Feddy Setio Pribadi

Konsep Pemrograman Komputer dengan C++ dan Python

Konsep Pemrograman Komputer dengan C++ dan 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).

Konsep Pemrograman Komputer dengan C++ dan Python

Feddy Setio Pribadi

deepublish
glorify and develop the intellectual of human's life

KONSEP PEMROGRAMAN KOMPUTER DENGAN C++ DAN PYTHON

Feddy Setio Pribadi

Desain Cover :
Herlambang Rahmadhani

Ilustrator :
Herlambang Rahmadhani

Tata Letak :
Titis Yuliyanti

Proofreader :
Mira Muarifah

Ukuran :
x, 51 hlm, Uk: 15.5x23 cm

ISBN :
978-623-02-3295-4

Cetakan Pertama :
Agustus 2021

PENERBIT DEEPUBLISH
(Grup Penerbitan CV BUDI UTAMA)
Anggota IKAPI (076/DIY/2012)

Jl.Rajawali, G. Elang 6, No 3, Drono, Sardonoharjo, Ngaglik, Sleman
Jl.Kaliurang Km.9,3 – Yogyakarta 55581
Telp/Faks: (0274) 4533427
Website: www.deepublish.co.id
www.penerbitdeepublish.com
E-mail: cs@deepublish.co.id

PENGANTAR

Alhamdulillah, puji syukur kami panjatkan kepada Allah Swt., dengan rahmat dan karunia-Nya maka buku ini dapat selesai ditulis. Buku ini mencoba menjelaskan konsep dasar pemrograman komputer. Pemahaman terhadap konsep dasar ini akan memberikan bekal dalam mengembangkan sebuah program komputer yang kompleks.

Buku ini merupakan buku ajar yang digunakan sebagai salah satu rujukan dalam memahami materi pada mata kuliah Pemrograman Komputer Dasar untuk tengah semester pertama. Penekanan buku ini adalah pada konsep pemrograman yaitu bagaimana pembaca akan diantarkan bagaimana memahami alur instruksi atau kode program.

Konsep pemrograman yang dibahas pada buku ajar ini adalah pengenalan dan pemahaman terhadap variabel dan tipe data, tiga struktur dasar pemrograman yaitu struktur terurut, struktur percabangan, dan struktur pengulangan. Konsep terakhir yang dibahas pada buku ajar ini adalah tentang konsep fungsi (*function*).

Sebagai pengantar awal tentang konsep pemrograman buku ajar ini menggunakan dua bahasa pemrograman untuk memberikan pemahaman kepada pembaca tentang konsep yang tengah dipelajari. Penggunaan dua bahasa pemrograman sebagai implementasi dari konsep diharapkan akan memberikan gambaran yang lebih jelas.

Penulis menyadari bahwa buku ajar ini masih perlu perbaikan dan penambahan, dengan keterbatasannya semoga buku ajar ini dapat menjadi salah satu media dalam mengantarkan pembaca dalam memahami konsep pemrograman komputer.

PENDAHULUAN

Pembahasan bahasa pemrograman pada buku ini lebih ditekankan pada bahasa pemrograman tingkat tinggi yaitu bahasa C++ dan Python. Dua bahasa ini menjadi pengantar untuk para pembaca dalam memahami konsep atau teknik pemrograman komputer. Menurut Pandey [1] Pada dasarnya konsep dasar atau teknik dasar pemrograman terdiri dari 3 struktur yaitu *sequencial* (terurut), *conditional* (pencabangan), dan *looping* (pengulangan). Sebuah program komputer yang kompleks pada dasarnya tersusun dari 3 struktur tersebut. Pada buku ini menekankan pada 3 struktur tersebut, sehingga diharapkan dengan memahami 3 struktur dasar tersebut pembaca dapat memahami dan mengembangkan program komputer yang kompleks.

Bahasa C++ dan Python menjadi bahasa pengantar yang digunakan pada buku ini untuk memahamkan konsep pengembangan sebuah program komputer. Penggunaan dua bahasa pemrograman yang disajikan pada buku ini dimaksudkan untuk memberikan ilustrasi bagaimana cara melakukan translasi antar bahasa pemrograman. Kemampuan translasi ini akan membantu bagi pada pembaca ketika suatu saat akan mempelajari bahasa pemrograman yang lain. Berikut adalah tampilan yang diberikan pada buku in dalam mengimplementasikan materi ke dalam bentuk bahasa program C++ dan Python.

Baris	C++	Python
1	#include <iostream>	p = 4
2	Using namespace std;	l = 5
3		
4	int main()	keliling = (2*p)+(2*l)
5	{	print(“Kelilingnya: “, keliling)
6	int p = 4;	
7	int l = 5;	
8	int keliling;	
9		
10	keliling = (2*p)+(2*l);	

Baris	C++	Python
12	cout << “kelilingnya” << keliling << endl;	
13	}	
Output		
18		

Buku ini mencoba menampilkan kode program dengan C++ dan Python dengan menggunakan tampilan tabel seperti ditunjukkan di atas. Tabel tersebut terdiri dari 3 kolom yaitu kolom baris berupa urutan angka, kolom C++ merupakan kode program bahasa C++ dan kolom Python merupakan kode program bahasa Python. Kolom baris dimaksudkan sebagai alat bantu bagi pembaca yang akan mempraktikkan kode program di atas yang mana setiap angka menunjukkan bahwa kode tersebut dituliskan pada satu baris.

Penyajian dua kode program dalam bentuk tabel tersebut juga sebagai pembanding tata cara penulisan (sintaks) antara bahasa C++ dan Python. Kode program dalam bahasa pemrograman Python terlihat lebih sedikit dan lebih sederhana. C++ membutuhkan 13 baris kode untuk mengimplementasikan program menghitung keliling sebuah bentuk persegi, sementara itu Python hanya membutuhkan 5 baris kode berikut dengan jeda baris kosong.

DAFTAR ISI

BAB I

MENGENAL BAHASA PEMROGRAMAN

Pada bab ini membahas tentang pengertian bahasa pemrograman komputer, pengelompokan dan macam-macam bahasa pemrograman komputer. Setelah membaca bab ini diharapkan pembaca dapat membedakan macam-macam bahasa pemrograman sesuai dengan peruntukannya.

PENGERTIAN BAHASA PEMROGRAMAN KOMPUTER

Bahasa merupakan alat komunikasi, kita mengenal bahasa Indonesia, bahasa Inggris, bahasa Arab dll., yang mana semua itu berfungsi sebagai alat untuk menyampaikan pesan. Menganalogikan bahasa pemrograman dengan bahasa yang digunakan oleh manusia (bahasa manusia), bahasa pemrograman juga merupakan alat komunikasi, hanya saja bahasa pemrograman digunakan untuk menyampaikan pesan kepada mesin dalam hal ini adalah komputer.

Setiap bahasa tersebut mempunyai sintaks (cara penulisan) yang berbeda-beda akan tetapi secara umum mempunyai struktur yang sama. Coba kita tengok lagi bahasa manusia, pada tabel berikut ini mencoba membandingkan dua kalimat dengan menggunakan bahasa Inggris dan bahasa Indonesia.

Tabel. 1.1. Struktur bahasa manusia

I	Love	You
Subject/Subjek	*Verb*/Predikat	*Object*/Objek
Aku	Cinta	Kamu

Baris pertama dan ketiga Tabel 1.1 menampilkan dua kalimat yang pertama adalah "*I love you*" dan kalimat kedua adalah "Aku cinta kamu". Kedua kalimat tersebut mempunyai arti yang sama akan tetapi ditulis

menurut tata cara bahasa Inggris dan bahasa Indonesia. Perhatikan juga dengan saksama baris kedua Tabel 1.1 tersebut, jika diperhatikan kedua kalimat tersebut meskipun mempunyai tulisan yang berbeda akan tetapi mempunyai struktur yang sama yaitu bahwa kedua kalimat tersebut tersusun atas kata pertama sebagai subjek, kata yang di tengah sebagai kata kerja atau disebut pula dengan predikat (*verb*) dan kata terakhir berlaku sebagai objek.

Analogi dengan bahasa manusia, bahasa pemrograman pun mempunyai karakteristik yang sama. Seperti halnya bahasa manusia, bahasa pemrograman juga mempunyai banyak varian, seperti bahasa *Assembly*, C, C++, Python, Java, PHP, dsb. Bahasa-bahasa pemrograman tersebut juga mempunyai sintaks yang berbeda-beda akan tetapi secara umum mempunyai struktur yang sama. Perhatikan Tabel 1.2 berikut ini.

Tabel 1.2. Contoh sintaks bahasa pemrograman C++ dan Python

	C++	**Python**
Menampilkan tulisan	<< cout (“Hello World”)	print (“Hello World”)
Perintah *looping*	for (int n=0; n<5; n++)	for n in range(5)

Tabel 1.2 menampilkan dua sintaks bahasa pemrograman yaitu bahasa C++ dan bahasa Python. Pada baris kedua Tabel 1.2 memperlihatkan perintah yang keduanya (C++ dan Python) mempunyai tujuan yang sama yaitu menampilkan tulisan “Hello world” pada layar monitor. Pada bahasa C++ perintah untuk menampilkan tulisan dimonitor menggunakan perintah “cout” sedangkan pada Python menggunakan perintah “print”. Baris kedua Tabel 1.2 menunjukkan sintaks penulisan struktur *looping* dengan *keyword* “for”. Kedua sintaks tersebut mempunyai fungsi yang sama yaitu melakukan proses *looping* sebanyak 5 kali.

Bahasa pemrograman mempunyai banyak sekali variannya, akan tetapi secara umum semua bahasa tersebut mempunyai pola atau struktur dasar yang sama. Sehingga apabila kita telah memahami struktur dasar salah satu bahasa pemrograman, maka diharapkan kita akan dapat dengan mudah memahami bahasa pemrograman yang lain.

Bahasa pemrograman yang ditunjukkan pada Tabel 1.2 yaitu C++ dan Python merupakan bahasa pemrograman yang termasuk dalam

kelompok bahasa pemrograman *High Level Languages*. Menurut Zak [2] Perkembangan bahasan pemrograman komputer dapat dipetakan menjadi tiga yaitu Bahasa Mesin (*Machine Languages*), Bahasa *Assembly* (*Assembly Languages*), dan Bahasa Level Tinggi (*High Level Languages*). Secara umum bahasa pemrograman dapat dikategorikan menjadi 2 yaitu bahasa tingkat rendah (*Low Level Languages*) dan Bahasa Level Tinggi (*High Level Languages*). Bahasa Mesin dan Bahasa *Assembly* masuk dalam kategori bahasa tingkat rendah dan bahasa tingkat tinggi merupakan kategori tersendiri.

Pengelompokan bahasa pemrograman ke dalam kelompok bahasa tingkat rendah dan bahasa tingkat tinggi dilihat dari kemudahan orang (*programmer*) dalam membaca kode bahasa program tersebut. Sebuah bahasa pemrograman masuk dalam kategori bahasa pemrograman tingkat rendah jika bahasa pemrograman tersebut sulit untuk dibaca atau dipahami oleh manusia, sedangkan jika mudah dibaca atau dipahami oleh manusia maka termasuk dalam bahasa tingkat tinggi.

BAHASA PEMROGRAMAN MENURUT APLIKASI

Secara umum bahasa pemrograman yang tergolong ke dalam bahasa pemrograman tingkat tinggi dapat dibagi menjadi tiga kelompok dengan melihat dari produk atau aplikasi yang dihasilkan yaitu bahasa pemrograman desktop, bahasa pemrograman web dan bahasa pemrograman perangkat bergerak (*mobile*). Akan tetapi pembagian ini juga tidak terlalu kaku karena terdapat beberapa bahasa pemrograman yang mampu menghasilkan yang berjalan pada lebih dari satu platform.

Bahasa pemrograman desktop berarti bahwa bahasa pemrograman tersebut diperuntukkan untuk mengembangkan program aplikasi yang berjalan pada sebuah perangkat komputer (laptop atau PC). Aplikasi berbentuk desktop biasanya berupa aplikasi *stand olone* (berdiri sendiri). Aplikasi ini biasanya harus diinstall ke dalam perangkat komputer dan berjalan secara mandiri tanpa harus terhubung ke dalam jaringan komputer. Bahasa pemrograman ini yang pertama kali dikenal pada dunia komputasi. Bahasa pemrograman desktop di antaranya adalah BASIC, COBOL, PL/1, PASCAL, ALGOL, PROLOG, C.

Bahasa-bahasa tersebut merupakan bahasa pemrograman awal yang digunakan oleh *programmer* dalam mengembangkan program komputer atau aplikasi. Beberapa aplikasi yang dikembangkan dengan menggunakan bahasa tersebut masih tampil pada layar *monochrome* (*layer* hitam putih). Bahasa-bahasa pemrograman berbasis desktop pada dekade awal 80 telah berkembang dengan memanfaatkan pemrograman visual. Pemrograman visual ini mulai dikenal seiring dengan munculnya Sistem Operasi berbasiskan Jendela (Windows). Bahasa pemrograman visual yang populer di antaranya DELPHI, VISUAL BASIC, BORLAND C++ BUILDER, dll.

Seiring dengan perkembangan sistem informasi, program aplikasi berbasis Web menjadi sangat popular. Program aplikasi ini cenderung lebih mudah digunakan saat melakukan *sharing* aplikasi dan data antara server dan *client*. Aplikasi berbasis web berjalan di atas web *browser* (seperti Chrome, Firefox, Internet Explorer, Safari). Orang yang menjalankan aplikasi web tidak perlu melakukan instalasi, cukup membuka *browser* dan mengetikkan alamat URLS maka aplikasi tersebut akan tampil dan berjalan pada komputer tersebut. Aplikasi yang berjalan di web biasanya digunakan untuk mempermudah dalam berbagi data dan informasi. Bahasa pemrograman yang popular untuk mengembangkan aplikasi web di antaranya adalah PHP, ASP, dan Java Script.

Seiring perkembangan sistem dan teknologi informasi, sebuah *handphone* yang tadinya hanya sebagai alat komunikasi saat ini bertransformasi menjadi perangkat cerdas (*smartphone* dan tablet). Munculnya sistem operasi IOS dan Android memungkinkan kita dapat menjalankan aplikasi seperti yang berjalan di sebuah komputer. Bahasa pemrograman yang digunakan untuk membuat aplikasi Android yaitu Java dan bahasa pemrograman yang digunakan untuk mengembangkan aplikasi pada IOS yaitu C# dan Objective C.

Pada kenyataannya terdapat beberapa bahasa pemrograman yang dapat digunakan untuk mengembangkan aplikasi pada platform yang berbeda seperti bahasa Java yang dapat digunakan untuk mengembangkan aplikasi berbasis desktop dan *mobile*, Python dapat digunakan untuk mengembangkan aplikasi berbasis desktop dan web. Melalui penambahan kode tertentu beberapa bahasa pemrograman seperti C dan C++ dapat digunakan untuk membuat aplikasi yang berjalan pada *browser*, akan

tetapi cara-cara seperti ini hanya digunakan untuk membuat sebuah *library* untuk meningkatkan performa dari sistem yang tengah dikembangkan.

Selain istilah bahasa pemrograman, pada pengembangan aplikasi juga mengenal istilah "script". Script adalah bahasa pemrograman sederhana yang terkadang telah menyertakan banyak fungsi sehingga memudahkan dalam mengembangkan program. Script digunakan sebagai alat pendukung dalam mengembangkan suatu program aplikasi, seperti membuat animasi, memberikan peringatan jika terjadi kesalahan-kesalahan, mempercantik tampilan dll., contoh Script adalah Java Script dan VB script.

Bahasa pemrograman saat ini berkembang sangat cepat. Bisa kita lihat bagaimana sebuah aplikasi yang berjalan di desktop sekarang sudah dapat berjalan smartphone. *Browser* yang tadinya hanya menyajikan informasi berupa teks, saat ini juga dapat digunakan untuk menjalankan aplikasi office. Sebuah proses komputasi yang membutuhkan pemrograman yang kompleks saat ini juga dapat dilakukan pada media Cloud. Tidak menutup kemungkinan perkembangan yang ada saat ini, akan memicu perkembangan bidang komputasi yang lain seiring dengan tuntutan masyarakat.

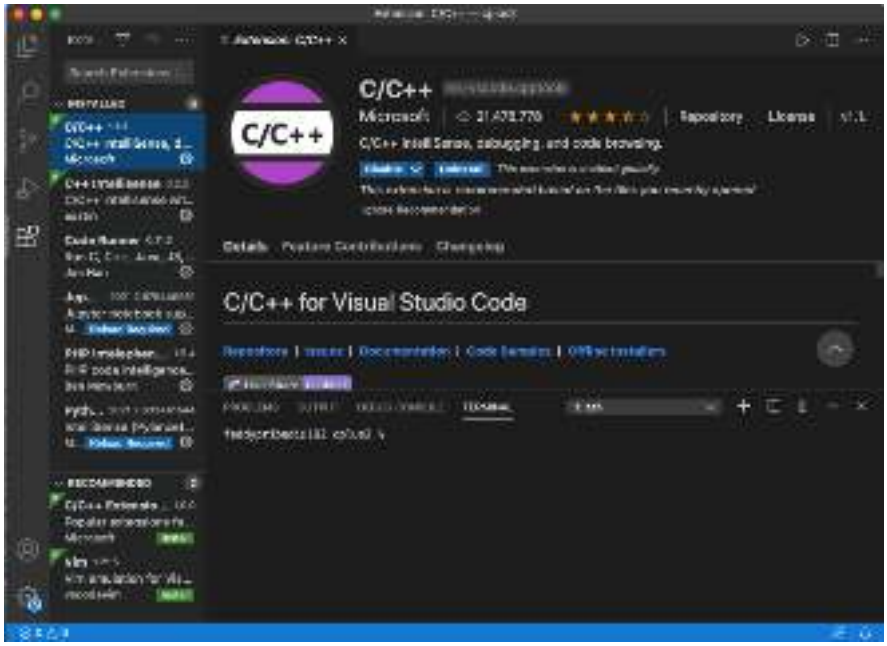
BAB II
EDITOR PROGRAM

Pada bab ini menyajikan penjelasan tentang penggunaan perangkat lunak (*software*) yang digunakan untuk mengembangkan aplikasi atau disebut juga sebagai *text editor* atau *code editor*. Istilah *text editor* biasanya digunakan untuk memberikan istilah pada perangkat lunak yang digunakan untuk mengembangkan aplikasi berbasis web, sedangkan untuk *code editor* digunakan untuk mengembangkan aplikasi berbasis desktop atau *mobile*. Perbedaan *text editor* dan *code editor* juga terletak pada lingkungan perangkat lunak tersebut. *Text editor* biasanya tidak dilengkapi dengan *interpreter* dan *compiler*, sedangkan *code editor* biasanya sudah dilengkapi dengan keduanya. *Interpreter* yang menerjemahkan bahasa yang dapat dibaca manusia ke dalam kode mesin yang bergantung pada komputer [3]. *Compiler* menerjemahkan kode sumber ke dalam bentuk perantara. Langkah ini disebut kompilasi, dan menghasilkan *file* objek. Kompiler kemudian memanggil *linker*, yang menggabungkan *file* objek ke dalam program yang dapat dieksekusi [1] [3]. Pada buku ini hanya akan dikenalkan dua perangkat lunak yang cukup popular saat ini sebagai *text editor* atau *code editor* yaitu Visual Studio Code dan Anaconda.

VISUAL STUDIO CODE

Visual Studio Code (VS Code) merupakan salah satu *code editor* yang cukup populer saat ini. VS Code memiliki fitur yang cukup lengkap, bahkan cukup dengan VS Code kita dapat mengembangkan aplikasi dengan berbagai macam bahasa pemrograman. VS Code kaya akan *library*, kita tinggal menambahkan *library* yang dibutuhkan dengan cara melakukan pencarian *library* yang telah disediakan pada lingkungan VS Code. VS Code dapat di-*download* melalui alamat https://code.visual

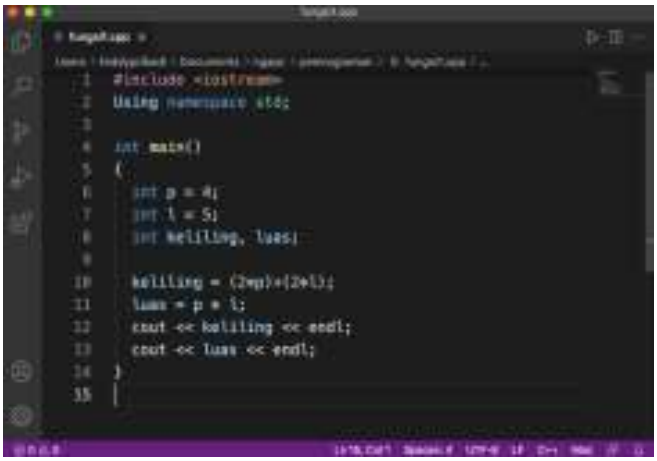
studio.com/download. Gambar 2.1 berikut ini menampilkan perangkat lunak dari VS Code.



Gambar 2.1. Tampilan Perangkat Lunak VS Code.

Gambar 2.1 menunjukkan tampilan ketika Anda akan melakukan instalasi pada VS Code sehingga dapat digunakan untuk mengembangkan program dengan bahasa C++. Anda cukup memasukkan *keyword*s C++ pada kotak pencarian "search extensions" yang terletak pada pojok kiri atas maka VS Code akan menampilkan daftar extensions yang bisa Anda pilih. Setelah mendapatkan extension yang sesuai, Anda tinggal menekan tombol Install yang terletak di bawah logo extensions.

VS Code memberikan kemudahan bagi kita, Hanya dengan menginstal VS Code Anda dapat membuat program komputer atau aplikasi sesuai dengan berbagai macam bahasa pemrograman. Melalui kotak pencarian "search extensions" maka Anda dapat melakukan instalasi beberapa bahasa pemrograman pada VS Code, seperti Python, PHP dan lain-lain.



Gambar 2.2. Tampilan Perangkat Lunak VS Code.

Gambar 2.2 menunjukkan kode program bahasa C++ yang ditulis pada VS Code. Setelah selesai menuliskan kode, untuk menjalankan program tersebut, Anda dapat melakukan klik pada tombol segitiga yang berada di pojok kanan atas. Hasil eksekusi kode tersebut akan muncul pada terminal.

Gambar 2.3. Hasil Eksekusi Kode Program pada Terminal

Anda harus terlebih dahulu melakukan pengaturan pada VS Code sehingga dapat terintegrasi dengan terminal. Proses integrasi antara VS Code dan terminal dapat Anda temukan pada situs resmi VS Code atau beberapa tutorial yang telah tersedia di internet.

JUPYTER NOTEBOOK

Edito program kedua yang digunakan pada buku ini adalah Jupyter Notebook. Jupyter Notebook merupakan editor yang khusus digunakan untuk mengimplemen bahasa Python. Jupyter Notebook dapat Anda unduh melalui halaman https://www.anaconda.com/.

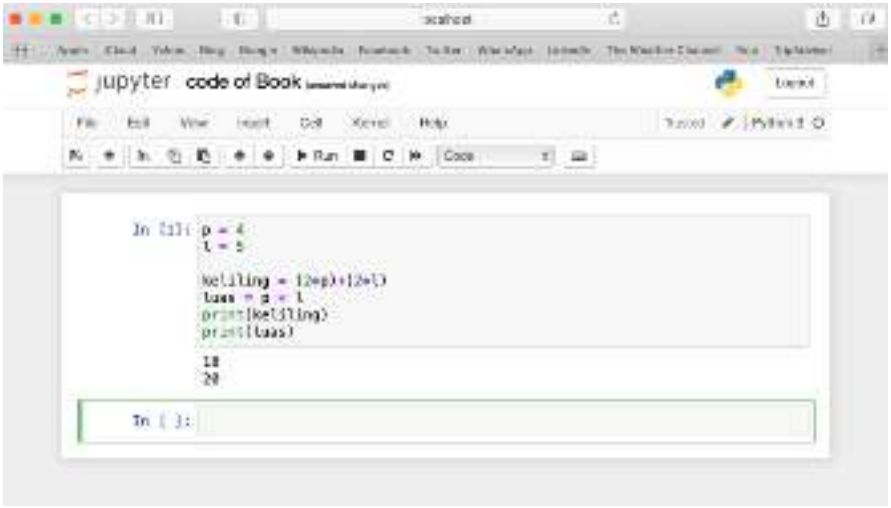
Gambar 2.4. Tampilan Halaman anaconda.com

Seperti ditunjukkan pada Gambar 2.4 terdapat banyak pilihan produk yang disediakan pada situs anaconda ini. Produk Individual Edition merupakan produk gratis yang dapat diunduh dan digunakan sebagai salah satu alat untuk mempelajari bahasa Python.

Gambar 2.4 Tampilan Halaman Instalasi Anaconda

Gambar 2.4 menampilkan halaman web setelah Anda mengklik menu "Individual Edition". Gulunglah *layer* ke bawah hingga mendapatkan tampilan seperti pada Gambar 2.4. Klik tombol "Install Anaconda" maka dengan serta-merta Anda akan mengunduh aplikasi Anaconda. Jika proses instalasi Anda selesai dan berhasil maka Anda akan mendapatkan tampilan aplikasi seperti yang terdapat pada Gambar 2.4. Jupyter Notebook merupakan salah satu menu yang terdapat pada aplikasi Anaconda.

Gambar 2.5 merupakan tampilan dari aplikasi Jupyter Notebook. Jupyter Notebook menyediakan sel untuk memasukan kode program. Tiap sel bisa berisi banyak baris program dan tiap-tiap sel bisa merupakan program yang berbeda, sehingga Ketika Anda akan menuliskan program baru tidak perlu membuat *file* yang baru cukup mengetikkan kode program pada sel yang lain. Sebagai catatan untuk membuat program yang baru usahakan menggunakan variabel yang lain kecuali Anda memang akan memanfaatkan atau menghubungkan kode antar sel, Anda dapat menggunakan variabel yang sudah ada. Anda dapat menjalankan kode program yang telah Anda selesai tulis dengan mengklik tombol Run yang terletak pada menu Bar.



Gambar 2.5 Tampilan Aplikasi Jupyter Notebook

Mungkin Anda bertanya bukankah VS Code bisa digunakan untuk mengimplementasikan bahasa Python, mengapa harus menggunakan Jupyter Notebook? Kelebihan dengan menggunakan Jupyter Anaconda adalah kemudahan dalam menambahkan *library* tambahan ketika mengembangkan sebuah aplikasi. Kelebihan yang lain adalah kita tidak perlu membuat *file* baru setiap kali kita akan menambahkan program baru karena setiap sel bisa di jalankan tanpa tergantung dengan sel yang lain. Terdapat juga fasilitas untuk memberikan komentar atau catatan dengan fasilitas "markdown". Masih ada lagi ini kelebihannya, Anda dapat mencetak kode program dalam bentuk *file* PDF yang tentu memudahkan dalam membuat dokumentasi aplikasi.

BAB III

VARIABEL DAN TIPE DATA

Pada bab ini membahas sebuah dasar dalam membuat kode program. Kode program dengan menggunakan bahasa pemrograman apapun pastilah mengenal yang namanya variabel. Variabel didefinisikan sebagai tempat yang digunakan untuk menyimpan suatu data bisa berupa angka atau teks. Pada bab ini kita akan mempelajari bagaimana kita menyiapkan tempat tersebut.

VARIABEL

Definisi tentang Variabel banyak ditemukan pada buku-buku tentang bahasa pemrograman komputer. Zak [2] menyatakan bahwa Variabel adalah alamat dalam memori komputer yang digunakan untuk menyimpan nilai. Davis [4] memberikan definisi sangat sederhana pada sebuah variabel, Variabel diibaratkan sebuah kotak yang dapat digunakan untuk menyimpan sesuatu.

Secara konseptual, lokasi memori di dalam unit memori dianalogikan seperti sebuah tabel yang terdiri dari baris dan kolom. Layaknya sebuah tabel setiap memori dalam unit memori memiliki alamat sel (*record*) yang berbeda antara satu dengan yang lain. Sebelum munculnya bahasa pemrograman tingkat tinggi seperti C, lokasi memori secara individu di ketahui dari alamatnya, sebagai contoh untuk menyimpan nilai integer 10 dan 100 dalam unit memori di simpan dalam alamat memori 0x7ffe atau 398800. Alamat memori tersebut akan berbeda antara komputer satu dengan yang lain.

Dalam bahasa tingkat tinggi penempatan suatu nilai dalam alamat memori tertentu diwakili oleh sebuah nama yang diambil dari bahasa alami. Penamaan simbol yang mewakili alamat memori disebut sebagai variabel. Menganalogikan dengan sebuah kompleks perumahan, konsep

variabel ini serupa dengan pemasangan papan nama pemilik rumah nomor sekian. Dalam kehidupan sehari-hari penunjukan sebuah rumah akan lebih mudah dengan menunjuk nama pemiliknya misalnya rumah Bapak Hadi atau rumah Bapak Eko dari pada menyebutkan nomor rumahnya misalnya rumah nomor 365A atau rumah nomor 101B.

Analogi yang sama juga dapat digunakan dalam memahami konsep variabel yaitu dalam bahasa matematika seperti x = y + z, yang mana x, y, dan z adalah nama-nama yang diberikan untuk mewakili nilai tertentu. Pada bahasa matematika x, y, dan z juga dikenal sebagai variabel. Melihat kembali pada program yang tertuang pada Gambar 2.3, Kita dapat memperhatikan potongan kode program tersebut:

```
P = 4;
L = 5
keliling = (2*P) + (2*L)
```

P dan L adalah nama variabel yang diberikan oleh *programmer* untuk menyimpan nilai 4 dan 5 ke dalam memori komputer yang lokasinya misal berada pada 1652 dan 2548. Terdapat aturan umum yang berlaku pada bahasa pemrograman ketika membuat atau mendefinisikan variabel yaitu sebagai berikut

1. Pengenal harus diawali dengan huruf (A … Z, a … z) atau karakter garis bawah (_).
2. Selanjutnya dapat berupa huruf, angka (0 .. 9), karakter garis bawah, atau simbol seperti dolar ($), %, dll.
3. Pengenal tidak boleh mengandung spasi
4. Panjang pengenal boleh terdiri lebih dari 32 karakter, akan tetapi hanya 32 karakter pertama yang dianggap berarti.

Contoh penamaan nama variabel yang dianggap valid dan tidak valid tersaji pada Tabel 3.1 berikut ini

Tabel 3.1. Tata Cara Penamaan Variabel

No.	variabel	valid	Tidak	No.	Variabel	valid	tidak
1	Jumlah	√		8	var1	√	
2	jumlah	√		9	1n		√

No.	variabel	valid	Tidak	No.	Variabel	valid	tidak
3	JUMLAH	√		10	var		√
4	jum_1	√		11	%persen		√
5	n	√		12	$_harga		√
6	n1	√		13	float		√
7	_satu	√		14	main		√

Tabel 3.1. menunjukkan tata cara pembuatan variabel yang dianggap valid dan yang tidak valid. Pada beberapa bahasa pemrograman termasuk C++ dan Python menganut aturan *case sensitive*, sebagai contoh variabel pada nomor 1, 2, dan 3 merupakan variabel yang berbeda meskipun hanya berbeda satu huruf. Selain menganut aturan pada ketentuan nomor 1 hingga 4, pembuatan nama variabel juga memperhatikan *keyword* atau pustaka pada bahasa pemrograman yang sedang digunakan. Pada Tabel 3.1. no. 13 dan 14 merupakan variabel yang tidak valid. Variabel *float* pada no. 13 merupakan nama dari tipe data yang digunakan pada bahasa C, C++, Python, Sementara itu no, 14 merupakan *keyword*s pada bahasa pemrograman C yang merujuk ke fungsi utama.

TIPE DATA

Tipe data merupakan batasan yang diberikan kepada variabel yang mana akan menentukan besarnya alokasi dari memori yang digunakan untuk menyimpan data [4] [2]. Penulisan tipe data pada variabel ditunjukkan pada Program 3.1 berikut ini.

Program 3.1. Pendefinisian variabel dan tipe data

Baris	C++	Python
1	#include <iostream>	p = 4
2	Using namespace std;	l = 5
3		
4	int main()	keliling = (2*p)+(2*l)
5	{	print(“Kelilingnya: “, keliling)
6	int p = 4;	
7	int l = 5;	
8	int keliling;	
9		
10	keliling = (2*p)+(2*l);	

Baris	C++	Python
12	cout << “kelilingnya” << keliling << endl;	
13	}	
Output		
18		

Program 3.1 menunjukkan bagaimana penulisan tipe data antara C++ dan Python. Program 3.1 menunjukkan penggunaan tipe data interger untuk membatasi nilai yang disimpan pada variabel p dan l, dan keliling. Tipe data biasanya dituliskan didepan nama variabelnya. Python tidak membutuhkan pendefinisian tipe data pada variabelnya. Seperti terlihat pada Program 3.1 pada kolom Python, kode program Python hanya menuliskan nama variabel dan nilai yang dimasukkan di dalamnya. Melalui informasi tersebut Python dapat memahami bahwa yang dimasukkan pada variabel berupa angka interger sehingga nilai keluaran yang dihasilkan juga berupa angka bertipe integer.

Beberapa bahasa pemrograman yang lain seperti Java, c, dan PHP menggunakan konsep yang serupa dengan C++ pada proses pendefinisian tipe data pada variabelnya. Meskipun Python tidak membutuhkan pendefinisian variabel di awal akan tetapi sebagai seorang *programmer* haruslah diwajibkan memahami tentang tipe data. Contoh-contoh program pada bab-bab selanjutnya akan memperjelas bagaimana secara prinsip seorang *programmer* harus memahami tipe data meski pada Python tipe data tidak secara eksplisit dinyatakan di awal pendefinisian sebuah variabel.

Pada dasarnya tipe data dapat digolongkan menjadi tiga yaitu Boolean, angka, dan teks. Penjelasan lebih detail terkait 3 jenis tipe data akan dijelaskan pada bagian di bawah ini.

BOOLEAN

Boolean hanya berisi dua nilai yaitu True dan False atau 1 dan 0. Tipe data ini biasa digunakan sebagai penanda pada sebuah urutan pada intruksi atau kode. Jika variabel bernilai 0 maka program akan mengeksekusi kode A sebagai misal, dan jika variabel bernilai 1 maka akan mengeksekusi kode B. Contoh penggunaan variabel Boolean pada sebuah program

Program 3.2. Pendifinisian Tipe Data Boolean

Baris	C++	Python
1	#include <iostream>	x = 5
2	Using namespace std;	y = False
3		z = 3*(bool(x) + bool(y))
4	int main()	print(z)
5	{	
6	bool x = 5;	
7	bool y = False;	
8	Cout << 3*(x+y) << endl;	
9	}	
Output		
3		

Program 3.2 akan menghasilkan nilai 3 jika program tersebut di eksekusi. Bagaimana program tersebut dapat menghasilkan angka 3? Sekarang coba kita runtut per kode, seperti terlihat program tersebut mempunyai tiga variabel yaitu x, y, dan z. Variabel x berisi angka 5, y berisi teks "False" dan z merupakan variabel yang digunakan untuk menyimpan hasil operasi. Operasi aritmatika (baris 3 kolom Python dan baris 8 kolom C++) mempunyai operasi 3 * (1 + 0) = 3. Variabel x akan bernilai 1 pada operasi aritmatika tersebut karena x bertipe data Boolean dan bernilai "True". Variabel y akan bernilai 0 karena bertipe Boolean dan mempunyai nilai "False".

Pendefinisian tipe data pada Python sedikit unik karena setiap variabel harus didefinisikan terlebih dahulu sebelum dieksekusi. Jika x dan y pada Python pada program 3.2 tidak dedifinikasi tipe datanya, maka variabel x secara default akan bertipe integer dan variabel y akan bertipe Boolean. Sebagai pengayaan terhadap pengaruh dari variabel bertipe Boolean, modifikasilah variabel menjadi y = true maka *output* program akan bernilai 6.

NUMERIC

Pada tipe numerik dikenal beberapa macam tipe data seperti integer, *float*, dan *double*. Berbeda dengan C++, tipe data yang dimiliki Python pada kategori numerik ini hanya ada dua yaitu *float* dan integer [5]. Tipe data integer digunakan untuk menandai variabel sehingga dapat digunakan untuk menyimpan bilangan bulat baik positif maupun negatif, sementara

itu untuk *float* dan *double* digunakan untuk menyimpan bilangan desimal. Perbedaan *float* dan *double* terletak pada besarnya data (digit yang dapat ditampung) yang mampu disimpan pada variabel yang ditandainya.

Program 3.3. Operasi Aritmatika Variabel Bertipe Integer dan Float

Baris	C++	Python
1	#include <iostream>	x = 5
2	Using namespace std;	y = 2
4	int main()	r1 = x + y
5	{	r2 = x / y
6	int x = 5;	print (r1)
7	int y = 2;	print (r2
8	Int z = 2;	
9	int r1, r2; float r3	
10	r1 = x + y;	
11	r2 = x/y;	
12	r3 = x/z;	
13	Cout << r1 << endl;	
14	Cout << r2 << endl;	
15	Cout << r3 << endl;	
16	}	
Output		
	7 2 2.5	7 2.5

Seperti yang diperlihatkan pada Program 3.3, C++ harus mendefinisikan dengan tepat tipe data untuk tiap variabelnya. Variabel r2 menghasilkan nilai keluaran yang tidak tepat dikarenakan penggunaan tipe data yang tidak tepat. Perlu menjadi catatan bahwa baik variabel yang dioperasikan maupun variabel yang digunakan untuk menampung/ menyimpan hasil operasi, harus mempunyai tipe data yang sesuai dengan nilai yang ditampungnya.

Python sekali lagi sedikit ajaib, pada Program 3.3 menunjukkan bahwa tanpa memberikan pendefinisian tipe data didepan variabel, Python mampu memberikan hasil yang tepat pada operasi aritmatika tersebut. Variabel pada Python secara cerdas mampu memahami nilai yang ditampung di dalamnya baik itu bilangan bulat, desimal, maupun teks.

TEKS

Tipe data berbentuk teks dapat dibedakan menjadi dua jenis yaitu karakter (huruf) dan *string* (kata/kalimat). Karakter adalah huruf alfabet, tanda baca, simbol, dan angka. Ketentuan dalam dalam mendefinisikan data bertipe karakter harus diapit oleh tanda petik tunggal atau pun petik ganda, contoh, 'h', 'Y','9". "9" meski berupa angka jika diapit oleh tanda petik maka angka sembilan tersebut berlaku sebagai karakter 9.

Program 3.4. Operasi Aritmatika Variabel Bertipe Integer dan Float

Baris	C++	Python
1	#include <iostream>	kata = 'saya'
2	Using namespace std;	kalimat = "saya makan"
4	int main()	print (kata)
5	{	print (kalimat)
6	char k[5] = "saya";	
7	string nama = "saya makan";	
8	cout << k << endl;	
9	cout << nama << endl;	
16	}	
Output		
saya saya makan		

Program 3.4 menunjukkan penggunaan tipe data teks. Bahasa C++ mempunyai dua tipe data yang dapat digunakan untuk menandai variabel sehingga dapat menyimpan data berbentuk teks yaitu *char* dan *string*. *Char* dengan diikuti dengan tanda kurung kotak yang di dalamnya berisi angka menunjukkan bahwa variabel k mampu digunakan untuk menampung 5 huruf atau karakter. Hal ini akan menjadi error jika teks yang dimasukkan pada variabel k melebihi jumlah nilai pada kurung kotak yang dipersyaratkan. C++ mempunyai tipe data *string* untuk menyimpan data dalam bentuk teks yang relatif lebih panjang.

Bahasa Python mempunyai cara yang lebih sederhana dalam menandai variabel yang bertipe *string* yaitu dengan memberikan tanda petik tunggal atau ganda pada data yang akan disimpan pada sebuah variabel.

Sebagai catatan bahwa setiap data yang diberi tanda petik akan diperlakukan sebagai teks meskipun data tersebut berupa angka, sehingga apabila dijadikan bentuk teks maka angka tersebut tidak dapat digunakan dalam operasi matematika.

BAB IV

STRUKTUR BERURUTAN (*SEQUENTIAL*)

Pada bab ini akan menjelaskan tentang struktur pemrograman terurut, akan tetapi sebelumnya akan didahului dengan membahas tentang operator dan fungsi-fungsi matematika standar yang terdapat pada *library* bahasa pemrograman.

OPERATOR DAN FUNGSI MATEMATIKA

Mengenal operator dan fungsi matematika dalam memahami konsep pemrograman terstruktur merupakan hal mendasar dalam memahami konsep pemrograman secara keseluruhan. Beberapa buku pemrograman akan mengawali contoh programnya dengan membuat program Hello World atau dengan program aritmatika sederhana. Berikut daftar operator aritmatika yang digunakan dalam pemrograman bahasa C dan Python.

Tabel 4.1. Simbol Operator Aritmatika dalam C++ dan Python

Arti	C++	Python
Penambahan	+	+
Pengurangan	-	-
Perkalian	*	*
Pembagian	/	/
Sisa Hasil Bagi (Modulus)	%	%
Pangkat kuadrat	^	**

Beberapa operasi matematika yang agak kompleks dalam bahasa pemrograman biasanya telah dibuatkan beberapa fungsi bawaan sehingga pengguna tinggal memanfaatkan fungsi tersebut dalam pemrograman, beberapa fungsi matematika yang disediakan oleh C++ dan Python :

Tabel 4.2 Fungsi matematika yang disediakan C

Fungsi	Arti
pow()	Fungsi untuk kuadrat
abs()	Fungsi untuk mendapatkan nilai absolut
sqrt()	Fungsi untuk mendapatkan nilai akar kuadrat
sin()	Fungsi untuk mendapatkan nilai Sinus
cos()	Fungsi untuk mendapatkan nilai Cosinus

PEMROGRAMAN TERURUT (*SEQUENTIAL*)

Pemrograman *sequential* atau pemrograman terurut adalah struktur pemrograman yang paling umum digunakan dalam membuat kode program komputer. Secara *default*, semua kode program tersusun sebagai struktur yang terurut dari atas ke bawah. Pada bagian ini membahas tentang struktur kode program yang tersusun secara *sequential*.

Program 4.1. Contoh 1 Program dengan Struktur terurut

Baris	C++	Python
1	#include <iostream>	x = 5
2	using namespace std;	y = False
3		z = 3 * (bool(x) + y)
4	int main()	print (z)
5	{	
6	int roti = 5; // x = 1;	
7	int kalori = 125; // y = 0;	
8	int dimakan = 2;	
9	int total_dimakan=0;	
10		
11	total_dimakan = total_dimakan + dimakan;	
12	roti = roti - dimakan;	
13	cout << "yang telah dimakan: " << total_dimakan << "; " <<"Roti yang sisa: " << roti << endl;	
14	cout << "total kalori: " << kalori * total_dimakan << endl;	
	return 0;	
15	}	
16		
Output		
..?..		

Program 4.2. Contoh 2 Program dengan Struktur terurut

Baris	C++	Python
1	#include <iostream>	rd = input(“Radiusnya? “)
2	Using namespace std;	t = input(‘Tingginya? “)
4	int main()	vol = 3.14*float(rd)*float(rd)*float(t)
5	{	print (“Volume Bangun Tersebut = “, vol)
6	float radius, tinggi, volume;	
7	cout << “Radiusnya? “;	
8	cin >> radius << endl;	
9	cout << “Tingginya? “ <<endl;	
10	cin >> tinggi << endl;	
11		
12	volume = 3.14*radius*radius*tinggi Cout << “Volume bangun tersebut adalah” << volume << endl;	
13	}	
Output		
..?..		

Program 4.3. Contoh 3 Memanfaatkan fungsi bawaan pada Struktur terurut

Baris	C++	Python
1	#include <iostream>	import math
2	Using namespace std;	
3	#include <cmath>	sd = input(“Sudutnya? “)
4	int main()	hasil = math.sin(sd)
5	{	
6	float sudut, hasil;	print (“Sinus Sudut tersebut”, hasil)
7	cout << “Sudutnya? “;	
8	cin >> radius << endl;	
9	hasil = sin(sudut)	
10	Cout << ‘Sinus Sudut tersebut” << hasil << endl;	
11	}	
Output		
..?..		

Program 4.1, 4.2, dan 4.3 merupakan contoh dari program komputer yang mempunyai struktur terurut. Secara *default* ketika sebuah kode program komputer di eksekusi, kode tersebut akan dieksekusi urut satu persatu dari atas ke bawah. Ketikkan sintaks program tersebut sehingga Anda akan bisa mengetahui keluaran dari ketiga contoh program tersebut.

Hal yang perlu diperhatikan dalam menyusun kode program adalah setiap variabel yang terlibat dalam sebuah operasi matematika atau operasi yang lain harus sudah menyimpan nilai ketika baris kodenya akan dieksekusi. Sebagai ilustrasi baris ke 14 pada Program 4.1 dipindahkan ke baris 10 pada kolom C++. Setelah Anda ubah program tersebut akan mengeluarkan nilai 0 (nol) pada perhitungan "kalori". Perhitungan tersebut didapat dari operasi matematika 125 * 0 (nilai awal variabel kalori dan nilai awal total_dimakan).

Kode program komputer pada hakikatnya bekerja secara berurutan dan menjalankan tiap baris kode atau instruksi. Kesalahan peletakan variabel pada saat pengoperasian variabel tersebut bisa menyebabkan hasil yang dikeluarkan tidak sesuai dengan yang diharapkan. Seorang *programmer* harus selalu memeriksa setiap nilai keluaran yang dihasilkan ketika program dieksekusi, karena bisa jadi program tidak error akan tetapi hasil yang dikeluarkan dari proses eksekusi tidak sesuai dengan perhitungan.

Program 4.3 jika Anda eksekusi akan terjadi error, mengapa bisa demikian, Anda perlu memperhatikan tentang tipe data. Python akan menganggap nilai yang dimasukkan melalui keyboard melalui perintah input akan dianggap sebagai *string* (teks) sehingga nilai masukkan tersebut tidak dapat dikenai operasi matematika.

BAB V

STRUKTUR PEMILIHAN / PERCABANGAN (*CONDITIONAL*)

Pada bab ini akan menjelaskan tentang struktur pemrograman kondisional. Sebelum membahas tentang struktur pemrograman kondisional, penjelasan pada bab ini didahului dengan membahas tentang operator relasional dan operator logika.

OPERATOR RELASIONAL DAN LOGIKA

Operator relasional melambangkan hubungan antara dua entitas, Entitas tersebut bisa berupa variabel, konstanta maupun fungsi, Nilai dari kedua hubungan tersebut hanya bernilai dua yaitu TRUE atau FALSE. Adapun Operator operasional dalam pemrograman berbentuk sebagai berikut:

Tabel 5.1. Simbol Operator Relasional yang digunakan dalam C dan Python

Operator	Arti
>	Lebih Besar
>=	Lebih Besar atau Sama Dengan
<	Lebih Kecil
<=	Lebih Kecil atau Sama Dengan
==	Sama Dengan
!=	Tidak Sama Dengan

Operator tersebut akan mempunyai dua nilai True atau False jika digunakan dalam operasi matematika, sebagai contoh

```
3 > 1   : True  , 5 < 8 : True  , 3==5 : False , 3!=5  : True
```

Operator logika juga merupakan operator yang menyatakan kondisi True atau False dengan kondisi logika tertentu. Adapun operator logika dalam pemrograman berbentuk sebagai berikut:

Tabel 5.2. Simbol Operator Logika dalam C++ dan Python

Arti	C++	Python
Logika AND	&&	and
Logika OR	||	or
Logika NOT	!	!

Tabel 5.3. Operasi Logika AND

Variabel 1	Variabel 2	Hasil
True	True	True
True	False	False
False	True	False
False	False	False

Tabel 5.4. Operasi Logika OR

Variabel 1	Variabel 2	Hasil
True	True	True
True	False	True
False	True	True
False	False	False

Seperti dijelaskan pada bab sebelumnya bahwa teknik pemrograman yang pertama adalah pemrograman berurutan. Hal ini berarti bahwa kode program atau instruksi yang disusun akan dieksekusi satu demi satu secara berurutan mulai dari baris pertama hingga baris terakhir. Hal ini tidak berlaku pada struktur pemrograman pemilihan, pada struktur ini ada baris program yang tidak dieksekusi. Proses eksekusi akan menjalankan kode program dengan mengacu pada kondisi tertentu. Struktur percabangan mempunyai 3 bentuk yaitu percabangan tunggal, percabangan majemuk atau multi dan terakhir percabangan bersarang (*nested*).

STRUKTUR PERCABANGAN TUNGGAL

Bahasa pemrograman secara umum menyediakan *keyword* **if else** untuk menyusun kode program yang melibatkan struktur percabangan.

Contoh lain dalam kehidupan sehari-hari di mana kita diharuskan memeriksa suatu kondisi tertentu untuk membuat keputusan selanjutnya.

1. Jika Andi mendapatkan nilai >= 70 **maka** Andi Lulus
2. Jika Andi mendapatkan nilai < 70 **maka** Andi tidak lulus

Pernyataan tersebut di atas jika diterjemahkan dalam bentuk algoritma menjadi sebagai berikut:

If Nilai >= 70 --------- >(Ekspresi)
Lulus ---------- >(Pernyataan 1)
Else
Tidak Lulus ---- >(Pernyataan 2)

Program 5.1. Program Struktur Percabangan Tunggal

Baris	C++	Python
1	#include <iostream>	nilai = int(input("Nilainya? "))
2	Using namespace std;	
3		If (nilai >= 70):
4	int main()	print("Lulus")
5	{	else:
6	float nilai;	print ("Tidak Lulus")
7	cout << "Nilainya? ";	
8	cin >> nilai << endl;	
	If (nilai >= 70)	
9	cout << "Lulus";	
	else	
10	cout << "Tidak Lulus";	
11	}	
Output		
..?..		

Keyword **if else** untuk menyeleksi kondisi pernyataan, jika pernyataan tersebut bernilai benar atau sesuai dengan kondisi yang diinginkan maka pernyataan tersebut akan dijalankan. Sebaliknya jika penyataan yang diperiksa tidak sesuai dengan kondisi maka pernyataan tersebut tidak dijalankan dan menjalankan penyataan yang lain.

Kasus pada Program 5.1 variabel nilai akan diperiksa oleh *keyword* if else, jika variabel nilai memegang angka 70 atau lebih maka proses

eksekusi kode akan menjalankan perintah cout atau print “Lulus”, kode else tidak pernah dieksekusi oleh sistem, demikian sebaliknya jika variabel nilai memegang angka kurang dari 70 maka kode else yang akan dieksekusi. Berikut adalah contoh algoritma dengan melibatkan lebih dari satu pernyataan untuk tiap bloknya.

```
If Nilai >= 85 --------- >(Ekspresi)
{
Lulus
Predikat sangat memuaskan (blok pernyataan 1)
}
else
{
Tidak Lulus
Mengulang Semester Depan (blok pernyataan 2)
}
```

Program 5.2. Program Struktur Percabangan Tunggal

Baris	C++	Python
1	#include <iostream>	nilai = int(input(“Nilainya? “))
2	Using namespace std;	
3		If (nilai >= 70):
4	int main()	print(“Lulus”)
5	{	print(“Sangat Memuaskan”)
6	float nilai;	else:
7	cout << “Nilainya? “;	print (“Tidak Lulus”)
8	cin >> nilai << endl;	print (“Mengulang”)
9	If (nilai >= 70)	
10	{	
11	cout << “Lulus”;	
12	cCout << “Sangat Memuaskan”;	
13	else	
14	{	
15	cout << “Tidak Lulus”;	
16	cout << “Mengulang”;	
17	}	
18	}	
Output		
..?..		

Sebagai catatan pada sintaks penulisan percabangan pada C++ dan Python perlu mendapatkan perhatian. Sintaks C++ tidak membutuhkan

format indentasi, sedangkan pada Python wajib menggunakan format indentasi. C++ memerlukan membutuhkan tanda kurung kurawal untuk menandai blok pernyataan yang lebih dari satu, sementara itu pada Python semua pernyataan harus pada posisi menjorok ke dalam. Sintaks Python pada bagian ekspresi diakhiri dengan tanda titik dua, hal ini akan menyebabkan error jika tidak diberikan.

STRUKTUR MULTI PERCABANGAN

Pada sintaks struktur percabangan tunggal baik C++ maupun Python mempunyai sepasang *keyword*s yaitu If – Else Multi percabangan, pada struktur multi percabangan ini akan mempunyai banyak pasangan if – else yang diwujudkan pada dengan sintak elif atau else if yang diikuti dengan ekspresi. Sebagai contoh perhatikan pernyataan di bawah ini

```
Jika Mahasiswa mempunyai nilai lebih dari atau sama dengan 85
Maka Mahasiswa tersebut mendapat nilai A
Jika Mahasiswa mempunyai nilai antara 70 sampai 85
Maka Mahasiswa tersebut mendapat nilai B
Jika Mahasiswa mempunyai nilai antara 60 sampai 70
Maka Mahasiswa tersebut mendapat nilai C
```

Jika diperhatikan ada beberapa rentang nilai seperti 70 sampai 85, 60 sampai 70 yang harus diperiksa untuk mendapatkan kesimpulan yang benar, berikut adalah algoritma untuk mengevaluasi pernyataan di atas

```
if (nilai >= 85 )
    {
      Nilai A
    }
else if ( (nilai >= 70 ) and (nilai < 85) )
    {
      Nilai B
     }
else ( (nilai >= 60 ) and (nilai < 07) )
    {
      Nilai C
     }
```

Program yang melibatkan suatu kondisi pemilihan juga dapat dibuat secara bertingkat, pemeriksaan kondisi bertingkat juga dapat dilakukan dengan melibatkan operator logika seperti operator AND (&&) dan operator OR (||).

Program 5.3. Program Struktur Multi Percabangan

Baris	C++	Python
1	#include <iostream>	nilai = int(input("Nilainya? "))
2	Using namespace std;	
3		If (nilai >= 85):
4	int main()	print("Nilai A")
5	{	elif (nilai >=70 and nilai < 85):
6	float nilai;	print("Nilai B")
7	cout << "Nilainya? ";	else
8	cin >> nilai << endl;	print ("Nilai C")
9	If (nilai >= 85)	
10	cout << "Nilai A";	
11	else if (nilai >= 70 && nilai < 85)	
12	cout << "Nilai B";	
13	else	
14	cout << "Nilai C";	
18	}	
Output		
..?..		

Program 5.3 menunjukkan implementasi struktur multi pada C++ dan Python. Secara struktur dua bahasa tersebut mempunyai aturan yang sama hanya sintaksnya saja yang berbeda yaitu pada "else if" dan "elif". Semakin banyak ekspresi yang harus diperiksa maka semakin banyak struktur else if atau elif-nya. Perlu diperhatikan bahwa pemeriksaan terakhir pada struktur multi percabangan adalah dengan menggunakan else. Else yang terakhir ini menyatakan jika semua ekspresi yang di atasnya tidak memenuhi maka kode program yang dieksekusi terakhir adalah kode program pada struktur Else yang terakhir.

STRUKTUR PERCABANGAN BERSARANG (*NESTED*)

Program yang melibatkan suatu kondisi pemilihan juga dapat dibuat secara bertingkat atau yang lebih dikenal dengan istilah if else bersarang (*nested*). Format penulisan struktur percabangan bersarang ini merupakan bentuk lain dari penulisan percabangan dengan melibatkan operator logica seperti AND atau OR.

Program 5.4. Program Struktur Percabangan Bersarang

Baris	C++	Python
1	#include <iostream>	nilai = int(input("Nilainya? "))
2	Using namespace std;	
3		if (nilai >= 80):
4	int main()	print("Lulus")
5	{	if (nilai >=95):
6	float nilai;	print("Anak Jenius")
7	cout << “Nilainya? “;	else:
8	cin >> nilai << endl;	print("Anak Pandai")
9	If (nilai >= 80)	else:
10	{	print("Belum Lulus")
11	cout << “Lulus”;	
12	If (nilai >=95)	
13	{	
14	cout << “Anak Jenius”;	
15	}	
16	else	
17	{	
18	cout << “Anak Pandai”;	
19	}	
20	}	
21	else	
22	{	
23	cout << “Belum Lulus”;	
24	cout << “Mengulang”;	
25	}	
Output		
..?..		

Struktur percabangan bersarang mempunyai cara penulisan yang bertingkat. Pada sintaks bahasa Python terlihat jelas susunan yang bertingkat pada penulisan percabangan bersarang ini. Melalui penulisan yang mengharuskan menggunakan indentasi struktur if di dalam if menjadi lebih mudah untuk dibaca dan dipahami.

Pada program 5.4 ditunjukkan sebuah kasus yang mana pada kasus tersebut akan dilakukan secara bertingkat terhadap sebuah angka yang dimasukkan pada variabel nilai. Cara kerja program tersebut adalah pertama angka yang terdapat pada variabel nilai akan diperiksa pada blok if yang pertama yaitu memeriksa apakah nilainya sama dengan atau lebih dari 80. Jika nilainya kurang dari 80 maka proses eksekusi kode langsung menuju pada kode else pada baris ke 9 untuk Python dan baris ke 21 untuk C++. Sebaliknya jika angka pada variabel nilai lebih dari atau sama dengan 80 maka eksekusi program masuk pada baris ke 4 pada Python dan baris ke 11 pada C++. Pada tahap selanjutnya terdapat pemeriksaan lagi terhadap variabel nilai, apakah angka pada variabel nilai mempunyai nilai sama dengan atau lebih dari 95, jika "ya" maka program akan menampilkan "Lulus" dan "Anak Jenius", akan tetapi jika nilai kurang dari 95 program akan menampilkan "Lulus" dan "Anak Pandai".

BAB VI

PEMROGRAMAN BERULANG (*LOOPING*)

Proses pengulangan pada suatu program merupakan salah satu kekuatan yang dimiliki ketika kita mengembangkan suatu program komputer. Proses pengulangan merupakan suatu proses yang dilakukan oleh komputer secara berulang-ulang hingga kondisi yang diinginkan terpenuhi. Dalam beberapa buku proses pengulangan diistilahkan sebagai proses *looping* atau kalang. Pada umumnya bahasa pemrograman komputer mempunyai dua struktur untuk melakukan proses pengulangan yaitu struktur for dan while. Pada bahasa C++ terdapat tiga struktur *looping* yaitu struktur for, struktur while—do, struktur do—while.

Sebelum membahas lebih jauh tentang proses *looping*, terlebih dahulu akan dibahas mengenai operator decrement (--) dan operator increment (++). Operator decrement adalah operator di mana variabel yang dirujuk akan melakukan pengurangan secara otomatis satu angka untuk setiap kali proses *looping*. Operator increment adalah sebaliknya akan menambah angka satu pada variabel yang dirujuk setiap kali proses *looping*. Pada pemakaiannya operator ini dapat dituliskan dibelakang ataupun di depan sebuah variabel yang dirujuk sebagai contoh

```
i ++ , i-- penulisan setelah variabel
++ i , -- I penulisan sebelum variabel
```

Penerapan kedua operator di atas akan dibahas lebih lanjut dalam penggunaannya dalam proses pengembangan program dengan melibatkan proses *looping*.

LOOPING DENGAN MENGGUNAKAN STRUKTUR FOR

Looping dengan menggunakan struktur For merupakan instruksi yang sering dijumpai pada program komputer. Berikut adalah struktur *looping* For secara umum.

```
for (batas awal; batas akhir; increment/decrement)
        {

        }
```

Batas awal mengisyaratkan bahwa proses pengulangan dimulai pada angka 1, 2, atau 3, batas akhir mengisyaratkan batas atas dari suatu kondisi di mana proses pengulangan harus berhenti. increment/decrement adalah perintah atau proses untuk melakukan penambahan/pengurangan terhadap sebuah *counter* (angka yang diguanakan sebagai penanda berapa kali program harus dieksekusi)
Berikut adalah contoh penggunaan stuktur for dalam pemrograman

Program 6.1. Program Pengulangan 1 dengan Struktur for

Baris	C++	Python
1	#include <iostream>	for i in range(1,6,1):
2	Using namespace std;	print("belajar program")
3		
4	int main()	
5	{	
6	for (int i=1; i<=5; i++)	
7	{	
8	cout << “belajara program”;	
9	}	
10	}	

Output
belajar program
belajar program
belajar program
belajar program
belajar program

Program 6.2. Program Pengulangan 2 dengan Struktur for

Baris	C++	Python
1	#include <iostream>	for i in range(1,6,1):
2	Using namespace std;	print(i)
3		
4	int main()	
5	{	
6	for (int i=1; i<=5; i++)	
7	{	
8	cout << i;	
9	}	
10	}	
Output		
1		
2		
3		
4		
5		

Program 6.1 dan 6.2 menunjukkan bagaimana penggunaan struktur for untuk menampilkan tulisan belajar program sebanyak 5 kali dan angka dari 1 hingga 5. Berikut adalah penjelasan program 6.2. *Looping* for mengisyaratkan bahwa program akan menampilkan angka di layar monitor dengan dimulai dengan angka 1 melaui perintah i=1, kemudian digit akan berangsur – angsur meningkat setiap kelipatan 1 melalui perintah i++, sedangkan perintah i<=5 mengisyaratkan kapan berakhirnya proses *looping*. Pada sintaks Python batas awal, akhir dan increment ditulis dengan menggunakan kode (1,6,1).

Terdapat perbedaan angka pada batas akhir C++ dan Python, pada C++ batas akhir menggunakan logika <=5, akan tetapi pada Python menggunakan angka 6. Angka 6 ini berarti bahwa *looping* akan diakhiri jika nilai pada variabel i kurang dari 6. Python tidak menggunakan operator logika pada perintah *looping* sehingga jika kita menentukan batas atas adalah angka 5 maka batas bawah dimulai dari angka 0 bukan 1 untuk melakukan proses *looping* sebanyak 5 kali.

Program 6.3. Program Pengulangan 2 dengan Struktur for

Baris	C++	Python
1	#include <iostream>	angka=int(input(“angka: "))
2	Using namespace std;	print("jumlah deret dari: ", angka)
3		jumlah = 0
4	int main()	for i in range(1,angka+1,1):
5	{	jumlah = jumlah + i
6	int angka, jumlah;	print(jumlah)
7	cout << “Angkanya: “;	
8	cin >> angka	
9	Jumlah = 0	
10	for (int i=1; i<=angka; i++)	
11	{	
12	jumlah = jumlah + I;	
13	cout << jumlah << endl;	
14	}	
15	}	

Output
1
3
6
10
15

Pada Program 6.3 di atas menghitung jumlah deret dari nilai yang dimasukan, semisal nilai yang dimasukan adalah 10 maka program akan melakukan proses penjumlahan yang berulang sebagai berikut; pertama variabel jumlah bernilai 0, pada instruksi jumlah = jumlah + I menyatakan proses perhitungan sebagai berikut jumlah = 0 + 1 untuk proses *looping* yang pertama. Hasil dari perhitungan tersebut di simpan dalam variabel jumlah, sehingga variabel jumlah yang semula 0 sekarang bernilai 1. Program berlanjut untuk *looping* yang kedua, instruksi jumlah = jumlah + i sekarang menyatakan proses perhitungan jumlah = 1 + 2, variabel jumlah dari proses *looping* yang pertama mengakibatkan variabel tersebut menyimpan angka 1 dan i pada proses *looping* yang kedua bernilai 2. Proses *looping* yang kedua akan disimpan dalam variabel jumlah sehingga yang tadinya bernilai 1 pada proses *looping* yang kedua akan bernilai 3. Demikian proses *looping* terus berlanjut hingga kondisi i <= 5 terpenuhi sebagai akhir dari *looping*.

LOOPING DENGAN MENGGUNAKAN STRUKTUR WHILE

Pernyataan *looping* dengan menggunakan *keyword* **while** pada umumnya diekspresikan dalam bentuk umum sebagai berikut:

```
while (ekspresi)
pernyataan
```

Ekspresi yang dikandung di dalam tanda kurung akan di uji seperti halnya ekspresi pada struktur pemrograman if – else. Jika ekspresi tersebut bernilai benar maka pernyataan di bawah struktur while akan dijalankan, jika sebaliknya maka pernyataan tersebut tidak akan pernah dijalankan. Berikut adalah proses yang dijalankan oleh komputer ketika untuk menguji *while statement*.

1. Uji ekspresi yang diberikan, Jika ekspresi bernilai benar / true /1 maka
   a. Jalankan pernyataan
   b. kembali ke langkah 1 untuk kembali menguji ekspresi setelah menjalankan pernyataan
2. else (jika ekspresi yang diuji bernilai salah / false / 0 maka keluar dari proses *looping* while do statement

Program 6.4 berikut mencoba mengimplementasikan *looping* dengan menggunakan struktur while.

Program 6.4. Program Pengulangan dengan Struktur while

Baris	C++	Python
1	#include <iostream>	angka=int(input(“angka: "))
2	Using namespace std;	print("jumlah deret dari: ", angka)
3		jumlah = 0
4	int main()	for i in range(1,angka+1,1):
5	{	jumlah = jumlah + i
6	int angka, jumlah, i;	print(jumlah)
7	cout << “Angkanya: “;	
8	cin >> angka	
9	Jumlah = 0	
10	i = 1;	
	while (i <= angka)	
11	{	

Baris	C++	Python
12	jumlah = jumlah + i;	
13	i++;	
14	}	
	cout << “Jumlah total = “	
	<< jumlah << endl;	
15	}	
Output		
Jumlah total = 15		

Pada Program 6.4 terlebih dahulu akan menginisialisasi variabel jumlah = 0, dan i = 1, program akan menampilkan perintah pada layar monitor kepada pengguna untuk memasukan angka yang akan dihitung jumlah deretnya, angka yang dimasukan oleh pengguna akan ditangkap oleh variabel *counter*. Pada instruksi selanjutnya program akan menguji dengan struktur pengulangan while. Pernyataan while (i <= counter) mengisyaratkan pengecekan apakah I yang bernilai 1 yang diketahui pada saat inisialisasi awal akan dibandingkan dengan nilai yang tertera pada variabel angka. Sebagai contoh pengguna memasukan angka 5, melalui pernyataan while akan diuji apakah 1 <= 5. Ketika logika dalam pernyataan tersebut bernilai benar maka instruksi jumlah = jumlah + i akan dijalankan, dari instruksi ini akan didapatkan hasil penjumlahan dari deret angka 1 sampai 5. Instruksi i++ mengisyaratkan variabel i melakukan proses increment dengan menambahkan nilai 1 setiap *looping*-nya. Sebagai contoh pada proses *looping* ke dua i akan bernilai 2, nilai ini kemudian akan dimasukan dalam pernyataan while (i <= counter) yang selanjutnya akan diuji apakah 2 <= 10, logika pernyataan tersebut masih bernilai benar maka program akan menjalankan instruksi di bawahnya. Instruksi i++ akan menyebabkan i bertambah dari 2 menjadi 3, nilai 3 tersebut kemudian diperiksa lagi dalam pernyataan while (i <= angka), demikian seterusnya hingga logika while (i <= angka) bernilai salah, sebagai contoh 6 <= 5. Ketika logika pernyataan bernilai salah maka proses *looping* akan berakhir.

LOOPING DENGAN MENGGUNAKAN STRUKTUR DO-WHILE

Secara khusus C++ mempunyai struktur *Looping* yang tidak dipunyai oleh Python yaitu struktur do while. Perbedaan dengan struktur

for dan while adalah keduanya akan memeriksa suatu kondisi pengulangan di awal program, sedangkan pada struktur do – while proses pemeriksaan kondisi berada di akhir sintaks *looping*.

for (ekspresi)	while (ekspresi)
…..	…..
……	……
……	……

Sedangkan struktur do while akan memeriksa kondisi pengulangan pada akhir program,

```
do
…….
…….
…….
While (ekspresi)
```

Struktur ini akan memungkinkan program untuk di jalankan minimal sekali sebelum pernyataan dari program tersebut diperiksa oleh kondisi **while**, jika kondisi yang diperiksa bernilai benar maka pernyataan di atas **while** akan di jalankan kembali hingga kondisi salah tercapai. Berikut ini adalah contoh penggunaan struktur do-while untuk menyelesaikan jumlah deret:

Program 6.5. Program Pengulangan dengan Struktur do-while

Baris	C++
1	#include <iostream>
2	Using namespace std;
3	
4	main()
5	{
6	int counter, jumlah, i;
7	
8	jumlah = 0; i = 1;
9	cout << "Masukan angka: ";
10	cin >> counter;
11	do

Baris	C++
12	{
13	jumlah = jumlah + i;
14	cout << jumlah;
	i++;
	}
	while(i<=counter);
	cout << jumlah << endl;
15	}
Output	
15	

Program 6.5 menunjukkan penggunaan struktur do while pada program untuk menghitung jumlah deret. Secara prinsip hasil keluaran yang dihasilkan antara struktur for, while, dam do while mempunyai nilai keluaran yang sama, akan tetapi dari ketiga struktur tersebut struktur tersebut struktur do while akan mengerjakan proses *looping* minimal satu kali meski setelah diperiksa ekspresinya bernilai False.

BAB VII

FUNGSI (*FUNCTION*)

Bab ini menjelaskan tentang konsep fungsi (*function*) atau dalam bahasa pemrograman yang lain disebut sebagai *routine*. Fungsi pada dasarnya adalah penggalang dari sebuah program. Ada beberapa tujuan pembuatan sebuah fungsi, di antaranya adalah untuk menyederhanakan kode program, menghindari penulisan kode program yang sama berulang-ulang, memecah program ke dalam beberapa bagian sehingga mudah dalam pembuatan dan pemeliharaan.

Di beberapa literatur ada yang membedakan antara pengertian fungsi dan *procedure*. *Procedure* diartikan sebagai suatu bagian program yang menjalankan tugas tertentu yang kemudian dipanggil oleh fungsi utama. Fungsi juga merupakan bagian dari program yang menjalankan tugas tertentu akan tetapi fungsi mempunyai nilai yang akan di kembalikan ke fungsi utama ketika di panggil. Secara umum struktur penamaan sebuah *procedure* dan fungsi adalah sebagai berikut:

`void Nama_prosedure (void)`	:	struktur umum penamaan *procedure*
`int nama_fungsi (tipe_data parameter)`	:	struktur umum penamaan fungsi

Procedure lebih merupakan sebuah fungsi yang tidak memiliki parameter keluaran. *Procedure* hanya digunakan untuk memecah program sehingga lebih mudah dalam pengelolaan dan pengembangan selanjutnya. *Procedure* lebih ditekankan pada pemecahan kode program komputer yang besar menjadi kode program yang lebih kecil-kecil. Proses pemecahannya ini juga dilakukan untuk memudahkan dalam pengembangan dan pemeliharaan program komputer itu sendiri. Pada proses pengembangan program dapat dikembangkan secara bertahap, setiap bagian kode program

dapat diperiksa apakah sudah berjalan dengan baik atau masih terdapat kesalahan.

Fungsi biasanya mempunyai parameter masukan atau keluaran yang nantinya akan dikeluarkan pada saat fungsi tersebut di panggil. Secara umum fungsi dapat dibedakan menjadi empat bentuk yaitu fungsi tanpa argument tanpa return, fungsi dengan argument tanpa return, fungsi tanpa argument dengan return dan fungsi dengan argument dengan return.

Sebagai pembanding awal program 7.1 merupakan program yang dikembangkan tanpa melibatkan fungsi. Program 7.1 ini akan menjadi rujukan bagaimana kita dapat mengembangkan program dengan melibatkan fungsi sehingga lebih efektif dan efisien dalam pengembangan dan pemeliharaannya.

Program 7.1. Program Tanpa Fungsi

Baris	C++	Python
1	#include <iostream>	p = 4
2	Using namespace std;	l = 5
3		
4	int main()	keliling = (2*p)+(2*l)
5	{	luas = p * l
6	int p = 4;	
7	int l = 5;	print(keliling)
8	int keliling, luas;	print(luas)
9		
10	keliling = (2*p)+(2*l);	
11	luas = p * l;	
12	cout << keliling << endl;	
13	cout << luas << endl;	
14	}	
Output		
	18 20	

Program 7.1 merupakan ilustrasi program yang dikembangkan tanpa menggunakan. Bentuk umum penulisan fungsi adalah seperti yang ditunjukkan pada program 7.2 berikut ini.

Program 7.2. Program Tanpa Fungsi

Baris	C++	Python
1	#include <iostream>	def nama_fungsi(argument):
2	Using namespace std;	… kode program
3	int nama_fungsi(argument);	…….
4	int main()	…….
5	{	
6	… Deklarasi variabel	
7	nama_fungsi;	nama_fungsi
8	… kode program	… kdoe program
9		
10	}	
11		
12	int nama_fungsi(argument)	
13	{	
14	kode program	
15	}	

Pada bahasa C++ sebelum kita membuat fungsi, terlebih dahulu harus mendefinisikan prototipe fungsi yang diletakkan sebelum sintaks int main(). Lain halnya dengan bahasa Python, sebuah fungsi bisa langsung dikodekan tanpa harus membuat prototipe fungsinya terlebih dahulu.

Struktur penulisan fungsi pada C++ menggunakan empat atusan yang akan dijelaskan pada bagian selanjutnya pada bab ini. Sementara itu, struktur penulisan fungsi pada Python didahului dengan *keyword* "def" kemudian diikuti dengan nama fungsinya.

FUNGSI TANPA ARGUMEN TANPA RETURN

Bentuk umum fungsi tanpa argument dan tanpa return adalah sebagai berikut:

Tabel 7.1 Sintaks Umum Fungsi tanpa Argument, tanpa Return

C++	Python
void nama_fungsi() { kode ….. …… }	fef nama_fungsi(): kode …. …….

Pengimplementasian struktur fungsi pada Tabel 7.1 dapat dilihat pada Program 7.1. Pada program tersebut pendefinisian fungsi didahului dengan *keyword* void kemudian diikuti dengan nama fungsi dalam hal ini keliling dan luas kemudian diakhiri dengan tanda kurung yang di dalamnya tidak mengandung argumen.

Program 7.1. Program tanpa Argumen tanpa Return

Baris	C++	Python
1	#include <iostream>	def keliling():
2	Using namespace std;	p = 4
3	void keliling();	l = 5
4	void luas();	k = (2*p)+(2*l)
5		print(k)
6	int main()	
7	{	def luas():
8	keliling();	p = 4
9	luas();	l = 5
10	}	L = p*l
11		print(L)
12	void keliling()	
13	{	keliling()
14	int p = 4;	luas()
15	int l = 5;	
16	int k	
17	k= (2*p)+(2*l)	
18	cout << k << endl;	
19	}	
20	void luas()	
21	{	
22	int p = 4;	
23	int l = 5;	
24	int L;	
25	L= p * l;	
26	cout << L << endl;	
27	}	

Output
18
20

FUNGSI DENGAN ARGUMEN TANPA RETURN

Bentuk umum fungsi dengan argument dan tanpa return adalah sebagai berikut:

Tabel 7.2 Sintaks Umum Fungsi dengan Argumen, tanpa Return

C++	Python
void nama_fungsi(argumen) { kode .... }	def nama_fungsi(agumen): kode ....

Pengimplementasian struktur fungsi pada Tabel 7.2 dapat dilihat pada Program 7.2. Pada program tersebut pendefinisian fungsi didahului dengan *keyword* void kemudian diikuti dengan nama fungsi dalam hal ini adalah "sisi" kemudian diikuti dengan tanda kurung yang di dalamnya terdapat argumen. Argumen pada dasarnya adalah variabel yang juga didefinisikan bentuk tipe datanya.

Program 7.2. Program dengan Argumen tanpa Return

Baris	C++	Python
1	#include <iostream>	def sisi(pj,lb):
2	Using namespace std;	keliling = (2*pj)+(2*lb)
3	void sisi(int pj, int lb);	luas = pj * lb
4		
5	int main()	print(keliling, luas)
6	{	
7	int p=4; int l = 5;	p = 4;
8		l = 5;
9	sisi(p,l);	
10	}	sisi(p,l)
11		
12	void sisi(int pj, int lb)	
13	{	
14	int keliling, luas;	
15	keliling=(2*pj)+(2*lb);	
16	Luas = pj * lb;	
17	k= (2*p)+(2*l);	
18	cout << keliling << endl;	
19	cout << luas << endl;	
20	}	

Baris	C++	Python
Output		
	18 20	

FUNGSI TANPA ARGUMEN DENGAN RETURN

Bentuk umum fungsi tanpa argument dan dengan return adalah sebagai berikut:

Tabel 7.3 Sintaks Umum Fungsi tanpa Argumen, dengan Return

C++	Python
Tipe data nama_fungs() { kode .... return(variabel) }	def nama_fungsi(): kode .... return(variabel)

Pengimplementasian struktur fungsi pada Tabel 7.3 dapat dilihat pada Program 7.3. Pada program tersebut pendefinisian fungsi didahului dengan pendefinisian tipe data kemudian diikuti dengan nama fungsi dalam hal ini adalah “sisilb” dan “sisilpj” kemudian diikuti dengan tanda kurung yang di dalamnya tidak terdapat argument apapun.

Program 7.3. Program tanpa Argumen dengan Return

Baris	C++	Python
1	#include <iostream>	def sisipj(): p = int(input("panjangnya: ")) return p
2	Using namespace std;	
3	int sisilb();	
4	int sisipj();	
5		def sisilb(): l = int(input("lebarnya: ")) return l
6	int main()	
7	{	
8	int kel, lu, lb, pj;	
9	lb = sisilb();	pj = sisipj()
10	pj = sisipj();	lb = sisilb()
11		keliling = (2*pj)+(2*lb)
12	kel = (2*pj)+(2*lb);	luas = pj * lb
13	lu = pj * lb;	print(keliling)
14	cout << “keliling” << kel;	Print(luas)
15	Cout << “luas” << lu;	

Baris	C++	Python
16	}	
17		
18	int sisipj()	
19	{	
20	int panjang;	
21	cout << “panjangnya “;	
22	cin >> panjang;	
	return panjang;	
23	}	
24		
25	int sisilb()	
26	{	
27	int lebar;	
28	cout << “lebarnya “;	
29	cin >> lebar;	
	return lebar;	
30	}	
Output		
18		
20		

Pada Program 7.3 ditampilkan sebuah implementasi program dengan melibatkan fungsi tanpa argument dengan return. *Keyword* return di sini berlaku sebagai kode yang memerintahkan bahwa hasil operasi dari fungsi akan dapat digunakan oleh fungsi yang lain, pada contoh di atas dimanfaatkan oleh fungsi main(). Perbedaan dengan C++ Python tidak memerlukan pendefinisian jenis tipe data yang akan dihasilkan oleh operasi dari fungsi tersebut. Seperti halnya pada bab pengenalan variabel dan tipe data, python mampu memahami nilai yang dimasukkan pada variabelnya. Pada Python seorang *programmer* bisa leluasa mengganti jenis variabel sesuai dengan jenis data yang dibutuhkan.

FUNGSI DENGAN ARGUMEN DENGAN RETURN

Bentuk umum fungsi tanpa argument dan dengan return adalah sebagai berikut:

Tabel 7.4 Sintaks Umum Fungsi dengan Argumen, dengan Return

C++	Python
Tipe data nama_fungs(argumen) { kode .... return(variabel) }	def nama_fungsi(argumen): kode .... return(variabel)

Pengimplementasian struktur fungsi pada Tabel 7.4 dapat dilihat pada Program 7.4. Pada program tersebut pendefinisian fungsi didahului dengan pendefinisian tipe data kemudian diikuti dengan nama fungsi dalam hal ini adalah “kel” dan “lu” kemudian diikuti dengan tanda kurung yang di dalamnya terdapat argumen.

Program 7.4. Program dengan Argumen dengan Return

Baris	C++	Python
1	#include <iostream>	def keliling(p, l):
2	Using namespace std;	kel = (2*p)+(2*l)
3	int kel(int pj, in lb);	return kel
4	int lu(int pj, int lb);	
5		def luas(p, l):
6	int main()	luas = p * l
7	{	return luas
8	int K, L, l, p;	
9	cout << “Panjanya: “;	pj = int(input("Panjangnya: "))
10	cin >> p;	lb = int(input("lebarnya: "))
11	cout << “Lebarnya: “;	K = keliling(pj,lb)
12	cin >> l	L = luas(pj,lb)
13		print(K)
14	K = kel(p,l);	print(keliling)
15	L = lu(p * l);	
16	cout << “keliling” << K;	
17	Cout << “luas” << L;	
18	}	
19		
20	int kel(int pj, int lb)	
21	{	

Baris	C++	Python
22	int keliling;	
23	keliling=(2*pj)+(2*lb);	
24	return keliling;	
25	}	
26		
27	int lu(int pj, int lb)	
28	{	
29	int luas;	
30	luas=pj * lb;	
31	return luas;	
32	}	

Output
18
20

Fungsi yang berbentuk dengan argument dengan return mempunyai jenis tipe data tertentu pada nilai keluaran yang diberikan oleh fungsi tersebut. Mengacu pada bentuk umum dari fungsi ini, pendefinisian fungsi diawali dengan menentukan jenis tipe data apa yang akan dikeluarkan oleh fungsi tersebut, contoh pada Program 7.4 jenis tipe data yang dikeluarkan oleh fungsi kel dan lu adalah integer.

Pada C++ proses pendefinisian jenis tipe data pada nilai keluaran fungsi sangat penting sekali dan merupakan hal yang mutlak, akan tetapi tidak demikian halnya dengan bahasa Python. Jika dilihat dari keempat bentuk fungsi tersebut secara umum antara C++ dan Python sama-sama mempunyai perilaku yang sama terhadap keempat bentuk fungsi tersebut.

PENGAYAAN

Pengayaan 1

Baris	C++	Python
1	#include <iostream>	?
2	using namespace std;	
3		
4	int kabisat(int th)	
5	int main()	
6	{	
7	int tahun	
8	cout << “Masukkan Tahun: “;	
9	cin >> tahun;	

Baris	C++	Python
11	kabisat(tahun); }	
12		
13		
14	int kabisat(int th)	
15	{	
16	If (th%4==0)	
17	cout << “Tahun Kabisat”;	
18	else	
19	cout << “Bukan Kabisat”;	
20	}	
Output		
....		

Pengayaan 2

Baris	C++	Python
1	?	def faktorial(fk):
2		if(fk==0):
3		f = 1
4		if(fk> 0):
5		f=1
6		for i in range (1,int(fk)+1):
7		f = f*i
8		return f
9		nilai = input("Angkanya? : ")
11		output = faktorial(int(nilai))
12		print(output)
13		
14		
15		
16		
17		
18		
19		
20		
21		
22		
23		
24		
25		
26		
27		
28		
29		
Output		
....		

Pengayaan 3

Baris	C++	Python
1	#include<iostream>	?
2	#include<math.h>	
3	using namespace std;	
4		
5	int main()	
6	{	
7	int n, nilai[10], m;	
8	float med;	
9		
10	cout << "masukkan banyaknya data: ";	
11	cin >> n;	
12		
13	for (int i=0; i<n; i++)	
14	{	
15	cout << "masukkan data ke-" << i << ": ";	
16	cin >> nilai[i];	
17	}	
18		
19	for (int i=0; i<n; i++)	
20	{	
21	if (n%2!=0)	
22	{	
23	m = n/2;	
24	med = nilai[m];	
25		
26	}	
27	if (n%2==0)	
28	{	
29	m = n/2;	
30	med = (nilai[m-1]+nilai[m]);	
31	med = med/2;	
32	}	
33	}	
34	cout << "median = " << med;	
35	}	
Output		
....		

DAFTAR PUSTAKA

[1] H. M. Pandey. *Object-Oriented Programming C++ Simplified*. New Delhi, India: University Science Press, 2008.

[2] D. Zak. *An Introduction to Programming with C++* (eighth edtion). Boston, USA: Cengage Learning, 2016.

[3] J. J. L. a. Bradley. *Teach Yourself C++ in 21 Days*. USA: Sams Publishing, 2005.

[4] S. R. Davis. *C++ fo Dummies*. New Jersey: John Wiley & Sons, Inc, 2014.

[5] J. Shovic and A. Simpson. *Python All-In-One fo Dummies 7 Books in One*. New Jersey: John Willey & Son, 2019.

[6] V. S. Code, Visual Studio Code, [Online]. Available: https://code.visualstudio.com/learn. [Accessed july 2021].

Konsep Pemrograman Komputer dengan C++ dan Python

Buku ini merupakan buku ajar yang digunakan sebagai salah satu rujukan dalam memahami materi pada mata kuliah Pemrograman Komputer Dasar untuk tengah semester pertama. Penekanan buku ini adalah pada konsep pemrograman yaitu bagaimana pembaca akan diantarkan bagaimana memahami alur instruksi atau kode program.

Konsep pemrograman yang dibahas pada buku ajar ini adalah pengenalan dan pemahaman terhadap variabel dan tipe data, tiga struktur dasar pemrograman yaitu struktur terurut, struktur percabangan, dan struktur pengulangan. Konsep terakhir yang dibahas pada buku ajar ini adalah tentang konsep fungsi (*function*).

Bahasa C++ dan Python menjadi bahasa pengantar yang digunakan pada buku ini untuk memahamkan konsep pengembangan sebuah program komputer. Penggunaan dua bahasa pemrograman yang disajikan pada buku ini dimaksudkan untuk memberikan ilustrasi bagaimana cara melakukan translasi antarbahasa pemrograman.

Feddy Setio Pribadi

Seorang pengajar pada Jurusan Teknik Elektro FT Unnes. Penulis menyelesaikan studi jenjang sarjana di Jurusan Teknik Elektro FT Unnes tahun 2002 dan studi pascasarjana S-2 tahun 2007 dan S-3 tahun 2019 di Jurusan Teknik Elektro FT UGM. Penulis tertarik pada pemrograman komputer, *data mining*, dan sistem cerdas.

Penerbit Deepublish (CV BUDI UTAMA)
Jl. Kaliurang Km 9,3 Yogyakarta 55581
Telp/Fax : (0274) 4533427
Anggota IKAPI (076/DIY/2012)
cs@deepublish.co.id
Penerbit Deepublish
@penerbitbuku_deepublish
www.penerbitdeepublish.com

Kategori : Pemrograman
ISBN 978-623-02-3295-4