



# **PEMROGRAMAN DASAR (C2) KELAS X**

**Penulis :  
Eko Subiyantoro, S.Pd, S.ST, M.T**

**PT. KUANTUM BUKU SEJAHTERA**

# PEMROGRAMAN DASAR

## SMK/MAK Kelas X

Penulis	: Eko Subiyantoro, S.Pd, S.ST, M.T
Editor	: Tim Quantum Book
Perancang sampul	: Tim Quantum Book
Perancang letak isi	: Tim Quantum Book
Penata letak	: Tim Quantum Book
Ilustrator	: Tim Quantum Book
Tahun terbit	: 2019
ISBN	: 978-623-7398-30-1
Alamat	: Jl. Pondok Blimbing Indah Selatan X N6 No 5 Malang - Jawa Timur

Tata letak buku ini menggunakan program Adobe InDesign CS3, Adobe Illustrator CS3, dan Adobe Photoshop CS3.

Font isi menggunakan Myriad Pro (10 pt)

B5 (17,6 × 25) cm

vi + 168 halaman

© Hak cipta dilindungi oleh undang-undang.  
Dilarang menyebarluaskan dalam bentuk apapun  
tanpa izin tertulis

Undang-Undang Republik Indonesia Nomor 19 Tahun 2002 Tentang Hak Cipta Pasal 72 Ketentuan Pidana Sanksi Pelanggaran.

1. Barang siapa dengan sengaja dan tanpa hak melakukan perbuatan sebagaimana dimaksud dalam Pasal 2 ayat (1) atau Pasal 49 ayat (1) dan ayat (2) dipidana dengan pidana masing-masing paling singkat 1 (satu) bulan dan/atau denda paling sedikit Rp1.000.000,00 (satu juta rupiah), atau pidana penjara paling lama 7 (tujuh) tahun dan/atau denda paling banyak Rp5.000.000.000,00 (lima miliar rupiah).
2. Barang siapa dengan sengaja menyiarkan, memamerkan, mengedarkan, atau menjual kepada umum suatu ciptaan atau barang hasil pelanggaran Hak Cipta atau Hak Terkait sebagaimana dimaksud pada ayat (1) dipidana dengan pidana penjara paling lama 5 (lima) tahun dan/atau denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah).

Pembelajaran abad 21 memiliki karakteristik atau prinsip-prinsip: 1) pendekatan pembelajaran berpusat pada siswa; 2) siswa dibelajarkan untuk mampu berkolaborasi; 3) Materi Pembelajaran (Menggunakan metode pembelajaran STEM) dikaitkan dengan permasalahan yang dihadapi dalam kehidupan sehari-hari, pembelajaran harus memungkinkan siswa terhubung dengan kehidupan sehari-hari mereka; dan 4) dalam upaya mempersiapkan siswa menjadi warga negara yang bertanggung jawab.

Salah satu pendekatan pembelajaran yang dapat mengakomodir karakteristik pembelajaran abad 21 tersebut adalah pendekatan *Science, Technology, Engineering, and Mathematics* atau disingkat dengan STEM. STEM merupakan suatu pendekatan di mana sains, teknologi, enjiniring, dan matematika diintegrasikan dengan fokus pada proses pembelajaran pemecahan masalah dalam kehidupan nyata. Pembelajaran STEM memperlihatkan kepada siswa bagaimana konsep-konsep, prinsip-prinsip sains, teknologi, enjiniring, dan matematika digunakan secara integrasi untuk mengembangkan produk, proses, dan sistem yang memberikan manfaat untuk kehidupan manusia.

Untuk menyiapkan siswa Indonesia memperoleh keterampilan abad 21, yaitu keterampilan cara berpikir melalui berpikir kritis, kreatif, mampu memecahkan masalah dan mengambil keputusan serta cara bekerja sama melalui kolaborasi dan komunikasi, maka pendekatan STEM diadopsi untuk menguatkan implelementasi Kurikulum 2013. Pendekatan STEM diyakini sejalan dengan ruh Kurikulum 2013 yang dapat diimplementasikan melalui penggunaan model pembelajaran berbasis proyek (*Project Based Learning*).

Buku Pemrograman Dasar Kelas X ini disusun berdasarkan tuntutan paradigma pengajaran dan pembelajaran kurikulum 2013 dan dipakai sebagai sumber belajar karena isinya yang lengkap, padat informasi, dan mudah dipahami.

Pada setiap materi yang disajikan dengan bahasa yang lugas, ilustrasi gambar dan soal latihan. Serta tugas proyek untuk memudahkan dalam setiap memahami setiap pembahasan dari pembahasan umum ke pembahasan khusus. Pokok bahasan yang 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 Penggunaan Fungsi
- Bab 7 Graphics User interface
- Bab 8 Aplikasi Graphics User interface
- Bab 9 Teknik Debugging dan Paket Installer

Penulis

# Daftar Isi

<b>BAB 1</b>	<b>Logika dan Algoritma .....</b>	<b>1</b>
	A. Pengertian Algoritma.....	3
	B. Aspek Penting dalam Algoritma .....	3
	C. Ragam Struktur Algoritma.....	4
	D. Metode Penyajian Algoritma.....	4
	E. Flowchart Struktur Urut.....	9
	E. Flowchart Struktur Percabangan .....	10
	G. Flowchart Struktur Perulangan.....	13
	Uji Kompetensi.....	18
<b>BAB 2</b>	<b>Struktur Dasar Bahasa Pemrograman .....</b>	<b>21</b>
	A. Perangkat Lunak Bahasa Pemrograman.....	23
	B. Struktur Bahasa Pemrograman C++ .....	33
	Uji Kompetensi.....	44
<b>BAB 3</b>	<b>Tipe Data dan Operator .....</b>	<b>47</b>
	A. Penerapan Data, Variabel, dan Konstanta.....	49
	B. Operasi Aritmetika dan Logika .....	52
	Uji Kompetensi.....	62
<b>BAB 4</b>	<b>Struktur Kontrol .....</b>	<b>65</b>
	A. Struktur Kontrol Percabangan atau Pengambilan Keputusan .....	67
	B. Struktur Kontrol Perulangan .....	74
	Uji Kompetensi.....	82
<b>BAB 5</b>	<b>Penerapan Array.....</b>	<b>87</b>
	A. Deklarasi dan Inisialisasi Array .....	89
	B. Penerapan Array .....	91
	Lembar Kerja Siswa .....	97
<b>BAB 6</b>	<b>Penggunaan Fungsi .....</b>	<b>99</b>
	A. Penggunaan Fungsi .....	101
	B. Pemanggilan Fungsi .....	103
	Lembar Kerja Siswa .....	111
<b>BAB 7</b>	<b>Graphics User interface .....</b>	<b>113</b>
	A. Dasar-Dasar Graphics User interface .....	115
	B. Event Handling .....	126
	Lembar Kerja Siswa .....	135
<b>BAB 8</b>	<b>Aplikasi Graphics User interface .....</b>	<b>137</b>
	A. Graphics Desain Interface.....	139
	B. Aplikasi Graphics User interface.....	146
	Lembar Kerja Siswa .....	154

**BAB 9 Teknik Debugging dan Paket Installer ..... 155**  
A. Teknik Debugging ..... 157  
B. Pembuatan Paket Instaler..... 160  
Lembar Kerja Siswa ..... 163

**Daftar Pustaka ..... 164**  
**Glosarium ..... 165**  
**Biodata Penulis ..... 168**



**BAB**

**1**

# Logika dan Algoritma

## Kompetensi Dasar

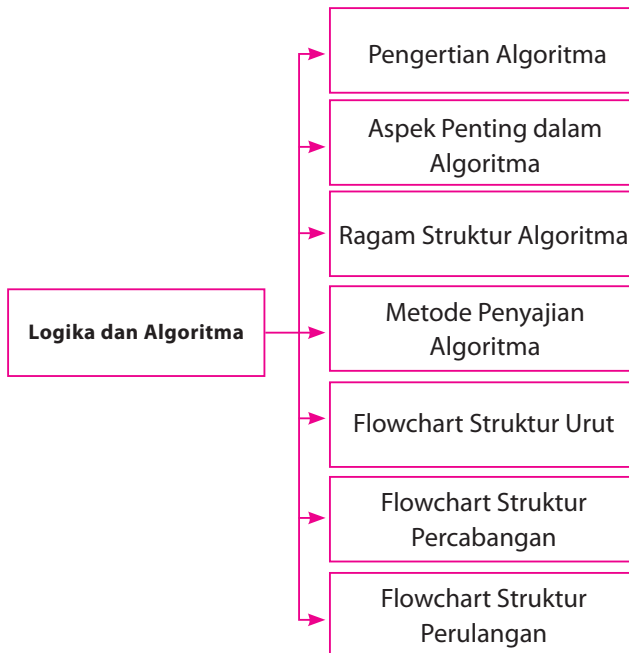
- 3.1 Menerapkan alur logika pemrograman komputer.
- 4.1 Membuat alur logika pemrograman komputer.

## Tujuan Pembelajaran

Setelah mempelajari bab ini, siswa diharapkan mampu:

1. menerapkan alur logika pemrograman komputer,
2. menyajikan alur logika pemrograman komputer dengan bahasa natural,
3. menyajikan alur logika pemrograman komputer dengan pseudocode,
4. menyajikan alur logika pemrograman komputer dengan flowchart,
5. menguji coba dan mengevaluasi rancangan alur penyajian logika pemrograman dengan bahasa natural, pseudocode, dan flowchart,
6. memperbaiki rancangan alur penyajian logika pemrograman dengan bahasa natural, pseudocode, dan flowchart, serta
7. mengkomunikasikan hasil pembuatan algoritma dengan baik dan persuasif.

## Peta Konsep





# Materi Pembelajaran

## A. Pengertian Algoritma

Saat pertama mempelajari bahasa pemrograman, mendapatkan kesan bahwa bagian sulit dari menyelesaikan masalah di komputer adalah menerjemahkan ide-ide ke dalam bahasa spesifik yang akan dimasukkan ke dalam komputer. Bagian tersulit dalam menyelesaikan masalah di komputer adalah menemukan metode solusinya. Setelah menemukan metode solusi, kemudian menerjemahkan metode ke dalam bahasa yang diperlukan, baik itu C++ atau bahasa pemrograman lainnya. Oleh karena itu sangat membantu untuk mengabaikan sementara waktu bahasa pemrograman dan berkonsentrasi pada merumuskan langkah-langkah solusi dan menuliskannya dalam bahasa yang sederhana. Seolah-olah instruksi harus diberikan kepada manusia bukan ke komputer. Urutan instruksi yang dinyatakan dengan cara ini sering disebut algoritma.

Istilah algoritma berasal dari seorang ilmuwan terkenal dari Persia yaitu Abu Ja'far Muhammad Ibnu Musa Al-Khuwarizmi. Al-Khuwarizmi dibaca orang barat menjadi Algorism. Al-Khuwarizmi menulis kitab atau buku yang berjudul *Al Jabr Wal-Muqabala* atau *The Book of Restoration and Reduction*. Dari buku tersebut diperoleh akar kata "Aljabar" (Algebra). Perubahan kata dari Algorism menjadi Algorithm muncul karena kata Algorism sering dikelirukan dengan *arithmetic*, sehingga akhiran -ism berubah menjadi -ithm.

Algoritma merupakan suatu prosedur, urutan langkah-langkah atau tahapan-tahapan sistematis, jelas, dan logis untuk menyelesaikan permasalahan. Langkah logis dalam algoritma harus dapat ditentukan, bernilai salah atau benar. Algoritma merupakan alur pemikiran dalam menyelesaikan suatu pekerjaan yang dituangkan secara tertulis. (Moh Sjukani, "Algoritma dan Struktur Data dengan C, C++, dan Java").

## B. Aspek Penting dalam Algoritma

Beberapa hal yang harus diperhatikan dalam membuat algoritma antara lain algoritma harus mengikuti suatu urutan aturan tertentu dan tidak boleh melompat-lompat. Algoritma seseorang dengan orang yang lain dapat berbeda-beda karena mempunyai alur pikir yang berbeda-beda pula, meskipun untuk menyelesaikan permasalahan yang sama. Langkah demi langkah secara eksak harus dapat memecahkan suatu masalah. Algoritma dapat diwujudkan dalam berupa kalimat, gambar atau tabel tertentu. Menurut Donald E. Knuth, algoritma harus mempunyai lima ciri penting, yaitu sebagai berikut.

1. *Finiteness*. Algoritma harus berhenti setelah mengerjakan sejumlah langkah tertentu atau 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 sejumlah waktu yang masuk akal.

### C. Ragam Struktur Algoritma

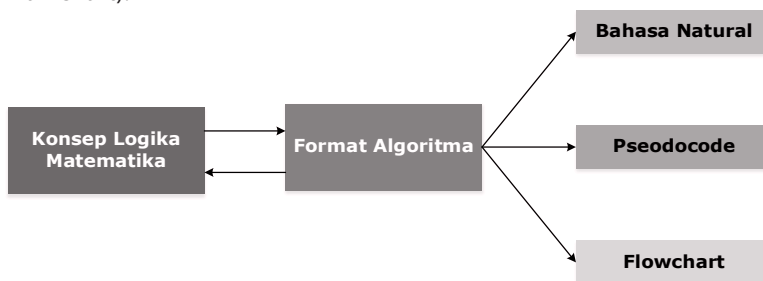
Algoritma berisi langkah-langkah penyelesaian suatu masalah. Langkah-langkah tersebut dapat berupa runtunan aksi (*sequence*), pemilihan aksi (*selection*), pengulangan aksi (*iteration* atau *looping*), atau kombinasi dari ketiganya. Struktur dasar penyajian algoritma dibedakan menjadi tiga, yaitu sebagai berikut.

1. Struktur runtunan (*sequence*), struktur algoritma yang mempunyai pernyataan secara berurutan atau sequential.
2. Struktur pemilihan atau percabangan (*selection*), struktur algoritma atau program yang menggunakan pemilihan atau penyeleksian kondisi.
3. Struktur perulangan, struktur algoritma atau program yang pernyataannya akan dieksekusi berulang-ulang sampai kondisi tertentu.

Pada algoritma, tidak dipakai simbol-simbol atau sintaks dari suatu bahasa pemrograman tertentu, melainkan bersifat umum dan tidak tergantung pada suatu bahasa pemrograman apapun juga. Notasi-notasi algoritma dapat digunakan untuk seluruh bahasa pemrograman mana pun.

### 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 dan mirip dengan kode bahasa pemrograman (misal bahasa Pascal, C, C++) sehingga lebih tepat digunakan untuk menggambarkan algoritma yang akan dikomunikasikan kepada pemrogram (programmer). Sedangkan algoritma yang disajikan dengan gambar dapat berbentuk diagram alur (flowchart) atau struktogram (Nassi Schneiderman chart).



Gambar 1.2 Hubungan dan jenis algoritma

## 1. Penyajian Algoritma Menggunakan Bahasa Natural (Alami)

Notasi penulisan algoritma dengan menggunakan bahasa deskriptif bisa juga disebut dengan notasi alami. Dilakukan dengan cara menuliskan instruksi-instruksi yang harus dilaksanakan dalam bentuk untaian kalimat deskriptif dengan menggunakan bahasa yang jelas. Dasar dari notasi bahasa deskriptif adalah bahasa Inggris, tetapi dapat dimodifikasi dengan bahasa sehari-hari termasuk Bahasa Indonesia. Karena tidak ada aturan baku dalam menuliskan algoritma dengan notasi deskriptif maka tiap orang dapat membuat aturan penulisan dan notasi algoritma sendiri. Hal ini dapat dimengerti karena teks algoritma tidak sama dengan teks program. Program adalah implementasi algoritma dalam notasi bahasa pemrograman tertentu. Tetapi, agar notasi algoritma mudah ditranslasi ke dalam notasi bahasa pemrograman, maka sebaiknya notasi algoritma tersebut berkoresponden dengan notasi bahasa pemrograman pada umumnya. Kata kerja adalah jenis kata yang biasa digunakan dalam penulisan bahasa deskriptif, contohnya tulis, baca, hitung, tampilkan, ulangi, bandingkan, dan lain-lain. Notasi jenis ini cocok untuk algoritma yang pendek. Tapi untuk masalah algoritma yang panjang, notasi ini kurang efektif. Cara penulisan algoritma dengan notasi bahasa deskriptif paling mudah dibuat, tetapi demikian cara ini paling sulit untuk diterjemahkan ke dalam bahasa pemrograman. Pada dasarnya teks algoritma dengan bahasa deskriptif disusun oleh tiga bagian utama yaitu: bagian judul (header), bagian deklarasi (kamus) dan bagian deskripsi.

Setiap bagian disertai dengan komentar untuk memperjelas maksud teks yang dituliskan. Komentar 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. Di bagian ini juga digunakan untuk menentukan apakah teks algoritma yang dibuat tersebut adalah program, prosedur, atau fungsi. Nama algoritma sebaiknya singkat tetapi cukup menggambarkan apa yang dilakukan oleh algoritma tersebut. Di bawah nama algoritma disertai dengan penjelasan singkat (intisari) tentang apa yang dilakukan oleh algoritma. Penjelasan di bawah nama algoritma sering dinamakan juga spesifikasi algoritma yang dituliskan dalam kurung kurawal ({}). Algoritma harus ditulis sesuai dengan spesifikasi yang didefinisikan. Gambar 2.1 adalah contoh judul algoritma menghitung luas lingkaran yang disertai dengan penjelasan singkat.

```
Algoritma Luas_Lingkaran ← Judul Algoritma {Menghitung luas lingkaran  
untuk ukuran jari-jari tertentu. Algoritma menerima masukan jari-jari lingkaran,  
menghitung luasnya, lalu cetak luasnya ke piranti keluaran} ← Spesifikasi.
```

### b. Bagian deklarasi

Di dalam algoritma, deklarasi atau kamus adalah bagian untuk mendefinisikan semua nama yang dipakai di dalam algoritma. Nama tersebut dapat berupa nama variabel, nama konstanta, nama tipe, nama prosedur atau nama fungsi. Semua nama tersebut baru dapat digunakan di dalam algoritma jika telah didefinisikan terlebih dahulu didalam bagian deklarasi. Penulisan sekumpulan nama dalam bagian deklarasi sebaiknya dikelompokkan menurut jenisnya. Pendefinisian nama konstanta sekaligus memberikan nilai konstanta. Pendefinisian nama fungsi atau

prosedur sekaligus dengan pendefinisian spesifikasi dan parameternya.

Deklarasi:

radius = real {tipe data bilangan pecahan}

luas = real {tipe data bilangan pecahan}

PHI = 3.14

**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 dari langkah paling atas hingga langkah paling bawah. Urutan penulisan menentukan urutan pelaksanaan perintah. Seperti yang telah dijelaskan bahwa penyusun atau struktur dasar algoritma adalah langkah-langkah. Suatu Algoritma dapat terdiri atas tiga struktur dasar, yaitu runtunan, pemilihan dan pengulangan. Ketiga jenis langkah tersebut membentuk konstruksi suatu algoritma. Pada bagian deskripsi inilah letak tiga struktur algoritma tersebut.

Deskripsi:

1. Baca radius
2. Hitung luas = radius \* radius \* PHI
3. Tampilkan luas ke layar
4. Selesai

Bentuk contoh lengkap penyajian algoritma dengan bahasa natural sebagai berikut.

Algoritma Luas Lingkaran

{Menghitung luas lingkaran untuk ukuran radius tertentu. Algoritma menerima masukan radius lingkaran, menghitung luasnya, lalu cetak luasnya ke piranti keluaran}

Deklarasi:

radius = real {tipe data bilangan pecahan}

luas = real {tipe data bilangan pecahan}

PHI = 3.14

Deskripsi:

1. Baca radius
2. Hitung luas = radius \* radius \* PHI
3. Tampilkan luas ke layar
4. Selesai

Gambar 1.3 Contoh penyajian algoritma menggunakan bahasa natural

## 2. Penyajian Algoritma Menggunakan Pseudocode

Pseudocode adalah cara penulisan algoritma yang menyerupai bahasa pemrograman tingkat tinggi. Pseudocode menggunakan bahasa yang hampir menyerupai bahasa pemrograman. Biasanya pseudo code menggunakan bahasa yang mudah dipahami secara universal dan juga lebih ringkas daripada algoritma. Pseudocode berisi deskripsi dari algoritma pemrograman komputer yang menggunakan struktur sederhana dari beberapa bahasa pemrograman tetapi bahasa tersebut hanya ditujukan agar dapat dibaca manusia. Sehingga pseudocode tidak dapat dipahami oleh komputer. Supaya

notasi pseudocode bisa dipahami oleh komputer maka harus diterjemahkan terlebih dahulu menjadi sintaks bahasa pemrograman komputer tertentu.

Dalam pseudocode, tidak ada sintaks standar yang resmi. Oleh karena itu, pseudocode ini dapat diterapkan dalam berbagai bahasa pemrograman. Disarankan untuk menggunakan keyword yang umum digunakan seperti *if*, *then*, *else*, *while*, *do*, *repeat*, *for*, dan lainnya. Keuntungan menggunakan notasi pseudocode adalah kemudahan mentranslasi ke notasi bahasa pemrograman, karena terdapat korespondensi antara setiap pseudocode dengan notasi bahasa pemrograman. Tabel 2.1. menunjukkan perbandingan beberapa kata yang biasa digunakan dalam penulisan algoritma dengan menggunakan kalimat deskriptif dan pseudocode.

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

Kalimat Deskriptif	Pseudocode
Masukkan panjang	<i>Input</i> panjang
	Read panjang
	Baca panjang
Hitung luas dengan rumus panjang x lebar	luas = panjang * lebar
Tampilkan luas	<i>Output</i> luas
	<i>Print</i> luas
	<i>Write</i> luas
Jika sudah selesai, cetak luas	if kondisi_selesai == true then <i>Print</i> luas
Nilai B ditambah 5	B ← B+5
Jika nilai A lebih kecil dari 5 maka nilai B dibagi 3	if A<5 then B ← B/3
Jika nilai A lebih besar dari nilai B maka tampilkan A, jika A lebih kecil dari B maka tampilkan nilai B	if A>B then <i>Print</i> A else <i>Print</i> B

Struktur penulisan pseudocode secara umum sama dengan struktur penulisan algoritma dengan menggunakan kalimat deskriptif yaitu dimulai dari judul/*header*, deklarasi/kamus dan diakhiri dengan deskripsi. Berikut contoh pseudocode menentukan bilangan terbesar dari 3 masukan bilangan

**Algoritma : Bilangan\_Maksimum**

{Dibaca tiga buah bilangan dari piranti masukan. Carilah bilangan bulat maksimum diantara ketiga bilangan tersebut}

**Deklarasi :**

Bil1,Bil2,Bil3: integer {bilangan yang dicari maksimumnya}

Max: integer {variabel bantu}

**Deskripsi:**

1. Read (Bil1,Bil2)
2. If Bil1 >= Bil2 then
3. Bil1 = Max
4. Else Bil2 = Max
5. Read (Bil3)

6. If Bil3 >= Max then
7. Bil3 = Max
8. Write (Max)

Gambar 1.4 contoh pseudocode menentukan bilangan terbesar dari 3 masukan bilangan

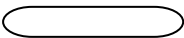
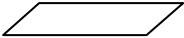
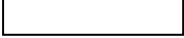
### 3. Penyajian Algoritma Menggunakan Flowchart

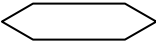
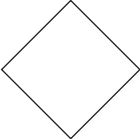

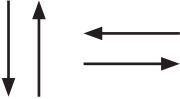
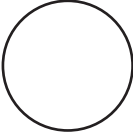

Flowchart adalah cara penulisan algoritma dengan menggunakan notasi grafis. Flowchart merupakan gambar atau bagan yang memperlihatkan urutan atau langkah-langkah dari suatu program dan hubungan antarproses beserta pernyataannya. Gambaran ini dinyatakan dengan simbol. Dengan demikian setiap simbol menggambarkan proses tertentu. Sedangkan antara proses digambarkan dengan garis penghubung. Dengan menggunakan flowchart akan memudahkan kita untuk melakukan pengecekan bagian-bagian yang terlupakan dalam analisis masalah. Di samping itu, flowchart juga berguna sebagai fasilitas untuk berkomunikasi antara pemrogram yang bekerja dalam tim suatu proyek. Flowchart menolong analis dan programmer untuk memecahkan masalah ke dalam segmen-segmen yang lebih kecil dan menolong dalam menganalisis alternatif lain dalam pengoperasian.

Pada dasarnya terdapat berbagai macam flowchart, di antaranya yaitu Flowchart Sistem (*System Flowchart*), Flowchart Paperwork/Flowchart Dokumen (*Document Flowchart*), Flowchart Skematik (*Schematic Flowchart*), Flowchart Program (*Program Flowchart*), Flowchart Proses (*Process Flowchart*). Untuk keperluan pembuatan program maka digunakan Flowchart Program.

Flowchart program menggambarkan urutan instruksi yang digambarkan dengan simbol tertentu untuk memecahkan masalah dalam suatu program. Dalam flowchart program mengandung keterangan yang lebih rinci tentang bagaimana setiap langkah program atau prosedur seharusnya dilaksanakan. Flowchart ini menunjukkan setiap langkah program atau prosedur dalam urutan yang tepat saat terjadi. Programmer menggunakan flowchart program untuk menggambarkan urutan instruksi dari program komputer. Analis Sistem menggunakan flowchart program untuk menggambarkan urutan tugas-tugas pekerjaan dalam suatu prosedur atau operasi.

**Tabel 1.2 Ragam simbol yang sering dipakai dalam Program Flowchart**

Notasi/Symbol/ Komponen	Nama Notasi/Symbol	Keterangan
	<i>Terminal Point</i>	Untuk memulai dan mengakhiri proses.
	<i>Input-Output</i>	Untuk memberikan nilai data dari perangkat masukan atau menampilkan data ke perangkat keluaran.
	<i>Processing</i>	Untuk melakukan proses atau instruksi.

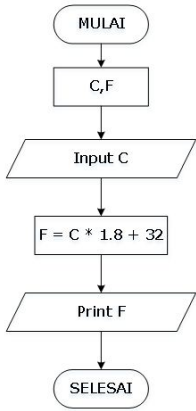
	<i>Preparation</i>	Untuk memberikan nilai awal variabel, menyiapkan penyimpanan dalam storage biasanya menggunakan kode program.
	<i>Decision/keputusan</i>	Untuk memilih proses atau keputusan berdasarkan kondisi yang ada. Simbol ini biasanya ditemui pada flowchart program.
	<i>Predefined Process/Symbol Proses Terdefinisi</i>	Untuk menunjukkan pelaksanaan suatu bagian prosedur ( <i>sub-proses</i> ). Dengan kata lain, prosedur yang terinformasi di sini belum detail dan akan dirinci di tempat lain.
	<i>Flow Direction Simbol/ Simbol Arus</i>	Untuk menghubungkan antara simbol yang satu dengan simbol yang lain ( <i>connecting line</i> ). Simbol ini juga berfungsi untuk menunjukkan garis alir dari proses.
	<i>Connector (On-page)</i>	Untuk menyederhanakan hubungan antar simbol yang letaknya berjauhan atau rumit bila dihubungkan dengan garis dalam satu halaman.
	<i>Connector (Off-page)</i>	Sama seperti <i>On-page Connector</i> , untuk menghubungkan simbol dalam halaman berbeda. label dari simbol ini dapat menggunakan huruf atau angka.

## E. Flowchart Struktur Urut

Sebuah runtunan terdiri atas satu atau lebih instruksi. Tiap instruksi dikerjakan secara berurutan sesuai dengan urutan penulisannya, yakni sebuah instruksi dilaksanakan setelah

instruksi sebelumnya selesai dikerjakan. Urutan dari instruksi menentukan hasil akhir dari suatu algoritma. Bila urutan penulisan berubah maka mungkin juga hasil akhirnya berubah. Sebagai contoh perhatikan operasi aritmateka berikut ini,  $(4+3)*7=49$ , tetapi bila urutan aksinya diubah maka hasil keluaran akan berbeda menjadi  $4+(3*7)=25$ .

Contoh kasus: Buatlah flowchart untuk mengonversi suhu dari Celcius ke Fahrenheit, Suhu dalam satuan Celcius dimasukkan oleh pengguna. Suhu dalam satuan Fahrenheit ditampilkan ke layar.

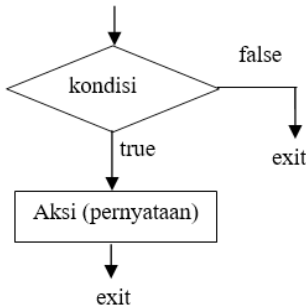


Gambar 1.5 Contoh flowchart struktur urut

## E. Flowchart Struktur Percabangan

### 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 menempati pada satu keluaran pemilihan yaitu untuk kondisi terpenuhi (hasil ekspresi kondisi benar) atau kondisi tidak terpenuhi (hasil ekspresi kondisi salah). Struktur *flowchart* satu kasus diperlihatkan dalam gambar di bawah ini.



Gambar 1.6 Struktur pemilihan dengan satu kasus



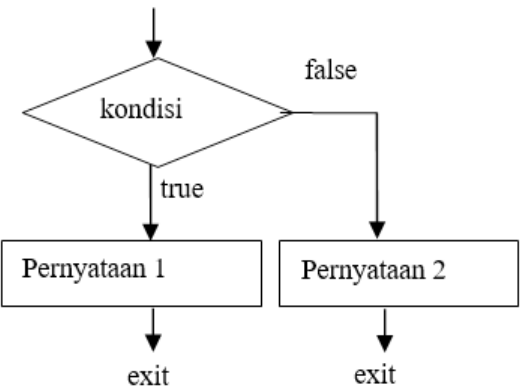
“Kondisi” yang ada pada simbol belah ketupat merupakan ekspresi kondisi yang terdiri atas operator yang meliputi: operator perbandingan (<, >, ==, .....), logika (OR, AND...), operand dapat berupa variabel atau nilai variabel. Tabel di bawah ini menjelaskan ragam contoh ekspresi kondisi.

**Tabel 1.3 Ragam jenis contoh ekspresi kondisi**

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

**2. Struktur Pemilihan atau Percabangan Dua Kasus**

Algoritma menggunakan *flowchart* dua kasus ini hanya terdiri satu ekspresi kondisi dan dua 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. Pernyataan atau instruksi yang akan dilakukan menempati pada dua keluaran pemilihan yaitu satu pernyataan pada kondisi terpenuhi (hasil ekspresi kondisi benar). Satu pernyataan lainnya menempati pada kondisi tidak terpenuhi (hasil ekspresi kondisi salah). Pernyataan atau instruksi yang akan dieksekusi tergantung dari data masukan dan hasil ekspresi kondisi. Struktur *flowchart* dua kasus diperlihatkan dalam gambar di bawah ini:

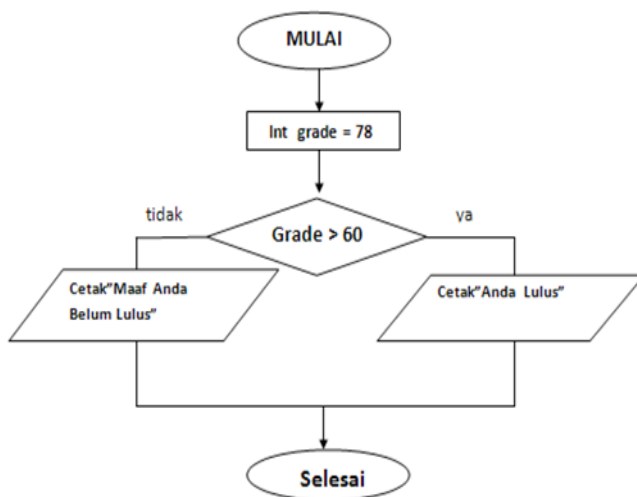


Gambar 1.7 Struktur pemilihan dengan dua kasus

Gambar di atas 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 dengan nilai dimasukkan dari *keyboard* dan ketentuan sebagai berikut,

**Tabel 1.4 Predikat nilai dengan dua grade**

Nilai	Grade
< 60	“Maaf Anda belum lulus”
>60	“Anda Lulus”

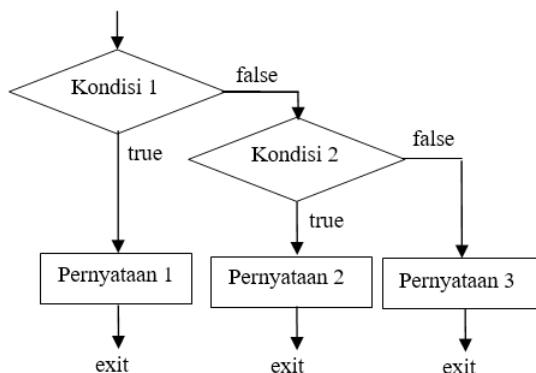


Gambar 1.8 Flowchart penentuan kelulusan dengan dua grade

Gambar di atas menjelaskan *flowchart* yang digunakan untuk menentukan kelulusan dengan kriteria jika nilai grade < 60 maka algoritma akan menampilkan "Maaf Anda belum Lulus." Jika nilai grade > 60 maka program akan menampilkan "Anda Lulus."

### 3. Struktur Pemilihan atau Percabangan TigaKondisi

Algoritma menggunakan *Flowchart* dengan tiga kasus ini terdiri dua ekspresi kondisi dan tiga pernyataan atau instruksi (aksi). Notasi yang digunakan untuk menyatakan struktur pemilihan tiga kasus ini adalah 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 berikut:



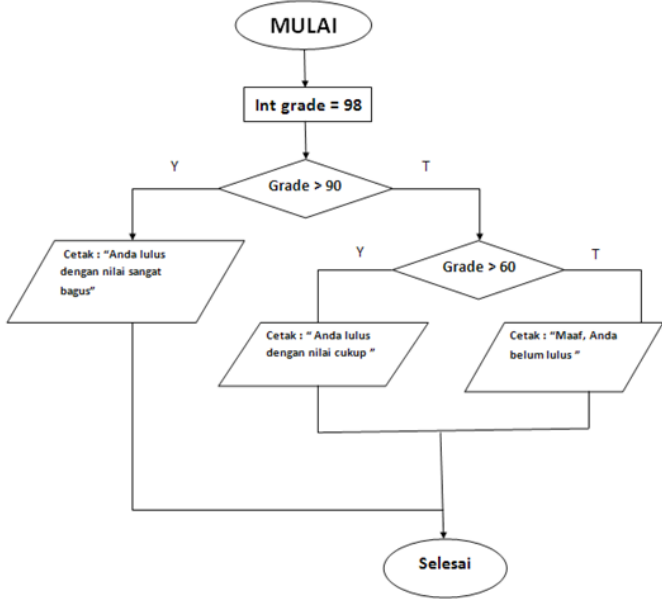
Gambar 1.8 Struktur pemilihan dengan tiga kondisi

Gambar di atas 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 1.5 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"



Gambar 1.9 Flowchart penentuan kelulusan dengan tiga grade

Gambar di atas menjelaskan *flowchart* yang digunakan untuk menentukan kelulusan dengan kriteria jika nilai Grade > 90 maka algoritma akan menampilkan "Anda lulus dengan nilai sangat bagus." Jika nilai grade antara 60 dan 90 maka program akan menampilkan "Anda lulus dengan nilai cukup." Jika nilai grade antara < 60 maka program akan menampilkan " maaf Anda belum lulus."

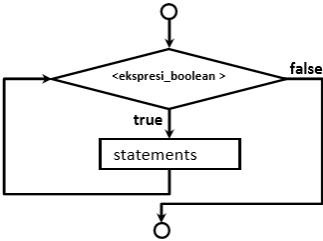
## G. Flowchart Struktur Perulangan

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: while, do-while, dan for.

### 1. 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. Bedanya dengan perintah do – while,

untuk instruksi *while* kondisi akan dicek dahulu selanjutnya pernyataan dikerjakan kemudian. Penulisan dengan notasi flowchartnya adalah sebagai berikut:

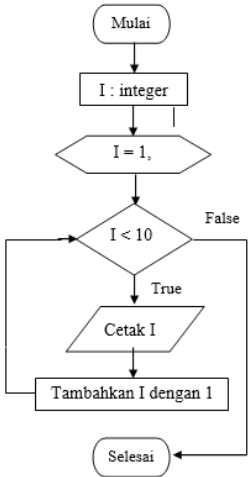


Gambar 1.9 Struktur flowchart menggunakan instruksi *while*

Contoh flowchart Gambar 10 akan menampilkan deret angka dari 1 sampai 9 dengan proses-prosesnya dapat dilihat dalam tabel berikut.

**Tabel 1.3 Analisis flowchart cetak deret 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



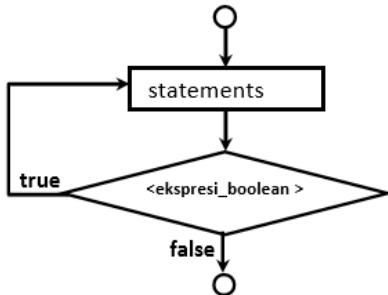
Gambar 1.10 Flowchart mencetak deret angka 1 sampai 9 menggunakan struktur instruksi *while*

## 2. Struktur Perulangan Menggunakan Instruksi Do-While

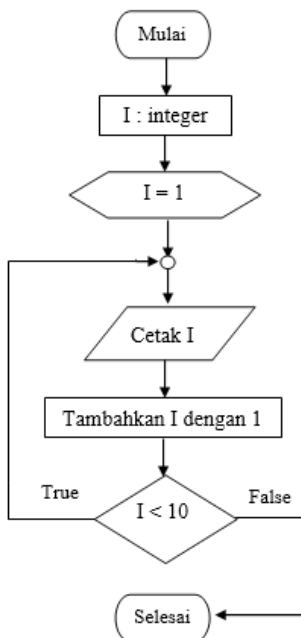
Sebagai mana instruksi *while* instruksi *do-while* merupakan proses perulangan yang akan berjalan jika ekspresi-boolean masih bernilai benar dan perulangan akan

dihentikan jika kondisinya sudah bernilai salah. Perbedaannya dengan instruksi *while* adalah terletak pada kondisi yang diperiksa, yaitu sebagai berikut.

- Pada perintah *while*, kondisi yang diperiksa terletak di awal perulangan, sehingga sebelum masuk ke dalam perulangan *while* kondisi harus bernilai benar.
- Pada perintah *do ... while*, kondisi diperiksa di akhir perulangan. Ini berarti bahwa paling sedikit sebuah perulangan akan dilakukan oleh perintah *do ... while*, karena untuk masuk ke dalam perulangan tidak ada kondisi yang harus dipenuhi.



Gambar 1.11 Struktur flowchart menggunakan instruksi *do...while*

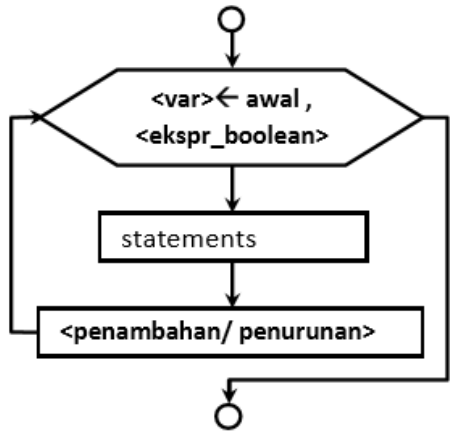


Gambar 1.12 Flowchart mencetak deret angka 1 sampai 9 menggunakan struktur instruksi *do ... while*

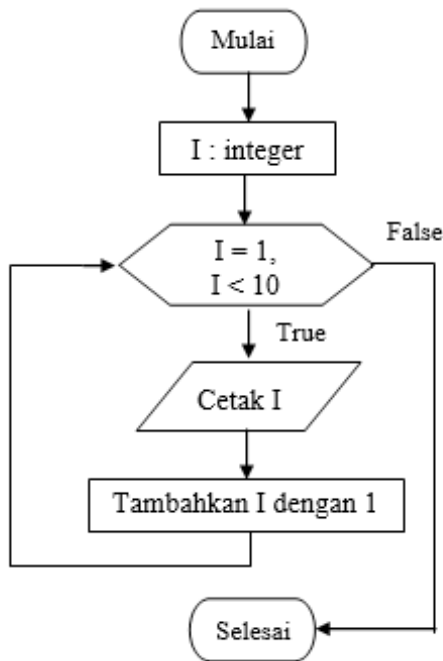
### 3. Struktur Perulangan Menggunakan Instruksi For

Struktur perulangan menggunakan *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 tiap kali sebuah perulangan dilaksanakan tergantung perintah yang akan diberikan.



Gambar 1.13 Struktur flowchart menggunakan instruksi for

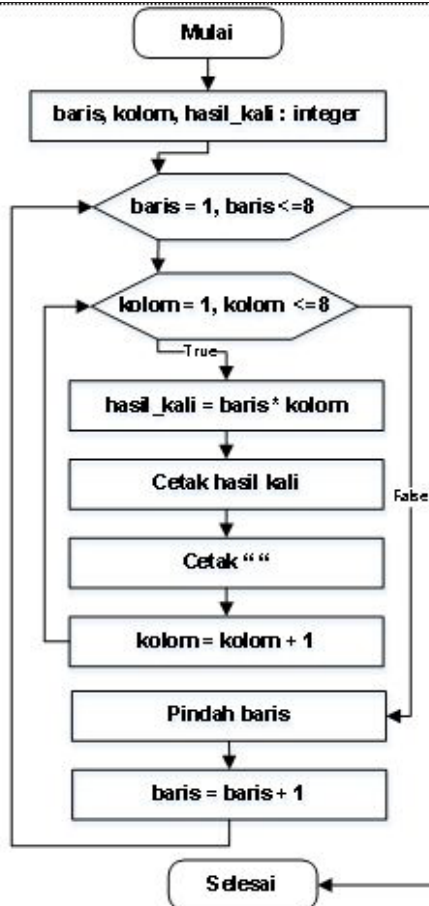


Gambar 1.14 Flowchart mencetak deret angka 1 sampai 9 menggunakan struktur instruksi for

#### 4. Struktur Perulangan Menggunakan Instruksi Do-While

Pada suatu perulangan bisa terdapat perulangan yang lain. Suatu struktur perulangan (*inner loop*) yang terdapat dalam perulangan lainnya (*outer loop*) ini sering disebut dengan istilah *nested loop*. Salah satu contoh penerapan algoritma *nested loop* adalah algoritma untuk tabel perkalian sebagai berikut.

	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



Gambar 1.15 Flowchart mencetak tabel perkalian 1 sampai 8

## Rangkuman

Alur logika pemrograman atau dikenal dengan nama algoritma merupakan suatu prosedur, urutan langkah-langkah atau tahapan-tahapan sistematis, jelas dan logis untuk menyelesaikan permasalahan. Penyajian algoritma dapat dibedakan menjadi dua yaitu: 1) berbentuk Tulisan dan 2) berbentuk Gambar. Algoritma yang disajikan dengan bentuk tulisan dapat menggunakan aturan bahasa natural (alami) dan *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. Pilihlah jawaban yang 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*
2. Bagian teks algoritma pseudocode sebagai tempat untuk mendefinisikan, nama tipe data, konstanta dan variabel adalah ....
  - a. *header*
  - b. *description*
  - c. *comment*
  - d. *dictionary*
3. Alur logika pemrograman atau algoritma harus berhenti setelah mengerjakan sejumlah langkah tertentu atau terbatas, aspek tersebut dikenal dengan istilah ....
  - a. *effectiveness*
  - b. *finiteness*
  - c. *definiteness*
  - d. *completeness*
4. Penulisan algoritma yang menyerupai bahasa pemrograman tingkat tinggi adalah ....
  - a. *natural*
  - b. *pseudocode*
  - c. *comment*
  - d. *scriptcode*
5. Instruksi perulangan yang umum digunakan dalam bahasa pemrograman, kecuali ....
  - a. *for*
  - b. *while*
  - c. *do ...while*
  - d. *do ... for*



## Lembar Kerja Siswa Berbasis STEM

1. Gambarkan Toleransi pada gambar susunan!
2. Gambarkan Ketinggian sepuluh titik Ra dari ketidakrataan!
3. Gambarkan Posisi keterangan-keterangan permukaan pada lambang!