya saja menghasilkan nilai. Nilai yang dimaksud adalah terdapat sebuah perhitungan seperti tambah atau perkalian kemudian dari hasil tersebut memberikan nilai balik.

Berikut ini adalah contoh penggunaan return:

```
D: > return.py > ...
1  #membuat Fungsi
2  def menghitung_luas_persegipanjang(panjang, lebar):
3      luas = (panjang * lebar)
4      return luas
5
6  print("Luas Persgi Panjang = ", menghitung_luas_persegipanjang(10, 5))
```

Hasil Outputnya:

```
Luas Persgi Panjang =  50
```

Pada kode program diatas, fungsi menghitung_luas_persegipanjang hanya utnuk menghasilkan nilai perhitungan panjang dikali lebar. Selanjutnya, diluar fungsi, kita bisa tampilkan nilai itu untuk menggunakan print seperti biasa.

4. Bekerja dengan Variabel dalam fungsi

Di python ada beberapa jenis variabel. Yaitu variable global dan variable lokal. Variabel Global adalah variabel yang dapat diakses dan digunakan di seluruh kode program. Sedangkan untuk variabel lokal adalah variabel yang dapat di akses hanya pada fungsi dimana variabel itu berada.

Di python variabel yang di akses terlebih dahulu adalah variable local kemudian variable global. Jika pada dalam pengaksesan variabel lokal di temukan, maka variabel lokal tersebut yang di akses terlebih dahulu.

Berikut adalah contoh kode program penggunaan variable global dan variabel lokal:

```
asal = "Dari global"

def pesan1():
    asal = "Dari local #1"
    print(asal)

def pesan2():
    print(asal)

pesan1()
pesan2()
```

C. Kesimpulan

1. *Function* (fungsi) merupakan salah satu operator dalam python untuk mempermudah penulisan yang menggunakan kode yang berulang-ulang dengan nilai yang berbeda-beda.

D. Latihan

1. Jelaskan pengertian fungsi menurut pemahaman Anda?
2. Jelakan cara mengakses fungsi pada python?
3. Buatlah kode program menghitung luas segitiga menggunakan fungsi?
4. Buatlah kode program utnuk menyelesaiakn permasalahan matematika menggunakan operator fungsi dengan parameter dan tanpa return?
5. Buatlah kode program utnuk menyelesaiakn permasalahan matematika menggunakan operator fungsi dengan tanpa parameter dan tanpa return?

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas

Ilmu Pendidikan UMJ.

- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.
- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB XII
DICTIONARY

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran
   Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu menganalisis konsep tipe data dictionary dan membuktikannya dalam program aplikasi.
b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami penggunaan tipe data, variabel, Algoritma teknik runtutan, Algoritma fungsi percabangan dan Algoritma fungsi perulangan dalam bahasa pemrograman python.
c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya
d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk memahami konsep tipe data jenis dictionary serta mengimplementasikan tipe data dictionary dalam sebuah contoh kasus.

B. Pokok Bahasan

- **Pengertian Dictionary**

  Bagian lain dari python yang juga dapat digunakan untuk pengolahan struktur data adalah Dictionary. Struktur data bekerja menggunakan metode mapping, dimana nilai-nilai dirujuk dengan nama-nama yang spesifik. Nama-nama tersebut kemudian disebut dengan istilah key, sedangkan nilai-nilainya disebut dengan definition.

  Konsep dictionary pada python dapat dikembangkan menjadi aplikasi-aplikasi praktis, seperti penggolongan warna buah dimana nama buah sebagai key dan warnanya

sebagai definition, pembuatan buku telepon yang mengaitkan nama(key) dan nomor telepon (definition). Bisa juga nama mahasiswa dan nilai ujiannya.

Berikut cara penulisan sebuah Dictionary pada Python:

variabel_dictionary = { “key” : ”value” }

Atau

variabel_dictionary =

{
“key” : ”value”
}

Cara membuat dictionary pada python ada 2 cara:

1. Menggunakan kurung {}
2. Menggunakan konstruktor .dict( ).

Contoh penulisan dictionary dengan tanda kurawal pada python:

```
# Contoh dictionary sederhana
data_mahasiswa = {
    "nama": "Andi",
    "nim": "220101001",
    "jurusan": "Informatika",
    "ipk": 3.75
}

# Akses nilai dalam dictionary
print(data_mahasiswa["nama"])      # Output: Andi
print(data_mahasiswa["ipk"])       # Output: 3.75
```

Contoh diatas merupakan 1 baris. Jika kita ingin menambahkan lebih dari satu baris data, maka caranya sebagi berikut :

```
# Tambah lebih dari satu data (baris demi baris)
data_mahasiswa["jurusan"] = "Teknik Informatika"
data_mahasiswa["ipk"] = 3.75
data_mahasiswa["angkatan"] = 2022

print(data_mahasiswa)
```

Jadi, pada kode diatas kita membuat variabel mahasiswa dengan tipe dictionary yang mempunyai key dan value

Key -> nama, value -> vaniamonica
Key -> alamat, value -> pamekasan

Contoh penulisan dictionary dengan tanda kurawal pada python:

```
# Dictionary berisi data mahasiswa
mahasiswa = {
    "nama": "Sari",
    "nim": "210401002",
    "jurusan": "Sistem Informasi",
    "ipk": 3.92
}

# Menampilkan seluruh isi dictionary
print(mahasiswa)
```

Kita coba contoh aplikasi sederhana yang menggunakan dictionary:

D: > dictionarynilai.py > ...

```
# dictionary
nilai_mahasiswa = {
    "umam" : 80,
    "desy" : 90,
    "fania" : 70,
    "anton" : 100,
    "siska" : 85,
}

nama = input("masukkan nama mahasiswa: ")
if nama in nilai_mahasiswa:
    print("nilai UAS Mahasiswa", nama, " adalah", nilai_mahasiswa[nama])
else:
        print("data Mahasiswa tidak ditemukan.")
        print("berikut nama-nama Mahasiswa:")

        for i in nilai_mahasiswa.keys():
            print(i)
```

Hasil Outputnya:

```
masukkan nama mahasiswa: umam
nilai UAS Mahasiswa umam  adalah 80

masukkan nama mahasiswa: fania
nilai UAS Mahasiswa fania  adalah 70

masukkan nama mahasiswa: anas
data Mahasiswa tidak ditemukan.
berikut nama-nama Mahasiswa:
umam
desy
fania
anton
siska
```

Pada kode program diatas, apabila data mahasiswa tidak ditemukan maka seluruh nama siswa yang tercacat di dalam dictionary akan di tampilkan.

2. Mengakses Nilai Item Dari Dictionary

Kita bisa mengakses item pada dictionary dengan dua cara:

1. dengan menggunakan kurung siku ([]).
2. atau dengan menggunakan fungsi get ().

Perhatikan contoh berikut:

```
buku = {
    "judul": "Algoritma dan Pemrograman",
    "penulis": "Bonifacius Vicky",
    "tahun": 2023
}
```

Hasil Outputnya:

```
# Akses nilai berdasarkan key
print(buku["judul"])     # Output: Algoritma dan Pemrograman
print(buku["penulis"])   # Output: Bonifacius Vicky
print(buku["tahun"])     # Output: 2023
```

```
# .get() tidak akan error jika key tidak ada
print(buku.get("judul"))        # Output: Algoritma dan Pemrograman
print(buku.get("penerbit"))     # Output: None (karena key tidak ada)
```

Keunggulan get adalah Kita bisa mengatur nilai default jika key pada dictionary yang kita panggil tidak ditemukan. Hal itu akan mencegah sistem untuk menampilkan error.

3. Perulangan Untuk Dictionary

Kita dapat menampilkan dictionary dengan menggunakan perulangan. Berikut ini contohnya:

```
mahasiswa = {
    "nama": "Rina",
    "nim": "220301005",
    "jurusan": "Informatika",
    "ipk": 3.85
}
```

```
for k in mahasiswa:
    print(k)
```

```
for k in mahasiswa:
    print(k, ":", mahasiswa[k])
```

Hasil Output:

```
nama
nim
jurusan
ipk
```

```
nama : Rina
nim : 220301005
jurusan : Informatika
ipk : 3.85
```

4. Operasi di Dictionary

Operasi Dictionary antara lain

- Menambah data
- Menghapus data
- Meng*update* data
- Menghitung jumlah data

1) Menambah Data Item Dictionary

Untuk menambahkan key dan item baru, caranya sebagai

berikut:

- Kalau *key* yang kita definisikan ternyata sudah ada, sistem akan me-*replace* item yang lama dengan yang baru.
- Tapi jika *key* yang kita definisikan ternyata tidak ada, maka sistem akan menambahkannya sebagai item baru.

```
# Dictionary awal
mahasiswa = {
    "nama": "Andi",
    "nim": "220101001"
}

# Menambah data item baru
mahasiswa["jurusan"] = "Teknik Informatika"
mahasiswa["ipk"] = 3.75

# Cetak hasilnya
print(mahasiswa)
```

Hasil Outputnya:

```
{'nama': 'Andi', 'nim': '220101001', 'jurusan': 'Teknik Informatika', 'ipk': 3.75}
```

Pada kode program diatas, pertama key dictionarynya hanya nama dan alamat, kemudian di tambahkan *key* hoby.

2) Menghapus Data Item Dictionary

Ada dua metode untuk menghapus data di Dictionary yaitu:

- Menggunakan statement del < dict[key] >.
- Atau menggunakan fungsi dictionary.pop ().

Berikut ini adalah contoth hapus data dengan menggunakan perintah del:

```
mahasiswa = {
    "nama": "Andi",
    "nim": "220101001",
    "jurusan": "Informatika"
}

# Hapus item 'jurusan'
del mahasiswa["jurusan"]

print(mahasiswa)
```

Hasil Outputnya:

```
{'nama': 'Andi', 'nim': '220101001'}
```

Berikut ini adalah contoth hapus data dengan menggunakan perintah pop:

```
ipk = mahasiswa.pop("ipk", "Tidak ada")  # 'ipk' akan dihapus jika ada

print(mahasiswa)
print("IPK yang dihapus:", ipk)
```

3) Mengupdate Data Item Dictionary

Seperti yang telah kita singgung di atas bahwasanya item pada dictionary bersifat changeable alias bisa diubah.

Untuk mengubah nilai item pada suatu dictionary, caranya simpel seperti mengubah variabel pada umumnya.

```
mahasiswa = {
    "nama": "Rina",
    "nim": "220301005",
    "ipk": 3.65
}

# Update nilai IPK
mahasiswa["ipk"] = 3.85

print(mahasiswa)
```

Hasil Outputnya:

```
{'nama': 'Rina', 'nim': '220301005', 'ipk': 3.85}
```

4) Menghitung Data Item Dictionary

kita bisa juga menghitung jumlah key yang terdapat pada sebuah dictionary dengan fungsi len().

Berikut ini contohnya:

```
mahasiswa = {
    "nama": "Dina",
    "nim": "220301008",
    "jurusan": "Teknik Informatika",
    "ipk": 3.90
}

# Menghitung jumlah item (key-value pairs)
jumlah_item = len(mahasiswa)

print("Jumlah item dalam dictionary:", jumlah_item)
```

Hasil Outputnya:

```
Jumlah item dalam dictionary: 4
```

C. Kesimpulan

1. *Dictionary* adalah bagian lain dari python yang juga dapat digunakan untuk pengolahan struktur data.

D. Latihan

1. Jelaskan apakah yang membedakan antara *list* dan *tuple* dan *dictionary*?
2. Buatlah suatu dictionary bernama kota dengan 3 string ini sebagai elemennya: “Jakarta”, “Bandung”, “Surabaya”.
    - Cetak dictionary?
    - Tambahkan elemen “Banten” dari kota dan cetak dictionary?
    - Hitung jumlah elemen sekarang pada dictionary kota?
3. Buatlah sebuah dictionary mahasiswa dengan menampilkan nama, prodi, pembimbing kemudian tampilkan hasilnya?
4. Buatlah sebuah dictionary mahasiswa dengan menampilkan nama, prodi, pembimbing dan gunakan perulangan for untuk menampilkan hasilnya?

5. Dari soal nomor 4, update elemen dictionary dari “Banten” menjadi “Semarang” kemudian tampilkan hasilnya?

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.
- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

BAB XIII
FILE

A. Pendahuluan

a. Tujuan dan Capaian Pembelajaran
   Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu memahami file serta operasi-operasi yang digunakan pada file dalam bahasa pemrograman python.
b. Kompetensi yang Diharapkan;
   Mahasiswa telah memahami penggunaan tools bahasa pemrograman python dan memahami karakter data yang akan diolah.
c. Keterkaitan materi dengan materi yang lain;
   Materi pendahuluan sebelum materi lainnya agar memahami materi selanjutnya
d. Pentingnya mempelajari isi bab;
   Setelah membaca dan mempelajari bab ini, mahasiswa diharapkan mampu untuk membaca sebuah file untuk mengolah data atau menulis ke dalam file untuk menyimpan data yang kita olah.

B. Pokok Bahasan

- **Pengertian File di Python**

  Python mampu membuka semua jenis file selama itu tidak terenkripsi. Atau gampangnya file jika file bisa dibuka menggunakan aplikasi semacam notepad, maka file tersebut juga bisa dibuka pada python.

  Selain .txt python juga dapat membaca dan menulis dengan ekstensi .csv. .md, json, dan sebagainya. Kode untuk membaca dan menulis file dalam python adalah sebagai berikut:

  <nama_variabel> = open (“<nama_file>”, mode)

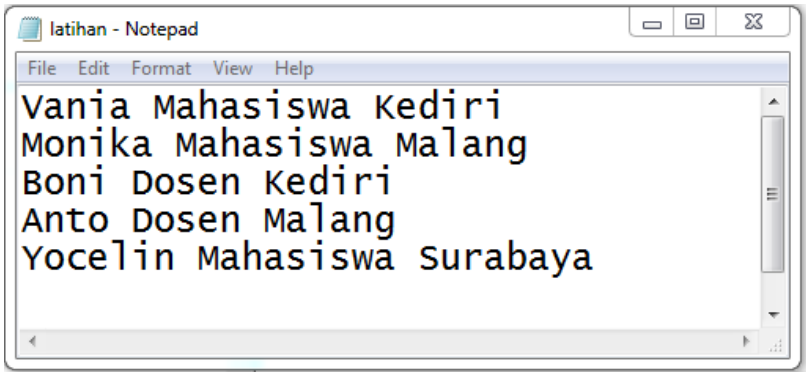
Beberapa mode yang dapat digunakan python untuk membaca dan menulis dalam file, yaitu :

MODE AKSES	DESKRIPSI
r	Membuka file untuk mode baca
w	Membuka file untuk mode tulis
a	Membuka file untuk mode tambah
rb	Membuka file untuk mode baca dalam format binari
wb	Membuka file untuk mode tulis dalam format binari
ab	Membuka file untuk mode tambah dalam format binari
r+	Membuka file untuk mode baca dan tulis
w+	Membuka file untuk mode tulis dan baca
a+	Membuka file untuk mode tambah dan baca
rb+	Membuka file untuk mode baca dan tulis dalam format binari
wb+	Membuka file untuk mode tulis dan baca dalam format binari
ab+	Membuka file untuk mode tambah dan baca dalam format binari

File teks akan dapat langsung dibaca apabila file berada dalam satu direktori bersama script python yang akan membaca file teks tersebut. Tetapi apabila file teks tidak berada dalam satu direktori dengan script python yang akan membaca file teks, maka harusnya dituliskan path file teks nya.

2. Membaca File di Python

Sebagai contoh kita akan membuat sebuah file dengan format text file (.txt) di notepad dan di save dengan nama file latihan1.txt, seperti berikut ini:



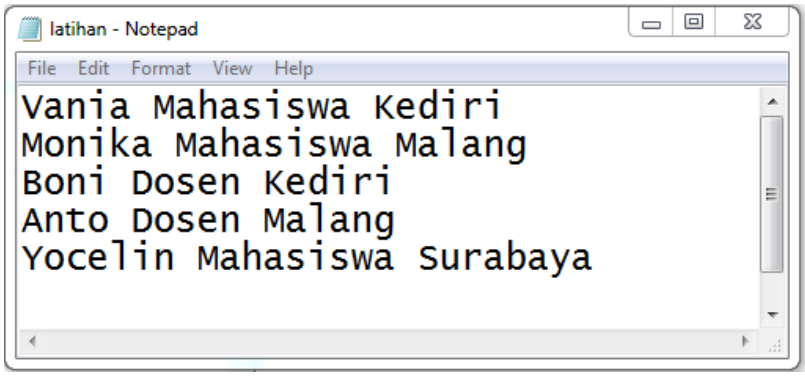
Data di atas memiliki 3 kolom yang dipisahkan oleh spasi " ". Pemisah ini penting, karena akan digunakan untuk memisahkan kolom data tanpa perlu mengetahui berapa jumlah kolomnya.
Kemudian kita akan buat program dalam script sebagai berikut:

bacafile.py ● latihan.txt

D: > bacafile.py

```
1  #membuka file latihan.txt dengan mode read
2  data = open("D:\\latihan.txt")
3
4  #mencetak isi latihan.txt
5  print(data.read())
```

Hasil Outputnya:

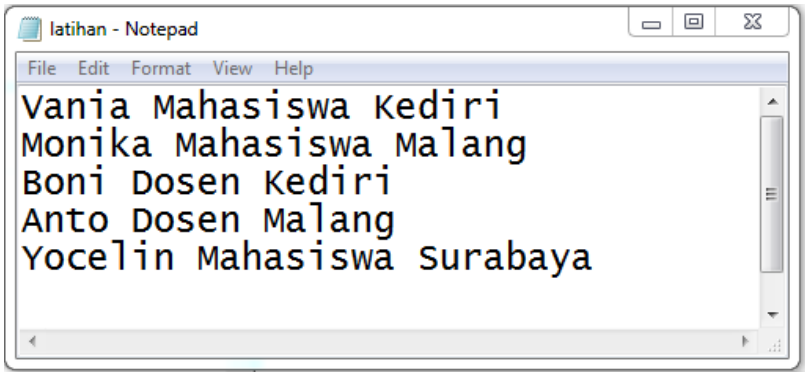


"latihan.txt" adalah nama file yang akan dibuka. Nama file selalu ditulis sebagai string atau menggunakan tanda petik (" "). Setelah tanda koma ada "r" yang merupakan singkatan dari "read". Huruf tersebut menunjukkan apa yang akan dilakukan terhadap file tadi, yaitu membacanya.
Untuk memudahkan pembacaan seluruh baris data, python memiliki sebuah fitur loop menggunakan for. Berikut ini adalah script untuk membaca seluruh baris data.

D: > fileloop.py

```
1  #membuka file latihan.txt dengan mode read
2  data = open("D:\\latihan.txt")
3
4  #mencetak isi latihan.txt dengan menggunakan perulangan
5  for i in data:
6      print(i)
```

Hasil Outputnya:



for i in data adalah variabel i akan terisi dengan data satu baris pada setiap perulangannya. "Fajar Mahasiswa Pamekasan" adalah baris pertama, kemudian berulang membaca seluruh baris file hingga "Meri Mahasiswa Surabaya" pada baris terakhir.

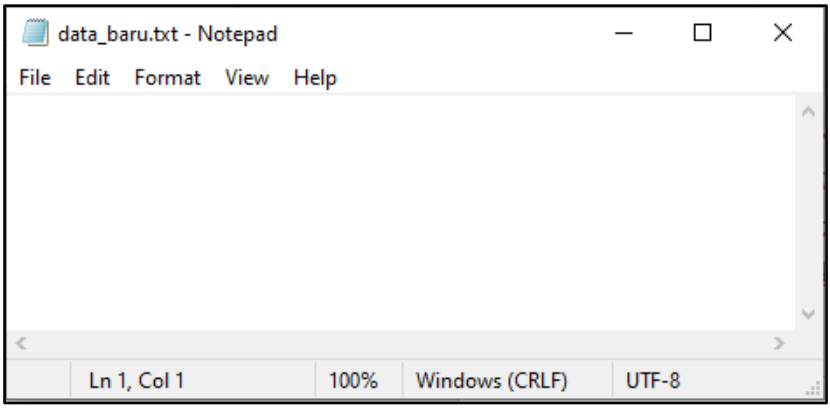
Teknik ini akan membaca satu baris data menjadi satu string. Padahal, satu baris data terdiri dari tiga kolom, yang kurang lebih berisi Nama, Pekerjaan, dan Asal, yang masing- masing dipisahkan dengan spasi " ".

3. Menulis Data File di Python

Untuk menuliskan ke dalam sebuah file pada python, kita dapat menggunakan mode ('w') saat membuka filenya. Atau juga dapat menggunakan mode ('a') untuk menambah isi file dari akhir file awal, tanpa menghapus atau menimpa isinya terlebih dahulu.

Kita harus sangat berhati – hati dalam menggunakan mode ('w') hal ini dikarenakan bisa menimpa file jika filenya sudah ada datanya. Semua data di file yang tertimpa akan terhapus dan diganti dengan isi baru.

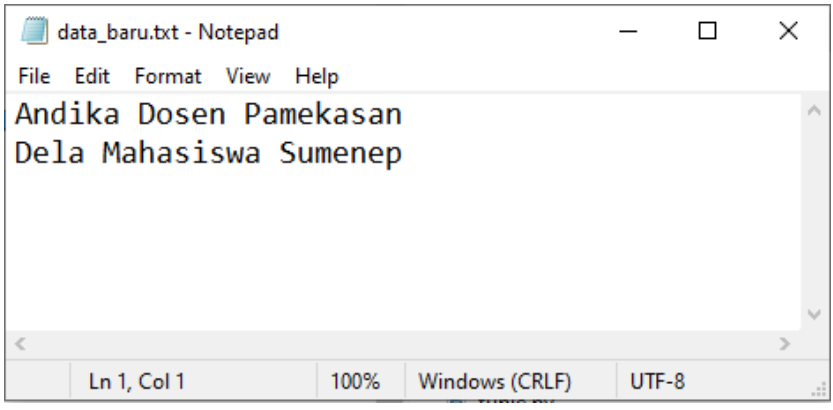
Kita buat file baru dengan nama data_baru.txt yang isinya kita kosongkan.



Kemudian selanjutnya kita buat kode program untuk menulis pada data_baru.txt tersebut.

```
D: > writefile.py
1   #membaca file databaru.txt dengan mode write
2   data = open("D:\\data_baru.txt", "w")
3
4   #menulis kalimat di file databaru.txt
5   data.write("Andika Dosen Pamekasan\n")
6   data.write("Dela Mahasiswa Sumenep\n")
7   data.close()
```

Jalankan kode program diatas dan hasil outputnya:



Data_baru.txt yang awalnya kosong sekrang sudah terisi sesuai apa yang kita tulis atau masukkan pada kode program diatas.

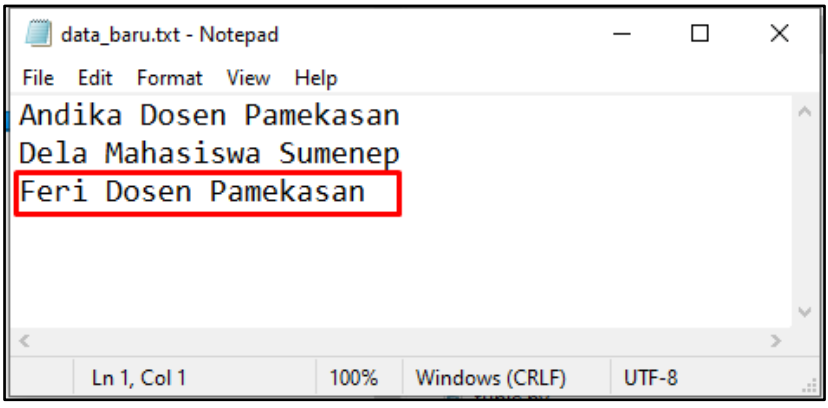
Bagaimana jika kita ingin menuliskan pada sebuah file namun tidak dengan menghilangkan data yang sudah ada pada file txt tersebut. berikut ini adalah caranya.

Apabila kita ingin memasukkan tulisan kedalam sebuah file tanpa menghilangkan isi dari file sebelumnya, atau dengan kata lain ingin menyisipkan data ke dalam file, maka dalam program gunakan mode “a” (append).

Sebagai contoh kita ingin menambahakan sebuah data yang “Feri Dosen Bangkalan” ke dalam data_baru.txt. Langkahnya sebagai berikut:

```
D: > appendfile.py
1   #membaca file databaru.txt dengan mode write
2   data = open("D:\\data_baru.txt", "a")
3
4   #menambahkan kalimat di file databaru.txt
5   data.write("Feri Dosen Pamekasan\n")
6   data.close()
```

Hasil Outputnya:



Mode append akan menulis tanpa menghapus teks yang sudah ada di file sedangkan mode write akan mereplace ulang teks di file

Jadi, ketika kita ingin menulis kembali di file databaru.txt dengan isi teks yang berbeda maka isi file sebelumnya akan hilang diganti dengan kalimat baru

4. Menghapus File di Python

Untuk menghapus file kita gunakan library os dengan fungsinya yaitu remove () Misal kita ingin menghapus file databaru.txt maka kode akan seperti ini.

```
D: > hapusfile.py
1   #import library os
2   import os
3
4   #perintah menghapus file
5   os.remove("data_baru.txt")
```

Jalankan kode program di atas, maka data_baru.txt akan di hapus.

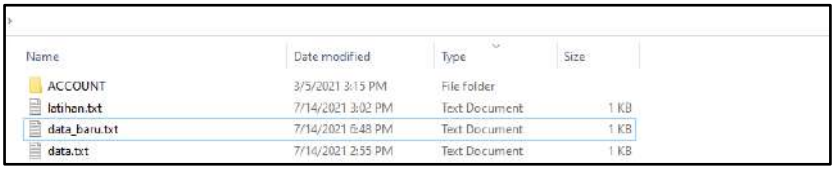
Mengganti Nama File

Selain dapat membca, menulis, menyisipkan data dalam file, python juga dapat mengganti nama file, menggunakan modul os.

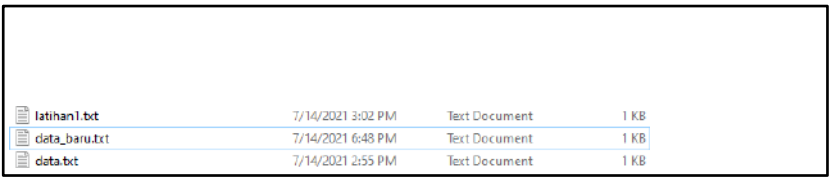
Sebagai contoh akan diubah nama file latihan.txt menjadi latihan1.txt.

```
D: > renamefile.py
1   import os
2   os.rename("D:\\latihan.txt", "D:\\latihan1.txt")
```

Sebelum kode program dijalankan file bernama latihan.txt



Setelah kode program dijalankan, Outputnya adalah file berganti nama latihan1.txt



C. Kesimpulan

1. File adalahsebuah data yang ada pada komputer, baik itu berupa teks, angka, gambar, video, suara dan lain sebagainya.

2. Python dapat terintegrasi dengan aplikasi lain seperti .txt (notepad). Kode program untuk membaca, menulis, mengubah dan lainnya tergantung mode. Dengan kode program: <nama_variabel> = open ("<nama_file>", mode)

D. Latihan

1. Buatlah kode program untuk menyelesaikan permasalahan matematika menggunakan python dengan memanfaatkan file?
2. Buatlah sebuah file dengan tipe CSV kemudian buat kode program di python untuk mengakses file CSV tersebut?

E. Rujukan

- Munir, Rinaldi .2005. *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.
- Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.
- ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.
- Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.
- Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python*.
- Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.
- Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan- python/.
- Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

F. Referensi Penunjang

- Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

DAFTAR PUSTAKA

Cormen, Thomas H. dkk. *Introduction to Algorithms*.3nd Edition. London England. The MIT Press. 2009.

Dianta, Ava (2019), *Logika dan Algoritma Pemrograman.*

ISMAH. 2017. *Pemrograman Komputer*. Jakarta: Fakultas Ilmu Pendidikan UMJ.

Jubilee Enterprise.2019. *Python untuk Programer Pemula*.Yoyakarta:PT Elex Komputindo.

Munir, Rinaldi (2005), *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 1, Edisi Ketiga, Penerbit Informatika Bandung.

Munir, Renaldi. 2016. *Algoritma Dan Pemrograman*. Bandung: INFORMATIKA Bandung.

Mohit, Da, B.N. (2017). *Learn Python in 7 Days*. Birmingham: Packt.

Padmanabhan, T.R. (2016). *Programming with Python*. Singapore: Springer.

Pythonindo.2019. Diakses pada 28 Juni 2020, dari https://www.pythonindo.com/program-pertama-dengan-python/.

Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python.*

Rafiqi, Aufa (2019), *Belajar Algoritma Pemrograman dengan Python.*

Swastika, W.2018. *Belajar Algoritma Pemrograman Dengan Menggunakan Python*. Ashera Publisher.

Tutorial Dasar Python untuk Pemula. Diakses pada 5 July 2021, dari https://www.petanikode.com/tutorial/python/.

DAFTAR ISTILAH

Algoritma diartikan sebagai urutan langkah-langkah yang disusun secara logis dan sistematis dalam menyelesaikan suatu masalah.

Asembly adalah bahasa pemrograman tingkat rendah yang dipergunakan dalam pemrograman komputer. Bahasa ini juga dikenal dengan istilah bahasa mesin

Index didefinisikan sebagai suatu simbol atau angka yang digunakan untuk mengidentifikasikan nilai atau angka tertentu dalam sebuah deret atau himpunan unsur-unsur yang sama.

Kompleks adalah suatu kesatuan yang terdiri dari sejumlah bagian, khususnya yang memiliki bagian yang saling berhubungan dan saling tergantung.

Sequence Unpacking merupakan fitur yang berfungsi dalam melakukan mengekstraki isi dari tuple ke dalam variabel-variabel tunggal secara berurutan.

Slicing list adalah teknik untuk memotong nilai pada list atau mengambil beberapa nilai dari anggota list dengan mendefinisikan indeks kiri dan indeks kanan.

Tuple Bersarang adalah istilah yang menggambarkan penggunaan tupple didalam tupple.

Operator Bitwise adalah operator khusus untuk menangani operasi logika bilangan biner dalam bentuk bit 0 dan 1.

Operator *assignment* atau operator penugasan adalah operator yang digunakan untuk memasukkan suatu nilai ke dalam variabel.

Programer adalah seseorang yang mempunyai kemampuan atau keahlian untuk merancang dan menulis kode program komputer menggunakan bahasa pemrograman komputer (Python, Java, Php, Javascript dll).

Platfrom adalah kombinasi antara perangkat lunak dan perangkat keras untuk menjalankan sebuah program atau aplikasi.

Variabel Global adalah variabel yang dapat digunakan atau dipanggil atau dikenali oleh semua fungsi / prosedur / dikenali diseluruh program.

TENTANG PENULIS

Bonifacius Vicky Indriyono, S.Kom., M.Kom, Lahir di Kediri pada tanggal 18 Februari 1979. Menempuh pendidikan S1 jurusan Sistem Informasi di Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Kadiri dan lulus tahun 2012, Kemudian melanjutkan pendidikan S2 jurusan Teknik Informatika dengan konsentrasi Sistem Informasi di AMIKOM Yogyakarta dan lulus tahun 2015.

Syahril Wahyu Ramadhan, S.Par, M.M, Lahir di Nganjuk pada tanggal 04 Februari 1997 Menempuh pendidikan S1 jurusan Hospitality and Tourism Business di Universitas Ciputra Surabaya dan lulus tahun 2019 Kemudian melanjutkan pendidikan S2 jurusan Magister Manajemen di Universitas Merdeka Malang dan lulus tahun 2023.

Motto Hidup Penulis "Hidup adalah proses yang disediakan Tuhan...tetaplah bersabar dan lalui dengan keiklasan...karena ada tujuan indah dibalik proses Tuhan"