



FUTURE SCIENCE

ALGORITMA DAN PEMROGRAMAN DENGAN BAHASA PEMROGRAMAN PYTHON

Editor :

Tundo, S.Kom., M.Kom.

Penulis :

Ade Oktarino | Santi Prayudani | Kuswandi
Yogi Kristiyanto | Sugiyono | Hidayatus Sibyan
Muhammad Rizqy Septyandy | Pahrul Irfan
Erna Hudianti Pujiarini | Nur Oktavin Idris
Rudy Cahyadi | Ariesta Damayanti

Bunga Rampai

**ALGORITMA DAN PEMROGRAMAN
DENGAN BAHASA PEMROGRAMAN PYTHON**

UU No 28 tahun 2014 tentang Hak Cipta

Fungsi dan sifat hak cipta Pasal 4

Hak Cipta sebagaimana dimaksud dalam Pasal 3 huruf a merupakan hak eksklusif yang terdiri atas hak moral dan hak ekonomi.

Pembatasan Pelindungan Pasal 26

Ketentuan sebagaimana dimaksud dalam Pasal 23, Pasal 24, dan Pasal 25 tidak berlaku terhadap:

- i. Penggunaan kutipan singkat Ciptaan dan/atau produk Hak Terkait untuk pelaporan peristiwa aktual yang ditujukan hanya untuk keperluan penyediaan informasi aktual;
- ii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk kepentingan penelitian ilmu pengetahuan;
- iii. Penggandaan Ciptaan dan/atau produk Hak Terkait hanya untuk keperluan pengajaran, kecuali pertunjukan dan Fonogram yang telah dilakukan Pengumuman sebagai bahan ajar; dan
- iv. Penggunaan untuk kepentingan pendidikan dan pengembangan ilmu pengetahuan yang memungkinkan suatu Ciptaan dan/atau produk Hak Terkait dapat digunakan tanpa izin Pelaku Pertunjukan, Produser Fonogram, atau Lembaga Penyiaran.

Sanksi Pelanggaran Pasal 113

1. Setiap Orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf i untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan/atau pidana denda paling banyak Rp100.000.000 (seratus juta rupiah).
2. Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf c, huruf d, huruf f, dan/atau huruf h untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan/atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah).

ALGORITMA DAN PEMROGRAMAN DENGAN BAHASA PEMROGRAMAN PYTHON

Penulis:

Ade Oktarino
Santi Prayudani
Kuswandi
Yogi Kristiyanto
Sugiyono
Hidayatus Sibyan
Muhammad Rizqy Septyandy
Pahrul Irfan
Erna Hudianti Pujiarini
Nur Oktavin Idris
Rudy Cahyadi
Ariesta Damayanti

Editor:

Tundo, S.Kom, M.Kom.



ALGORITMA DAN PEMROGRAMAN DENGAN BAHASA PEMROGRAMAN PYTHON

Penulis:

Ade Oktarino
Santi Prayudani
Kuswandi
Yogi Kristiyanto
Sugiyono
Hidayatus Sibyan
Muhammad Rizqy Septyandy
Pahrul Irfan
Erna Hudianti Pujiarini
Nur Oktavin Idris
Rudy Cahyadi
Ariesta Damayanti

Editor: **Tundo, S.Kom., M.Kom.**

Desain Cover: **Nada Kurnia, S.I.Kom.**

Tata Letak: **Samuel, S.Kom.**

Halaman: **A5 Unesco (15,5 x 23 cm)**

Ukuran: **xii, 199**

e-ISBN: **978-634-7037-82-4**

Terbit Pada: **Februari 2025**

Hak Cipta 2025, Pada Penulis

Isi diluar tanggung jawab percetakan

Copyright © 2025 by Future Science Publisher

All Right Reserved

Hak cipta dilindungi undang-undang
Dilarang keras menerjemahkan, memfotokopi, atau
memperbanyak sebagian atau seluruh isi buku ini
tanpa izin tertulis dari Penerbit.

**PENERBIT FUTURE SCIENCE
(CV. FUTURE SCIENCE)**

Anggota IKAPI (348/JTI/2022)

Jl. Terusan Surabaya Gang 1 A No. 71 RT 002 RW 005, Kel. Summersari, Kec. Lowokwaru, Kota
Malang, Provinsi Jawa Timur.
www.futuresciencepress.com

KATA PENGANTAR

Puji syukur kami panjatkan kepada Tuhan Yang Maha Esa atas rahmat dan karunia-Nya sehingga buku ALGORITMA DAN PEMROGRAMAN DENGAN BAHASA PEMROGRAMAN PYTHON ini dapat terselesaikan dengan baik. Buku ini diharapkan dapat memberikan kontribusi yang berarti bagi pengembangan ilmu dan praktik dalam bidang ilmu komputer di Indonesia.

Buku ini merupakan panduan komprehensif yang dirancang untuk pemula yang ingin belajar bahasa pemrograman python beserta struktur algoritmanya. Terdiri dari 12 bab: Pengantar Algoritma, Struktur Dasar Algoritma, Notasi Algoritma Menggunakan (Kalimat Deskriptif, Flowchart, dan Pseudo Code), Pengantar Bahasa Pemrograman Python, (Tipe Data, Operator, dan Ekspresi pada Python), Perintah Percabangan pada Python, Perintah Perulangan pada Python, Manipulasi String dan Operasinya, Array, Pemrograman Modular dengan Fungsi, Pointer pada Python, Operasi File.

Kami ingin mengucapkan terima kasih yang sebesar-besarnya kepada Penerbit Future Science yang telah memberikan dukungan dan kesempatan untuk menerbitkan buku ini. Dukungan dari penerbit sangat berarti bagi kami dalam proses penyusunan dan penerbitan buku ini. Ucapan terima kasih juga kami sampaikan kepada para penulis yang telah berkontribusi dengan pengetahuan, pengalaman, dan dedikasi mereka.

Semoga buku ini dapat memberikan manfaat yang besar bagi para pembaca dan menjadi referensi yang berharga dalam penguasaan pemrograman bahasa python untuk konsep algoritma dan pemrograman.

Jakarta, Desember 2024

Editor,

Tundo

DAFTAR ISI

KATA PENGANTAR.....	v
DAFTAR ISI	vii
BAB 1 PENGANTAR ALGORITMA	1
Ade Oktarino.....	1
PENDAHULUAN	1
KARAKTERISTIK ALGORITMA.....	1
TIPE DATA NUMERIK	3
JENIS TEKS	5
JENIS SQUENCE.....	5
JENIS PEMETAAN	6
JENIS SET	6
TIPE BOOLEAN	7
OPERATOR	7
KESIMPULAN.....	9
BAB 2 STRUKTUR DASAR ALGORITMA	13
Santi Prayudani	13
PENDAHULUAN	13
STRUKTUR URUT (SEQUENCE)	13
STRUKTUR PEMILIHAN (SELECTION)	16
STRUKTUR PENGULANGAN (REPETITION).....	21
KESIMPULAN.....	28
BAB 3 NOTASI ALGORITMA MENGGUNAKAN KALIMAT DESKRIPTIF, FLOWCHART, DAN PSEUDOCODE	31

	Kuswandi	31
	PENDAHULUAN	31
	PENYAJIAN ALGORITMA.....	31
	KESIMPULAN	44
BAB 4	Pengantar Bahasa Pemrograman Python	45
	Yogi Kristiyanto.....	45
	PENDAHULUAN	45
	PERSIAPAN AWAL DENGAN PYTHON	47
	MENGGUNAKAN EDITOR UNTUK BAHASA PEMROGRAMAN PYTHON	51
	LATIHAN MEMBUAT SCRIPT DENGAN PYTHON	57
	KESIMPULAN	58
BAB 5	TIPE DATA, OPERATOR, DAN EKSPRESI PADA PYTHON	61
	Sugiyono	61
	TIPE DATA	61
	OPERATOR.....	67
	EKSPRESI	72
	KESIMPULAN	73
BAB 6	PERINTAH PERCABANGAN PADA PYTHON	77
	Hidayatus Sibyan	77
	PENDAHULUAN	77
	STRUKTUR IF	79
	STRUKTUR IF-ELSE	80
	STRUKTUR IF-ELIF-ELSE	83
	PERCABANGAN BERSARANG (NESTED IF)	86

	PENGGUNAAN OPERATOR LOGIKA DALAM IF	88
	KESIMPULAN.....	91
BAB 7	PERINTAH PERULANGAN PADA PYTHON.....	95
	Muhammad Rizqy Septyandy	95
	PENDAHULUAN	95
	KESIMPULAN.....	108
BAB 8	MANIPULASI STRING DAN OPERASINYA.....	111
	Pahrul Irfan	111
	DASAR-DASAR STRING DI PYTHON	111
	METODE BAWAAN UNTUK MANIPULASI STRING.....	113
	FORMAT STRING DAN F-STRINGS.....	115
	PENGGUNAAN REGULAR EXPRESSIONS PADA STRING	117
	PENKODEAN DAN DEKODE STRING.....	119
	KESIMPULAN.....	122
BAB 9	ARRAY	127
	Erna Hudianti Pujiarini	127
	PENDAHULUAN	127
	DEKLARASI ARRAY	129
	OPERASI PADA ARRAY	137
	MENGURUTKAN DAN MENCARI ELEMEN ARRAY	144
	KESIMPULAN.....	147
BAB 10	PEMROGRAMAN MODULAR DENGAN FUNGSI.....	151
	Nur Oktavin Idris	151
	PENGANTAR PEMROGRAMAN MODULAR.....	151

FUNGSI DALAM PYTHON	153
VARIABEL LOKAL DAN GLOBAL	156
FUNGSI REKURSIF	157
FUNGSI LAMBDA	159
MODUL DAN PAKET	160
MEMBANGUN APLIKASI MODULAR.....	163
KESIMPULAN	166
BAB 11 POINTER PADA PYTHON.....	169
Rudy Cahyadi.....	169
PENDAHULUAN	169
KONSEP DASAR POINTER.....	170
MANAJEMEN MEMORI DI PYTHON	172
MUTABLE DAN IMMUTABLE OBJECT	174
MANIPULASI OBYEK MENGGUNAKAN REFERENSI	180
KESIMPULAN.....	181
BAB 12 OPERASI FILE.....	185
Ariesta Damayanti.....	185
PENDAHULUAN	185
OPERASI FILE.....	187
MEMBUKA FILE	189
MEMBACA FILE.....	191
MENULIS KE FILE	192
MENAMBAHKAN PADA FILE	193
MENUTUP FILE.....	194

MENGUBAH NAMA FILE.....	195
MENGHAPUS FILE	195
MENULIS FILE CSV PADA PYTHON.....	197
KESIMPULAN.....	198

BAB 1

PENGANTAR ALGORITMA

Ade Oktarino
Universitas Adiwangsa Jambi, Kota Jambi
E-mail: adeoktarino@unaja.ac.id

PENDAHULUAN

Algoritma adalah serangkaian instruksi atau langkah-langkah yang dirancang untuk menyelesaikan suatu masalah atau melakukan tugas tertentu secara sistematis. Dalam ilmu komputer, algoritma adalah inti dari pemrograman dan bertanggung jawab untuk memecahkan masalah secara efisien. Algoritma dapat diimplementasikan dalam berbagai bahasa pemrograman dan digunakan dalam berbagai aplikasi, mulai dari pencarian data hingga pemrosesan gambar dan pengembangan kecerdasan buatan.

Kata "algoritma" berasal dari nama ilmuwan Persia abad ke-9, Muhammad ibn Musa al-Khwarizmi, yang menulis buku tentang penyelesaian masalah matematika menggunakan metode sistematis. Algoritma telah berkembang pesat sejak itu, terutama dengan kemajuan teknologi komputer di abad ke-20 dan ke-21.

KARAKTERISTIK ALGORITMA

Sebuah algoritma yang baik harus memiliki karakteristik dengan demikian berikut penjelasan mengenai karakteristik algoritma:

a) Definiteness (Ketegasan)

Definiteness atau ketegasan dalam algoritma merujuk pada kejelasan dan ketepatan setiap langkah atau instruksi dalam algoritma tersebut. Setiap langkah dalam algoritma harus spesifik, jelas, dan tidak boleh ambigu sehingga dapat

diikuti tanpa adanya kebingungan atau interpretasi yang berbeda.

Adapun Pentingnya Definiteness sebagai berikut:

1) Kejelasan Pelaksanaan:

Instruksi yang jelas memastikan bahwa algoritma dapat diimplementasikan secara konsisten oleh siapa pun atau oleh mesin tanpa kesalahan interpretasi.

2) Kesalahan Minimal:

Dengan instruksi yang tegas dan jelas, kemungkinan kesalahan selama eksekusi dapat diminimalkan, yang sangat penting untuk mencapai hasil yang akurat dan dapat diandalkan.

3) Reproducibility (Reproduksi):

Algoritma yang tegas dapat dijalankan berulang kali dengan hasil yang sama, asalkan input yang sama diberikan. Ini penting untuk keperluan pengujian dan validasi.

Contoh Kejelasan Instruksi

Untuk memahami konsep ketegasan, berikut adalah contoh sederhana dari algoritma untuk menemukan nilai maksimum dalam sebuah daftar angka:

b) Finiteness (Keterbatasan)

Finiteness atau keterbatasan dalam konteks algoritma berarti bahwa algoritma harus selalu selesai setelah sejumlah langkah tertentu. Algoritma tidak boleh berjalan tanpa henti; ia harus memiliki titik akhir yang jelas, di mana hasil yang diinginkan tercapai atau masalah dinyatakan tidak dapat diselesaikan.

Adapun pentingnya Finiteness (Keterbatasan) sebagai berikut:

- Keberakhiran: Algoritma yang memiliki finiteness memastikan bahwa proses akan berhenti setelah

sedangkan operator memungkinkan berbagai operasi pada data tersebut. Dengan menguasai kedua konsep ini, Anda dapat menulis kode Python yang efisien, logis, dan mudah dipahami.

DAFTAR PUSTAKA

- Afriansyah, A., Oktarino, A., & Turnip, A. (2019). Expert System for Diagnosing Children Allergic Diseases Through Web Forward Chaining. *Internetworking Indonesia Journal*, 11(2), 23–27.
- Gagniuc, P. A. (2023). *Data Types and Statements BT - An Introduction to Programming Languages: Simultaneous Learning in Multiple Coding Environments* (P. A. Gagniuc (Ed.); hal. 79–104). Springer International Publishing. https://doi.org/10.1007/978-3-031-23277-0_5
- Husni, A. F., & Oktarino, A. (2021). Aplikasi Jambiku Bersih Dengan Permodelan Menggunakan UML. *Jurnal Ilmiah Media Sisfo*, 15(1), 1. <https://doi.org/10.33998/mediasisfo.2021.15.1.932>
- Liben-Nowell, D. (Ed.). (2022). Basic Data Types. In *Connecting Discrete Mathematics and Computer Science* (2 ed., hal. 5–78). Cambridge University Press. <https://doi.org/DOI: 10.1017/9781009150484.002>
- Oktarino, A., Afriansyah, A., & Turnip, A. (2020). Design and Implementation of Android-Based Village Fund Monitoring Application. *Internetworking Indonesia Journal*, 12(1), 17–21.
- Turnip, M., Dharma, A., Andrian, Afriansyah, A., Oktarino, A., & Turnip, A. (2021). *Integration of FIR and Butterworth Algorithm for Real-Time Extraction of Recorded ECG Signals BT - Cyber Physical, Computer and Automation System: A Study of New Technologies* (E. Joelianto, A. Turnip, & A. Widyotriatmo (Ed.); hal. 49–58). Springer Singapore. https://doi.org/10.1007/978-981-33-4062-6_5

PROFIL PENULIS



Ade Oktarino, S.Kom., M.S.I

Penulis merupakan alumni S1 Teknik Informatika Universitas Surakarta Tahun 2010, Alumni S2 Magister Sistem Informasi Pada Sekolah Tinggi Ilmu Komputer Dinamika Bangsa Jambi Tahun 2013 dan saat ini sedang menempuh Program Doctoral Information Technology di Universitas Putra Indonesia YPTK Padang pada Tahun 2022, sebelumnya pernah sebagai dosen tetap di Politeknik Jambi hingga Tahun 2014 dan sekarang aktif mengajar di Universitas Adiwangsa Jambi pada Program Studi Teknologi Informasi Fakultas Teknik dan Ilmu Komputer. Sebelumnya Dosen Tugas Tambahan sebagai Wakil Rektor II Universitas Adiwangsa periode 2017 – 2023 dan saat ini sebagai Dosen Tugas Tambahan pada posisi Dekan Fakultas Keguruan Ilmu Pendidikan Universitas Adiwangsa Jambi. Aktif dalam organisasi Relawan TIK Provinsi Jambi sebagai salah satu pengurus dan juga aktif sebagai konsultasi IT di instansi Polda Jambi, Dinas Lingkungan Hidup Kota Jambi dan Kantor Imigrasi Provinsi Jambi. Kosentrasi penelitian dengan topik artificial intelegency, machine learning, deep learning dengan bidang data science, computer vision dan natural language processing.

BAB 2

STRUKTUR DASAR ALGORITMA

Santi Prayudani
Politeknik Negeri Medan, Medan
E-mail: santiprayudani@polmed.ac.id

PENDAHULUAN

Algoritma yang terdiri dari serangkaian instruksi untuk menyelesaikan masalah, dapat dikategorikan ke dalam beberapa struktur dasar, yaitu Struktur Urut (*Sequence*), Struktur Pemilihan (*Selection*), dan Struktur Pengulangan (*Repetition*) (Resmawan, 2019). Algoritma dibangun dari berbagai struktur dasar yang berfungsi sebagai landasan dalam merancang penyelesaian. Struktur dasar algoritma adalah kerangka atau pola yang menjadi dasar dalam menyusun algoritma untuk menyelesaikan suatu masalah secara logis dan terstruktur (Esabella et al, 2021). Pemahaman yang baik terhadap ketiga struktur dasar ini sangat penting dalam memprogram dan menyusun algoritma yang efektif dan efisien.

STRUKTUR URUT (*SEQUENCE*)

Struktur urut adalah struktur dasar yang paling sederhana, dimana instruksi dalam algoritma dijalankan secara berurutan, satu per satu. Setiap instruksi dilaksanakan setelah instruksi sebelumnya selesai tanpa ada pengulangan atau percabangan. Misalnya dalam sebuah algoritma yang memiliki 20 langkah, setiap instruksi tersebut dilaksanakan secara berurutan, dimulai dari langkah 1 hingga langkah 20, tanpa ada instruksi yang terlewat (Esabella et al, 2021).

Sebagai contoh struktur urut dalam kehidupan sehari-hari adalah mengirim Pesan Whatsapp baru, algoritma dapat ditulis seperti ini:

Algoritma Mengirim Pesan Whatsapp Baru:

1. Mulai
2. Buka aplikasi Whatsapp pada *handphone*
3. Buka Halaman chat baru pilih nomor *handphone* Whatsapp
4. Input pesan yang akan dikirim
5. Klik tombol kirim
6. Chat Whatsapp akan terkirim
7. Selesai

Dalam sains, struktur urut sering diterapkan untuk menyelesaikan masalah yang melibatkan perhitungan dengan rumus sederhana. Perhitungan ini biasanya menggunakan operator dasar seperti penjumlahan, pengurangan, dan perkalian. Contoh penerapan logika struktur urut mencakup berbagai kasus, seperti menghitung jarak tempuh, luas persegi panjang, luas lingkaran, atau menentukan upah pegawai, semuanya berdasarkan rumus dasar yang berfungsi secara linear tanpa percabangan atau pengulangan (Resmawan, 2019). Sebagai contoh untuk menghitung luas segitiga, algoritma dapat ditulis seperti ini:

Algoritma Menghitung Luas Segitiga:

1. Mulai
2. Masukkan data alas
3. Masukkan data tinggi
4. Hitung luas = alas * tinggi * 0.5
5. Tampilkan luas
6. Selesai

ulang instruksi yang sama, menjadikannya sangat penting untuk mengatasi masalah yang melibatkan proses berulang.

DAFTAR PUSTAKA

- Esabella, S., & Haq, M. (2021). *Dasar-Dasar Pemrograman*. Sumbawa Besar: Olat Maras Publishing.
- Kani, M. (2020). *Algoritma dan Pemrograman*. Tangerang Selatan: Universitas Terbuka.
- Kaswidjanti, W. (2021). *Algoritma dan Pemrograman*. Universitas Pembangunan Nasional “Veteran” Yogyakarta Learning.
<http://learning.upnyk.ac.id/mod/resource/view.php?id=1409>
- Putri, P., M., Barovih, G., Azdy, R., A., dkk. (2022). *Algoritma dan Struktur Data*. Bandung: Penerbit Widina Bhakti Persada Bandung.
- Putro, S. S., Anamisa, D. R., & Mufarroha, F. A. (2022). *Algoritma Pemrograman*. Malang: Media Nusa Creative (MNC Publishing).
- Resmawan. (2019). *Resmawan [ADP] - Struktur Dasar Algoritma*. Universitas Negeri Gorontalo Repository.
<https://repository.ung.ac.id/data/cari/ADP>.
- Zein, A., & Eriana, E. S. (2022). *Algoritma dan Struktur Data*. Tangerang Selatan: Unpam Press.

PROFIL PENULIS



Santi Prayudani

Lahir di Kotamadya Binjai Sumatera Utara pada tanggal 28 Maret 1986. Penulis menempuh pendidikan S1 di Prodi Ilmu Komputer Universitas Sumatera Utara pada tahun 2004. Kemudian melanjutkan kembali pendidikan S2 di Prodi Teknik Informatika Universitas Sumatera Utara pada tahun 2011. Memulai karir sebagai guru di SDS Al Azhar Medan pada tahun 2010. Kemudian mengajar juga sebagai dosen di AMIK Harapan dan Universitas Pembangunan Panca Budi dari tahun 2011 sampai tahun 2014. Saat ini penulis diberi amanah oleh negara untuk mengabdikan sebagai dosen di Politeknik Negeri Medan dari tahun 2015.

BAB 3

NOTASI ALGORITMA MENGGUNAKAN KALIMAT DESKRIPTIF, FLOWCHART, DAN PSEUDOCODE

Kuswandi
Politeknik LP3I Jakarta, Jakarta
E-mail: kuswandi@lp3i.id

PENDAHULUAN

Menurut (Sismoro, 2005, 29) Algoritma merupakan sekumpulan instruksi atau langkah-langkah yang dituliskan secara sistematis dan digunakan untuk menyelesaikan masalah / persoalan logika dan matematika dengan bantuan komputer. Instruksi dalam algoritma merupakan perintah untuk melakukan aksi tertentu, seperti instruksi menulis hasil perhitungan, membaca masukan, memanggil sebuah prosedur, dan sebagainya. Instruksi algoritma dinyatakan sebagai pernyataan yang dapat disajikan dalam bentuk tulisan. Tulisan pernyataan instruksi algoritma haruslah menggunakan bahasa yang mudah dibaca dan dipahami oleh manusia. Penyajian penulisan algoritma ini dapat dilakukan menggunakan untaian kalimat deskriptif, pseudocode dan flowchart.

PENYAJIAN ALGORITMA

1. Notasi Kalimat Deskriptif

Pada dasarnya, algoritma merupakan susunan langkah-langkah sistematis yang bisa digunakan untuk menyelesaikan permasalahan tertentu. Penulisan notasi algoritma dapat disajikan dalam bentuk kalimat deskriptif dalam istilah lain disebut dengan notasi alami.

Penulisan langkah-langkah dalam notasi algoritma deskriptif dilakukan dalam bentuk kalimat deskriptif dengan menggunakan bahasa yang mudah dimengerti, Notasi ini biasanya digunakan pada algoritma yang pendek karena apabila untuk algoritma yang panjang notasi deskriptif kurang efektif. karena tidak memiliki aturan yang baku untuk menuliskan algoritma deskriptif maka instruksi pada algoritma deskriptif ini dapat menggunakan bahasa sehari – hari.

Penulisan algoritma dengan notasi kalimat deskriptif biasanya menggunakan kata kerja seperti mulai, baca, tampilkan, tulis, jika, maka dan ulangi. adapun dalam Bahasa Inggris seperti read, write, print, if, end, dan sebagainya. penulis bebas dalam menuliskan notasi yang penting tulisan tersebut jelas dan mudah dimengerti oleh pembaca. Pada saat menuliskan notasi kalimat deskriptif ini terbagi menjadi 3 bagian, yaitu:

1. **Judul** : terdiri dari nama algoritma dan penjelasan tentang algoritma yang akan dibuat.
2. **Deklarasi** : mendefinisikan semua nama yang digunakan pada algoritma, seperti tipe data, variable, konstanta dan fungsi.
3. **Deskripsi** : memuat uraian langkah-langkah dalam penyelesaian permasalahan.

Contoh notasi kalimat deskriptif adalah sebagai berikut:

Contoh 1:

Judul : Algoritma_KirimEmail

Deklarasi :

Deskripsi :

Mulai

Pengirim membuka browser

Buka Email pengirim

Klik tombol compose

KESIMPULAN

Instruksi algoritma merupakan perintah untuk melakukan aksi tertentu, seperti instruksi menulis hasil perhitungan, membaca masukan, memanggil sebuah prosedur, dan sebagainya. Instruksi algoritma dinyatakan sebagai pernyataan yang dapat disajikan dalam bentuk tulisan, Penyajian penulisan algoritma ini dapat dilakukan menggunakan 3 jenis notasi algoritma yaitu untaian kalimat deskriptif, pseudocode dan flowchart

DAFTAR PUSTAKA

- Kadir, Abdul. 2022. “Dasar Logika Pemrograman Komputer”, Elex Media Komputindo, Jakarta.
- Meidyan Permata Putri, Guntoro Barovich, Rezania Agramanisti Azdy, Yuniansyah, Andri Saputra, Yesi Sriyeni, Arsia Rini, Fadhila Tangguh Admojo. 2022. “Algoritma dan Struktur Data”. Penerbit Widina.
- Zarman, Wendi & Mochamad Fajar Wicaksono. 2020. “Implementasi Algoritma dalam Bahasa Python”. Informatika, Bandung.

BAB 4

PENGANTAR BAHASA PEMROGRAMAN PYTHON

Yogi Kristiyanto
Universitas IPWIJA, Jakarta
E-mail: yogi.kristiyanto@gmail.com

PENDAHULUAN

Python adalah bahasa pemrograman tingkat tinggi yang diciptakan oleh Guido van Rossum dan pertama kali dirilis pada tahun 1991. Python dirancang untuk mudah dibaca dan ditulis, dengan sintaks yang bersih dan minimalis. Berikut adalah beberapa fitur utama Python:

1. **Sederhana dan Mudah Dibaca:** Sintaks Python menyerupai bahasa Inggris, yang membuatnya mudah dipahami dan ditulis.
2. **Dinamika dan Interaktif:** Python adalah bahasa pemrograman yang dinamis, artinya variabel tidak perlu dideklarasikan dengan tipe tertentu. Python juga menyediakan mode interaktif untuk eksperimen langsung.
3. **Berorientasi Objek:** Python mendukung pemrograman berorientasi objek, memungkinkan penggunaan kembali kode dan modularitas.
4. **Perpustakaan Standar yang Kaya:** Python memiliki perpustakaan standar yang luas, yang mencakup berbagai modul dan paket untuk berbagai keperluan, mulai dari pengolahan data hingga pengembangan web.
5. **Kompatibilitas Lintas Platform:** Program Python dapat dijalankan pada berbagai sistem operasi seperti Windows, MacOS, dan Linux tanpa perubahan kode yang signifikan.

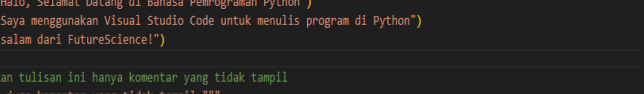


Sumber: (Van Rossum, 2024)

Gambar 4.1. Foto Guido Van Rossum
Penemu Bahasa Pemrograman Python

Nama “Python” dipilih oleh Guido van Rossum bukan dari nama ular, tetapi dari grup komedi Inggris “Monty Python's Flying Circus”. Setelah rilis pertamanya, Python terus berkembang dengan masukan dari komunitas pengguna dan kontribusi dari pengembang lainnya. Beberapa tonggak penting dalam perkembangan Python meliputi:

1. **Python 1.0 (1994)**: Versi ini memperkenalkan beberapa fitur penting seperti lambda, map, filter, dan reduce. Ini adalah langkah besar pertama menuju bahasa yang lebih fungsional.
2. **Python 2.0 (2000)**: Python 2.0 menambahkan fitur seperti garbage collection berbasis siklus dan dukungan untuk Unicode. Versi ini juga memperkenalkan list comprehensions, yang memungkinkan cara yang lebih deklaratif untuk membuat dan memanipulasi list.
3. **Python 3.0 (2008)**: Python 3.0, juga dikenal sebagai "Python 3000" atau "Py3k", adalah pembaruan besar yang tidak kompatibel dengan versi sebelumnya. Tujuan



The screenshot shows the Visual Studio Code interface. The editor window displays a file named `hello.py` with the following Python code:

```
1 print("Halo, Selamat Datang di Bahasa Pemrograman Python")
2 print("Saya menggunakan Visual Studio Code untuk menulis program di Python")
3 print("salam dari FutureScience!")
4
5 # Abaikan tulisan ini hanya komentar yang tidak tampil
6 """ Ini juga komentar yang tidak tampil """
```

Below the editor, the **TERMINAL** tab is active, showing the command prompt output:

```
PS C:\Users\ACER\AppData\Local\Programs\Python\Python312\python.exe "c:\Users\ACER\Downloads\Buku Algoritma dan Pemrograman\Latihan Python\hello.py"
Halo, Selamat Datang di Bahasa Pemrograman Python
Saya menggunakan Visual Studio Code untuk menulis program di Python
salam dari FutureScience!
PS C:\Users\ACER\Downloads\Buku Algoritma dan Pemrograman\Latihan Python>
```

Sumber: Pustaka Penulis (2024)

Gambar 4.12. Hasil menjalankan file `hello.py` pada terminal

KESIMPULAN

Python adalah bahasa pemrograman yang kuat dan serbaguna, cocok untuk pemula dan pengembang berpengalaman. Dengan memahami dasar-dasar Python dan melakukan persiapan awal yang tepat, Anda dapat memulai perjalanan Anda dalam pemrograman dengan langkah yang kuat.

DAFTAR PUSTAKA

- Foundation, P. S. (2024). *Python 3.12.4 documentation*. Retrieved from 3.12.4 Documentation: <https://docs.python.org/3/>
- Microsoft. (2023). *Getting Started with Python in VS Code*. Retrieved from Getting Started with Python in VS Code: <https://code.visualstudio.com/docs/python/python-tutorial>
- Van Rossum, G. (2024). *Guido van Rossum - Personal Home Page*. Retrieved from Guido van Rossum - Personal Home Page: <https://gvanrossum.github.io/>

PROFIL PENULIS



Yogi Kristiyanto

Penulis adalah seorang dosen dan praktisi dalam bidang teknologi data dan keamanan informasi, di mana ia berafiliasi dan berkontribusi dalam pengembangan kurikulum dan pengajaran di bidang ilmu komputer pada Program Studi S1 Sistem Informasi dan Informatika di Universitas IPWIJA (2022). Sebagai alumni

Universitas Gunadarma dengan gelar Sarjana Komputer (2009) dan Magister Manajemen Sistem Informasi (2019), Yogi Krisityanto aktif terlibat dalam komunitas teknologi informasi, baik di dalam maupun di luar kampus. Ia sering menjadi pembicara dalam seminar dan workshop, berbagi pengetahuan dan pengalaman dengan mahasiswa, profesional, dan masyarakat umum. Selain itu, ia juga membimbing mahasiswa dalam proyek-proyek penelitian dan pengembangan perangkat lunak, membantu mereka mengaplikasikan pengetahuan teoretis ke dalam praktik nyata.

BAB 5

TIPE DATA, OPERATOR, DAN EKSPRESI PADA PYTHON

Sugiyono

Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika, Jakarta

E-mail: inosoguy007@gmail.com

TIPE DATA

Tipe data adalah konsep fundamental dalam pemrograman, termasuk di Python, yang menentukan jenis nilai yang dapat disimpan dan dioperasikan oleh suatu variabel. Memahami tipe data adalah langkah pertama yang sangat penting dalam mempelajari Python karena setiap tipe data memiliki karakteristik, operasi, dan perilaku tertentu yang berperan besar dalam bagaimana data diproses dalam program. Python mendukung berbagai tipe data yang dapat digunakan untuk menyimpan dan memanipulasi berbagai jenis informasi, termasuk angka, teks, dan struktur data yang lebih kompleks. Berikut adalah penjelasan rinci mengenai tipe data di Python:

1. Tipe Data Numerik

Tipe data numerik adalah tipe data yang digunakan untuk menyimpan angka. Dalam Python, tipe data numerik terbagi menjadi tiga jenis utama: Integer, Float, dan Complex.

a) Inteteger

Integer adalah tipe data untuk bilangan bulat, baik positif maupun negatif, tanpa bagian desimal. Ini adalah tipe data numerik paling sederhana di Python. Bilangan bulat dapat sebesar dan sekecil yang kita inginkan, karena Python secara otomatis memperbesar atau memperkecil kapasitas penyimpanan yang dibutuhkan sesuai dengan ukuran bilangan. Contoh:

```
x = 10  # Bilangan bulat positif  
y = -3  # Bilangan bulat negatif  
z = 0   # Nol juga merupakan integer
```

Dalam operasi matematika, bilangan bulat sering digunakan untuk melakukan operasi dasar seperti penjumlahan, pengurangan, perkalian, dan pembagian. Karena integer tidak memiliki bagian desimal, hasil dari pembagian antara dua integer mungkin berupa float, kecuali jika menggunakan pembagian bulat (floor division).

b) Float

Float adalah tipe data untuk bilangan desimal, atau lebih tepatnya bilangan yang memiliki bagian pecahan. Bilangan float sering digunakan untuk perhitungan yang membutuhkan presisi lebih dari sekadar bilangan bulat, seperti dalam aplikasi ilmiah, keuangan, atau teknik. Contoh:

```
pi = 3.14159  # Bilangan desimal positif  
neg_float = -2.718  # Bilangan desimal negatif
```

lam matematika, float digunakan untuk operasi yang membutuhkan ketelitian tinggi, seperti perhitungan geometri atau penghitungan rata-rata. Penting untuk diingat bahwa representasi float memiliki keterbatasan presisi karena mereka diimplementasikan menggunakan format floating-point biner, yang dapat menyebabkan hasil perhitungan tertentu menjadi kurang akurat jika melibatkan angka yang sangat besar atau sangat kecil.

kebutuhan.

2. Operator:

Python menyediakan berbagai jenis operator untuk memanipulasi data, termasuk operator aritmatika (seperti +, -, *, /), operator perbandingan (seperti ==, >, <), operator logika (and, or, not), operator bitwise, serta operator keanggotaan dan identitas. Operator ini digunakan untuk mengolah variabel dan nilai untuk membentuk ekspresi dan logika program.

3. Ekspresi:

Ekspresi adalah kombinasi dari nilai, variabel, dan operator yang dievaluasi menjadi satu nilai. Ekspresi dapat berupa operasi aritmatika, logika, atau pemanggilan fungsi, yang digunakan untuk memproses data dan mengontrol alur program. Ekspresi juga dapat bersifat sederhana maupun kompleks, serta mendukung konsep bersarang, penggabungan string, dan pemrosesan koleksi data seperti list dan tuple.

Dengan memahami konsep dasar tipe data, operator, dan ekspresi, seorang programmer akan mampu menulis program Python yang efisien, logis, dan terstruktur. Ketiga elemen ini merupakan fondasi penting yang menjadi dasar pengembangan keterampilan pemrograman yang lebih lanjut.

DAFTAR PUSTAKA

- Foundation, P. S. (2024). *Python 3.12.4 documentation*. Retrieved from 3.12.4 Documentation: <https://docs.python.org/3/>
- Van Rossum, G. (2024). *Guido van Rossum - Personal Home Page*. Retrieved from Guido van Rossum - Personal Home Page: <https://gvanrossum.github.io/>

PROFIL PENULIS



Sugiyono

Penulis Buku Pemula, membuat dan menghasilkan penulisan buku yang berkontribusi dalam bidang Teknologi dan Infomasi (TI), dan diantara buku tersebut yang telah kami tulis adalah: Panduan Teknik Komputer Untuk Pemula, Teknik Merakit Komputer PC, dan Technical English. Dengan tujuan untuk berbagi

ilmu pengetahuan (*Knowledge Sharin*) melalui buku kepada publik dan khususnya para mahasiswa yang berhubungan dengan Ilmu Pengetahuan Teknologi dan Inforamsi (TI),

BAB 6

PERINTAH PERCABANGAN PADA PYTHON

Hidayatus Sibyan
Universitas Sains Al-Qur'an, Wonosobo
E-mail: hsibyan@unsiq.ac.id

PENDAHULUAN

Percabangan merupakan konsep dasar dari pemrograman yang memungkinkan kita dapat membuat program untuk mengambil keputusan berdasarkan kondisi tertentu. Kondisi ini akan dievaluasi dengan nilai benar (*true*) atau salah (*false*). Jika kondisi bernilai benar, maka blok kode yang sesuai akan dijalankan. Namun jika salah, program akan menjalankan blok kode yang lain (Fahrudin & S ST, 2023). Dalam mengevaluasi kondisi pada percabangan biasanya menggunakan ekspresi logika yang melibatkan operator perbandingan berikut (ST Wadi, 2018)

Tabel 6.1. Operator Perbandingan

Operator	Contoh	Penjelasan
== (sama dengan)	$x == y$	Jika nilai x sama dengan nilai y , maka kondisi bernilai benar
!= (tidak sama dengan)	$x != y$	Jika nilai x tidak sama dengan nilai y , maka kondisi bernilai benar
< (lebih kecil dari)	$x < y$	Jika nilai x kurang dari nilai y , maka kondisi bernilai benar
<= (lebih kecil atau sama dengan)	$x <= y$	Jika nilai x kurang dari atau sama dengan nilai y , maka kondisi bernilai benar
> (lebih besar dari)	$x > y$	Jika nilai x lebih dari nilai y , maka kondisi bernilai benar

>= (lebih besar atau sama dengan)	x >= y	Jika nilai x lebih dari atau sama dengan nilai y, maka kondisi bernilai benar
-----------------------------------	--------	---

Selain itu, dalam percabangan ada kalanya juga menggunakan operator logika untuk menggabungkan beberapa kondisi. Berikut adalah beberapa operator logika yang bisa digunakan (Irmawati et al., 2023).

Tabel 6.2. Operator Logika

Operator	Contoh	Penjelasan
AND	True AND True (hasil True) True AND False (hasil False) False AND True (hasil False) False AND False (hasil False)	Akan bernilai <i>True</i> jika kedua kondisi bernilai <i>True</i> . Selain itu akan bernilai <i>False</i>
OR	True OR True (hasil True) True OR False (hasil True) False OR True (hasil True) False OR False (hasil False)	Akan bernilai <i>False</i> jika kedua kondisi bernilai <i>False</i> . Selain iut akan bernilai <i>True</i>
NOT	NOT True (hasil False)	Akan bernilai kebalikannya

Dalam bahasa pemrograman python, percabangan biasanya diimplementasikan menggunakan pernyataan if, if-else, if-elif-else, dan if bersarang (*nested if*).

- e. Operator logika (*and*, *or*, *not*) sangat berguna untuk menggabungkan beberapa kondisi sekaligus, memungkinkan kita untuk membuat keputusan yang lebih kompleks dalam satu pernyataan *if*.

Pemahaman yang kuat tentang percabangan adalah langkah penting dalam mengembangkan logika program yang efektif dan efisien. Dengan menguasai berbagai bentuk percabangan, kita dapat mengembangkan solusi yang lebih adaptif dan fleksibel dalam menyelesaikan masalah yang lebih rumit.

DAFTAR PUSTAKA

- Ardiansah, I., & Permana, R. H. (2023). *Memulai Python: Belajar Python dari Nol*. CV. Cendekia Press.
- Fahrudin, T. M., & S ST, M. T. (2023). *Algoritma dan Pemrograman Dasar dalam Bahasa Pemrograman Python*. Thalibul Ilmi Publishing & Education.
- Hartatik, H., Gaffara, G. R., Nasution, H. A., Ardiansyah, A., Arsana, I. N. A., Jannah, U. M., & Iwan Adhichandra, S. T. (2023). *PENGENALAN PEMROGRAMAN DASAR DUNIA KODING*. PT. Sonpedia Publishing Indonesia.
- Irmawati, I., El Rayeb, A., Carudin, C., Meilani, B. D., Halim, A., Indriyani, T., & Singadji, M. (2023). *Buku Ajar Pemrograman*. PT. Sonpedia Publishing Indonesia.
- Nurhadi, N., Jatmiko, A. R., Legito, L., Saputra, E. A., Surianto, D. F., Komalasari, R., Mukhlis, I. R., Sulistyowati, S., Lorosae, T. A., & Zain, N. N. L. E. (2023). *BUKU AJAR LOGIKA & ALGORITMA*. PT. Sonpedia Publishing Indonesia.
- Putro, S. S., Anamisa, D. R., & Mufarroha, F. A. (2022). *Algoritma Pemrograman*. Media Nusa Creative (MNC Publishing).

- Setiawan, G. A., & Vania, E. (2022). *Praktek Pemrograman C++ dan Python*. SCU Knowledge Media.
- ST Wadi, H. (2018). *Pemrograman Python: Untuk Pelajar & Mahasiswa* (Vol. 1). TR Publisher.
- Swastika, W. (2019). *Pengantar Algoritma dan Penerapannya pada Python*. Ma Chung Press.

PROFIL PENULIS



Hidayatus Sibyan

Lahir di Wonosobo pada tanggal 30 November 1989, adalah seorang akademisi di bidang Teknik Informatika. Menyelesaikan studi S1 Teknik Informatika di Universitas Sains Al-Qur'an (UNSIQ) Wonosobo pada tahun 2012. Mendapatkan gelar Magister Komputer (M.Kom.) pada program studi Magister Teknik Informatika STMIK Amikom Yogyakarta pada tahun 2016. Telah mengabdikan dirinya di Universitas Sains Al-Qur'an (UNSIQ) Wonosobo sebagai tenaga pendidik sejak tahun 2016 sampai saat ini. Mata kuliah yang diampu diantaranya Algoritma dan Dasar Pemrograman, Algoritma dan Struktur Data, Basis Data, Sains Data.

BAB 7

PERINTAH PERULANGAN PADA PYTHON

Muhammad Rizqy Septyandy
Universitas Mulawarman, Samarinda
E-mail: rizqyseptyandy@ft.unmul.ac.id

PENDAHULUAN

Bab ini menyimpulkan pembahasan terkait pernyataan prosedural Python dengan menyajikan dua pernyataan konstruksi perulangan utama bahasa tersebut yang mengulang suatu tindakan berulang-ulang. Yang pertama, pernyataan `while`, menyediakan cara untuk membuat kode perulangan umum. Yang kedua, pernyataan `for`, dirancang untuk menelusuri item-item dalam urutan atau objek yang dapat diulang lainnya dan menjalankan blok kode untuk masing-masing item.

Sekarang, sebagian besar pemula pasti bertanya-tanya apa itu pernyataan kontrol. Pernyataan kontrol tidak lain hanyalah serangkaian pernyataan yang diikuti oleh suatu program untuk mendapatkan hasil yang diinginkan. Pertama-tama mari dicoba pahami apa itu pernyataan kontrol dan mengapa pernyataan itu merupakan bagian penting dari bahasa pemrograman apa pun. Masyarakat mungkin pernah menggunakan mesin ATM untuk menarik uang. Apa yang mereka lakukan saat memasukkan kartu debit? Hal pertama yang mereka lihat di layar adalah opsi untuk dipilih dari menu. Sekarang mereka harus melakukan tindakan tertentu, jika tidak, kartu mereka akan ditolak setelah beberapa saat, jika mereka membiarkan komputer menunggu lama. Saat mereka memilih untuk menarik uang, mereka harus memberikan jumlah yang benar, yang harus ditarik dari mesin. Jika mereka memberikan jumlah yang salah, komputer akan segera memberikan pesan "Silakan masukkan jumlah yang

benar". Apa yang terjadi di sini? Apakah mereka memperhatikan? Di sini komputer hanya mengikuti instruksi. Komputer telah diperintahkan untuk hanya menerima jumlah yang benar dan memunculkan pesan kesalahan saat memasukkan jumlah yang salah. Ini adalah salah satu skenario yang mana mereka dapat melihat penggunaan pernyataan kontrol.

Mungkin ada situasi ketika suatu kondisi mungkin perlu dijalankan satu blok kode beberapa kali, dalam situasi tersebut, loop akan berguna. Loop berguna dalam situasi seperti mengulang struktur data dalam bahasa pemrograman apa pun atau menelusuri set data yang besar untuk menyaring data sampah, diikuti oleh kata kunci tertentu kemudian diikuti juga oleh karakter alfanumerik serta diikuti juga oleh karakter khusus tertentu.

1. Tipe Perulangan (*Loop*)

Perulangan for adalah iterator generik dalam Python: ia dapat melangkah melalui item dalam urutan yang diurutkan atau objek yang dapat diulang lainnya. Pernyataan for berfungsi pada string, daftar, tupel, dan iterabel bawaan lainnya, serta objek baru yang ditentukan pengguna yang akan dipelajari cara membuatnya nanti dengan kelas.

Ada dua jenis perulangan, yaitu:

- 1. Terbatas:** Dalam kasus ini, kode blok dijalankan dalam jumlah waktu yang ditentukan (Mohit & Das, 2017). Ini berguna ketika programmer tahu persis berapa kali tugas akan dieksekusi atau anggap saja dia tahu jumlah elemen dalam struktur data. Misalnya, kekuatan kelas.
- 2. Tidak terbatas:** Dalam kasus ini, kode blok berjalan hingga kondisinya benar (Mohit & Das, 2017). Ini berguna jika jumlahnya tidak diketahui. Misalnya, mencoba mencari

KESIMPULAN

Bab ini dimulai dengan memahami konsep pernyataan kontrol dan mengapa mereka penting dalam masalah pemrograman sehari-hari. Kemudian berkesempatan untuk melihat ke dalam dua fungsi `range()` dan `xrange()` yang digunakan untuk menghasilkan daftar dengan cepat. Selanjutnya belajar tentang loop terbatas dan loop tidak terbatas. Lalu juga belajar tentang perbedaan antara loop terbatas dan tidak terbatas. Di bawah loop terbatas, hal yang dipelajari tentang berbagai pernyataan kontrol seperti `if`, `if...else`, `if...elif...else`, dan `for` loop. Sementara, dalam loop tidak terbatas dipelajari pernyataan `while` dan bagaimana dipelajari `while` loop untuk bekerja sebagai loop terbatas. Penggunaan `while` dengan pernyataan `break` dan tanpa pernyataan `break` serta kerja pernyataan `continue` dan `pass`.

DAFTAR PUSTAKA

- Lutz, M. (2013). *Learning Python, 5th Edition*. Sebastopol: O'Reilly Media, Inc.
- Mohit & Das, B. N. (2017). *Learn Python in 7 Days*. Birmingham: In Packt Publishing.

PROFIL PENULIS



Muhammad Rizqy Septyandy, S.Si., M.T., C.Ed., C.ESP.

Lahir di Jakarta pada tanggal 26 September 1990. Lulus S1 di Program Studi Fisika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Indonesia (FMIPA UI) tahun 2012, lulus S2 di Program Studi Teknik Geologi Fakultas Ilmu dan Teknologi Kebumihan (FITB) Institut Teknologi Bandung (FITB ITB) tahun 2016.

Mendapatkan sertifikat keahlian sebagai editor dari LPPSDM ECHA PROGRES Makassar tahun 2023 serta sertifikat keahlian sebagai pakar di bidang publikasi ilmiah dari ESAS Edukasi Indonesia di

tahun yang sama. Saat ini merupakan dosen tetap Program Studi Teknik Geologi Fakultas Teknik Universitas Mulawarman. Mengampu mata kuliah Geologi Kebencanaan, Geologi Lingkungan, Mitigasi Bencana Kebumihan, serta Sistem Informasi dan Pemodelan Bencana Geologi di beberapa program studi seperti teknik geologi, teknik lingkungan, geologi, dan geofisika di Universitas Mulawarman dan Universitas Indonesia. Pernah jadi dosen tetap di Program Studi Teknik Geologi, Institut Teknologi Sumatera tahun 2016-2017 serta dosen tidak tetap di Program Studi Geologi, Universitas Indonesia dari tahun 2017-2022. Aktif menulis artikel baik artikel ilmiah di jurnal internasional dan jurnal nasional serta artikel populer di media elektronik kumparan.com khususnya yang berkaitan dengan kecerdasan artifisial, penginderaan jauh, dan mitigasi bencana geologi.

BAB 8

MANIPULASI STRING DAN OPERASINYA

Pahrul Irfan
Universitas Mataram, Nusa Tenggara Barat
E-mail: irfan@staff.unram.ac.id

DASAR-DASAR STRING DI PYTHON

String merupakan tipe data fundamental dalam Python yang digunakan untuk merepresentasikan teks. Di Python, string dapat didefinisikan menggunakan tanda kutip tunggal (`'`), ganda (`"`), atau kutip tiga (`"` atau `'''`) untuk string multi-baris. Pemahaman dasar tentang string sangat penting bagi programmer Python karena banyak operasi dan manipulasi data yang melibatkan teks.

Contoh Pembuatan String:

```
1  satu = 'Hello, World!'
2  dua = "Python Programming"
3  tiga = """Ini adalah string
4  multi-baris dalam Python."""
```

Indeks dan Slicing: Setiap karakter dalam string memiliki indeks yang dimulai dari 0 untuk karakter pertama. Python mendukung operasi slicing yang memungkinkan pengambilan sebagian string berdasarkan rentang indeks.

```
1  contoh = "Python"
2  print(contoh[0]) # Output: P
3  print(contoh[2:5]) # Output: thr
4  print(contoh[-1]) # Output: n
```


- **Indeks Positif:**

Dimulai dari 0 hingga n-1 (n adalah panjang string).

- **Indeks Negatif:**

Dimulai dari -1 hingga -n (dimana -1 adalah karakter terakhir).

Immutabilitas String: String di Python bersifat immutable, artinya karakter dalam string tidak dapat diubah setelah string dibuat. Jika ingin memodifikasi string, perlu membuat string baru.

```
1 kata = "Python"
2 # kata[0] = 'p' # Akan menghasilkan TypeError
3 kata_baru = 'p' + kata[1:]
4 print(kata_baru) # Output: python
```

Immutabilitas ini memungkinkan optimasi memori dan keamanan data, namun juga memerlukan pendekatan yang berbeda saat melakukan modifikasi string.

Operasi Dasar pada String:

- **Konkatenasi:** Menggabungkan dua atau lebih string menggunakan operator +.

```
1 a = "Hello"
2 b = "World"
3 print(a + " " + b) # Output: Hello World
```

- **Pengulangan:** Mengulang string menggunakan operator *.

```
1 print("Python " * 3) # Output: Python Python Python
```

- **Pemeriksaan Keberadaan Substring:** Menggunakan keyword in atau not in.

```
1 teks = "Belajar Python"
```

Kesimpulan Utama:

Manipulasi string adalah keterampilan esensial yang harus dikuasai oleh setiap programmer Python. Dari dasar-dasar pembuatan dan pengolahan string hingga penggunaan metode bawaan, format string yang canggih, regular expressions yang kompleks, serta pengelolaan encoding dan decoding, setiap aspek ini berkontribusi pada kemampuan kita untuk mengelola dan memanipulasi data teks dengan efisien. Dengan menguasai konsep-konsep ini, kita dapat mengembangkan aplikasi yang lebih kuat, fleksibel, dan dapat diandalkan dalam berbagai konteks pengembangan perangkat lunak.

DAFTAR PUSTAKA

- Budhi Gustiandi. (2023). *Langkah Awal Menguasai Bahasa Pemrograman Python*. In *Langkah Awal Menguasai Bahasa Pemrograman Python*. <https://doi.org/10.55981/brin.656>
- Enterprise, J. (2019). *Python untuk Programmer Pemula*. Elex Media Komputindo.
- Kurniawan, D. (2021). *Pengenalan machine learning dengan python*. Gramedia.
- R. Sianipar. (2015). *Pemrograman Python: teori dan implementasi*. Bandung: Informatika.
- Raharjo, B. (2019). *Mudah Belajar Python Untuk Aplikasi Dekstop Dan Web Edisi Revisi*. Penerbit Informatika.
- Rahman, S., Sembiring, A., Siregar, D., Khair, H., Gusti Prahmana, I., Puspadini, R., & Zen, M. (2023). *Python : Dasar Dan Pemrograman Berorientasi Objek*. Penerbit Tahta Media.
- Wahyono, T. (2018). *Fundamental of Python for Machine Learning: Dasar-Dasar Pemrograman Python untuk Machine Learning dan Kecerdasan Buatan*. Gava Media.

PROFIL PENULIS



Pahrul Irfan

Penulis adalah seorang profesional berpengalaman dengan lebih dari 7 tahun di bidang Pengembangan Web. Saat ini penulis bekerja sebagai Dosen pengajar berbagai matakuliah Pemrograman di termasuk Web Programming, Big Data, Machine Learning dan Kecerdasan Buatan (AI) di Universitas Mataram. Penulis juga fokus pada penerapan

Python dan PHP dalam pengembangan aplikasi web modern serta Machine Learning. Pengalaman dalam pengembangan aplikasi web, analisis data besar, dan kecerdasan buatan menjadi landasan kuat untuk berkontribusi menulis buku Python yang bertujuan membantu pembaca dari berbagai tingkatan. Buku ini dirancang untuk memberikan panduan praktis dan aplikatif, memudahkan pemula hingga profesional dalam menggunakan bahasa pemrograman Python untuk berbagai kebutuhan, mulai dari pemrograman dasar hingga pengembangan aplikasi kompleks. Dengan komitmen terhadap pengajaran dan kontribusi di dunia pemrograman, penulis terus berupaya menginspirasi generasi baru pengembang untuk menguasai bahasa pemrograman Python serta berinovasi di dunia teknologi.

BAB 9

ARRAY

Erna Hudianti Pujiarini
Universitas Teknologi Digital Indonesia, Yogyakarta
E-mail: ernahudi@utdi.ac.id

PENDAHULUAN

Array dalam Python merupakan struktur data yang digunakan untuk menyimpan sekumpulan elemen yang memiliki tipe data yang sama. Meskipun Python tidak memiliki tipe data array bawaan, konsep array dapat dicapai melalui penggunaan list, modul array, atau library NumPy, library Pandas (Lutz, 2013).

Array biasanya dirancang untuk menyimpan elemen dengan tipe data yang sama, seperti integer atau float, yang memungkinkan pengolahan data yang lebih efisien. Setiap elemen dalam array dapat diakses menggunakan indeks numerik, yang memudahkan pengguna dalam mengambil dan memanipulasi data. Array mendukung operasi matematis dasar, dan dalam konteks modul array atau pustaka NumPy, pengguna dapat melakukan operasi vektor dan matriks yang kompleks.

Array memiliki keunggulan, terutama dalam hal efisiensi dan kecepatan, tetapi juga memiliki kelemahan terkait fleksibilitas dan kemudahan penggunaan. Pilihan antara menggunakan array atau struktur data lain harus didasarkan pada kebutuhan (Chen, 2020). Berikut ini keunggulan dari struktur data array :

1. Array menggunakan memori secara lebih efisien dibandingkan dengan struktur data lainnya (seperti list), terutama saat menyimpan data numerik dengan tipe yang sama.

2. Elemen dalam array dapat diakses dengan cepat menggunakan indeks, yang memungkinkan operasi yang lebih cepat dibandingkan dengan struktur data lain.
3. Array mendukung operasi vektor dan matriks, yang sangat berguna dalam komputasi ilmiah dan analisis data. Library seperti NumPy memungkinkan pengguna untuk melakukan operasi ini secara langsung pada seluruh array.
4. Keberadaan tipe data yang sama dalam array memudahkan optimasi kinerja dan pengolahan data, serta mengurangi kemungkinan kesalahan tipe (type errors).
5. Array dapat dengan mudah dibuat dalam bentuk multidimensi, memungkinkan representasi yang lebih kompleks seperti matriks, yang penting dalam machine learning dan statistik.

Kelemahan struktur data array :

1. Dalam banyak implementasi, ukuran array harus ditentukan saat dibuat dan tidak dapat diubah. Ini membatasi fleksibilitas dalam penanganan data yang dinamis.
2. Array hanya dapat menyimpan elemen dengan tipe data yang sama, yang dapat menjadi kendala jika perlu menyimpan berbagai tipe data dalam satu struktur.
3. Array (terutama yang diimplementasikan melalui modul array) memiliki metode yang lebih terbatas dibandingkan dengan list, yang memiliki banyak fungsi untuk manipulasi data.
4. Mengubah ukuran atau menambah elemen ke dalam array (terutama dalam array statis) memerlukan pembuatan array baru dan penyalinan data, yang dapat memakan waktu dan sumber daya.

tipe data yang sama. Meskipun Python tidak memiliki tipe data array bawaan, konsep array dapat dicapai melalui penggunaan list, modul array, atau library NumPy, library Pandas.

Array biasanya dirancang untuk menyimpan elemen dengan tipe data yang sama, seperti integer atau float, yang memungkinkan pengolahan data yang lebih efisien. Setiap elemen dalam array dapat diakses menggunakan indeks numerik, yang memudahkan pengguna dalam mengambil dan memanipulasi data. Array mendukung operasi matematis dasar, dan dalam konteks modul array atau pustaka NumPy, pengguna dapat melakukan operasi vektor dan matriks yang kompleks.

Deklarasi array merupakan proses mendefinisikan nama array dan tipe data yang akan disimpan di dalamnya. Dalam Python, mendeklarasikan tipe data array tidak secara eksplisit seperti di beberapa bahasa pemrograman lain, tetapi array dapat dideklarasikan dengan menggunakan list, modul array, Library NumPy dan library Pandas.

DAFTAR PUSTAKA

- Chen, D. Y. (2020). *Hands-On Data Analysis with NumPy and Pandas*. Birmingham: Packt Publishing.
- Lutz, M. (2013). *Learning Python*. Sebastopol, California: O'Reilly Media.
- Python, S. F. (2024, 9 30). *array Efficient arrays of numeric values*. Retrieved from docs.python.org: <https://docs.python.org/3/library/array.html>
- Python, S. F. (2024, 9 30). *Data Structures*. Retrieved from docs.python.org: <https://docs.python.org/3/tutorial/datastructures.html>

PROFIL PENULIS



Erna Hudianti Pujiarini, S.Si, M.Si.

Lahir di Magetan, 28 September 1971. Lulus Sarjana di Program Studi Statistik Universitas Gadjah Mada tahun 1990, lulus program studi Matematika Program Pasca Sarjana Universitas Gadjah Mada tahun 2000. Mulai mengajar tahun 1996 di STMIK AKAKOM yang sekarang berubah menjadi Universitas Teknologi Digital Indonesia. Saat ini menjadi dosen pada program studi Informatika, fakultas Teknologi Informasi, Universitas Teknologi Digital Indonesia. Matakuliah yang diampu Statistika, Statistika Terapan, Matematika, Data Mining, Jaringan Syarat Tiruan. Bidang konsentrasi penelitian Statistika Terapan dan Data Science. Buku yang ditulis “SPSS: Analisis Data Statistik”, “Data Mining: Pola dibalik Angka”.

BAB 10

PEMROGRAMAN MODULAR DENGAN FUNGSI

Nur Oktavin Idris
Universitas Negeri Gorontalo, Kota Gorontalo
E-mail: nur.oktavin@ung.ac.id

PENGANTAR PEMROGRAMAN MODULAR

Pemrograman modular merupakan pendekatan yang terdapat dalam pengembangan perangkat lunak dengan membagi program tersebut menjadi bagian-bagian kecil yang disebut dengan modul. Setiap modul adalah unit kode yang terpisah dan independent yang dapat dikembangkan, diuji dan dipelihara secara terpisah. Pendekatan ini memungkinkan pengembang untuk fokus pada satu spesifik dari program tanpa harus memahami keseluruhan kode(Mulyawan, 2024).

Adapun manfaat dari pemrograman modular yaitu:

1. Pemeliharaan yang lebih mudah : dengan memisahkan program menjadi modul-modul yang lebih kecil, kesalahan dapat diidentifikasi dan diperbaiki dengan lebih mudah. Setiap modul dapat diuji secara independent, sehingga meminimalkan resiko kesalahan yang tersembunyi
2. Penggunaan kembali kode : modul yang telah dikembangkan dapat digunakan kembali pada proyek lainnya, sehingga dapat menghemat waktu dan usaha karena pengembang tidak perlu menulis ulang kode yang sama untuk aplikasi yang berbeda.
3. Kolaborasi yang lebih baik : programmer dapat bekerja pada modul yang berbeda secara bersamaan tanpa saling mengganggu, sehingga meningkatkan efisiensi tim dan memungkinkan proyek besar diselesaikan lebih cepat

4. Pengembangan yang lebih cepat : modul yang terpisah memungkinkan pengembangan parallel, mempercepat proses pengembangan perangkat lunak. Setiap modul dapat dikembangkan dan diuji secara independent sebelum digabungkan ke dalam sistem utama
5. Fleksibilitas dan skalabilitas : sistem modular lebih mudah untuk diperluas dan dimodifikasi. Modul baru dapat ditambahkan tanpa mempengaruhi modul yang sudah ada, dan perubahan pada satu modul tidak akan berdampak pada modul lainnya.

Pemrograman modular memiliki prinsip-prinsip dasar yaitu:

1. Enkapsulasi : merupakan prinsip yang memastikan bahwa setiap modul memiliki batasan yang jelas dan hanya mengungkapkan antarmuka yang diperlukan untuk berinteraksi dengan modul lain, sehingga dapat membantu menjaga integritas data dan mencegah akses yang tidak sah ke bagian dalam modul.
2. Independensi : modul harus dirancang sedemikian rupa sehingga perubahan dalam satu modul tidak mempengaruhi modul lain. Independensi ini memungkinkan pengembangan, pengujian, dan pemeliharaan modul secara terpisah, sehingga dapat meningkatkan efisiensi dan fleksibilitas pengembangan perangkat lunak.
3. Kohesi : mengacu pada sejauh mana fungsi dalam satu modul saling berkaitan. Modul dengan kohesi tinggi memiliki tanggung jawab utama yang membuat lebih mudah dipahami dan dipelihara. Kohesi yang baik juga meningkatkan kejelasan dan fokus dari setiap modul.
4. Kopling rendah : modul harus memiliki ketergantungan minimal pada modul lain. Ini meminimalkan dampak perubahan pada satu modul terhadap modul lainnya,

menjadikan proses pengembangan perangkat lunak lebih terencana dan sistematis.

Untuk mencapai pemrograman modular yang efektif, penting untuk memulai dengan perencanaan yang matang. Rancanglah arsitektur sistem dengan jelas, mendefinisikan tanggung jawab setiap modul secara spesifik. Gunakan penamaan yang deskriptif untuk modul, fungsi, dan variabel agar mudah dipahami oleh seluruh anggota tim. Minimalkan ketergantungan antar modul untuk memudahkan pemeliharaan di masa mendatang. Selalu lakukan pengujian unit terhadap setiap modul sebelum integrasi untuk memastikan bahwa setiap bagian berfungsi dengan baik.

DAFTAR PUSTAKA

- Ali, E. (2019). *Rekayasa Perangkat Lunak*. CV MFA.
- Anam, S., Widihiatmoko, F., Fitriah, Z., Yanti, I., & Guci, A. N. (2023). *Pengantar Algoritma dan Pemrograman dengan Python*. UB Press.
- Andre. (2020). *Tutorial Belajar Python Part 31: Pengertian Parameter dan Argumen Fungsi Python*. <https://www.duniailkom.com/tutorial-belajar-python-pengertian-parameter-dan-argumen-fungsi-python/>
- Beazley, D. M. (2009). *Python Essential Reference* (Fourth Edi). Addison-Wesley.
- Huda, N. (2021). *Python Dasar: Menampilkan Output (print, format, dll)*. Jago Ngoding. <https://jagongoding.com/python/dasar/output/>
- Lutz, M. (2013). *Learning Python* (R. Roumeliotis (ed.); 5th ed.). O'Reilly Media.
- Mulyawan, R. (2024). *Modular Programming*. <https://rifqimulyawan.com/literasi/modular-programming/>
- novitaa, dian. (2024). *Apa Itu Lambda? Panduan Lengkap tentang Lambda dalam Komputasi dan Matematika*.

- Teknokrat. <https://daftarkampus.spmb.teknokrat.ac.id/apa-itu-lambda-panduan-lengkap-tentang-lambda-dalam-komputasi-dan-matematika/>
- Python Software Foundation. (2024). *The Python Tutorial*. <https://docs.python.org/id/3/tutorial/index.html>
- Riswandi, M. R. (2023). *Pengenalan Fungsi Pada Python: Argumen dan Parameter*. Kakakiky. <https://www.kakakiky.id/2023/09/argumen-dan-parameter-python.html>
- Suryantara, I. G. N. (2024). *Python: Bahasa Pemrograman Era Digital untuk Implementasi Algoritmik dalam Era Industri 4.0 dan Society 5.0*. PT Elex Media Komputindo.
- Zarman, W., & Wicaksono, M. F. (2020). *Implementasi Algoritma dalam Bahasa Python*. Informatika.

PROFIL PENULIS



Nur Oktavin Idris, M.Eng.

Penulis merupakan dosen tetap dalam bidang ilmu Teknologi Informasi dan mengajar mata kuliah Perancangan Basis Data, Pemrograman Berorientasi Objek, Big Data, serta Algoritma dan Pemrograman pada Program Studi Teknologi Rekayasa Perangkat Lunak Program Vokasi Universitas Negeri Gorontalo. Penulis menempuh pendidikan D3 Manajemen Informatika di Universitas Negeri Gorontalo, S1 Sistem Informasi di STMIK Ichsan Gorontalo serta S2 Teknologi Informasi di Universitas Gadjah Mada. Penulis juga aktif dalam kegiatan penelitian dan pengabdian masyarakat, dan telah menerbitkan buku, jurnal nasional serta berpartisipasi dalam konferensi internasional bereputasi.

BAB 11

POINTER PADA PYTHON

Rudy Cahyadi
Politeknik Negeri Media Kreatif, Jakarta
E-mail: masrudyc@gmail.com

PENDAHULUAN

Pointer merupakan salah satu konsep penting dalam banyak bahasa pemrograman, terutama yang dekat dengan *hardware* seperti C dan C++. Pointer adalah variabel yang menyimpan alamat memori dari variabel lain. Namun, Python, sebagai bahasa pemrograman tingkat tinggi, menyembunyikan kompleksitas manajemen memori dari penggunaannya. Python menggunakan referensi dan garbage collection untuk mengelola memori secara otomatis (Ramalho, 2022).

Salah satu fitur penting dalam pemrograman adalah penggunaan pointer, yang secara eksplisit terlihat di bahasa seperti C atau C++. Namun, Python menangani pointer secara berbeda. Meskipun Python tidak memiliki dukungan pointer secara langsung seperti di C/C++, Python menggunakan referensi obyek dan model manajemen memori yang mirip dengan penggunaan pointer dalam praktik. Pada bab ini akan dibahas konsep pointer dalam Python, bagaimana Python secara implisit menangani referensi obyek dalam memori, serta cara kerjanya yang berbeda dengan bahasa pemrograman lainnya. Pemahaman mengenai cara Python dalam menangani obyek, referensi, dan alokasi memori dapat membantu dalam penggunaan bahasa ini dengan lebih efisien.

KONSEP DASAR POINTER

Pointer adalah variabel yang menyimpan alamat memori dari variabel lain. Mereka memungkinkan pemrogram untuk langsung mengakses dan memanipulasi memori, yang dapat meningkatkan efisiensi dan fleksibilitas program. Dalam bahasa seperti C dan C++, pointer digunakan untuk berbagai tujuan, termasuk manipulasi array, alokasi memori secara dinamis, dan pengelolaan struktur data. Namun, penggunaan pointer yang tidak hati-hati dapat menyebabkan masalah seperti kebocoran memori dan dereferensi pointer null. Secara umum perbandingan antara C/C++, Java, dan Python adalah sebagai berikut :

1. Kontrol Memori:
 - C/C++: Memberikan kontrol penuh atas memori, memungkinkan penggunaan pointer untuk manipulasi memori langsung.
 - Java: Mengabstraksi manajemen memori dengan menggunakan referensi dan pengumpulan sampah otomatis.
 - Python: Menggunakan referensi obyek dan pengumpulan sampah otomatis, mirip dengan Java.
2. Keamanan
 - C/C++: Kurang aman karena pointer dapat menyebabkan masalah seperti kebocoran memori dan dereferensi pointer null.
 - Java dan Python: Lebih aman karena tidak ada pointer eksplisit dan pengelolaan memori dilakukan secara otomatis.
3. Kemudahan Penggunaan:
 - C/C++: Lebih kompleks karena programmer harus mengelola memori secara manual.
 - Java dan Python: Lebih mudah digunakan karena manajemen memori dihandle oleh runtime.

data besar tanpa membuat salinan data yang tidak diperlukan.

5. Aplikasi Praktis Pointer dalam Python

Meskipun tidak memiliki pointer eksplisit, konsep pointer secara fungsional direpresentasikan melalui penggunaan referensi. *Names* dalam Python bertindak seperti "pointer" dalam arti bahwa mereka menunjuk ke lokasi objek di memori, dan perilaku ini sangat relevan dalam kasus pengelolaan data, seperti penanganan list, dict, atau objek kompleks lainnya.

DAFTAR PUSTAKA

Python Software Foundation. Python Documentation.
<https://docs.python.org/>

Ramos, Leodanis Pozo , (2023) Python's Mutable vs Immutable Types: What's the Difference? Retrieved 13 September, 2024, from <https://realpython.com/python-mutable-vs-immutable-types/>

Ramalho, Luciano. Fluent Python 2nd Edition: Clear, Concise, and Effective Programming. O'Reilly Media, 2022

Tyas, Seyma. (21/03/2024). Garbage Collection and Memory Management in Python. *Reference list: Electronic sources (web publications)*. Retrieved 12 September, 2024, from <https://builtin.com/articles/garbage-collection-in-python>

PROFIL PENULIS



Rudy Cahyadi, S.Si, MT

Lahir di Palu, 19 Maret 1975. Lulus Sarjana pada program studi Elektronika & Instrumentasi Universitas Gadjah Mada tahun 1998 dan menyelesaikan S2 pada program studi Teknik Elektro Program Pasca Sarjana Universitas Gadjah Mada tahun 2002. Saat ini menjadi dosen pada program studi Teknologi Permainan, Politeknik Negeri Media Kreatif Jakarta. Mata kuliah yang diampu antara lain

Dasar Pemrograman, Pemrograman Game Beroientasi Obyek, Augmented dan Virtual Reality, dan Metodologi Penelitian. Bidang peminatan adalah Pemrograman, Kecerdasan Buatan, Sains Data.

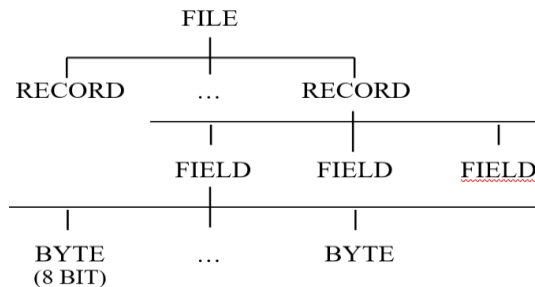
BAB 12

OPERASI FILE

Ariesta Damayanti
Universitas Teknologi Digital Indonesia, Yogyakarta
E-mail: ariesta@utdi.ac.id

PENDAHULUAN

Data yang ada pada komputer, seperti angka, teks, gambar, suara, video, dan lain-lain serta di simpan secara permanen di media penyimpanan seperti hardisk dikenal dengan file. Karena memori (RAM) komputer bersifat volatile atau tidak stabil, data akan hilang saat komputer dimatikan, sehingga file harus dibuka lagi kemudian. File juga disebut sebagai unit penyimpan logika, yang diekstrak dari perangkat penyimpan oleh sistem operasi. Disk magnetik, tape magnetik, dan disk optik adalah contoh penyimpan sekunder di mana informasi disimpan. Informasi yang ada dalam file diatur oleh pembuatnya. Tipe file akan menentukan struktur file. Jenis file dapat berupa data numerik, karakter, atau biner dan pada program, seperti misalnya source program, objek program, dan program executable. Struktur yang ada pada file ditunjukkan pada gambar 12.1.



Gambar 12.1. Struktur data pada file

Dalam pembuatan program yang sesungguhnya, operasi pada file memegang peranan penting karena selama program berjalan, data harus disimpan dalam file. Sebagian besar aplikasi menggunakan media disk untuk membaca atau merekam data. Data yang tersimpan di dalam disk akan disimpan dalam bentuk kesatuan yang disebut file. Pengertian dari file sendiri adalah kumpulan byte-byte yang disimpan dalam media penyimpanan. File merupakan komponen yang bertipe data sama, yang jumlahnya tidak tentu, yang dapat ditambah atau dikurangi jika dibutuhkan. File terdiri dari kumpulan *record* yang masing-masing terdiri dari satu atau beberapa field, dan setiap field terdiri dari satu atau beberapa byte, yang masing-masing byte merupakan susunan 8 bit.

Operasi yang ada pada file sendiri terdiri dari tiga komponen yaitu, membuka atau mengaktifkan file, melaksanakan proses pada file dan menutup file. File adalah komponen penting dalam bahasa pemrograman Python yang harus diperhatikan. Karena pasti akan dilakukan pembacaan file untuk pengolahan data atau menulis kode di dalamnya. Python memiliki mode operasi yang dapat digunakan oleh pengguna saat menangani file. mulai dengan menghapus, menulis, mengubah nama, dan menambahkan data ke file. Operasi pada file ini bekerja dengan semua jenis ekstensi file, termasuk file teks (.txt) dan file biner. Penanganan file telah berkembang dari bahasa pemrograman satu ke bahasa pemrograman lainnya, tetapi untuk implementasinya menjadi sulit ataupun panjang, sedangkan bahasa pemrograman Python mendukung penanganan file yang lebih mudah dan memungkinkan pengguna menangani berkas, seperti membaca dan menulis berkas, serta banyak opsi penanganan berkas lainnya. Seperti konsep Python lainnya, konsep ini juga sederhana dan singkat (Murugesh, T. S., Vasudevan, S. K., & Pulari, S. R. 2024).

Pesan Konfirmasi: Program mencetak pesan untuk mengkonfirmasi bahwa data telah ditulis dengan berhasil.

KESIMPULAN

Operasi file memegang peranan yang penting dalam pengelolaan file. Pada pembuatan pemrograman yang sesungguhnya operasi file memegang peranan yang sangat penting karena pada saat program dijalankan data berada pada file. Secara umum operasi file terdiri dari tiga tahapan yaitu, membuka atau mengaktifkan file, melaksanakan proses file dan menutup file. Python sebagai salah satu Bahasa Pemrograman yang mudah dipelajari dan digunakan memberikan beberapa alternatif untuk operasi file. Proses membaca, menulis, dan mengelola file dengan fungsi `open()`, `read()`, `write()`, `close()` dan beberapa fungsi lainnya. Sementara mode biner memungkinkan pengolahan file non-teks, pernyataan `with` memastikan file selalu tertutup dengan benar. Penanganan kesalahan adalah alat yang harus digunakan secara konsisten untuk memastikan bahwa program tetap berjalan meskipun terjadi masalah saat mengakses file.

DAFTAR PUSTAKA

- DoLab AI- Powered Learning. (27/12/2022). APA Style. *Reference list: Electronic sources (web publications). from* <https://dqlab.id/alternatif-mode-operasi-python-untuk-handlingfile#:~:text=Alternatif%20Mode%20Operasi%20Python%20untuk%20Handling%20File>
- Muruges, T. S., Vasudevan, S. K., & Pulari, S. R. (2024). *Python: A Practical Learning Approach*. CRC Press.
- Python Software Foundation. (n.d.). Python Documentation. Retrieved from [python.org](<https://docs.python.org/3/>)

- Ritonga, A., & Yahfizham, Y. (2023). Studi Literatur Perbandingan Bahasa Pemrograman C++ Dan Bahasa Pemrograman Python Pada Algoritma Pemrograman. *Jurnal Teknik Informatika dan Teknologi Informasi*, 3(3), 56-63.
- Shaw, Z. A. (2024). Learn Python the Hard Way. Addison-Wesley Professional.

PROFIL PENULIS



Ariesta Damayanti, S.Kom., M.Cs

Lahir di Banjarmasin, 20 April 1978. Lulus Sarjana pada jurusan Teknik Informatika STMIK AKAKOM Yogyakarta tahun 2002, menyelesaikan S2 pada program studi Ilmu Komputer Program Pasca Sarjana Universitas Gadjah Mada. Mulai mengajar tahun 2005 di STMIK AKAKOM yang sekarang berubah menjadi Universitas Teknologi Digital Indonesia. Saat ini menjadi dosen pada program studi Informatika, fakultas Teknologi Informasi, Universitas Teknologi Digital Indonesia. Mata kuliah yang diampu antara lain Kecerdasan Buatan, Pengenalan Pola, Machine Learning, Sistem Fuzzy, Data Mining, serta Jaringan Syarat Tiruan. Bidang konsentrasi penelitian adalah Kecerdasan Buatan, Machine Learning dan Data Mining.

ALGORITMA DAN PEMROGRAMAN DENGAN BAHASA PEMROGRAMAN PYTHON

Buku ALGORITMA DAN PEMROGRAMAN DENGAN BAHASA PEMROGRAMAN PYTHON merupakan panduan komprehensif yang dirancang untuk pemula yang ingin belajar bahasa pemrograman Python beserta struktur algoritmanya. Buku ini mencakup berbagai konsep Algoritma dan Pemrograman, mulai dari Pengantar dan Struktur Algoritma, Notasi, Pengenalan Python, hingga sampai konsep dasar dalam membuat program seperti variabel, tipe data, operator, array, dan pointer. Berdasarkan pendekatan yang praktis dan disertai dengan contoh-contoh kode pemrograman Python yang mudah diikuti, buku ini membantu pembaca memahami cara menulis dan menjalankan program Python dalam konsep Algoritma dan Pemrograman. Setiap bab dilengkapi dengan contoh pemrograman yang bertujuan untuk menguatkan pemahaman pembaca terhadap materi yang telah dipelajari. Selain itu, buku ini juga menyajikan operasi file pada python yang dapat membantu pembaca mengaplikasikan berbagai operasi file ke pemrograman Python yang telah dibuat.



FUTURE SCIENCE

Jl. Terusan Surabaya, Gang 1 A No. 71 RT 002 RW 005,
Kel. Sumbarsari, Kec. Lowokwaru, Kota Malang,
Provinsi Jawa Timur.
Website : www.futuresciencepress.com



IKAPI
IKATAN PENELITI INDONESIA

No. 348/JTI/2022

ISBN 978-634-7037-82-4 (PDF)



9

786347

037824