

PEMROGRAMAN DASAR

(C2) KELAS X

Febrianto Alqodri

PT KUANTUM BUKU SEJAHTERA

PEMROGRAMAN DASAR

SMK/MAK Kelas X

© 2020

Hak cipta yang dilindungi Undang-Undang ada pada Penulis.

Hak penerbitan ada pada **PT Kuantum Buku Sejahtera**.

Penulis	: Febrianto Alqodri
Editor	: Muhammad Afian Musthofa
Desainer Kover	: Achmad Faisal
Desainer Isi	: Putri Ari Kristanti
Tahun terbit	: 2020
ISBN	: 978-623-7591-74-0

Diterbitkan oleh

PT Kuantum Buku Sejahtera

Anggota IKAPI No. 212/JTI/2019

Jalan Pondok Blimbing Indah Selatan X N6 No. 5 Malang - Jawa Timur

Telp. (0341) 438 2294, Hotline 0822 9951 2221;

Situs web: www.quantumbook.id

*Dilarang memperbanyak atau memindahkan sebagian atau seluruh isi buku ini dalam bentuk apapun, baik secara elektronis maupun mekanis, termasuk memfotokopi, merekam atau dengan sistem penyimpanan lainnya, tanpa izin tertulis dari **PT Kuantum Buku Sejahtera**.*

Daftar Isi

Kata Pengantar	v
Prakata	vi
Bab 1 Logika dan Algoritma	1
A. Pengenalan Logika dan Algoritma	2
B. Aspek Penting Dalam Algoritma	3
C. Ragam Struktur Algoritma	3
D. Metode Penyajian Algoritma	3
E. Flowchart Struktur Urut	9
F. Flowchart Struktur Percabangan	10
G. Flowchart Struktur Perulangan	13
Uji Kompetensi.....	21
Bab 2 Struktur Dasar Bahasa Pemrograman	27
A. Perangkat Lunak Bahasa Pemrograman	28
B. Struktur Bahasa Pemrograman C++	36
C. Uji Kompetensi	50
Bab 3 Tipe Data dan Operator	55
A. Penerapan Data, Variabel, dan Konstanta	56
B. Operasi Aritmatika dan Logika	62
Uji Kompetensi.....	73
Bab 4 Struktur Kontrol	77
A. Struktur Kontrol Percabangan atau Pengambilan Keputusan.....	78
B. Struktur Kontrol Perulangan	87
Uji Kompetensi.....	103
Bab 5 Penerapan Array	109
A. Pengenalan Array	110
B. Jenis-jenis Array	113
Uji Kompetensi.....	126
Bab 6 Penggunaan Fungsi	131
A. Pengenalan Fungsi	132
B. Cara Pemanggilan Fungsi.....	135
C. Jenis-jenis Fungsi.....	139
Uji Kompetensi.....	146
Bab 7 Graphical User Interface	153
A. Dasar-Dasar Graphical User Interface	154
B. Event Handling.....	162
C. Graphics Device Interface (GDI)	170
Uji Kompetensi.....	179

Bab 8 Teknik Debugging dan Paket Installer 183
 A. Teknik Debugging..... 184
 B. Pembuatan Paket Installer 188
 Uji Kompetensi..... 193

Daftar Pustaka 199
Biodata Penulis 201
Biodata Konsultan 202
Tim Kreatif..... 203

Kata Pengantar

Puji dan syukur kami panjatkan kepada Tuhan yang Maha Esa sehingga buku-buku bahan ajar terbitan PT Kuantum Buku Sejahtera dapat tersusun untuk memenuhi kebutuhan dunia pendidikan khususnya Sekolah Menengah Kejuruan di Indonesia untuk bekal menuju dunia industri, usaha, maupun akademik. Kualitas pendidikan erat kaitannya dengan buku sebagai media pengajaran dan pembelajaran bagi pendidik untuk menyampaikan ilmu pengetahuan kepada peserta didik sebelum terjun ke dunia industri, usaha, maupun akademik. Buku berperan penting agar peserta didik mudah memahami mata pelajaran yang sedang ditempuh. Selain itu, buku merupakan salah satu faktor untuk mengoptimalkan penyampaian informasi kepada peserta didik.

Peran pendidik sangat penting untuk meningkatkan dan menyesuaikan daya serap siswa dengan ketersediaan kegiatan pada buku ini yang sudah terintegrasi ke dunia usaha maupun industri. Para pendidik pun dapat lebih kreatif dalam mengajarkan ilmu yang akan disampaikan ke peserta didik. Implementasi pengajaran disesuaikan dengan kurikulum 2013 yang dirancang untuk meningkatkan kualitas peserta didik dari sisi ilmu pengetahuan dan keterampilan sesuai dengan kebutuhan dunia industri maupun dunia usaha serta penyikapan permasalahan yang akan mereka hadapi di dunia kerja.

Proses penyusunan buku ini telah dikaji mendalam oleh tim ahli yang kompeten di bidangnya dan sudah terintegrasi ke dunia industri maupun dunia usaha, karena PT Kuantum Buku Sejahtera telah bekerja sama dengan banyak perusahaan untuk pengembangan ilmu pengetahuan. Namun buku ini memerlukan saran dan kritik untuk perbaikan edisi selanjutnya, sehingga dapat terus dimanfaatkan sebagai peningkatan kualitas pendidikan di Sekolah Menengah Kejuruan.

Kami PT Wonokoyo Group mengucapkan terima kasih atas kerja sama berbagai pihak mulai dari penulis naskah, tim editor serta tim-tim yang berkontribusi untuk kesempurnaan isi buku. Harapan kami buku ini dapat ikut serta memajukan dunia pendidikan serta dapat membentuk generasi yang berkualitas.

PT Wonokoyo Group mendukung dan bekerja sama dengan PT Kuantum Buku Sejahtera untuk memajukan pendidikan di Indonesia terutama dalam pengembangan *skill* di SMK Seluruh Indonesia. Semoga buku ini bermanfaat.

Malang, Februari 2020

PT Wonokoyo Group

Prakata

Pembelajaran abad 21 atau yang disebut dengan pembelajaran yang menyiapkan siswa mampu beradaptasi pada segala lingkungan, terutama untuk saat ini dapat *survive* dalam Revolusi Industri 4.0. Pembelajaran ini memiliki karakteristik berupa: 1) pendekatan yang berpusat pada siswa, 2) siswa belajar untuk berkolaborasi, 3) upaya mempersiapkan siswa untuk memiliki sikap yang baik, dan 4) materi pembelajaran (menggunakan metode *STEM*).

STEM merupakan suatu pendekatan untuk menyelesaikan masalah menggunakan integrasi ilmu dalam bidang sains, teknologi, keteknikan, dan matematika. Metode ini merepresentasikan suatu pembelajaran yang sistematis, terstruktur, dan berdekatan dengan kehidupan sehari-hari. Penggunaan metode STEM dapat mempermudah siswa untuk mengintegrasikan multidisiplin ilmu untuk mengembangkan konsep, produk, proses, dan sistem yang mampu mempermudah kehidupan banyak manusia.

Buku *Pemrograman Dasar* ini ditujukan untuk kelas X yang disusun berdasarkan rancangan Kurikulum 2013 yang telah direvisi karena isinya yang lengkap, padat, informatif, dan mudah dipahami. Pada setiap materi disajikan dengan bahasa yang lugas, jelas, informatif, *best practice*, gambar yang ilustratif, dan soal latihan yang membangkitkan semangat siswa untuk mengerjakannya. Pokok-pokok bahasan meliputi:

- Bab 1 Logika dan Algoritma
- Bab 2 Struktur Dasar Bahasa Pemrograman
- Bab 3 Tipe data dan Operator
- Bab 4 Struktur Kontrol
- Bab 5 Penerapan *Array*
- Bab 6 Penerapan Fungsi
- Bab 7 *Graphical User Interface*
- Bab 8 Teknik Debugging dan Paket *Installer*

Semoga buku ini dapat bermanfaat bagi pendidikan dan dapat digunakan sebagai pedoman pembelajaran bagi guru-guru SMK di Indonesia. Penulis dengan senang hati menerima kritik dan saran dari pembaca untuk perbaikan kelak. Terima kasih.

Penulis

BAB

1

Logika dan Algoritma

Kompetensi Dasar

- 3.1 Menerapkan alur logika pemrograman komputer
- 4.1 Membuat alur logika pemrograman komputer

Tujuan Pembelajaran

1. Melalui kegiatan diskusi dan pemberian masalah siswa dapat menerapkan alur logika pemrograman komputer.
2. Melalui kegiatan rancang bangun siswa dapat menyajikan alur logika pemrograman komputer dengan bahasa natural.
3. Melalui kegiatan rancang bangun siswa dapat menyajikan alur logika pemrograman komputer dengan *pseudocode*.
4. Melalui kegiatan rancang bangun siswa dapat menyajikan alur logika pemrograman komputer dengan diagram alur (*flowchart*).
5. Melalui kegiatan rancang bangun siswa dapat menguji coba dan mengevaluasi rancangan alur penyajian logika pemrograman dengan bahasa natural, *pseudocode*, dan diagram alur (*flowchart*).
6. Melalui kegiatan rancang bangun siswa dapat memperbaiki rancangan alur penyajian logika pemrograman dengan bahasa natural, *pseudocode*, dan diagram alur (*flowchart*).
7. Melalui kegiatan presentasi siswa dapat mengomunikasikan hasil pembuatan algoritma dengan baik dan persuasif.

Ketika Anda berjalan-jalan pada suatu daerah yang ramai pengunjung seperti alun-alun, pantai, mal, atau tempat lainnya, biasanya Anda melihat suatu atraksi yang ditunjukkan oleh air mancur (yang terdapat di bagian tengah destinasi) untuk memikat hati para pengunjung yang melihatnya. Pernahkah Anda memikirkan mengapa air mancur tersebut dapat mengeluarkan air dengan pola tertentu yang dikombinasikan oleh warna lampu sehingga menjadi pertunjukan yang menakjubkan bagi pengunjung? Mekanisme keluarnya air mancur tersebut memiliki urutan langkah-langkah yang dilakukan berdasarkan periode tertentu. Urutan langkah-langkah sesuai periode tertentu tersebut merupakan salah satu contoh implementasi algoritma dalam kehidupan sehari-hari.



Gambar 1.1 Air mancur
Sumber: Aquilla Devina, t.t.

Pada bab ini Anda akan belajar lebih jauh tentang apa yang dimaksud dengan algoritma dan bagaimana cara merepresentasikan algoritma. Setelah itu Anda akan belajar membuat suatu algoritma dengan konteks kehidupan nyata.

A. Pengenalan Logika dan Algoritma

Saat pertama mempelajari bahasa pemrograman, kesan yang dipikirkan oleh seseorang yang baru mempelajari bahasa pemrograman adalah bagaimana menyelesaikan permasalahan tersebut melalui bahasa-bahasa pemrograman tertentu (misal C, C++, Java, PHP, Python, dan sebagainya). Padahal jika ditelusuri lebih lanjut pemikiran tersebut kurang tepat, karena inti permasalahannya bukan pada pemilihan bahasa pemrograman, melainkan bagaimana menemukan metode pemecahan masalahnya (pencarian solusi). Logika adalah suatu teknik berpikir logis untuk menyelesaikan suatu permasalahan. Selain pada teknik berpikir logis, fokus awal dalam metode pemecahan masalah adalah berkonsentrasi pada urutan instruksi (langkah-langkah) yang efektif dan efisien untuk diterjemahkan menjadi bahasa yang sederhana dan dapat dipahami oleh manusia (seolah-olah instruksi harus diberikan kepada manusia), bukan kepada komputer. Urutan atau langkah-langkah tersebut disebut sebagai algoritma, sehingga logika dan algoritma merupakan kesatuan yang tidak dapat dipisahkan dalam pemecahan suatu masalah.

“Algoritma merupakan suatu prosedur, urutan langkah-langkah atau tahapan-tahapan sistematis, jelas, dan logis untuk menyelesaikan permasalahan.” (Microsoft Press, 1998). Pendapat lain menyatakan bahwa algoritma merupakan alur pemikiran dalam menyelesaikan suatu pekerjaan yang dituangkan secara tertulis (Moh. Sjukani, 2005). Berdasarkan kedua pendapat tersebut, dapat disimpulkan bahwa algoritma merupakan

urutan langkah-langkah yang sistematis dan jelas untuk menyelesaikan suatu pekerjaan tertentu. Langkah-langkah tersebut dapat direpresentasikan dalam bentuk tulisan maupun gambar (simbol). Penulisan algoritma harus mudah dimengerti oleh orang lain agar tidak terjadi mispersepsi terhadap algoritma yang dikembangkan.

B. Aspek Penting dalam Algoritma

Menurut Knuth (1997) suatu algoritma mempunyai lima ciri penting seperti berikut ini.

1. **Finiteness.** Algoritma harus berhenti setelah mengerjakan sejumlah langkah tertentu atau bersifat terbatas.
2. **Definiteness.** Setiap langkah harus didefinisikan secara tepat, tidak boleh membingungkan (ambiguous).
3. **Input.** Sebuah algoritma memiliki nol atau lebih input yang diberikan kepada algoritma sebelum dijalankan.
4. **Output.** Sebuah algoritma memiliki satu atau lebih output, yang biasanya bergantung kepada input.
5. **Effectiveness.** Setiap algoritma diharapkan memiliki sifat efektif. Setiap langkah harus sederhana sehingga dapat dikerjakan dalam rentang waktu yang masuk akal.

C. Ragam Struktur Algoritma

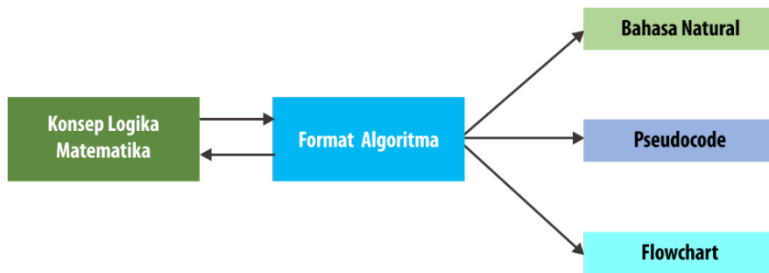
Terdapat tiga struktur dasar dalam penyajian algoritma seperti berikut ini.

1. Struktur runtut (*sequence*), artinya struktur algoritma tersebut berisi pernyataan-pernyataan secara berurutan atau *sequential*.
2. Struktur pemilihan atau percabangan (*selection*), artinya struktur algoritma tersebut menggunakan penyeleksian atau pemilihan kondisi.
3. Struktur perulangan (*iteration atau looping*), artinya struktur algoritma tersebut akan dieksekusi berulang-ulang sampai kondisi tertentu.

Tiga struktur algoritma di atas dapat digunakan satu struktur saja, ataupun kombinasi ketiga struktur tersebut. Representasi algoritma tidak menggunakan simbol-simbol (sintaks) dan notasi dari suatu bahasa pemrograman tertentu, melainkan bahasa yang bersifat umum.

D. Metode Penyajian Algoritma

Penyajian algoritma secara garis besar dapat dibedakan menjadi dua, yaitu berbentuk tulisan dan berbentuk gambar. Algoritma yang disajikan dengan bentuk tulisan dapat menggunakan aturan bahasa natural (alami) dan *pseudocode*. Bahasa natural menggunakan struktur bahasa tertentu (misalnya struktur bahasa Indonesia atau bahasa Inggris). *Pseudocode* adalah kode-kode tertentu yang mendekati kode dalam bahasa pemrograman (misal bahasa Pascal, C, C++, Java, Python, dan sebagainya), sehingga bermanfaat menggambarkan algoritma yang akan dikomunikasikan kepada pemrogram (*programmer*). Algoritma yang disajikan dengan gambar dapat berbentuk diagram alur (*flowchart*) atau struktogram. Hubungan dan jenis algoritma dapat dilihat pada Gambar 1.2.



Gambar 1.2 Hubungan dan jenis algoritma
Sumber: Febrianto Alqodri

1. Penyajian Algoritma Menggunakan Bahasa Natural (Alami)

Penyajian algoritma menggunakan bahasa natural (notasi alami) merupakan penyajian algoritma dengan menggunakan bahasa deskriptif yang jelas dan mudah dipahami manusia. Dasar dari penyajian bahasa deskriptif tersebut adalah bahasa Inggris, namun dapat dimodifikasi dengan bahasa sehari-hari termasuk bahasa Indonesia (karena tidak ada aturan baku). Hal ini dapat dibedakan karena teks algoritma tidak sama dengan teks program. Program merupakan implementasi algoritma dalam notasi bahasa pemrograman tertentu. Representasi kalimat deskriptif yang digunakan salah satunya adalah kata kerja, seperti tulis, baca, hitung, tampilkan, ulangi, bandingkan, dan lain-lain.

Pada dasarnya teks algoritma dengan bahasa deskriptif disusun dari tiga bagian utama, yaitu bagian judul (*header*), bagian deklarasi (kamus), dan bagian deskripsi. Bagian tersebut harus saling berkorelasi agar menjadi suatu teks algoritma yang runtut, jelas, dan logis. Setiap bagian disertai dengan komentar untuk memperjelas maksud teks yang dituliskan. Komentar tersebut adalah kalimat yang diapit oleh pasangan tanda kurung kurawal ('{' dan '}').

a) Judul algoritma

Judul Algoritma merupakan bagian yang terdiri atas nama algoritma dan penjelasan (spesifikasi) tentang algoritma tersebut. Bagian ini juga digunakan untuk menentukan apakah teks algoritma yang dibuat tersebut adalah program, prosedur, atau fungsi. Judul algoritma sebaiknya singkat namun cukup menggambarkan apa yang dilakukan oleh algoritma tersebut. Gambar 1.3 adalah contoh judul algoritma menghitung volume kubus disertai dengan penjelasan singkat.

Algoritma Volume_Tabung ← Judul Algoritma

{Menghitung volume tabung untuk ukuran tertentu. Algoritma menerima masukan berupa nilai jari-jari dan tinggi tabung, menghitung volumenya, lalu cetak volumenya ke piranti keluaran} ←Spesifikasi.

Gambar 1.3 Bagian Judul Algoritma Menghitung Volume Tabung
Sumber: Febrianto Alqodri

b) Bagian deklarasi

Di dalam algoritma, deklarasi atau kamus adalah bagian untuk mendefinisikan (menuliskan) semua nama yang akan dipakai di dalam algoritma. Nama tersebut dapat berupa nama variabel, nama konstanta, nama tipe, nama prosedur, dan nama fungsi. Penulisan pendefinisian deklarasi ini penting dilakukan karena

semua proses yang dilakukan oleh algoritma akan merujuk pada nilai yang disebutkan pada bagian ini. Pendefinisian dalam bagian deklarasi sebaiknya dikelompokkan menurut jenis dan nilainya masing-masing. Gambar 1.4 merupakan contoh bagian deklarasi algoritma untuk menghitung volume tabung.

Deklarasi:
 $r = \text{real \{tipe data bilangan pecahan\}}$
 $t = \text{real \{tipe data bilangan pecahan\}}$
 $v = \text{real \{tipe data bilangan pecahan\}}$
 $\text{PHI} = 3.14$

Gambar 1.4 Deklarasi algoritma menghitung volume tabung
Sumber: Febrianto Alqodri

c) Bagian deskripsi

Deskripsi adalah bagian inti dari struktur algoritma. Bagian ini berisi uraian langkah-langkah penyelesaian masalah. Langkah-langkah ini dituliskan dengan notasi yang lazim dalam penulisan algoritma. Setiap langkah algoritma dibaca secara urut dari langkah paling atas hingga langkah paling bawah. Suatu algoritma dapat terdiri dari tiga struktur dasar, yaitu runtunan, pemilihan, dan pengulangan. Gambar 1.5 merupakan contoh bagian deskripsi algoritma untuk menghitung volume tabung. Gambar 1.6 merupakan contoh penyajian algoritma secara lengkap untuk menghitung volume tabung menggunakan bahasa natural.

Deskripsi:
1. Masukan r
2. Masukan t
3. Baca PHI
2. Hitung $\text{volume} = \text{PHI} * r * r * t$
3. Tampilkan volume ke layar
4. Selesai

Gambar 1.5 Deskripsi algoritma menghitung volume tabung
Sumber: Febrianto Alqodri

Algoritma Volume Tabung

{Menghitung volume tabung untuk ukuran tertentu. Algoritma menerima masukan berupa nilai jari-jari dan tinggi tabung, menghitung volumenya, lalu cetak volumenya ke piranti keluaran}

Deklarasi:

r = real {tipe data bilangan pecahan}

t = real {tipe data bilangan pecahan}

v = real {tipe data bilangan pecahan}

$\text{PHI} = 3.14$

Deskripsi:

1. Masukan r
2. Masukan t
3. Baca PHI
2. Hitung volume = $\text{PHI} * r * r * t$
3. Tampilkan volume ke layar
4. Selesai

Gambar 1.6 Contoh penyajian algoritma menggunakan bahasa natural
Sumber: Febrianto Alqodri

2. Penyajian Algoritma Menggunakan *Pseudocode*

Pseudocode merupakan cara penyajian algoritma yang menyerupai bahasa pemrograman tingkat tinggi (bahasa pemrograman yang mudah dimengerti oleh manusia). *Pseudocode* menggunakan bahasa yang mudah dipahami secara umum dan lebih sederhana dibandingkan dengan algoritma. *Pseudocode* berisi urutan dari algoritma pemrograman komputer yang menggunakan struktur sederhana dari bahasa pemrograman, tetapi bahasa tersebut hanya ditujukan agar dapat dibaca manusia (tidak diprogramkan langsung pada komputer). Supaya notasi *pseudocode* dapat dipahami oleh komputer maka harus diterjemahkan terlebih dahulu menjadi sintaks bahasa pemrograman komputer tertentu.

Pseudocode tidak memiliki sintaks standar yang resmi, namun disarankan untuk menggunakan kata kunci yang umum digunakan dalam bahasa pemrograman seperti: *if, then, else, else if, while, do...while, repeat, go, for*, dan sebagainya. Tabel 1.1 menunjukkan perbandingan beberapa kata yang umum digunakan dalam penulisan algoritma dengan menggunakan kalimat deskriptif dan *pseudocode*.

Tabel 1.1 Perbandingan kata dalam algoritma dengan kalimat deskriptif dan *pseudocode*

Kalimat Deskriptif	<i>Pseudocode</i>
Masukkan jari-jari	Input jari-jari
	Read jari-jari
	Baca jari-jari
Hitung volume dengan rumus jari-jari x jari-jari x jari-jari	volume $\leftarrow$ jari-jari * jari-jari * jari-jari
Tampilkan volume	Output volume
	Print volume
	Write volume
Jika sudah selesai, cetak volume	if kondisi_end == true then print volume
Nilai A ditambah 5	A $\leftarrow$ A+5
Jika nilai A lebih kecil atau sama dengan 5, nilai B dibagi 2	if A $\leq$ 5 then B $\leftarrow$ B/2
Jika nilai A lebih besar atau sama dengan nilai B maka tampilkan nilai C, jika A lebih kecil dari B, tampilkan nilai B	if A $\geq$ B then print C else print B

Sumber: Febrianto Alqodri

Struktur penulisan *pseudocode* secara umum sama dengan struktur penulisan algoritma yang menggunakan kalimat deskriptif yaitu dimulai dari judul/header, deklarasi/kamus dan diakhiri dengan deskripsi. Gambar 1.7 adalah contoh *pseudocode* menentukan predikat nilai siswa.

Algoritma: Menentukan Predikat Nilai {Dibaca satu buah bilangan dari piranti masukan. Tentukan predikat yang cocok untuk nilai kelulusan yang dimasukan} Deklarasi: Nilai_Siswa : integer {variabel yang dimasukan} Batas_Minim : integer = 75 {variabel batas minimal lulus} Deskripsi: 1. Read (Nilai_Siswa) 2. If Nilai_siswa $\geq$ Batas_Minim then 3. Write ("Selamat, Nilai Anda Baik") 4. Else 5. Write ("Maaf, Nilai Anda Kurang Memuaskan")
--

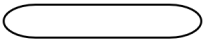
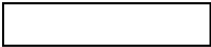
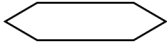
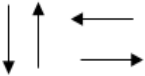
Gambar 1.7 Contoh *pseudocode* menentukan predikat nilai siswa
 Sumber: Febrianto Alqodri

3. Penyajian Algoritma Menggunakan Diagram Alur (*Flowchart*)

Flowchart adalah cara penulisan algoritma dengan menggunakan notasi grafis/gambar/bagan. Hal tersebut merepresentasikan urutan langkah dari suatu program yang memiliki hubungan antara proses satu dengan proses yang lain. Gambar/bagan *flowchart* ini dinyatakan dengan simbol yang menggambarkan suatu proses tertentu (simbolik) dan dihubungkan dengan suatu garis penghubung. *Flowchart* dapat membantu analis dan programmer untuk memecahkan masalah ke dalam segmen-segmen yang lebih kecil dan menolong dalam proses menganalisis alternatif solusi yang dapat digunakan. Selain itu, penggunaan *flowchart* bagi programmer bertujuan untuk menggambarkan runtutan instruksi yang diberikan kepada komputer, sedangkan bagi analis sistem *flowchart* digunakan untuk menggambarkan urutan tugas/pekerjaan dalam suatu prosedur atau operasi.

Pada dasarnya terdapat berbagai macam diagram alur (*flowchart*), di antaranya *flowchart* sistem (*system flowchart*), *flowchart* paperwork/*flowchart* dokumen (*document flowchart*), *flowchart* skematik (*schematic flowchart*), *flowchart* program (*program flowchart*), *flowchart* proses. Dalam kasus pembuatan program maka *flowchart* yang digunakan adalah *flowchart* program. *Flowchart* program menggambarkan urutan instruksi yang digambarkan dengan simbol tertentu untuk memecahkan masalah dalam suatu program. Tabel 1.2 menunjukkan simbol *flowchart* yang sering digunakan dalam pembuatan program.

Tabel 1.2 Ragam simbol yang sering dipakai dalam program *flowchart*

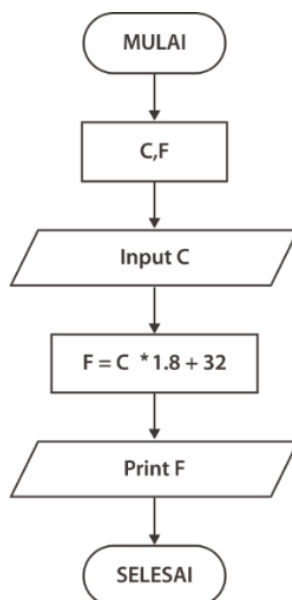
Notasi/Simbol/ Komponen	Nama Notasi/ Simbol	Keterangan
	Terminal Point	Untuk memulai dan mengakhiri proses.
	Input-Output	Untuk memberikan nilai data dari perangkat masukan atau menampilkan data ke perangkat keluaran.
	Processing	Untuk melakukan proses atau instruksi.
	Preparation	Untuk memberikan nilai awal variabel dan menyiapkan penyimpanan dalam storage.
	Decision/ Keputusan	Untuk memilih proses atau keputusan berdasarkan kondisi yang ada.
	Predefined Process/ Simbol Proses Terdefinisi	Untuk menunjukkan pelaksanaan suatu bagian prosedur (sub-proses).
	Flow Direction Symbol/ Simbol Arus	Untuk menghubungkan antara simbol yang satu dengan simbol yang lain (<i>connecting line</i>).

Notasi/Symbol/ Komponen	Nama Notasi/ Symbol	Keterangan
	Connector (On-Page)	Untuk menghubungkan dalam satu halaman.
	Connector (Off-Page)	Untuk menghubungkan berbeda halaman

Sumber: Febrianto Alqodri

E. Flowchart Struktur Urut

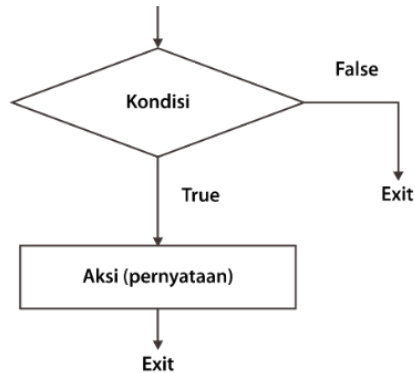
Flowchart struktur urut dilakukan untuk menyajikan suatu urutan instruksi yang harus dilakukan dalam memecahkan masalah tertentu secara berurutan. Proses pembacaan instruksi pasti dilakukan secara berurutan dan sesuai dengan urutan dan penulisannya (proses pembacaan dari atas ke bawah dan dari kiri ke kanan), dalam artian akan melakukan langkah 1 sebelum melakukan langkah 2. Kesalahan penulisan dalam struktur urutan dapat mengakibatkan kurang tepatnya output yang dihasilkan. Sebagai contoh perhatikan operasi aritmatika berikut ini, $(5+3)*7=45$, tetapi bila urutan aksinya diubah maka hasil outputnya akan berbeda menjadi $5+(3*7)=26$. Pada Gambar 1.8 disajikan *flowchart* struktur urut untuk mengonversi suhu dari Celcius ke Fahrenheit di mana suhu dalam satuan Celcius dimasukkan oleh pengguna dan setelah dilakukan proses kalkulasi secara otomatis akan menampilkan suhu dalam satuan Fahrenheit.



Gambar 1.8 Contoh *flowchart* struktur urut
Sumber: Febrianto Alqodri

1. Struktur Pemilihan atau Percabangan Satu Kasus

Algoritma menggunakan *flowchart* satu kasus ini hanya terdiri satu ekspresi kondisi dan satu pernyataan atau instruksi (aksi). Notasi yang digunakan untuk menyatakan struktur pemilihan satu kasus ini adalah satu buah belah ketupat dengan satu buah masukan dan dua buah keluaran. Sementara itu pernyataan atau instruksi yang akan dilakukan ditempatkan pada satu keluaran pemilihan, untuk kondisi terpenuhi (hasil ekspresi kondisi benar) atau kondisi tidak terpenuhi (hasil ekspresi kondisi salah). Gambar 1.9 merupakan contoh struktur *flowchart* satu kasus.



Gambar 1.9 Struktur pemilihan dengan satu 1 kasus
Sumber: Febrianto Alqodri

“Kondisi” yang ada pada simbol belah ketupat merupakan ekspresi kondisi yang terdiri dari operator yang meliputi: operator perbandingan (<, >, ==,) dan logika (OR, AND...), sedangkan operand dapat berupa variabel atau nilai variabel. Tabel 1.3 di bawah ini menjelaskan ragam contoh ekspresi kondisi.

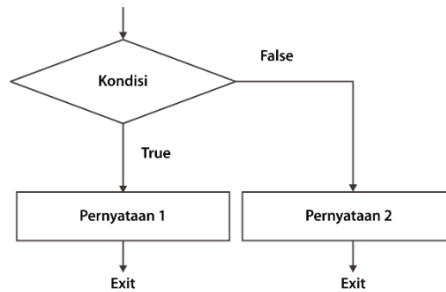
Tabel 1.3 Ragam jenis contoh ekspresi kondisi

Operand 1	Operand 2	Operator	Ekspresi	Hasil
1	4	<	1 < 4	false
4	2	>	5 > 2	true
A=5	B=5	==	A == B	true

Sumber: Febrianto Alqodri

2. Struktur Pemilihan atau Percabangan Dua Kasus

Algoritma menggunakan *flowchart* dua kasus ini hanya terdiri satu ekspresi kondisi dan dua pernyataan atau instruksi (aksi). Pernyataan atau instruksi yang akan dilakukan ditempatkan di dua keluaran pemilihan, yaitu satu pernyataan di kondisi yang terpenuhi (hasil ekspresi kondisi benar) dan satu pernyataan lainnya ditempatkan di kondisi yang tidak terpenuhi (hasil ekspresi kondisi salah). Struktur *flowchart* dua kasus diperlihatkan dalam Gambar 1.10.

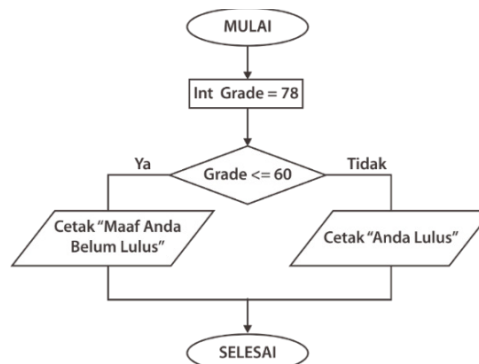


Gambar 1.10 Struktur pemilihan dengan dua kasus
Sumber: Febrianto Alqodri

Gambar 1.10 menjelaskan bahwa pernyataan 1 akan dieksekusi jika hasil ekspresi kondisi bernilai *true* atau benar dan pernyataan 2 akan dieksekusi jika hasil ekspresi kondisi bernilai *false* atau salah. Contoh kasus: Tulislah algoritma untuk menentukan ketuntasan atau kelulusan dari suatu nilai yang dimasukkan dari *keyboard* dengan ketentuan sebagai berikut.

Tabel 1.4 Predikat nilai dengan dua grade

Nilai	Grade
≤ 60	"Maaf Anda Belum lulus"
> 60	"Anda Lulus"

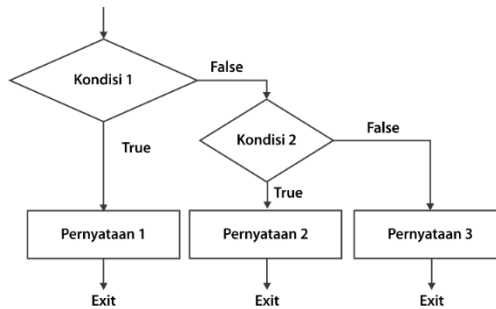


Gambar 1. 11 Flowchart penentuan kelulusan dengan dua grade
Sumber: Febrianto Alqodri

Gambar 1.11 menjelaskan *flowchart* yang digunakan untuk menentukan kelulusan dengan kriteria jika nilai Grade ≤ 60 maka algoritma akan menampilkan "Maaf Anda belum lulus", dan Jika nilai grade > 60 maka program akan menampilkan "Anda Lulus"

4. Struktur Pemilihan atau Percabangan Tiga Kasus

Algoritma menggunakan *Flowchart* dengan tiga kasus ini terdiri dari dua ekspresi kondisi dan tiga pernyataan atau instruksi (aksi). Notasi yang digunakan untuk menyatakan struktur pemilihan tiga kasus ini yaitu dua buah belah ketupat dengan satu buah masukan pada satu kondisi 1, satu keluaran kondisi 1 dan dua keluaran pada kondisi 2. Masing-masing keluaran memiliki satu pernyataan atau instruksi yang akan dieksekusi tergantung dengan hasil ekspresi kondisi 1 dan 2. Struktur *flowchart* tiga kasus diperlihatkan dalam Gambar 1.12.



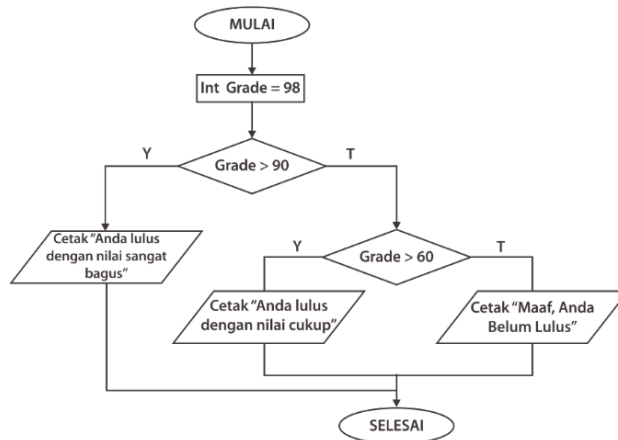
Gambar 1.12 Struktur pemilihan dengan tiga kondisi
Sumber: Febrianto Alqodri

Gambar1.12 menjelaskan pernyataan 1 akan dieksekusi jika hasil ekspresi kondisi 1 bernilai *true* atau benar. Pernyataan 2 akan dieksekusi jika hasil ekspresi kondisi 1 bernilai *false* atau salah dan hasil ekspresi kondisi 2 bernilai benar. Pernyataan 3 akan dieksekusi jika hasil ekspresi kondisi 1 bernilai *false* atau salah dan hasil ekspresi kondisi 2 bernilai salah. Contoh kasus: Tulislah algoritma untuk menentukan ketuntasan atau kelulusan dari suatu nilai dengan nilai diinputkan dari keyboard dengan kriteria berikut.

Tabel 1.4 Predikat nilai dengan tiga grade

Nilai	Grade
>90	"Anda lulus dengan nilai sangat bagus"
60 – 90	"Anda Lulus dengan nilai cukup"
<60	"Maaf Anda belum lulus"

Sumber: Febrianto Alqodri



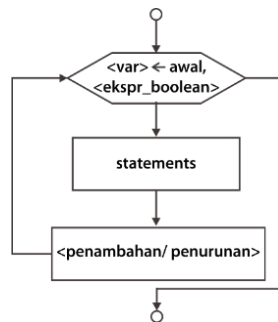
Gambar 1.13 Flowchart penentuan kelulusan dengan tiga grade
Sumber: Febrianto Alqodri

Gambar 1.13 menjelaskan *flowchart* yang digunakan untuk menentukan kelulusan dengan kriteria jika nilai Grade > 90 (lebih dari 90) maka algoritma akan menampilkan "Anda lulus dengan nilai sangat bagus", jika nilai grade antara 60 hingga 90 maka program akan menampilkan "Anda lulus dengan nilai cukup", dan jika nilai grade < 60 (kurang dari 60) maka program akan menampilkan "Maaf, Anda belum lulus".

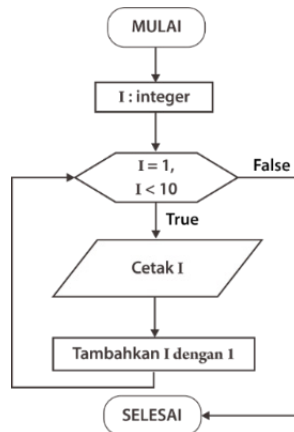
Struktur perulangan digunakan untuk mengulang suatu perintah sebanyak yang diinginkan tanpa harus menulis ulang. Instruksi perulangan yang umum digunakan dalam bahasa pemrograman antara lain adalah: *for*, *while*, dan *do-while*.

1. Struktur perulangan menggunakan instruksi *for*

Struktur perulangan *for* digunakan untuk mengulangi perintah dengan jumlah perulangan yang sudah diketahui. Pada *flowchart* dengan struktur ini perlu dituliskan nilai awal dari variabel, suatu kondisi untuk diuji yang berupa ekspresi *boolean* dan perintah yang dipakai untuk penghitung (*counter*). Nilai variabel penghitung akan secara otomatis bertambah atau berkurang setiap kali sebuah perulangan dilaksanakan tergantung perintah yang akan diberikan. Gambar 1.14 merupakan *flowchart* perulangan dengan instruksi *for*. Gambar 1.15 merupakan *flowchart* implementasi perulangan mencetak deret 1 sampai 9 menggunakan instruksi *for*.



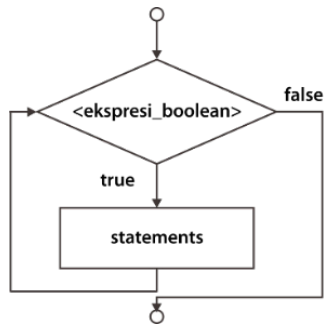
Gambar 1.14 Struktur *flowchart* menggunakan instruksi *for*
Sumber: Febrianto Alqodri



Gambar 1.15 *Flowchart* mencetak angka 1 sampai 9 menggunakan instruksi *for*
Sumber: Febrianto Alqodri

2. Struktur Perulangan Instruksi *WHILE*

Perintah *while* digunakan untuk mengulangi suatu perintah sampai kondisi tertentu. Perulangan akan terus berjalan selama kondisi masih bernilai "benar". Perulangan akan berhenti jika kondisi ekspresi bernilai "salah". *Flowchart* menggunakan struktur perulangan *while* ditunjukkan oleh Gambar 1.16.



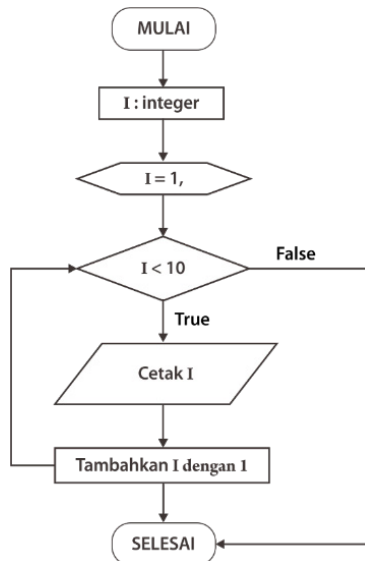
Gambar 1.16 Struktur *flowchart* menggunakan instruksi *while*
Sumber: Febrianto Alqodri

Contoh *flowchart* Gambar 1.17 akan menampilkan deret angka dari 1 sampai 9 dengan proses-prosesnya dapat dilihat dalam Tabel 1.5 berikut.

Tabel 1.5 Analisa *flowchart* cetak angka 1 sampai 9

i	i < 10	Hasil ekspresi i < 10	Keluaran
1	(1 < 10)	true	1
2	(2 < 10)	true	1 2
3	(3 < 10)	true	1 2 3
.	.	.	.
.	.	.	.
.	.	.	.
9	(9 < 10)	true	1 2 3 4 5 6 7 8 9
10	(10 < 10)	false	1 2 3 4 5 6 7 8 9

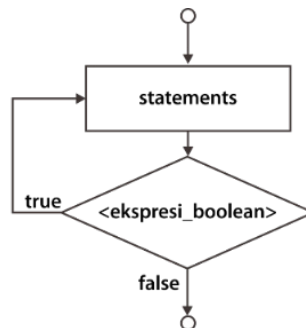
Sumber: Febrianto Alqodri



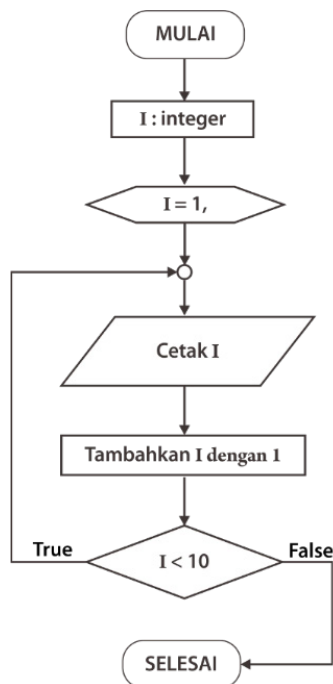
Gambar 1.17 *Flowchart* mencetak angka 1 sampai 9 menggunakan instruksi *while*
Sumber: Febrianto Alqodri

3. Struktur perulangan menggunakan instruksi **do-while**

Instruksi *do-while* merupakan proses perulangan yang akan berjalan jika ekspresi *boolean* masih bernilai "benar" dan perulangan akan dihentikan jika kondisinya sudah bernilai "salah". Perbedaannya dengan perulangan *while* yaitu jika perulangan *while* melakukan pengecekan kondisi di awal, namun perulangan *do-while* ini melakukan pengecekan di akhir perulangan. Gambar 1.18 merupakan *flowchart* perulangan menggunakan instruksi *do-while*. Gambar 1.19 merupakan *flowchart* perulangan mencetak deret angka 1 sampai 9 (seperti pada Tabel 6) menggunakan instruksi *do-while*.



Gambar 1.18 Struktur *flowchart* menggunakan instruksi *do-while*
Sumber: Febrianto Alqodri



Gambar 1.19 *Flowchart* mencetak deret angka 1 sampai 9 menggunakan instruksi *do-while*
Sumber: Febrianto Alqodri

4. Struktur perulangan menggunakan kombinasi perulangan (**nested loop**)

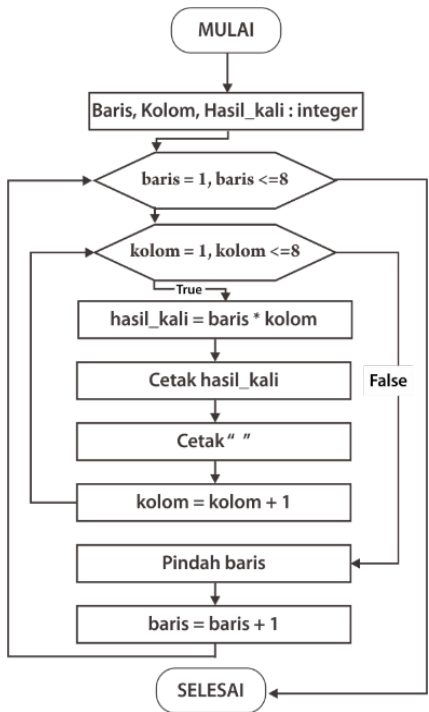
Dalam suatu perulangan dapat terdapat perulangan yang lain. Suatu struktur perulangan (*inner loop*) yang terdapat dalam perulangan lainnya (*outer loop*) disebut

dengan istilah *nested loop*. Salah satu contoh penerapan algoritma *nested loop* adalah algoritma untuk Tabel 1.6 perkalian sebagai berikut.

Tabel 1.6 Perkalian bilangan data integer

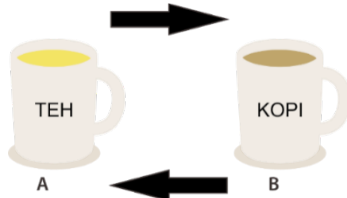
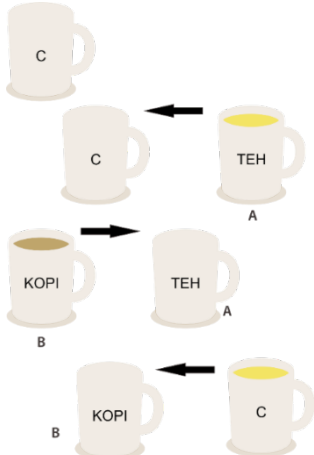
	1	2	3	4	5	6	7	8
1	1	2	3	4	5	6	7	8
2	2	4	6	8	10	12	14	16
3	3	3	9	12	15	18	21	24
4	4	8	12	16	20	24	28	32
5	5	10	15	20	25	30	35	40
6	6	12	18	24	30	36	42	48
7	7	14	21	28	35	42	49	56
8	8	16	24	32	40	48	56	64

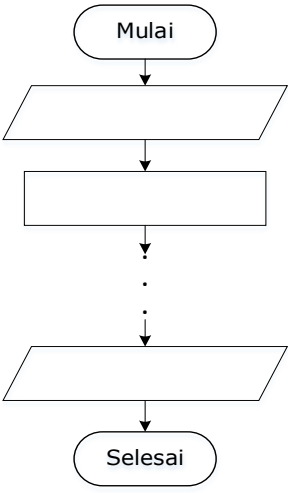
Sumber: Febrianto Alqodri



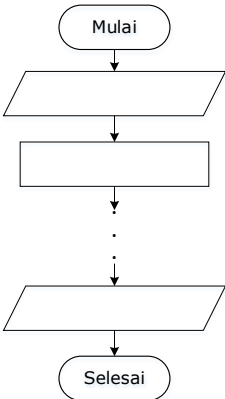
Gambar 1.20 Flowchart mencetak tabel perkalian 1 sampai 8
Sumber: Febrianto Alqodri

Tugas Kelompok (Lab. STEM)

KASUS 1	Membuat Algoritma Penukaran Data
A. Reflection - Permasalahan	
Diberikan dua buah gelas A dan B, gelas A berisi air teh dan gelas B berisi air kopi. Tukarkan isi gelas tersebut sehingga menghasilkan gelas A yang semula berisi air teh menjadi berisi air kopi dan gelas B yang semula berisi air kopi menjadi berisi air teh.	
B. Research	
Mengumpulkan informasi atau sumber bacaan yang terkait dengan penukaran data, salah satunya melalui ilustrasi seperti pada Gambar 1.21.	
 Gambar 1.21 Ilustrasi gelas berisi teh dan kopi Sumber: Febrianto Alqodri	
C. Discovery	
Salah satu cara menyelesaikan permasalahan penukaran data yaitu dengan mengikuti langkah-langkah seperti yang ditunjukkan dalam Gambar 1.22.	
 Gambar 1.22 Ilustrasi perpindahan isi Sumber: Febrianto Alqodri	
Input	Isi gelas A dengan “teh” dan isi gelas B dengan “kopi”
Output	Gelas A berisi “kopi” dan gelas B berisi “teh”.
Peralatan	Komputer / laptop Perangkat lunak microsoft word Perangkat lunak microsoft visio

D. Application	
1. Rancanglah algoritma penukaran data dengan penyajian:	
Bahasa Natural	Algoritma Penukaran Data {.....} Deklarasi : ... Deskripsi:
Pseudocode	Algoritma Penukaran Data {.....} Deklarasi: Deskripsi:
Flowchart	 <pre> graph TD A([Mulai]) --> B[/ /] B --> C[] C --> D[.] D --> E[/ /] E --> F([Selesai]) </pre> Gambar 1.23 Flowchart program Sumber: Febrianto Alqodri
Rancang ulang penyajian algoritma yang telah dibuat jika ada yang perlu disempurnakan. Uji kembali penyajian algoritma penukaran data. Buat laporan percobaan dengan format yang tersedia dan presentasikan.	
E. Communication	
Presentasikan penyajian algoritma penukaran data dengan bahasa natural, penyajian algoritma penukaran data dengan <i>pseudocode</i>, dan penyajian algoritma penukaran data dengan <i>flowchart</i>.	

KASUS 2	Membuat Algoritma Stop Kontak Listrik Pintar
A. Reflection - Permasalahan	
Salah satu ide project untuk <i>Internet of Thing</i> (IoT) adalah pembuatan stop kontak listrik pintar. Colokan listrik yang ada dihubungkan dengan Wi-Fi dan disambungkan dengan barang-barang elektronik seperti lampu atau oven. Jadi, Anda nantinya dapat menghidupkan maupun mematikan sambungan listrik dengan jangkauan internet pada ponsel yang terkoneksi ke colokan listrik tersebut. Dengan cara seperti ini ketika berada di luar rumah dan lupa mematikan lampu atau barang elektronik, Anda hanya perlu membuka ponsel dan mematikan koneksi listriknya dari jarak jauh.	
B. Research	
Mengumpulkan informasi atau sumber bacaan yang terkait dengan stop kontak listrik pintar melalui ilustrasi permasalahan stop kontak listrik pintar seperti dalam gambar 1.24. Gambar 1.24 Analogi Stop Kontak Listrik Pintar berbasis IoT Sumber: Febrianto Alqodri	
C. Discovery	
Stop kontak pintar ini berbasiskan modul Wifi di mana modul wifi tersebut dapat berfungsi sebagai wifi hotspot ataupun wifi client. Fungsi modul wifi tersebut dapat berfungsi sebagai wifi hotspot ataupun wifi client, sehingga untuk implementasi di stop kontak, wifi ini akan memiliki 2 mode pemakaian: 1. Mode Stop kontak sebagai wifi hotspot Dengan mode stop kontak sebagai wifi hotspot maka device yang akan terhubung dengan stop kontak wifi ini harus join ke wifi hotspot dari stop kontak wifi ini agar dapat mengakses IP address dari wifi hotspot. Stop kontak wifi hanya dapat diakses dalam jaringan lokal, dan tidak dapat diakses melalui jaringan internet karena jaringan wifi tersebut tidak terhubung ke internet. Untuk dapat menghidupkan ataupun mematikan perangkat elektronik dengan menekan icon pada android phone, gunakan setting sebagai berikut: a. Port1 disambungkan dengan kipas angin, b. Port2 disambungkan dengan lampu, dan c. Port3 disambungkan dengan TV. 2. Mode stop kontak sebagai wifi client Stop kontak ini harus diprogram untuk dapat bergabung kepada wifi hotspot yang akan dipergunakan (wifi hotspot harus memiliki koneksi internet agar stop kontak dapat diakses melalui jaringan internet), setelah stop kontak wifi terhubung ke jaringan hotspot maka langkah selanjutnya adalah mencari <i>IP address</i> dari stop kontak wifi tersebut dengan cara mencoba untuk mengakses <i>IP address</i> yang berada dalam satu subnet dengan android phone yang dipergunakan untuk mengontrol stop kontak. Untuk dapat menghidupkan ataupun mematikan perangkat elektronik dengan menekan icon pada android phone, gunakan seting sebagai berikut: a. Port1 disambungkan dengan kipas angin, b. Port2 disambungkan dengan lampu, dan c. Port3 disambungkan dengan TV.	
Input	Pemilihan icon kipas angin, lampu, dan TV.

Output	Terdapat 2 mode akses stop kontak pintar, mode 1 untuk akses dalam jaringan lokal, dan mode 2 untuk akses melalui internet dengan android phone.
Peralatan	Komputer/laptop Perangkat lunak microsoft word Perangkat lunak microsoft visio
D. Application	
1. Rancanglah algoritma stop kontak pintar:	
Bahasa Natural	Algoritma Stop Kontak Pintar {.....} Deklarasi: ... Deskripsi:
<i>Pseudocode</i>	Algoritma Stop Kontak Pintar {.....} Deklarasi: Deskripsi:
<i>Flowchart</i>	 Gambar 1.25 Flowchart Stop Kontak Listrik Pintar berbasis IoT Sumber: Febrianto Alqodri)
Rancang ulang penyajian algoritma yang telah dibuat jika ada yang perlu disempurnakan. Uji kembali penyajian algoritma stop kontak pintar. Buat laporan percobaan dengan format yang tersedia dan presentasikan.	
E. Communication	
Presentasikan penyajian algoritma stop kontak pintar dengan bahasa natural, penyajian algoritma stop kontak pintar dengan <i>pseudocode</i> , dan penyajian algoritma stop kontak pintar dengan <i>flowchart</i> .	

Rangkuman

Alur logika pemrograman yang dikenal dengan nama algoritma merupakan suatu prosedur, urutan langkah-langkah, atau tahapan-tahapan sistematis yang jelas dan logis untuk menyelesaikan suatu permasalahan. Penyajian algoritma dapat dibedakan menjadi dua yaitu berbentuk tulisan dan berbentuk gambar. Algoritma yang disajikan dengan bentuk tulisan dapat menggunakan aturan bahasa natural (alami) atau *pseudocode*, sedangkan algoritma yang disajikan dengan gambar dapat berbentuk diagram alur (*flowchart*) atau struktogram (*Nassi Schneiderman*). Langkah-langkah dalam algoritma dapat berupa struktur runtunan aksi (*sequence*), pemilihan aksi (*selection*), pengulangan aksi (*iteration* atau *looping*) atau kombinasi dari ketiganya.

Uji Kompetensi

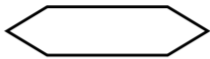
A. Soal Pilihan Ganda

Pilihlah jawaban yang paling tepat.

1. Komponen alur algoritma (*flowchart*) yang digunakan untuk menjelaskan pelaksanaan suatu bagian proses (sub-proses) adalah
 - a. *preparation*
 - b. *predefined process*
 - c. *processing*
 - d. *terminal point*
 - e. *connector*
2. Bagian teks algoritma *pseudocode* sebagai tempat untuk mendefinisikan nama tipe data, konstanta, dan variable adalah
 - a. *header*
 - b. *description*
 - c. *comment*
 - d. *dictionary*
 - e. *declaration*
3. Alur logika pemrograman atau algoritma harus berhenti setelah mengerjakan sejumlah langkah tertentu, aspek tersebut dikenal dengan istilah
 - a. *effectiveness*
 - b. *finiteness*
 - c. *definiteness*
 - d. *completeness*
 - e. *finishness*
4. Penulisan algoritma yang menyerupai bahasa pemrograman tingkat tinggi adalah

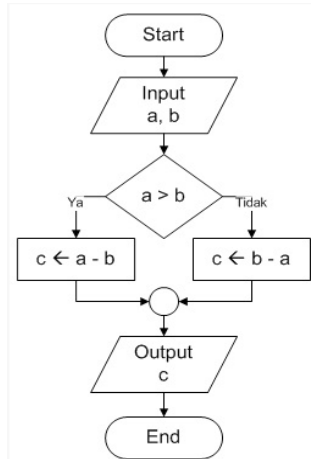
a. <i>natural</i>	d. <i>scriptcode</i>
b. <i>pseudocode</i>	e. <i>flowchart</i>
c. <i>comment</i>	

5. Berikut merupakan instruksi perulangan yang umum digunakan dalam bahasa pemrograman, kecuali
 - a. for
 - b. while
 - c. do ...while
 - d. do ... for
 - e. while...do
6. Jenis perulangan yang dapat digunakan jika diketahui banyaknya perulangan yang akan dilakukan yaitu....
 - a. do - while
 - b. if - else
 - c. for
 - d. switch case
 - e. do-for
7. Berikut ini yang bukan bagian struktur algoritma adalah....
 - a. iterasi
 - b. seleksi
 - c. sekuensial
 - d. rapid
 - e. skematik
8. Langkah-langkah yang digunakan sebagai bentuk pemecahan masalah dalam penciptaan program disebut....
 - a. *flowchart*
 - b. algoritma
 - c. program
 - d. *pseudocode*
 - e. deklarasi
9. Simbol berikut memiliki fungsi sebagai....



- a. masukan
 - b. keputusan
 - c. variabel
 - d. proses
 - e. keluaran
10. Banyaknya pernyataan yang dapat dieksekusi dalam sebuah *flowchart* adalah....
 - a. 5
 - b. 4
 - c. 3
 - d. 2
 - e. 1

11. Penulisan *pseudocode* hasil konversi *flowchart* di bawah ini adalah....



- a. INPUT a,b
IF a>b THEN
c = b-a
ENDIF
OUTPUT c
- b. INPUT a,b
IF a>b THEN
c = b-a;
ENDIF
OUTPUT a,b
- c. INPUT a,b
IF a>b THEN
c = b-a;
ENDIF
OUTPUT c
- d. INPUT a,b
IF a>b THEN
b-a = c
ENDIF
OUTPUT c
- e. INPUT a,b
IF a>b THEN
c = a-b
ENDIF
OUTPUT c

12. Output yang dihasilkan oleh sintaks berikut adalah....

```
int x = 1;
do
{
    printf("%i", x);
    x++;
}
while (x<0);
```

- a. 1
 - b. 1 2
 - c. 0 1 2
 - d. 0 1
 - e. 1 1 2 2
13. Berikut ini merupakan operator hubungan yang benar, **kecuali**....
- a. >=
 - b. ||
 - c. !=
 - d. ==
 - e. <=
14. Langkah yang benar dalam penyelesaian masalah adalah....
- a. definisi masalah – membuat model – rancang algoritma – tulis program – eksekusi – compile – dokumentasi
 - b. definisi masalah – membuat model – rancang algoritma – dokumentasi – tulis program – compile – eksekusi
 - c. definisi masalah – rancang algoritma – membuat model – tulis program – compile – dokumentasi – eksekusi
 - d. definisi masalah – memuat model – rancang algoritma – tulis program – compile – eksekusi – dokumentasi
 - e. definisi masalah – rancang algoritma – membuat model – tulis program – eksekusi – compile – dokumentasi
15. Perintah yang digunakan jika program menginginkan pengecekan kembali suatu kondisi sebelum langkah selanjutnya dikerjakan yaitu....
- a. if – else
 - b. while
 - c. switch case
 - d. do – while
 - e. for

B. Soal Esai

Jawaban dengan tepat dan benar.

1. Jelaskan perbedaan logika dan algoritma!
2. Apa ciri khas suatu algoritma?
3. Kapanakah notasi sub rutin pada *flowchart* dapat digunakan?
4. Jelaskan kondisi apa yang dapat menyatakan tugas algoritma telah selesai!
5. Gambarkan notasi *flowchart* untuk struktur dasar *sequential*!
6. Bagaimana jika terdapat algoritma yang tidak dapat diterjemahkan ke dalam *source code* program?
7. Sebutkan macam-macam notasi algoritma!
8. Uraikan kesalahan yang sering muncul pada penggunaan algoritma!
9. Berikan contoh sederhana penulisan deklarasi pada *pseudocode*!
10. Jelaskan mengapa pada *flowchart* tidak lagi menggunakan operator penugasan dalam memberikan perintah?

C. Soal Esai Uraian

Jawaban dengan ringkas dan benar.

1. Suatu algoritma terdiri dari tiga struktur dasar, yaitu runtunan, pemilihan, dan perulangan. Jelaskan masing-masing!
2. Buatlah algoritma untuk menghitung luas kerucut! Algoritma dibuat dalam bentuk kalimat deskriptif, *pseudocode*, dan *flowchart*.
3. Buatlah algoritma dalam bentuk kalimat deskriptif, *pseudocode* dan *flowchart* untuk menentukan kelayakan seseorang yang menjalani tes untuk masuk ke dalam suatu sekolah dengan nilai minimal 70 pada setiap tesnya! Di mana tes tersebut dilakukan secara berjenjang, misal gagal di salah satu tes maka otomatis gugur, misal seseorang tersebut lolos semua tes maka tampilkan "Selamat Anda Lulus", dan jika seseorang tersebut tidak lolos maka tampilkan "Maaf, Anda belum lulus".
4. Buatlah *flowchart* untuk menampilkan angka genap dari bilangan prima Antara 1-100!
5. Analisislah potongan algoritma di bawah ini! Apakah tujuan algoritma ini? Selanjutnya ubahlah potongan algoritma berikut ini menjadi sebuah *flowchart*.

```
1. const PI := 3.14
2. var radius, volume : real
3. put "please enter radius..."..
4. get radius
5. volume := (4/3)*PI*radius**3
6. put ""
7. put "The volume is", volume
```