

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET, DAN TEKNOLOGI  
REPUBLIK INDONESIA, 2023

Informatika  
untuk SMK/MAK Kelas X Semester 2

Penulis: Kusmadi, Imam Badrudin, Lisna Nurrohmawati, dan Bima Laksana Putra  
ISBN: 978-623-194-543-3 (no.jil.lengkap PDF)  
978-623-194-545-7 (jil.2 PDF)



# Bab 1 Algoritma dan Pemrograman

Bagaimana cara kalian menerapkan berbagai notasi algoritma ke dalam bahasa pemrograman prosedural?



## Tujuan Pembelajaran

Setelah mempelajari materi pada bab ini, kalian diharapkan mampu, menerapkan praktik baik konsep pemrograman prosedural dalam salah satu bahasa pemrograman prosedural dengan tepat dan mengembangkan program yang terstruktur dalam notasi algoritma atau notasi lain berdasarkan strategi algoritmik yang tepat.



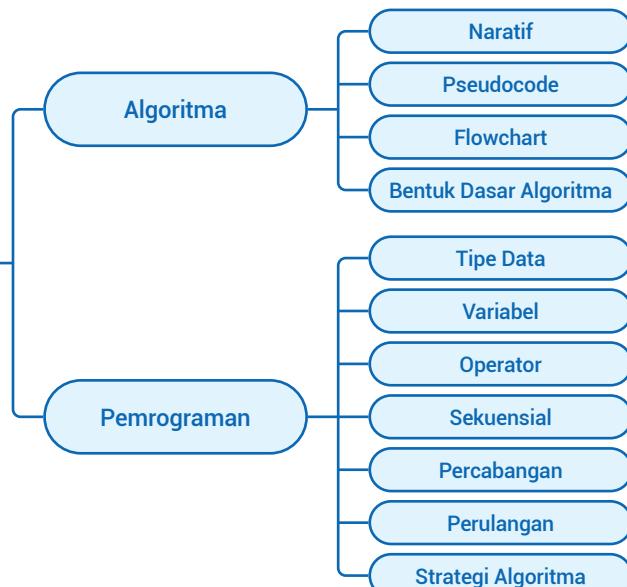
## Kata Kunci

Algoritma • Flowchart • Pseudocode • Pemrograman •  
Tipe Data • Operator • Percabangan • Perulangan • Variabel



## Peta Materi

Algoritma dan  
Pemrograman (AP)



Gambar 1.1 Peta Materi

Pernahkah kalian membuat teh manis atau membuat kopi susu di rumah? Coba kalian bayangkan bagaimana tahapan dalam membuat teh manis atau kopi susu di rumah.

Jika kalian membuat kopi susu, langkah yang harus kalian siapkan sebagai berikut.

1. Menyiapkan kopi, susu, gula, gelas, sendok, dan air panas.
2. Membuka bungkus kopi kemudian menuangkannya ke dalam gelas sesuai takaran yang dikehendaki.
3. Membuka bungkus susu kemudian menuangkannya ke dalam gelas sesuai takaran yang dikehendaki.
4. Menambah satu sendok gula ke dalam gelas.
5. Menuangkan air panas ke dalam gelas secukupnya.
6. Aduk kopi susu hingga bercampur dengan air.
7. Jika kopi susu kurang manis, silakan tambahkan lagi gulanya kemudian aduk lagi sampai bercampur dengan air.
8. Menghidangkan/menikmati kopi susu yang sudah siap.

Contoh membuat kopi susu di atas adalah salah satu bentuk algoritma.

## Algoritma

### 1. Pengertian

Sebelum hadir di sekolah, umumnya ada beberapa tahapan yang harus kalian lalui, antara lain membersihkan diri dengan mandi, memakai seragam sekolah, mempersiapkan alat tulis dan buku pelajaran, sarapan, perjalanan menuju sekolah, masuk gerbang sekolah, bertemu teman di sekolah, masuk ke kelas, mengikuti pelajaran yang diberikan oleh guru, istirahat kemudian di akhir pelajaran pulang menuju rumah masing-masing. Aktivitas tersebut dilakukan secara berurutan dari awal hingga akhir. Aktivitas tersebut disebut sebagai algoritma. Definisi algoritma adalah urutan atau aliran yang digunakan dalam komputasi sistematis atau pemecahan masalah dalam kegiatan pemrograman. Algoritma sering dianggap logis saat menentukan urutan program yang akan dieksekusi.

Setelah mengetahui pengertian dari algoritma, kalian juga harus mengetahui apa sebenarnya fungsi dari algoritma tersebut. Pada dasarnya, fungsi utama dari suatu algoritma adalah memecahkan suatu masalah. Algoritma pemrograman membawa manfaat dan fitur penting untuk aktivitas pemrograman. Berikut adalah algoritma yang perlu kalian ketahui.

- a. Pemrograman digunakan untuk memecahkan masalah kompleks dalam program komputer yang mungkin juga melibatkan perhitungan tingkat tinggi. Algoritma yang diterapkan pada pemrograman bertujuan untuk meminimalisir kesalahan yang mungkin terjadi.
- b. Algoritma pemrograman juga mampu menyederhanakan program, dari program yang besar menjadi program yang lebih sederhana.
- c. Fungsi dari algoritma ini bukan merupakan sekali pakai, artinya dapat digunakan secara berulang-ulang. Jadi, kalian tidak perlu menuliskan kembali kode program yang sama di lain waktu.
- d. Saat membuat program, kalian pasti akan menemukan beberapa kesalahan, itu sangat wajar. Dengan menerapkan fungsi algoritma, pencarian serta pemecahan masalah dapat diperbaiki dengan lebih mudah dan cepat.
- e. Algoritma pemrograman memiliki alur yang sangat jelas sehingga memudahkan untuk menemukan Error ketika terjadi kesalahan.

Menurut Donald E. Knuth (1997:4) dalam bukunya yang berjudul *The Art of Computer Programming*, Algoritma memiliki lima karakteristik yang perlu diketahui yaitu sebagai berikut.

- a. Terbatas (*Finiteness*)

Algoritma harus berhenti setelah mengerjakan sejumlah langkah terbatas (berhingga).

- b. Tidak ambigu (*Definiteness*)

Setiap langkah harus didefinisikan dengan tepat dan tidak memiliki makna ganda (*ambiguous*).

- c. Masukan (*Input*)

Algoritma memiliki nol atau lebih masukan (*input*).

- d. Keluaran (*Output*)

Algoritma mempunyai nol atau lebih keluaran (*output*).

- e. Efektif (*Effectiveness*)

Algoritma harus efektif dan efisien.

## 2. Bahasa Pemrograman

Bahasa pemrograman adalah seperangkat instruksi standar yang digunakan untuk memerintah komputer melakukan fungsi tertentu. Bahasa pemrograman terdiri atas kumpulan sintaks dan instruksi komputer dasar seperti perhitungan, penyimpanan data, dan lain-lain.

Fungsi bahasa pemrograman adalah menginstruksikan komputer untuk memproses data sesuai dengan yang diinginkan. Keluaran dari bahasa pemrograman ini berupa program atau aplikasi yang dapat dijalankan dan prosesnya sesuai dengan alur kerja yang diinginkan. Misal: lampu lalu lintas merupakan salah satu contoh penerapan bahasa pemrograman untuk membuat nyala lampu lalu lintas bergantian.



Gambar 1.2 Lampu Lalu Lintas

Sumber: Freepik/macrovector (2022)

Menurut Dasril Aldo dkk. (2020:30) dalam bukunya *Pengantar Teknologi Informasi*, generasi bahasa pemrograman terbagi menjadi lima generasi sebagai berikut.

a. Generasi Pertama, Bahasa Mesin

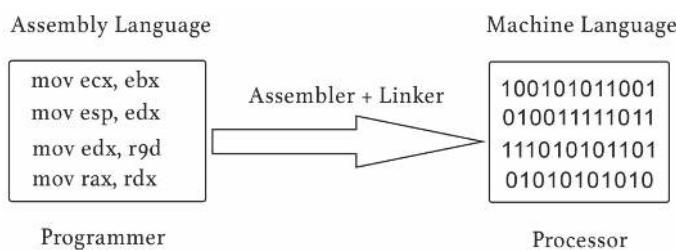
Generasi ini sering disebut bahasa mesin atau bahasa komputer asli. Bahasa pemrograman ini hanya mengenal bilangan biner (angka 1 dan 0). Jenis bahasa pemrograman ini jarang digunakan untuk membuat perangkat lunak karena tingkat kesulitannya dan biasanya diperkenalkan di lingkungan pendidikan untuk kepentingan pembelajaran dasar pemrograman komputer.

**Tabel 1.1** Bahasa Mesin

<b>Hexadecimal</b>	<b>Code in Machine Language</b>			
$(1FEF)_{16}$	0001	1111	1110	1111
$(240F)_{16}$	0010	0100	0000	1111
$(1FEF)_{16}$	0001	1111	1110	1111
$(241F)_{16}$	0010	0100	0001	1111
$(1040)_{16}$	0001	0000	0100	0000
$(1141)_{16}$	0001	0001	0100	0001
$(3201)_{16}$	0011	0010	0000	0001
$(2422)_{16}$	0010	0100	0010	0010
$(1F42)_{16}$	0001	1111	0100	0010
$(2FFF)_{16}$	0010	1111	1111	1111
$(0000)_{16}$	0000	0000	0000	0000

b. Generasi Kedua, Bahasa Assembler

Pada generasi ini sudah tidak menggunakan bilangan 0 dan 1 tetapi menggunakan kode-kode yang disebut dengan *mnemonic*. Sebagai pengganti kode-kode biner, digunakan kependekan dari kata-kata seperti “MOV” untuk menyatakan “MOVE” dan “JNZ” yang berarti “Jump Non-Zero”. Setiap instruksi Bahasa Assembler sebenarnya identik dengan satu instruksi bahasa mesin.



**Gambar 1.3** Bahasa Asembler

Sumber: assignmentstudio.net/assembly-language-assignment-help/

c. Generasi Ketiga, Bahasa Prosedural

Pada generasi ini, bahasa yang dipakai sudah menggunakan pendekatan prosedural sehingga bahasanya mudah dimengerti oleh manusia, hanya saja masih menyertakan simbol karakter seperti kurung kurawal, tanda tanya dan sebagainya. Contoh bahasa pemrograman generasi ini yaitu Basic, C, PASCAL, COBOL, dan FORTRAN.

d. Generasi Keempat, Bahasa 4GL

Pada generasi ini, dikenal dengan bahasa querie terstruktur (SQL) yang banyak digunakan untuk mengolah basis data seperti Oracle, MySQL, PostgreSQL, dan lain sebagainya. Generasi ini juga disebut dengan *Very High Level Language* atau *Problem Oriented Language* (bahasa yang berorientasi pada masalah), dengan harapan *programmer* dapat meningkatkan produktivitas dan menghasilkan program dalam waktu yang singkat.

e. Generasi Kelima, Bahasa Pemrograman Berorientasi Objek dan Pengembangan Website

Pada generasi ini, bahasa pemrograman lebih menekankan pada aspek efisiensi dalam penggunaan modul-modul yang telah dibuat dengan bahasa pemrograman tingkat tinggi tertentu. Pada generasi ini biasa disebut *intelligent programming* atau pemrograman kecerdasan yang menekankan aspek otomasi dalam setiap prosesnya.

Menurut Dasril Aldo, dkk. (2020:30) dalam bukunya Pengantar Teknologi Informasi, berdasarkan tingkatannya, bahasa pemrograman dibagi menjadi tiga tingkatan sebagai berikut.

a. Bahasa Tingkat Rendah (*low level language*)

Bahasa mesin adalah satu-satunya bahasa pemrograman yang dapat diproses langsung oleh komputer tanpa proses konversi atau penerjemahan. Saat ini hampir tidak ada programmer yang menggunakan bahasa pemrograman seperti itu karena tersedia bahasa pemrograman yang lebih tinggi dan mudah digunakan.

Programmer menggunakan bahasa mesin untuk menulis kode program dalam bentuk biner (1 dan 0), yang kemudian diubah oleh mesin menjadi desimal, oktal, atau heksadesimal.

b. Bahasa Tingkat Menengah (*middle level language*)

Bahasa yang digunakan pada level ini adalah bahasa Assembler. Assembler memiliki kode simbolis yang mewakili kode mesin yang dapat dibaca manusia, seperti move, loop, dan lain-lain. Tiap baris kode Assembler mewakili satu pernyataan kode mesin.

c. Bahasa Tingkat Tinggi (*high level language*)

Pada tingkatan ini, bahasa pemrograman yang banyak digunakan adalah C++, Visual Basic, Delphi, Pascal. Bahasa generasi ke-3 ini juga banyak digunakan pada generasi ke-5 meskipun dengan perubahan yang lebih sempurna. Contoh bahasa pemrograman lainnya adalah Java (Android), JSP, ASP, PHP, .Net, C#, Python, dan lain-lain.

Berikut proses pembuatan program.

- a. Menulis kode pada sebuah editor.
- b. Mengubahnya menjadi bahasa mesin yang dieksekusi oleh prosesor. Proses ini terdiri atas tiga bagian sebagai berikut.
  - 1) Kompilasi

Penerjemahan dan penggabungan kode sumber ke dalam format lain. Kompilasi adalah teknik membaca program yang ditulis dalam bahasa sumber dan kemudian menerjemahkannya ke bahasa lain. Contoh bahasa pemrograman yang menggunakan teknik ini adalah C dan C++.
  - 2) Interpretasi

Dalam bahasa pemrograman yang menggunakan teknik ini, kode sumber dieksekusi baris demi baris. Jika terjadi kesalahan selama eksekusi, program berhenti di baris kode sumber tempat kesalahan terjadi. Teknik ini digunakan pada bahasa pemrograman seperti Perl, Ruby, dan Python.
  - 3) Kompilasi sekaligus Interpretasi

Bahasa pemrograman menggunakan teknik ini secara bersamaan dalam menerjemahkan dan menginterpretasikan ke bahasa mesin. Contoh bahasa pemrograman ini adalah Java. Java mengubah kode sumber menjadi *bytecode* melalui proses kompilasi yang kemudian hasilnya dieksekusi oleh *interpreter* pada komputer.



## Aktivitas Pembelajaran



### Ayo Berlatih

Aktivitas individu untuk memperdalam pemahaman

Berdasarkan uraian di atas tentang algoritma dan bahasa pemrograman, coba amati aktivitas yang dilakukan di sekolah sebelum mulai pembelajaran sampai dengan pembelajaran selesai. Kemudian, jelaskan bagian mana saja yang termasuk dalam karakteristik algoritma berdasarkan hasil pengamatan kalian serta jelaskan pengertian bahasa pemrograman.



### Ayo Berdiskusi

Diskusi Kelompok dengan anggota 4-5 orang perkelompok

Berdasarkan uraian di atas tentang algoritma dan bahasa pemrograman, lakukan diskusi bersama teman kelompokmu untuk memperdalam pengetahuan tentang algoritma dan perkembangan bahasa pemrograman.

Untuk memudahkan pemetaan konsep dalam berdiskusi kalian dapat menggunakan tabel berikut sebagai kerangka kerja diskusi yang kolaboratif dan kreatif bersama teman kelompokmu!

Tabel 1.2 Tugas Kelompok Algoritma

Generasi Bahasa Pemrograman	Tingkatan Bahasa Pemrograman	Proses Pembuatan Program

## Naratif, Pseudocode, Flowchart

### 1. Naratif

Teks naratif adalah sebuah cerita yang menyajikan urutan peristiwa dalam urutan waktu, baik bersifat fiksi maupun nonfiksi. Sesuai definisinya, teks narasi memiliki ciri khas berupa urutan suatu kejadian dengan beberapa peristiwa kronologis. Naratif dalam algoritma merupakan cara menuliskan instruksi yang harus dilaksanakan secara berurutan dalam bentuk cerita atau uraian kalimat dengan menggunakan bahasa yang jelas seperti bahasa dalam kehidupan sehari-hari.

Contoh: menghitung luas persegi panjang menggunakan naratif.

Langkah ke-1: mulai

Langkah ke-2: baca nilai panjang

Langkah ke-3: baca nilai lebar

Langkah ke-4: hitung luas (Panjang x Lebar)

Langkah ke-5: cetak hasil luas

Langkah ke-6: selesai

### 2. Pseudocode

*Pseudocode* adalah metode penulisan algoritma yang hampir sama seperti bahasa pemrograman, namun *pseudocode* lebih sederhana dan ditulis menggunakan bahasa yang mudah dipahami manusia. Dalam struktur penulisannya, *pseudocode* dibagi menjadi tiga bagian yaitu sebagai berikut.

a. Bagian Judul

Bagian judul diawali dengan nama “PROGRAM” kemudian diikuti dengan nama algoritma.

b. Bagian Deklarasi

Deklarasi digunakan untuk mendeklarasikan variabel apa saja yang akan digunakan di dalam algoritma dan sejenisnya.

c. Bagian Isi

Bagian utama dari jalannya algoritma.

Contoh: *pseudocode* untuk menghitung luas persegi panjang.

PROGRAM Hitung Luas Persegi Panjang

DEKLARASI

int p, l

float luas

ALGORITMA

Baca p dan l

Hitung luas = p \* l

Tampilkan luas



*PSEUDOCODE* adalah metode penulisan algoritma yang menjelaskan urutan sebelum program ditulis ke dalam bahasa pemrograman yang diinginkan.

### 3. Flowchart

*Flowchart* adalah suatu bagan dengan simbol-simbol tertentu yang menggambarkan urutan proses secara mendetail dan hubungan antara suatu proses (instruksi) dengan proses lainnya dalam suatu program. Menggambar *flowchart* (diagram alir) tidak ada rumus atau standar yang mutlak. Diagram berikut merupakan contoh hasil berpikir dalam menganalisis suatu masalah di komputer. Secara umum setiap struktur *flowchart* selalu terdiri atas tiga bagian, yaitu input (masukan), proses, dan *output* (keluaran).



Gambar 1.4 *Input – Proses – Output*

Sumber: Kusmadi (2022)

Berikut simbol-simbol pada *flowchart*.

Tabel 1.3 Simbol *Flowchart*

Simbol	Nama	Penggunaan
	<i>Flow Direction</i>	Simbol yang digunakan untuk menghubungkan antara simbol yang satu dengan simbol yang lain. Simbol ini juga disebut sebagai <i>connecting line</i> .
	<i>Terminator</i>	Simbol untuk permulaan atau akhir dari suatu kegiatan.
	<i>Connector</i>	Simbol untuk menghubungkan diagram dalam lembar atau halaman yang sama.
	<i>Connector</i>	Simbol untuk menghubungkan diagram dalam lembar atau halaman yang berbeda.
	<i>Processing</i>	Simbol yang digunakan untuk menunjukkan pengolahan data oleh komputer.
	<i>Manual Operation</i>	Simbol yang digunakan untuk menunjukkan pengolahan data tidak dilakukan oleh komputer.
	<i>Decision</i>	Simbol pemilihan proses sesuai dengan kondisi yang ada.
	<i>Input-Output</i>	Simbol yang digunakan untuk menyatakan proses <i>input output</i> .
	<i>Manual Input</i>	Simbol untuk <i>input</i> data manual melalui <i>keyboard</i> .
	<i>Preparation</i>	Simbol untuk mempersiapkan penyimpanan yang akan digunakan sebagai tempat pengolahan di dalam <i>storage</i> .
	<i>Predefined Process</i>	Simbol untuk pelaksanaan suatu bagian ( <i>subprogram/prosedur</i> ).
	<i>Display</i>	Simbol yang menyatakan peralatan yang digunakan yaitu printer, plotter, dan lain sebagainya.
	<i>Disk and On-line Storage</i>	Simbol yang menyatakan <i>input</i> berasal dari disk atau disimpan pada disk.

Simbol	Nama	Penggunaan
	<i>Magnetic Tape Unit</i>	Simbol yang menyatakan <i>input</i> berasal dari peta magnetik atau dicetak pada pita magnetik.
	<i>Punch Card</i>	Simbol yang menyatakan <i>input</i> berasal dari kartu atau dicetak pada sebuah kartu.
	<i>Document</i>	Simbol yang menyatakan <i>input</i> berasal dari dokumen kertas atau dicetak pada sebuah kertas.



## Aktivitas Pembelajaran



### Ayo Berlatih

Aktivitas individu untuk memperdalam pemahaman

Berdasarkan uraian materi di atas, jelaskan pengertian teks naratif, *pseudocode* dan *flowchart* dalam algoritma!



### Ayo Berdiskusi

Diskusi Kelompok dengan anggota 4-5 orang perkelompok

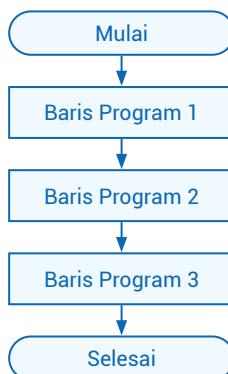
Berdasarkan uraian di atas tentang naratif, *pseudocode* dan *flowchart*, lakukan diskusi bersama teman kelompokmu untuk menentukan studi kasus dalam kehidupan sehari-hari yang ada unsur algoritmanya. Kemudian buatlah narasi seperti contoh membuat kopi susu pada bagian apersepsi, buatlah *pseudocode* beserta *flowchart* dan jelaskan bagian *input*, proses serta *output* yang terdapat pada *flowchart*.

# Bentuk Dasar Algoritma

## 1. Sekuensial

Pada dasarnya setiap aktivitas yang kalian kerjakan dilakukan secara sekuensial atau berurutan, yaitu dimulai dari langkah pertama, kedua, dan seterusnya hingga selesai. Sebagai contoh adalah bagaimana menghitung luas kamar tidur berbentuk peseri panjang di rumah yang kalian tempati jika diketahui panjang kamar tujuh meter dan lebar empat meter?

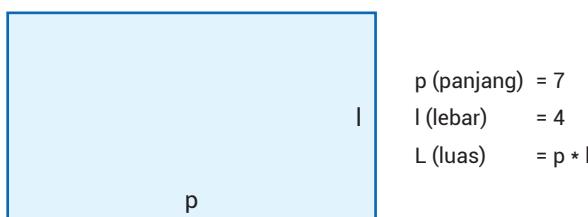
Jika dibuat dalam bentuk diagram, hasilnya menjadi seperti berikut.



Gambar 1.5 Sekuensial

Sumber: Kusmadi (2022)

Apabila dibuat dalam bentuk naratif, hasilnya menjadi seperti berikut.



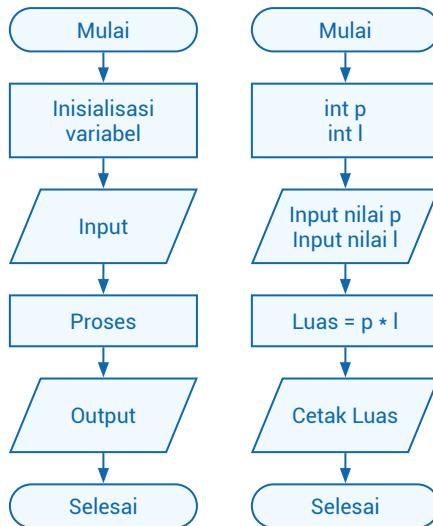
Gambar 1.6 Menghitung Luas Kamar Tidur Berbentuk Persegi Panjang

Sumber: Kusmadi (2022)

- Buat simbol *terminator* (*elipse*) sebagai tanda awal dimulainya program.
- Tulis keterangan *Mulai* untuk menunjukkan bahwa program telah berjalan.
- Buat simbol *input-output* untuk mendefinisikan variabel *input*.
- Masukkan nama *variabel* dan nilainya (bebas).
- Buatlah simbol *proses* untuk menentukan perhitungan luas kamar tidur.

- f. Tuliskan rumus untuk menghitung luas kamar tidur **Luas = p \* l**
- g. Buatlah simbol *input-output* untuk menampilkan hasil perhitungan luas kamar tidur.
- h. Tuliskan nama *variabel* yang akan ditampilkan sebagai keluaran/*output* pada layar.
- i. Buat lagi simbol *terminator* (*ellipse*) untuk mengakhiri program.
- j. Tulis keterangan *Selesai* sebagai tanda bahwa program telah berhenti.

Apabila dibuat dalam bentuk *flowchart*, hasilnya sebagai berikut!



**Gambar 1.7 Flowchart Menghitung Luas Kamar Tidur Berbentuk Persegi Panjang**

Sumber: Kusmadi (2022)

Dari contoh di atas *pseudocode* dapat dibuat seperti berikut.

1	<b>PROGRAM HitungLuasPersegiPanjang</b>
2	<b>DEKLARASI</b>
3	<b>int p, l</b>
4	<b>float luas</b>
5	
6	<b>ALGORITMA</b>
7	<b>baca p dan l</b>
8	<b>hitung luas = p * l</b>
9	<b>tampilkan ("Luas Persegi Panjang adalah " + luas)</b>

**Gambar 1.8 Pseudocode Menghitung Luas Kamar Tidur (Persegi Panjang)**

Sumber: Kusmadi (2022)

## 2. Percabangan

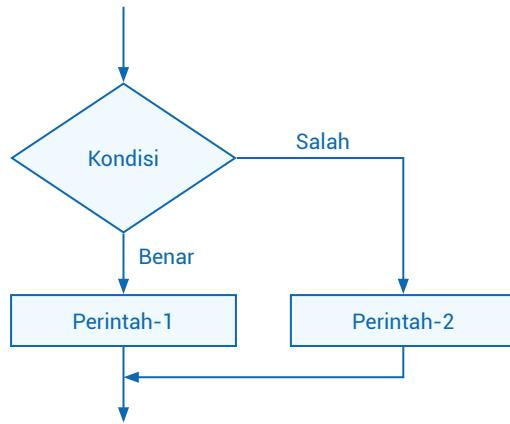
Percabangan/pemilihan adalah proses di mana suatu kondisi dinyatakan benar dan yang lainnya dinyatakan salah. Sebagai contoh, ketika kalian mengendarai sepeda motor di jalan raya dan pada saat yang sama lampu lalu lintas berwarna merah maka yang harus kalian lakukan adalah menghentikan sepeda motor tepat di belakang garis marka jalan sembari menunggu hingga lampu lalu lintas berwarna hijau.

Dari kondisi di atas, tahapan narasi yang sesuai adalah sebagai berikut.

- ▶ Mengeluarkan sepeda motor dari dalam rumah.
- ▶ Memakai perlengkapan keselamatan berkendara.
- ▶ Membawa surat izin mengemudi.
- ▶ Mengendarai sepeda motor melintasi jalan raya dengan tertib.
- ▶ Selama berkendara, apabila dari kejauhan lampu lalu lintas berwarna kuning, maka yang seharusnya dilakukan adalah mengurangi kecepatan laju kendaraan bermotor.
- ▶ Apabila telah sampai di lampu lalu lintas dan berwarna merah, maka kendaraan bermotor harus dihentikan sepenuhnya.
- ▶ Apabila lampu lalu lintas telah menyala warna hijau, maka tuas gas dapat diputar untuk melanjutkan perjalanan.

Dari contoh di atas dapat disimpulkan bahwa apabila kondisi yang telah ditetapkan oleh pemerintah melalui peraturan perundang-undangan lalu lintas tidak terpenuhi maka kalian dapat dikenakan denda tilang sesuai dengan peraturan yang berlaku.

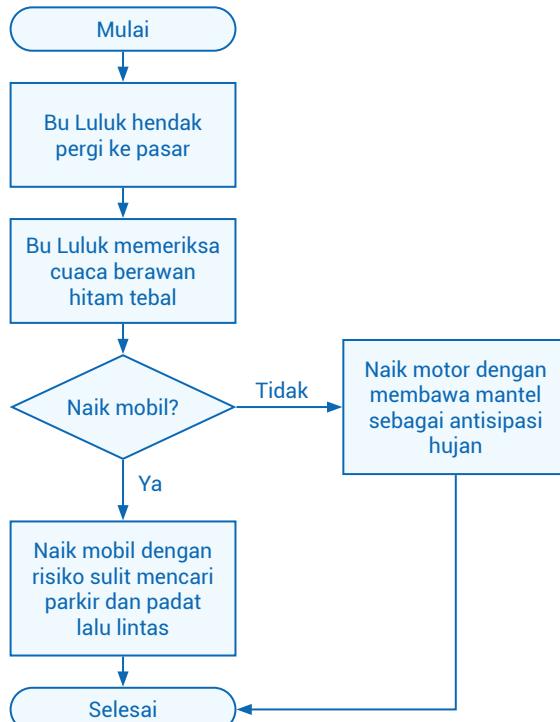
Dalam hidup kita sering menghadapi situasi di mana kita harus memilih salah satu dari dua pilihan atau lebih. Misalnya, jika kalian ingin pergi ke pasar, kalian dihadapkan pada dua pilihan dan harus memilih salah satu. Salah satu keputusan itu adalah apakah akan pergi ke pasar dengan mobil atau sepeda motor. Penentuan pilihan tersebut akan membuat langkah dilanjutkan ke proses selanjutnya.



**Gambar 1.9** Pemilihan

Sumber: Kusmadi (2022)

Contohnya ada seorang ibu rumah tangga bernama Luluk yang ingin pergi ke pasar. Saat hendak berangkat, tiba-tiba cuaca berubah menjadi gelap dan mendung. Bu Luluk mengira sebentar lagi akan hujan, jadi dia mulai berpikir untuk mengendarai mobil atau motor. Dalam situasi seperti itu, logikanya dapat digambarkan sebagai berikut.



**Gambar 1.10** Kondisi Percabangan Pergi ke Pasar

Sumber: Kusmadi (2022)

## Struktur Percabangan

Struktur percabangan adalah struktur program yang melakukan proses pengujian untuk memutuskan apakah akan memproses baris perintah/blok atau tidak. Dalam bahasa pemrograman, pernyataan bisa bernilai benar atau salah. Secara umum, percabangan terdiri atas operator-operator yang digabungkan, yaitu antara operator relasi dan operator logika. Contoh sebagai berikut.

Pernyataan 1:  $5 == 5$  ← benar karena nilai kiri dan kanan sama.

Pernyataan 2:  $9 < 32$  ← benar karena 9 hasilnya lebih kecil dari 32.

Pernyataan 3:  $8 == 4$  ← salah karena 8 tidak sama dengan 4.

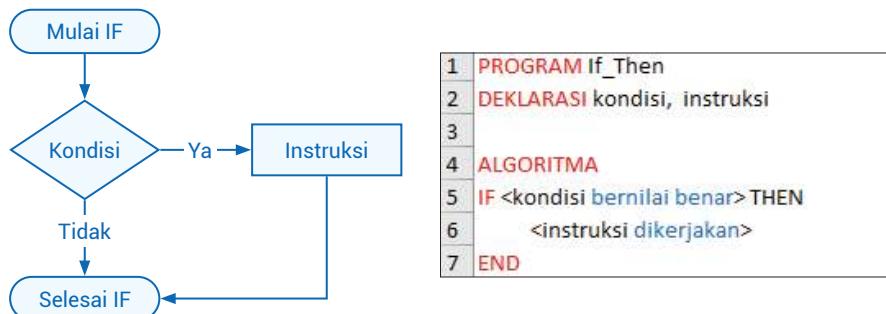


Penulisan struktur percabangan dalam program dituliskan dengan perintah IF.

Macam-macam struktur IF adalah sebagai berikut.

### a. Percabangan IF

Percabangan IF ialah percabangan yang hanya memiliki 1 blok pilihan instruksi pada saat kondisi bernilai benar. Apabila kondisi bernilai YA, maka instruksi dikerjakan sedangkan kondisi bernilai TIDAK, maka instruksi tidak dikerjakan.

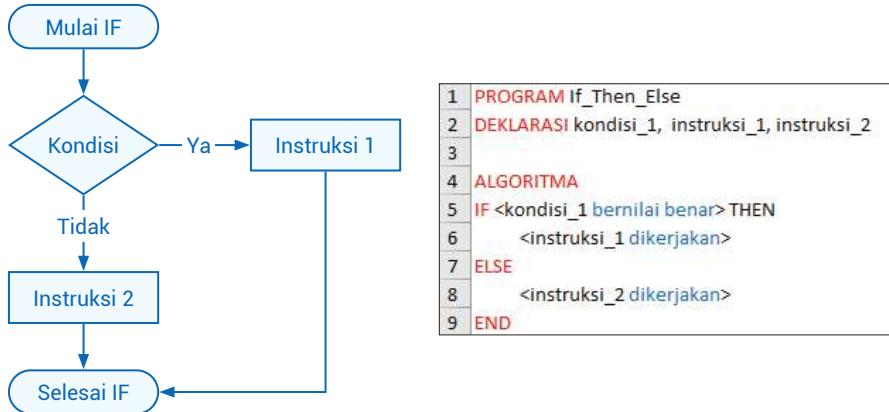


Gambar 1.11 Flowchart dan Pseudocode IF

Sumber: Kusmadi (2022)

### b. Percabangan IF – ELSE

IF – ELSE adalah percabangan dengan 2 blok pilihan instruksi. Dalam bentuk ini, Instruksi 1 dijalankan jika nilai kondisinya YA, dan Instruksi 2 dijalankan jika kondisinya TIDAK.

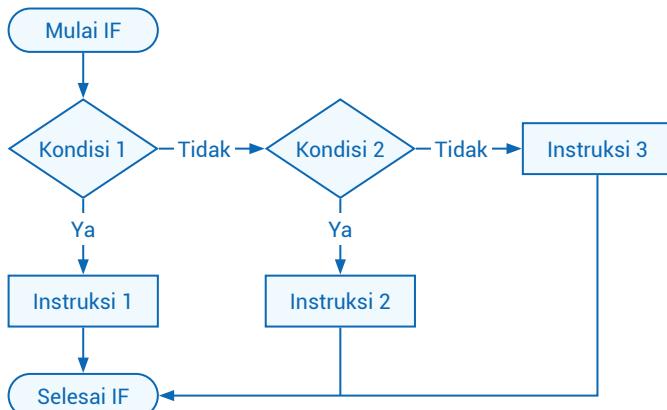


Gambar 1.12 Flowchart dan Pseudocode IF – ELSE

Sumber: Kusmadi (2022)

### c. Percabangan IF – ELSE – IF

IF – ELSE – IF adalah percabangan dengan lebih dari 2 blok yang dipilih. Dalam bentuk ini, instruksi 1 dijalankan jika kondisi 1 adalah YA. Instruksi 2 dijalankan jika kondisi 1 mengembalikan NO dan kondisi 2 mengembalikan YA. Instruksi 3 dijalankan jika kondisi 1 dan kondisi 2 TIDAK.



```

1 PROGRAM If_Else_IF
2 DEKLARASI kondisi_1, kondisi_2, instruksi_1, instruksi_2, instruksi_3
3
4 ALGORITMA
5 IF <kondisi_1 bernilai benar> THEN
6   <instruksi_1 dikerjakan>
7 ELSE IF <kondisi_2 bernilai benar> THEN
8   <instruksi_2 dikerjakan>
9 ELSE
10  <instruksi_3 dikerjakan>
11 END

```

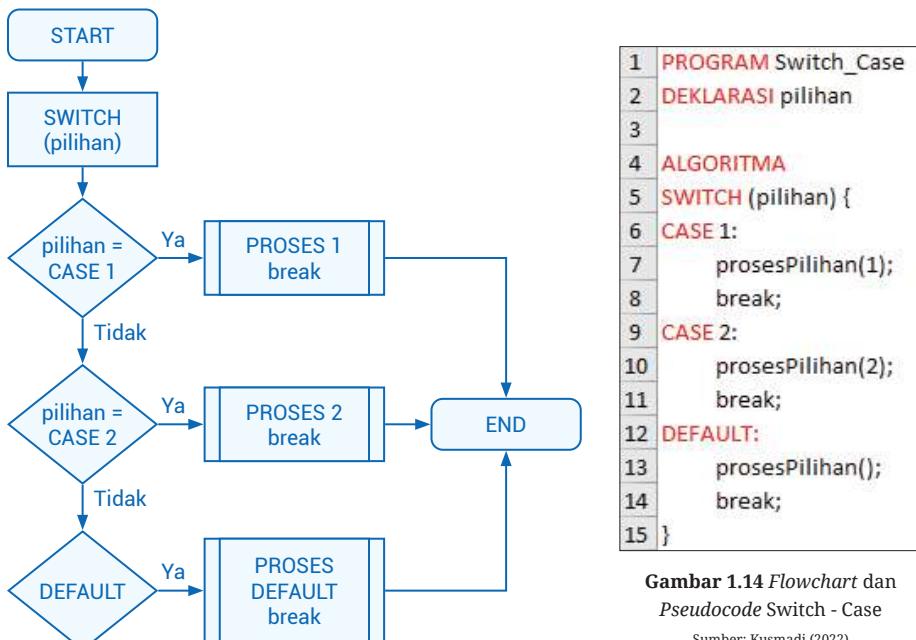
Gambar 1.13 Flowchart dan Pseudocode IF – ELSE – IF (Banyak Kondisi)

Sumber: Kusmadi (2022)

#### d. Percabangan Switch – Case

Percabangan Switch – Case merupakan bentuk lain dari percabangan IF – ELSE. Kondisi Switch Case adalah percabangan kode program yang membandingkan isi variabel multinilai. Jika proses perbandingan bernilai *true*, maka blok kode program akan diproses.

Kondisi SWITCH CASE terdiri atas dua bagian: perintah SWITCH yang berisi nama variabel yang akan diperiksa, dan satu atau lebih perintah CASE untuk setiap nilai yang akan diperiksa. Pilihan yang telah ditetapkan akan dibandingkan dengan kondisi yang ada. Jika perbandingan menghasilkan TRUE, maka sebuah proses akan dikerjakan.



Gambar 1.14 Flowchart dan Pseudocode Switch - Case

Sumber: Kusmadi (2022)



## Aktivitas Pembelajaran



### Ayo Berlatih

Aktivitas individu untuk memperdalam pemahaman

Cocokkan percabangan pada kolom A dengan penjelasan yang ada pada kolom B.

Tabel 1.4 Aktivitas Individu Percabangan

Kolom A		Kolom B
IF		Percabangan yang hanya memiliki 2 blok pilihan instruksi.
IF – ELSE		Bentuk lain dari percabangan IF - ELSE.
IF – ELSE – IF		Percabangan yang hanya memiliki 1 blok pilihan instruksi.
SWITCH – CASE		Percabangan yang hanya memiliki lebih dari 2 blok pilihan instruksi.



### Ayo Berdiskusi

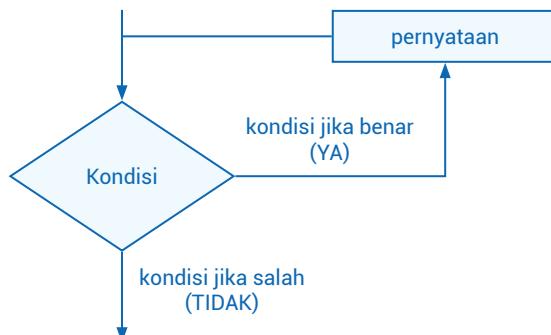
Diskusi Kelompok dengan anggota 4-5 orang perkelompok

Berdasarkan uraian materi percabangan, lakukan diskusi bersama teman kelompokmu untuk memperdalam pengetahuan tentang percabangan. Dengan cara membuat *flowchart* dan *pseudocode* serta narasi untuk menentukan *grade* dan predikat dari nilai hasil ulangan harian siswa dengan kriteria sebagai berikut. Jika nilai yang diperoleh antara 0 sampai 64 maka akan mendapatkan *grade* D dengan predikat "Kurang", jika nilai yang diperoleh antara 65-79 maka akan mendapatkan *grade* C dengan predikat "Cukup", jika nilai yang diperoleh antara 80-89 maka akan mendapatkan *grade* B dengan predikat "Baik", dan jika nilai yang diperoleh antara 90-100 maka akan mendapatkan *grade* A dengan predikat "Sangat Baik".

### 3. Perulangan

Pada dasarnya setiap aktivitas yang kalian kerjakan secara berulang-ulang dapat disebut sebagai perulangan, seperti rutinitas berangkat sekolah, menanam padi di sawah, menanak nasi, dan lain sebagainya.

Perhatikan diagram berikut!



Gambar 1.15 Perulangan

Sumber: Kusmadi (2022)

Sebuah proses atau langkah akan diulang ke langkah sebelumnya sampai kondisi pengulangannya terpenuhi atau bernilai benar (*true*).



Gambar 1.16 Teh Manis

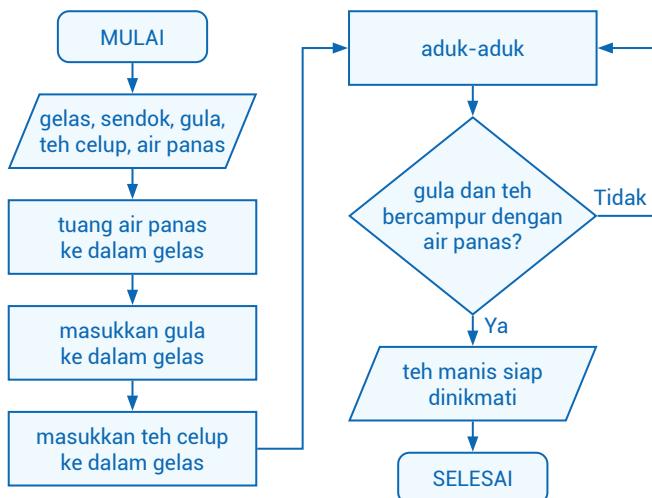
Sumber: Kompasiana (2022)

Contoh pada kegiatan ini misalnya kalian ingin membuat teh manis, dan agar gula yang dituangkan ke dalam gelas bisa larut maka harus diaduk sampai gula larut dalam air. Dalam studi kasus ini kalian menggunakan cara mengolah teh manis. Langkah-langkah pembuatan teh manis adalah sebagai berikut.

- Siapkan gelas, teh celup, gula pasir, dan air panas.
- Tuangkan air panas ke dalam gelas sampai mendekati bibir gelas.
- Tambahkan gula pasir ke dalam gelas secukupnya.
- Celupkan satu buah teh celup.
- Aduk-aduk.
- Aduk-aduk.
- Aduk-aduk.
- Aduk-aduk.
- Aduk-aduk.
- Teh manis siap dinikmati.

Tahapan 5-9 merupakan kegiatan yang bersifat *repetitive* (berulang-ulang) sehingga tidak efektif dalam kehidupan sehari-hari. Jadi kalian membutuhkan logika perulangan untuk membuat penulisan lebih efektif dan efisien.

Pembuatan diagram alirnya sebagai berikut.



1	PROGRAM Perulangan
2	DEKLARASI pilihan
3	
4	ALGORITMA
5	Siapkan gelas, sendok, gula
6	Siapkan teh_celup, air_panas
7	
8	Tuangkan air_panas kedalam gelas
9	Tuangkan gula kedalam gelas
10	Tuangkan teh_celup kedalam gelas
11	
12	Lakukan aduk_aduk sampai gula + air_panas + teh_celup telah bercampur
13	
14	teh manis siap dinikmati

Gambar 1.17 Flowchart dan Pseudocode Pembuatan Teh Manis

Sumber: Kusmadi (2022)

## Jenis Perulangan

Perulangan adalah instruksi khusus dalam bahasa pemrograman dan algoritma yang digunakan untuk mengulang beberapa perintah sesuai jumlah yang sudah ditentukan.

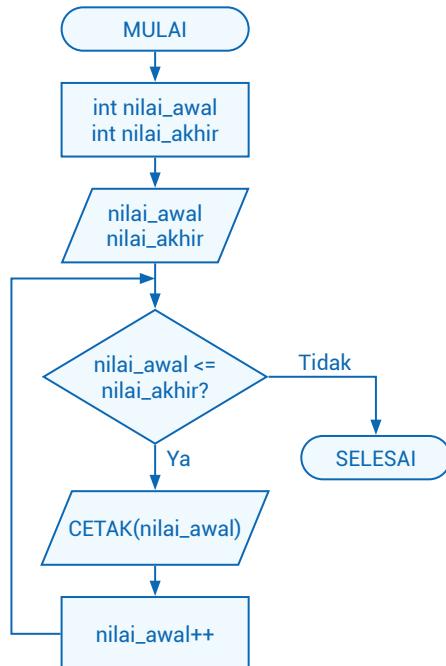
Instruksi pengulangan terdiri atas tiga bentuk seperti berikut.

### 1. Perulangan menggunakan instruksi FOR

Berikut bentuk perulangan FOR yang terdiri atas dua bentuk.

- Bentuk perulangan FOR to DO atau perulangan positif.

Bentuk perulangan dengan penghitung (*counter*) dari kecil ke besar disebut juga pertambahannya positif. Bentuk perulangan FOR to DO adalah sebagai berikut.



Gambar 1.18 FOR to DO Positif

Sumber: Kusmadi (2022)

Penulisannya sebagai berikut.

- a) Diawali dengan kata kunci FOR.
- b) Kemudian diikuti oleh variabel awal yang bernilai bilangan bulat, misal nilai\_awal=1.
- c) Kemudian diikuti oleh operator relasional.
- d) Kemudian diikuti oleh variabel akhir atau angka yang memiliki nilai lebih besar dari variabel nilai\_awal dengan nilai bilangan bulat, misal nilai\_akhir = 10.
- e) Apabila hasil perbandingan menghasilkan nilai benar, maka aksi pengulangan akan dikerjakan, dan apabila hasil perbandingan menghasilkan nilai salah, maka aksi pengulangan akan dihentikan.
- f) Setelah pernyataan dikerjakan, langkah berikutnya adalah menambah nilai variabel awal dengan angka yang telah ditentukan, misal nilai\_awal += 1 atau nilai\_awal++.
- g) Proses ini akan terus dilakukan hingga hasil perbandingan menghasilkan nilai salah. Ketika hal ini terjadi, perulangan akan dihentikan atau berakhir.



Dalam algoritma penambahan nilai disebut increment.

*Increment 1* (ditambah 1) dapat dituliskan dengan berbagai cara, seperti contoh berikut.

nilai\_awal++ artinya nilai variabel nilai\_awal + 1

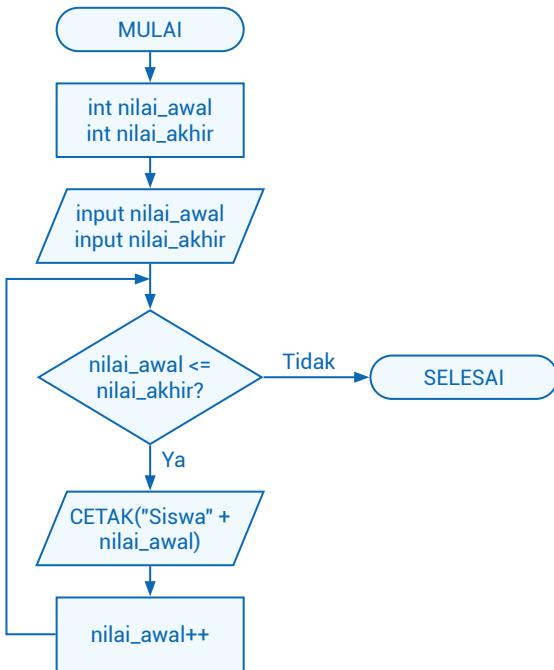
nilai\_awal+= 1

nilai\_awal = nilai\_awal + 1

perintah IF

Perhatikan studi kasus berikut!

Pak Bambang adalah seorang guru yang memiliki 5 orang siswa masing-masing bernama Siswa 1, Siswa 2, Siswa 3, Siswa 4 dan Siswa 5. Pak Bambang ingin membuat *flowchart* perulangan FOR to DO.



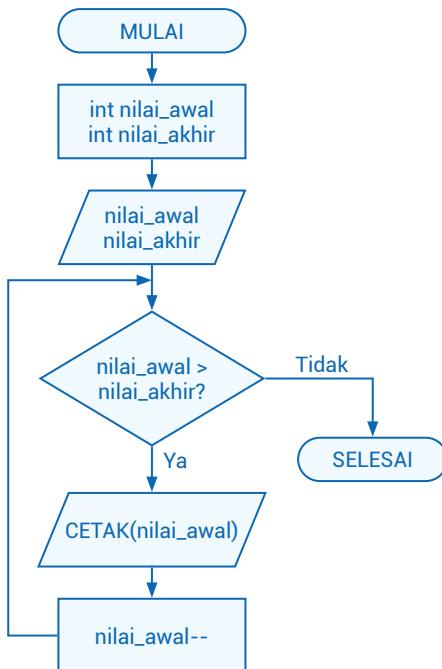
Gambar 1.19 Perulangan FOR Mencetak Nama Siswa

Sumber: Kusmadi (2022)

Penjelasannya sebagai berikut.

- Aktivitas dijalankan.
- Masukkan nilai bilangan bulat untuk variabel nilai\_awal =1 dan nilai\_akhir = 5.
- Memeriksa apakah nilai\_awal <= nilai\_akhir, jika YA maka nama siswa 1 akan dicetak.
- Kemudian nilai\_awal + 1 sehingga menjadi 2.
- Memeriksa apakah nilai\_awal <= nilai\_akhir, jika YA maka nama siswa 2 akan dicetak.
- Kemudian nilai\_awal + 1 sehingga menjadi 3.
- Memeriksa apakah nilai\_awal <= nilai\_akhir, jika YA maka nama siswa 3 akan dicetak.
- Kemudian nilai\_awal + 1 sehingga menjadi 4.
- Memeriksa apakah nilai\_awal <= nilai\_akhir, jika YA maka nama siswa 4 akan dicetak.
- Kemudian nilai\_awal + 1 sehingga menjadi 5.
- Memeriksa apakah nilai\_awal <= nilai\_akhir, jika YA maka nama siswa 5 akan dicetak.

- l) Kemudian nilai\_awal + 1 sehingga menjadi 6.
  - m) Memeriksa apakah nilai\_awal <= nilai\_akhir, jika TIDAK maka aktivitas dihentikan.
- Bentuk perulangan FOR down to DO atau perulangan negatif yaitu bentuk perulangan dengan penghitungan (*counter*) dari besar ke kecil atau disebut juga pertambahan negatif. Bentuk perulangan FOR down to DO adalah sebagai berikut.



Gambar 1.20 Perulangan FOR to DO Negatif

Sumber: Kusmadi (2022)

Penulisannya sebagai berikut.

- a) Diawali dengan kata kunci FOR.
- b) Kemudian diikuti oleh variabel awal yang bernilai bilangan bulat, misal nilai\_awal = 10.
- c) Kemudian diikuti oleh operator relasional.
- d) Kemudian diikuti oleh variabel akhir atau angka yang memiliki nilai lebih kecil dari variabel kondisi awal dengan nilai bilangan bulat, misal nilai\_akhir = 1.
- e) Apabila hasil perbandingan menghasilkan nilai benar maka *statement* akan dikerjakan, dan apabila hasil perbandingan menghasilkan nilai salah maka *statement* akan diabaikan.

- f) Setelah statement dikerjakan, langkah berikutnya adalah mengurangi nilai variabel awal dengan angka yang telah ditentukan, misal `nilai_awal = 1` atau `nilai_awal--`.
- g) Proses ini akan terus dilakukan hingga hasil perbandingan menghasilkan nilai salah. Ketika hal ini terjadi perulangan akan dihentikan atau berakhir.



Dalam algoritma pengurangan nilai disebut *decrement*.

*Decrement 1* (dikurangi 1) dapat ditulis dengan berbagai cara, seperti contoh berikut.

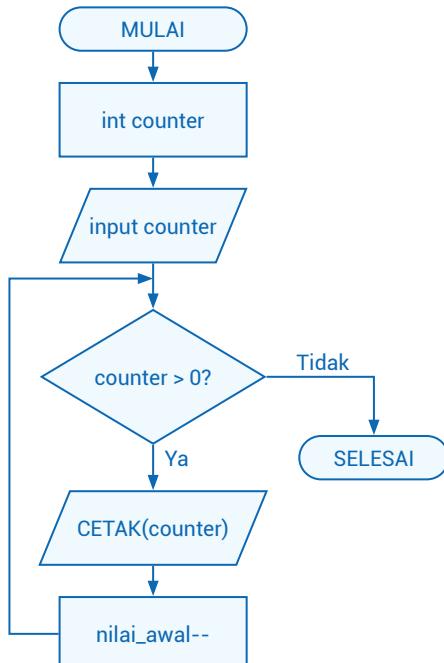
`nilai_awal--` artinya nilai variabel `nilai_awal - 1`

`nilai_awal= 1`

`nilai_awal = nilai_awal - 1`

Perhatikan studi kasus berikut!

Pak Abdul seorang guru olahraga di salah satu SMK di Indonesia. Beliau mengadakan kegiatan lomba lari sprint 100 meter kepada murid-muridnya dengan menghitung mundur dari tiga hingga satu. Jika dibuat dalam bentuk *flowchart* maka hasilnya adalah sebagai berikut.



Gambar 1.21 Perulangan FOR Mencetak COUNTER

Sumber: Kusmadi (2022)

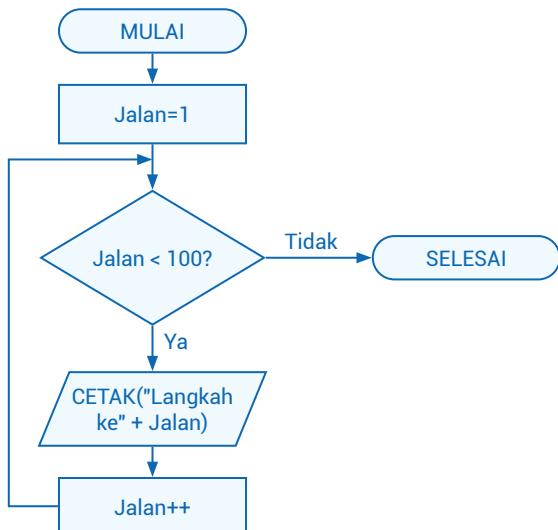
Berikut penjelasan *flowchart* di atas.

- a) Aktivitas dijalankan.
- b) Masukkan nilai bilangan bulat untuk variabel counter = 3.
- c) Memeriksa apakah counter > 0, jika YA maka angka 3 akan dicetak.
- d) Kemudian nilai awal - 1 sehingga menjadi 2.
- e) Memeriksa apakah counter > 0, jika YA maka angka 2 akan dicetak.
- f) Kemudian nilai awal - 1 sehingga menjadi 1.
- g) Memeriksa apakah counter > 0, jika YA maka angka 1 akan dicetak.
- h) Kemudian nilai awal - 1 sehingga menjadi 0.
- i) Memeriksa apakah counter > 0, jika TIDAK maka aktivitas dihentikan.

## 2. Perulangan menggunakan instruksi WHILE

WHILE, yaitu bentuk perulangan yang digunakan untuk melakukan pengulangan suatu *statement* atau blok *statement* selama kondisi bernilai benar. Dalam perulangan WHILE, suatu kondisi akan diperiksa terlebih dahulu sebelum sebuah aksi dilakukan.

Misalnya kalian harus berjalan 100 langkah, maka kalian harus melakukan aktivitas berjalan secara berulang-ulang. Jika aktivitas berjalan tersebut dibuat dalam bentuk *flowchart*, hasilnya menjadi seperti berikut.



Gambar 1.22 Perulangan WHILE

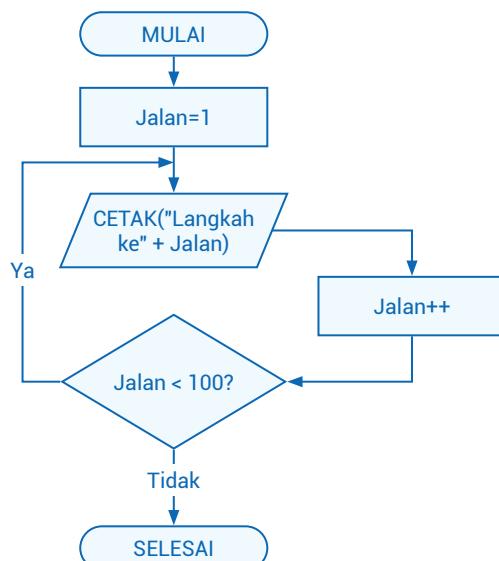
Sumber: Kusmadi (2022)

Penjelasannya adalah sebagai berikut.

- a. Ketika aktivitas dimulai, akan membuat inisialisasi variabel Jalan yang diisi angka.
  - b. Periksa apakah Jalan < 100.
  - c. Jika Ya, maka langkah akan dicetak kemudian Jalan ditambah 1.
  - d. Lakukan aktivitas b dan c hingga Jalan = 100.
  - e. Jika telah mencapai 100 maka aktivitas selesai.
3. **Perulangan menggunakan instruksi DO-WHILE**

Ini adalah jenis perulangan yang digunakan untuk mengulang pernyataan atau blok pernyataan selama suatu kondisi bernilai benar, dimulai dengan eksekusi pertama dari pernyataan awal. Kemudian dilanjutkan dengan pengecekan kondisi pernyataan bernilai benar atau salah untuk ke proses selanjutnya.

Berdasarkan contoh pada instruksi while jika diterapkan pada instruksi do-while maka bentuk *flowchart* yang dihasilkan seperti berikut!



Gambar 1.23 Perulangan DO-WHILE

Sumber: Kusmadi (2022)

Penjelasannya adalah sebagai berikut.

- a. Ketika aktivitas dimulai, inisialisasi variabel Jalan akan diisi angka 1.
- b. Langkah akan dicetak kemudian Jalan ditambah 1.
- c. Periksa apakah Jalan < 100.
- d. Jika Ya, maka lakukan aktivitas b dan c hingga Jalan = 100.
- e. Jika telah mencapai 100 maka aktivitas selesai.



## Aktivitas Pembelajaran



### Ayo Berlatih

Aktivitas individu untuk memperdalam pemahaman

Berdasarkan uraian materi di atas silahkan kalian jelaskan kembali tentang apa saja jenis perulangan yang dapat digunakan serta buatlah contoh tentang jenis perulangan yang pernah kalian gunakan dalam kehidupan sehari-hari!



### Ayo Berdiskusi

Diskusi Kelompok dengan anggota 4-5 orang perkelompok

Berdasarkan uraian materi perulangan, lakukan diskusi bersama teman kelompokmu untuk memperdalam pengetahuan tentang perulangan. Dengan cara membuat *flowchart* dan *pseudocode* dan narasi untuk mencetak bilangan ganjil pada rentang angka 0 sampai 20.

Gunakan skema pengulangan yang kalian pilih, serta manfaatkan notasi percabangan, sebab kalian hanya diminta mencetak bilangan dari 0 sampai 20, hanya jika bilangan tersebut ganjil.

## 1. Pengertian Pemrograman

Program komputer (biasa disebut program) adalah seperangkat instruksi yang ditulis untuk melakukan fungsi tertentu pada komputer. Sebuah komputer pada dasarnya membutuhkan kehadiran sebuah program untuk menjalankan fungsinya sebagai komputer dengan mengeksekusi serangkaian instruksi program pada prosesor. Program biasanya memiliki beberapa bentuk model eksekusi dan dapat dijalankan langsung di komputer.

Program yang sama dalam bentuk kode yang dapat dibaca manusia disebut kode sumber, yaitu suatu bentuk program yang memungkinkan pemrogram untuk menganalisis dan memverifikasi algoritma yang digunakan dalam program. Kode sumber akhirnya dikompilasi oleh utilitas bahasa pemrograman tertentu untuk membentuk sebuah program. Bentuk alternatif lain dari model eksekusi program adalah penggunaan interpreter. Dalam hal ini, kode sumber langsung dieksekusi oleh utilitas interpreter dari bahasa pemrograman yang digunakan.

Beberapa program komputer dapat berjalan secara bersamaan di komputer kalian. Kemampuan komputer untuk menjalankan beberapa program secara bersamaan disebut *multitasking*. Program komputer dapat diklasifikasikan menurut fungsinya.

Pemrograman komputer adalah proses berulang menulis dan mengedit kode sumber untuk membuat program. Pengeditan kode sumber meliputi pengujian, analisis, perbaikan bug, pengoptimalan algoritma, normalisasi kode, dan koordinasi satu pemrogram dengan pemrogram lain ketika program ditulis oleh banyak orang dalam satu tim. Praktisi dengan keahlian dalam menulis kode dalam bahasa pemrograman dikenal sebagai *programmer*, pengembang perangkat lunak atau *coders*. Istilah rekayasa perangkat lunak sering digunakan karena proses pembuatan program dianggap sebagai disiplin rekayasa.

## 2. Bahasa Pemrograman

Bahasa pemrograman adalah rangkaian kata berupa instruksi atau perintah yang dapat dimengerti oleh komputer, biasanya terdiri atas banyak baris. Pengembang harus menguasai bahasa pemrograman untuk membuat aplikasi atau perangkat lunak. Bahasa pemrograman juga digunakan untuk membuat aplikasi tertentu, tergantung dari kebutuhan aplikasi yang sedang dibuat.

Jumlah bahasa pemrograman sangat banyak. Tentu saja dari sekian banyak bahasa pemrograman yang ada saat ini tidak semuanya digunakan oleh developer. Saat mengembangkan perangkat lunak mereka hanya menggunakan satu atau dua bahasa pemrograman.

TIOBE adalah salah satu lembaga penelitian yang selama beberapa tahun mencoba menyusun peringkat bahasa pemrograman terpopuler di dunia, yang disajikan dalam bentuk TIOBE Programming Community Index. Pada edisi Oktober 2022, Java dan C terus menjadi bahasa pemrograman pertama dan terpopuler.

Python, Java, C, dan C++ telah menjadi empat bahasa teratas dalam indeks TIOBE selama beberapa waktu. Tahun lalu, keempat bahasa tersebut menguasai pangsa pasar 40%, dan tahun ini mencapai 55%.

Dalam buku ini, kalian akan mempelajari bahasa pemrograman C++ karena selain sebagai suksesor dari bahasa C, bahasa C++ juga mampu menangani pemrograman berorientasi objek dan struktural. Ini merupakan bekal yang cukup sebelum mempelajari bahasa pemrograman tingkat lanjut untuk pengembangan aplikasi desktop, website dan perangkat bergerak (Android/iOS).

### 3. Bahasa Pemrograman C++

#### a. Sejarah C++

C++ merupakan salah satu bahasa pemrograman tingkat tinggi yang paling populer saat ini. Bahasa pemrograman C++ banyak digunakan di berbagai bidang seperti pengembangan game, pengembangan perangkat lunak, dan keamanan informasi. Bjarne Stroustrup adalah orang yang membuat bahasa pemrograman C++. Orang Denmark ini lahir pada tanggal 30 Desember 1950. Dia adalah orang yang mengembangkan bahasa pemrograman C++.

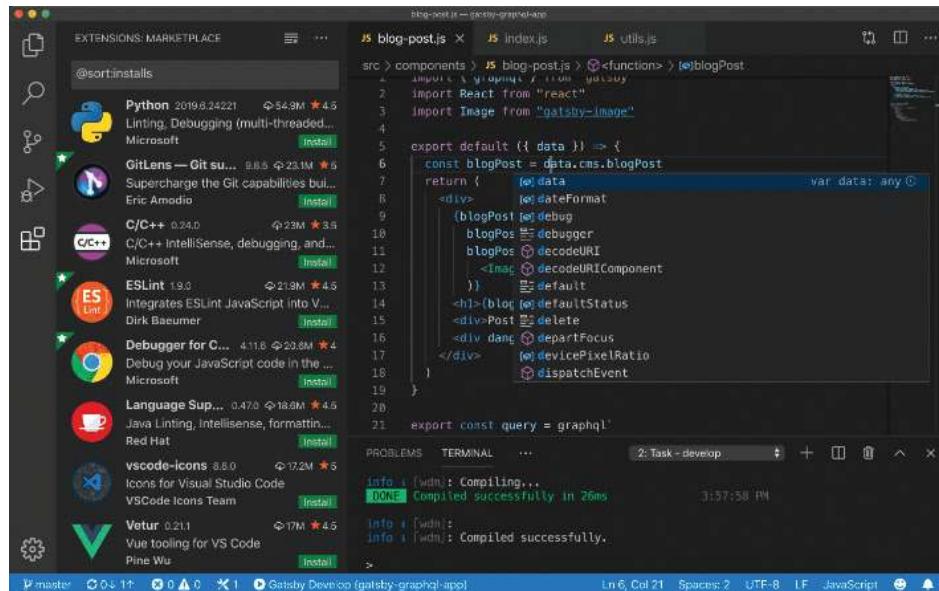
Stroustrup juga dikenal sebagai ilmuwan komputer ulung. Ia meraih gelar master dalam bidang matematika dan ilmu komputer dari Universitas Aarhus. Sedangkan PhD-nya (S3), dia mendapatkannya dari Universitas Cambridge di bidang yang sama.

Asal-usul bahasa pemrograman C++ berasal dari ide dan usahanya untuk membuat bahasa pemrograman C yang berjalan di ranah prosedural menjadi bahasa pemrograman berorientasi objek menggunakan Class.

## b. Aplikasi Editor C++

Aplikasi editor yang beredar di dunia maya sangat banyak jumlahnya. Dalam buku ini, kalian akan menggunakan 1 aplikasi editor pemrograman, yaitu Microsoft Visual Studio Code.

Aplikasi ini mendukung banyak bahasa pemrograman dengan fitur yang cukup lengkap dan banyak digunakan oleh berbagai developer di dunia.



Gambar 1.24 Microsoft Visual Studio Code (VSCode)

Sumber: Kusmadi (2022)

Sesuai dengan namanya, Microsoft Visual Studio Code (VSCode) dikembangkan oleh raksasa teknologi, Microsoft. Kalian dapat mengunduh aplikasi ini di laman <https://code.visualstudio.com>.

VSCode merupakan salah satu editor paling stabil dan kaya fitur yang tersedia untuk Windows, Linux, dan MacOS. Editor ini didasarkan pada kerangka Electron. Berbicara tentang fitur, VSCode memiliki semua fitur penting seperti penyelesaian kode cerdas, penyorotan sintaks, pemfaktoran ulang kode, dukungan cuplikan, kemampuan *debugging*, kontrol Git bawaan, dan banyak lagi. Informasi lebih detail dapat dilihat pada laman berikut <https://s.id/microsoft-visual-studio-code>.

## 4. Tipe Data

Tipe data adalah atribut yang terkait dengan sepotong data yang memberi tahu sistem komputer bagaimana menafsirkan nilai itu. Memahami tipe data memastikan bahwa data ditangkap dalam format yang tepat dan setiap properti memiliki nilai yang sesuai.

Berikut adalah jenis tipe data yang umum ada di setiap bahasa pemrograman.

Tabel 1.5 Tipe Data

Tipe Data	Definisi	Contoh
Integer (int)	Tipe data numerik untuk angka tanpa pecahan.	-707, 0, 707
Floating Point (float)	Tipe data numerik untuk angka dengan pecahan.	707.07, 0.7, 707.00
Character (char)	Huruf tunggal, digit, tanda baca, simbol, atau spasi kosong.	a, 1, !
String (str or text)	Urutan karakter, digit, atau simbol—selalu diperlakukan sebagai teks.	hello, +1-999-666-3333
Boolean (bool)	Nilai benar atau salah.	0 (false), 1 (true)
Enumerated type (enum)	Kumpulan kecil nilai unik yang telah ditentukan sebelumnya (elemen atau enumerator) yang dapat berbasis teks atau numerik.	rock (0), jazz (1)
Array	Daftar dengan sejumlah elemen dalam urutan tertentu—biasanya dari jenis yang sama.	rock (0), jazz (1), blues (2), pop (3)
Date	Tanggal dalam format YYYY-MM-DD (sintaks ISO 8601).	2021-09-28
Time	Waktu dalam format hh:mm:ss untuk waktu, waktu sejak acara, atau interval waktu antar peristiwa.	12:00:59
Datetime	Tanggal dan waktu bersama dalam format YYYY-MM-DD hh:mm:ss.	2021-09-28 12:00:59
Timestamp	Jumlah detik yang telah berlalu sejak tengah malam (00:00:00 UTC), 1 Januari 1970 (waktu Unix).	1632855600

## 5. Variabel

Variabel adalah alokasi memori komputer yang nilainya berubah-ubah. Secara teknis, variabel merujuk ke alamat di memori komputer (RAM). Membuat variabel mempersiapkan lokasi memori untuk menyimpan nilainya. Setiap variabel memiliki nama yang digunakan sebagai ID variabel.

Seperti namanya, isi variabel dapat diubah sepanjang kode program. Misalnya, jika kalian menulis program untuk menghitung luas persegi, kalian dapat mengatur variabel panjang dan lebar untuk diisi dengan angka 7 dan 4. Kalian kemudian dapat mengganti isi variabel panjang dan lebar dengan 17, 15, atau angka lainnya.

Variabel juga biasa digunakan untuk menyimpan nilai *input*. Misalnya, jika pengguna (pengguna aplikasi) perlu memasukkan nilai bujur dan lintang melalui aplikasi yang sedang berjalan.

### a. Aturan Penamaan Variabel

Bahasa pemrograman C++ menganut *case sensitive*, artinya huruf besar dan kecil dianggap berbeda. Dalam bahasa pemrograman C++ terdapat beberapa aturan yang harus dipatuhi dalam membuat sebuah variabel seperti berikut.

- 1) Variabel dapat terdiri atas huruf, angka, dan garis bawah (\_).
- 2) Karakter pertama variabel hanya boleh berupa huruf, garis bawah (\_ ) tidak boleh berupa angka. Meskipun penggunaan garis bawah diperbolehkan, tetapi jangan gunakan garis bawah di depan variabel karena dapat bertentangan dengan beberapa variabel konfigurasi program.
- 3) Variabel tidak boleh berupa kata kunci. Misalnya, kalian tidak dapat menggunakan kata int sebagai nama variabel karena int adalah kata kunci yang menunjukkan bahwa tipe datanya bilangan bulat.
- 4) Beberapa *compiler* C++ membatasi panjang variabel hingga maksimum 31 karakter. Nama variabel tidak boleh melebihi 31 karakter untuk keamanan.

## b. Cara Penulisan Variabel

Menulis variabel memiliki dua langkah di hampir semua bahasa pemrograman, yaitu deklarasi dan inisialisasi. Deklarasi adalah proses memberitahu *compiler* C++ bahwa kalian akan membuat sebuah variabel. C++ adalah bahasa pemrograman yang menggunakan konsep *strongly typed programming language*. Ini berarti bahwa untuk setiap variabel kalian harus menjelaskan jenis data apa yang dikandungnya. Apakah itu bilangan bulat (*integer*), desimal (*float/double*), karakter (*char*), atau yang lain.

Berikut tipe data yang biasanya digunakan.

- 1) Tipe data *integer*. Tipe data *integer* seperti 1, 5, 9. Tipe data *integer* dijelaskan menggunakan kata kunci *int*.
- 2) Tipe data *double*. Tipe data desimal seperti 3,14; 0,8; 22,7. Tipe data *double* dijelaskan menggunakan kata kunci *double*.
- 3) Tipe data *character*, yaitu tipe data karakter seperti 'A', 'a', 'Z'. Tipe data *character* dideskripsikan menggunakan kata kunci *char*.
- 4) Tipe data *string*. Tipe data untuk menyimpan set karakter. "belajar", "informatika", "belajar informatika di dunia C++", dan lain sebagainya. Tipe data *string* dijelaskan menggunakan kata kunci *String*.

Berikut adalah contoh cara mendeklarasikan variabel.

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int bilangan_bulat;
7     double phi;
8     char jenis_kelamin;
9     string sekolah;
10
11    return 0;
12 }
```

Gambar 1.25 Cara Mendeklarasikan Variabel

Sumber: Kusmadi (2022)

Penjelasannya adalah sebagai berikut.

- a. Pada baris ke-6, akan terbentuk sebuah variabel bernama bilangan\_bulat dengan tipe data int. Artinya, variabel bilangan\_bulat hanya bisa diisi oleh angka saja.
- b. Pada baris ke-7, akan terbentuk sebuah variabel bernama phi dengan tipe data double. Artinya, variabel phi dapat menerima angka pecahan, misalnya 3,14.
- c. Pada baris ke-8, akan terbentuk sebuah variabel bernama jenis\_kelamin dengan tipe data char. Artinya, variabel jenis\_kelamin hanya dapat menerima 1 karakter saja seperti p/l.
- d. Pada baris ke-9, akan terbentuk sebuah variabel bernama sekolah dengan tipe data string. Artinya, variabel sekolah dapat menerima semua karakter dari huruf, angka dan simbol.

Setelah sebuah variabel dideklarasikan, kalian dapat memberi nilai awal ke dalam variabel tersebut. Proses pemberian nilai awal ini disebut inisialisasi. Nilai yang diberikan harus sesuai dengan tipe data yang telah dideklarasikan, misalnya variabel dengan tipe data *int* harus mendapatkan nilai bilangan bulat sedangkan tipe data *char* harus mendapatkan nilai berupa 1 buah karakter. Lebih jelasnya, perhatikan contoh berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int bilangan_bulat;
7     double phi;
8     char jenis_kelamin;
9     string sekolah;
10
11     bilangan_bulat = 99;
12     phi = 3.14;
13     jenis_kelamin = 'p';
14     sekolah = "Kemendikbud";
15
16     return 0;
17 }
```

Gambar 1.26 Cara Menginisialisasi Variabel

Sumber: Kusmadi (2022)

Dari potongan kode di atas, terlihat setiap variabel telah memiliki nilai (lihat baris kode 11-14). Tanda sama dengan (=) berfungsi sebagai operator penugasan, yaitu operator yang akan memberikan nilai. Penulisan operator

penugasan ditulis dari kiri ke kanan. Phi = 3.14 artinya, variabel *phi* yang bertipe data *double* diberi nilai 3.14. Dalam pemrograman, simbol titik (.) berarti koma.

Untuk menampilkan nilai dari setiap variabel, kalian dapat menggunakan perintah cout seperti contoh berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int bilangan_bulat;
7     double phi;
8     char jenis_kelamin;
9     string sekolah;
10
11    bilangan_bulat = 99;
12    phi = 3.14;
13    jenis_kelamin = 'p';
14    sekolah = "Kemendikbud";
15
16    cout<<bilangan_bulat<<endl;
17    cout<<phi<<endl;
18    cout<<jenis_kelamin<<endl;
19    cout<<sekolah<<endl;
20
21    return 0;
22 }
```

Gambar 1.27 Cara Menampilkan Data ke Layar

Sumber: Kusmadi (2022)

Dari potongan kode di atas, jika dijalankan akan menghasilkan luaran sebagai berikut.

```
99
3.14
p
Kemendikbud
```

```

1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int bilangan_bulat;
7     double phi;
8     char jenis_kelamin;
9     string sekolah;
10
11    bilangan_bulat = 19;
12    phi = 3.14;
13    jenis_kelamin = 'p';
14    sekolah = "Kemendikbud";
15
16    cout<<"Usiamu adalah "<<bilangan_bulat<<" tahun"<<endl;
17    cout<<"Nilai PHI adalah "<<phi<<endl;
18    cout<<"Jenis Kelamin "<<jenis_kelamin<<endl;
19    cout<<"Asal Sekolah"<<sekolah<<endl;
20
21    return 0;
22 }

```

**Gambar 1.28** Cara Menampilkan Data ke Layar

Sumber: Kusmadi (2022)

Dari potongan kode di atas jika dijalankan akan menghasilkan luaran sebagai berikut.

Usiamu adalah 19 tahun  
 Nilai PHI adalah 3.14  
 Jenis Kelamin p  
 Asal Sekolah Kemendikbud

Dari kedua potongan kode di atas ada beberapa perbedaan, yaitu:

- *cout* adalah perintah untuk menampilkan data;
- *endl* atau “\n” adalah perintah untuk pindah baris;
- setiap perintah harus diakhiri oleh titik koma (;); dan
- untuk mencetak sembarang data (bertipe *string*) tanpa deklarasi variabel, harus diapit oleh dua tanda kutip.

Penulisan deklarasi variabel dan inisialisasi dapat pula dilakukan dalam satu baris. Agar lebih jelas, perhatikan contoh berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int bilangan_bulat = 19;
7     double phi = 3.14;
8     char jenis_kelamin = 'p';
9     string sekolah = "Kemendikbud";
10
11    cout<<"Usiamu adalah "<<bilangan_bulat<<" tahun"<<endl;
12    cout<<"Nilai PHI adalah "<<phi<<endl;
13    cout<<"Jenis Kelamin "<<jenis_kelamin<<endl;
14    cout<<"Asal Sekolah "<<sekolah<<endl;
15
16    return 0;
17 }
```

Gambar 1.29 Cara Mendeklarasikan dan Menginisialisasi Variabel secara Bersamaan

Sumber: Kusmadi (2022)

Dari potongan kode di atas, jika dijalankan akan menghasilkan luaran sebagai berikut.

Usiamu adalah 19 tahun  
Nilai PHI adalah 3.14  
Jenis Kelamin p  
Asal Sekolah Kemendikbud

Untuk memberi nilai pada program yang telah berjalan, perintah yang dapat digunakan adalah cin dan getline. Cin dapat menerima semua huruf dan angka, hanya saja khusus tipe data *string* hanya dapat menerima 1 kata. Berbeda dengan *getline*, selain dapat menangani semua yang ada pada *cin*, *getline* dapat menerima banyak kata dalam *input* (masukan).

Agar lebih jelas, perhatikan potongan kode berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     string nama, kelas;
7     int usia;
8
9     cout<<"Nama kamu siapa ?"<<endl;
10    cout<<"Nama saya ";
11    getline(cin, nama);
12    cout<<endl;
13
14    cout<<"Kamu kelas berapa ?"<<endl;
15    cout<<"Saya kelas ";
16    getline(cin, kelas);
17    cout<<endl;
18
19    cout<<"Berapa usiamu ?"<<endl;
20    cout<<"Usia saya ";
21    cin>>usia;
22    cout<<endl;
23
24    cout<<"Selamat Datang "<<nama<<endl;
25    cout<<"Waaaah usia "<<usia<<" tahun sudah kelas "<<kelas<<" ya"<<endl;
26    cout<<"Selamat Belajar C++"<<endl;
27
28    return 0;
29 }
```

Gambar 1.30 Memberi Nilai Variabel saat Program Berjalan

Sumber: Kusmadi (2022)

Hasil keluaran ketika program dijalankan adalah sebagai berikut.

```
Nama kamu siapa ?
Nama saya Kusmadi SmileCodes

Kamu kelas berapa ?
Saya kelas X SMK

Berapa usiamu ?
Usia saya 14

Selamat Datang Kusmadi SmileCodes
Waaaah usia 14 tahun sudah kelas X SMK ya
Selamat Belajar C++
```

## 6. Operator

Operator adalah sebuah simbol yang digunakan untuk melakukan operasi tertentu dalam pemrograman. Sebagai contoh, untuk menghitung luas persegi panjang  $P = 7$  dan  $L = 4$  diperlukan operator perkalian yang ditandai dengan simbol kali (\*).

Terdapat enam jenis operator dalam bahasa pemrograman C++, yaitu sebagai berikut.

### a. Operator Aritmatika

Yaitu operator yang digunakan untuk melakukan operasi aritmatika. Tugas operator ini tampak pada tabel 1.6.

Tabel 1.6 Operator Aritmatika

Nama Operator	Simbol
Penjumlahan	+
Pengurangan	-
Perkalian	*
Pembagian	/
Sisa Bagi	%

Cara pakai dari operator di atas dapat dilihat pada potongan kode berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int a, b, c;
7     a = 5;
8     b = 2;
9
10    //Penjumlahan
11    c = a + b;
12    cout<< a << " + " << b << " =" << c << endl;
13
14    //Pengurangan
15    c = a - b;
16    cout<< a << " - " << b << " =" << c << endl;
17
18    //Perkalian
19    c = a * b;
20    cout<< a << " * " << b << " =" << c << endl;
21
22    //Pembagian
23    c = a / b;
24    cout<< a << " / " << b << " =" << c << endl;
25
26    //Sisa Hasil Bagi
27    c = a % b;
28    cout<< a << " % " << b << " =" << c << endl;
29
30    return 0;
31 }
```

Gambar 1.31 Operator Aritmatika

Sumber: Kusmadi (2022)

Kode program di atas jika dijalankan akan menghasilkan luaran berikut.

```
5 + 2 = 7  
5 - 2 = 3  
5 * 2 = 10  
5 / 2 = 2  
5 % 2 = 1
```

### b. Operator Penugasan

Yaitu operator yang berfungsi untuk memberikan nilai pada variabel. Terdapat sebelas operator penugasan, yaitu:

Tabel 1.7 Operator Penugasan

Nama Operator	Simbol
Pengisian Nilai	=
Pengisian dan Penambahan	+=
Pengisian dan Pengurangan	-=
Pengisian dan Perkalian	*=
Pengisian dan Pembagian	/=
Pengisian dan Sisa Bagi	%=
Pengisian dan Shift Left	<<=
Pengisian dan Shift Right	>>=
Pengisian dan Bitwise AND	&=
Pengisian dan Bitwise OR	=
Pengisian dan Bitwise XOR	^=

Cara pakai dari operator di atas dapat dilihat pada potongan kode berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int a, b, c;
7
8     // Memberi nilai dengan operator =
9     a = 5;
10    b = 7;
11
12    // Memberi sekaligus penambahan
13    b += a; // ini sama seperti b = b + a
14    cout<< " Hasil b += a adalah " << b << endl;
15
16    // Memberi sekaligus pengurangan
17    b -= a; // ini sama seperti b = b - a
18    cout<< " Hasil b -= a adalah " << b << endl;
19
20    // Memberi sekaligus perkalian
21    b *= a; // ini sama seperti b = b * a
22    cout<< " Hasil b *= a adalah " << b << endl;
23
24    // Memberi sekaligus pembagian
25    b /= a; // ini sama seperti b = b / a
26    cout<< " Hasil b /= a adalah " << b << endl;
27
28    // Memberi sekaligus sisa hasil bagi
29    b %= a; // ini sama seperti b = b % a
30    cout<< " Hasil b %= a adalah " << b << endl;
31
32    return 0;
33 }
```

Gambar 1.32 Code Program Operator Penugasan

Sumber: Kusmadi (2022)

Code program di atas jika dijalankan akan menghasilkan keluaran berikut.

```
Hasil b += a adalah 12
Hasil b -= a adalah 7
Hasil b *= a adalah 35
Hasil b /= a adalah 7
Hasil b %= a adalah 2
```

### c. Operator Pembanding

Operator pembanding dikenal juga sebagai operator relasi yaitu operator yang digunakan untuk membandingkan dua buah nilai. Operator ini terdiri atas enam jenis sebagai berikut.

Tabel 1.8 Operator Pembanding

Nama Operator	Simbol
Lebih Besar	>
Lebih Kecil	<
Sama dengan	==
Tidak Sama dengan	!=
Lebih Besar Sama dengan	>=
Lebih Kecil Sama dengan	<=

Nilai yang dihasilkan dari operasi perbandingan adalah benar (*true*) atau salah (*false*). Dalam Bahasa pemrograman C++, nilai benar (*true*) sama saja dengan 1, sedangkan nilai salah (*false*) sama saja dengan 0. Cara pakai dari operator di atas dapat dilihat pada potongan kode berikut.

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai_akademik, perilaku;
7     bool hasil;
8
9     cout<<"Berapa nilai akademik Budi : ";
10    cin>>nilai_akademik;
11
12    cout<<"Bagaimana perilaku Budi (1/0) : ";
13    cin>>perilaku;
14    cout<<endl;
15
16    // menggunakan operator AND
17    hasil = nilai_akademik > 75 && perilaku ==1;
18    cout<< "nilai_akademik > 75 && perilaku ==1 : " << hasil << endl;
19
20    // menggunakan operator OR
21    hasil = nilai_akademik > 75 || perilaku ==1;
22    cout<< "nilai_akademik > 75 || perilaku ==1 : " << hasil << endl;
23
24    // menggunakan operator NOT
25    hasil = !perilaku;
26    cout<< "!perilaku : " << hasil << endl;
27
28    return 0;
29 }
```

Gambar 1.33 Operator Pembanding

Sumber: Kusmadi (2022)

*Code* program di atas jika dijalankan akan menghasilkan luaran berikut.

```
a = 7  
b = 4  
a > b = 1  
a < b = 0  
a >= b = 1  
a <= b = 0  
a == b = 0  
a != b = 1
```

#### d. Operator Logika

Operator berfungsi untuk membuat operasi logika. Terdapat tiga operator logika, yaitu `&&`, `||`, dan `!`.

Tabel 1.9 Operator Logika

Nama Operator	Simbol
Logika AND	<code>&amp;&amp;</code>
Logika OR	<code>  </code>
Negasi/kebalikan	<code>!</code>

Misal terdapat dua kondisi yang mana keduanya harus diperiksa untuk mencari tahu apakah kedua kondisi tersebut bernilai benar (*true*) atau salah ( ).

- Apakah nilai yang kamu dapat  $> 75$ ?      *true*
- Apakah nilai perilakumu baik?                *true*

Nilai yang kamu dapat  $> 75$  dan berperilaku baik jika kedua kondisi tersebut bernilai true dan akan menghasilkan nilai 1.

Agar lebih jelas, perhatikan potongan kode berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai_akademik, perilaku;
7     bool hasil;
8
9     cout<<"Berapa nilai akademik Budi : ";
10    cin>>nilai_akademik;
11
12    cout<<"Bagaimana perilaku Budi (1/0) : ";
13    cin>>perilaku;
14    cout<<endl;
15
16    // menggunakan operator AND
17    hasil = nilai_akademik > 75 && perilaku ==1;
18    cout<< "nilai_akademik > 75 && perilaku ==1 : " << hasil << endl;
19
20    // menggunakan operator OR
21    hasil = nilai_akademik > 75 || perilaku ==1;
22    cout<< "nilai_akademik > 75 || perilaku ==1 : " << hasil << endl;
23
24    // menggunakan operator NOT
25    hasil = !perilaku;
26    cout<< "!perilaku : " << hasil << endl;
27
28    return 0;
29 }
```

Gambar 1.34 Operator Logika

Sumber: Kusmadi (2022)

Code program di atas jika dijalankan akan menghasilkan luaran berikut.

```
Berapa nilai akademik Budi : 89
Bagaimana perilaku budi (1/0) : 1

nilai_akademik> 75 && perilaku ==1 : 1
nilai_akademik> 75 || perilaku ==1 : 1
!perilaku : 0
```

### e. Operator Bitwise

Operator Bitwise adalah operator yang digunakan untuk operasi biner pada sebuah nilai. Operator bitwise terdiri atas empat jenis, yaitu:

Tabel 1.10 Operator Bitwise

Nama Operator	Simbol
AND	&
OR	
XOR	^
NOT/komplemen	~

Misal kalian punya nilai 3 dan 5 dalam bentuk desimal, maka dalam operator bitwise nilai 3 dan 5 harus diubah dahulu menjadi bentuk biner. Jika diketahui nilai biner dari 3 dan 5 adalah 0011 dan 0101, maka apabila dilakukan bitwise `&` terhadap nilai tersebut akan menghasilkan nilai 1.

Detail perhitungannya sebagai berikut.

$$3_{10} = 0011_2$$

$$5_{10} = 0101_2$$

#### Bitwise AND ( & )

Tabel 1.11 Operator Bitwise AND

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

Hasil dari Bitwise 3 & 5 => 0011 & 0101 adalah 0001.

## Bitwise OR ( | )

Tabel 1.12 Operator Bitwise OR

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

Hasil dari Bitwise  $3 \mid 5 \Rightarrow 0011 \mid 0101$  adalah 0111.

## Bitwise XOR ( ^ )

Tabel 1.13 Operator Bitwise OR

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

Operator XOR akan menghasilkan nilai 1 manakala nilai X dan Y tidak sama.

Hasil dari Bitwise  $3 \wedge 5 \Rightarrow 0011 \wedge 0101$  adalah 0110.

## Bitwise NOT ( ~ )

Tabel 1.14 Operator Bitwise NOT

X	$\sim X$
0	1
1	0

Operator NOT dikenal juga operator komplemen, yaitu operator yang akan menghasilkan nilai terbalik dari nilai biner aslinya. Hasil dari Bitwise  $\sim 3$  (0011) adalah 1100.

Agar lebih jelas, perhatikan potongan kode berikut!

**Gambar 1.35** Code Program Operator Bitwise

Sumber: Kusmadi (2022)

## Output:

Bitwise a(00000011) & b(00000101) = 1

Bitwise a(00000011) | b(00000101) = 7

Bitwise  $a(00000011) \wedge b(00000101) = 6$

$$\text{Bitwise } \sim a(1100) = -4$$

### f. Operator Khusus

Selain operator di atas, ada beberapa operator yang perlu diketahui. Berikut beberapa operator yang perlu diketahui.

**Tabel 1.15** Operator Khusus

Nama Operator	Simbol	Keterangan
Alamat memori	&	untuk mengambil alamat memori
Pointer	*	untuk membuat pointer
Ternary	? :	untuk membuat kondisi
Increment	++	untuk menambah 1
Decrement	--	untuk mengurangi 1

Operator “&” digunakan untuk mengambil alamat memori, sedangkan operator “\*” digunakan untuk membuat pointer pada suatu variabel.

Agar lebih jelas, perhatikan potongan kode berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai = 5;
7     int *pointer_nilai = &nilai;
8
9     cout<<"Alamat memori variabel nilai adalah "<<&nilai<<endl;
10
11 //Mengubah nilai a dari pointer
12 cout<<"Nilai Awal = "<<nilai<<endl;
13 *pointer_nilai = 21;
14 cout << "Nilai Awal diubah = "<<nilai<<endl;
15
16 return 0;
17 }
```

Gambar 1.36 Operator Khusus

Sumber: Kusmadi (2022)

Output:

Alamat memori variabel nilai adalah 0x7ffee30fc9a8

Nilai Awal = 5

Nilai Awal diubah = 21

Kondisi *ternary* merupakan salah satu bentuk percabangan yang ditulis dalam satu baris. Perhatikan potongan kode berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai;
7
8     cout<<"Masukkan Nilai : ";
9     cin>>nilai;
10
11     string hasil = (nilai > 75)? "Selamat, kamu LULUS" : "Maaf, Silakan MENGULANG";
12     cout<< hasil <<endl;
13
14     return 0;
15 }
```

Gambar 1.37 Operator Ternary

Sumber: Kusmadi (2022)

Output percobaan 1:  
Masukkan nilai : 90  
Selamat, kamu LULUS

Output percobaan 2:  
Masukkan nilai : 75  
Maaf, Silakan MENGULANG



## Aktivitas Pembelajaran



### Ayo Berlatih

Aktivitas individu untuk memperdalam pemahaman

Untuk lebih memahami uraian materi di atas silahkan kalian lengkapi isian pada tabel berikut!

Tabel 1.16 Aktivitas Individu Tipe Data dan Operator

Tipe Data	Penjelasan
<i>Integer</i>	
<i>Double</i>	
<i>String</i>	
<i>Character</i>	
<i>Boolean</i>	

Operator	Simbol yang Digunakan
Aritmatika	
Pembanding	
Logika	
Penugasan	
Khusus	



## Ayo Berdiskusi

Diskusi Kelompok  
dengan anggota 4-5  
orang perkelompok

Berdasarkan uraian di atas tentang tipe data, variabel, dan operator, lakukan diskusi bersama teman kelompokmu untuk memperdalam pengetahuan tentang itu dengan studi kasus berikut ini. Pemilik toko Barokah ingin memberikan diskon sebesar 5% kepada pelanggannya yang belanja dengan total minimal Rp 75.000,00. Berdasarkan studi kasus tersebut, buatlah program penjualan sederhana untuk menghitung diskon dan total yang harus dibayarkan oleh pelanggan!

Lakukan identifikasi terlebih dahulu apa yang menjadi masukannya, bagaimana prosesnya, dan apa luarannya, kemudian identifikasi juga variabel dan tipe data yang diperlukan serta operator-operator yang terlibat. Setelah itu, buatlah algoritmanya dalam salah satu notasi.

## 7. Implementasi Sekuensial, Percabangan dan Pengulangan dalam Pemrograman

### a. Sekuensial

Sekuensial merupakan urutan pekerjaan dalam program. Setiap pekerjaan dikerjakan sesuai dengan tahapannya.

Misal:

Bambang diminta oleh ayahnya untuk menghitung luas dan keliling kamar tidurnya, jika diketahui panjang 7 m dan lebar 4 m. Bambang mengerjakannya menggunakan bahasa pemrograman C++. Berikut adalah hasil pekerjaannya!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int panjang, lebar;
7     int luas, keliling;
8
9     //langkah 1: memasukkan nilai panjang
10    cout<<"Masukkan panjang kamar : ";
11    cin>>panjang;
12
13    //langkah 2: memasukkan nilai lebar;
14    cout<<"Masukkan lebar kamar : ";
15    cin>>lebar;
16
17    //langkah 3: menghitung luas kamar tidur
18    luas = panjang * lebar;
19
20    //langkah 4: menghitung keliling kamar tidur
21    keliling = 2 * (panjang + lebar);
22
23    //Menampilkan luas ke layar
24    cout<<"Luas kamar tidur adalah "<<luas<<" m2"<<endl;
25
26    //Menampilkan keliling ke layar
27    cout<<"Keliling kamar tidur adalah "<<keliling<<" m"<<endl;
28
29    return 0;
30 }
```

Gambar 1.38 Sekuensial

Sumber: Kusmadi (2022)

*Output percobaan 1:*

Masukkan panjang kamar : 4

Masukkan lebar kamar : 3

Luas kamar tidur adalah 12 m<sup>2</sup>

Keliling kamar tidur adalah 14 m

*Output* percobaan 2:

Masukkan panjang kamar : 7

Masukkan lebar kamar : 5

Luas kamar tidur adalah 35 m<sup>2</sup>

Keliling kamar tidur adalah 24 m

### b. Percabangan

Percabangan atau juga dikenal sebagai *decision*, *control flow*, struktur kondisi, struktur IF dan lain sebagainya merupakan penggambaran untuk struktur program yang bercabang. Dalam Bahasa pemrograman C++ terdapat lima jenis percabangan sebagai berikut.

#### 1) Percabangan IF

Percabangan IF ialah percabangan yang hanya memiliki satu blok pilihan pada saat kondisi bernilai benar, maka instruksi dikerjakan. Agar lebih jelas, perhatikan kode program berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     unsigned int belanja;
7
8     cout<<"Total Belanja : ";
9     cin>>belanja;
10
11    if(belanja >= 100000){
12        double diskon = 0.1;
13        cout<<"Selamat, kamu mendapatkan diskon "<<(diskon*100)<<% sebesar : Rp. "<<(belanja*diskon)<<endl;
14        belanja = belanja - (belanja*diskon);
15    }
16
17    cout<<"Total belanja yang harus kamu bayar adalah Rp. "<<belanja<<endl;
18    cout<<"Terima Kasih Sudah Berbelanja";
19
20    return 0;
21 }
```

Gambar 1.39 Percabangan IF

Sumber: Kusmadi (2022)

*Output* percobaan 1:

Total Belanja : 120000

Selamat, kamu mendapatkan diskon 10% sebesar : Rp 12000

Total belanja yang harus kamu bayar adalah Rp 108000

Terima Kasih Sudah Berbelanja

*Output* percobaan 2:

Total Belanja : 90000

Total belanja yang harus kamu bayar adalah Rp 90000

Terima Kasih Sudah Berbelanja

## 2) Percabangan IF – ELSE

IF – ELSE adalah percabangan dengan dua blok pilihan. Dalam bentuk ini, pernyataan pertama dijalankan jika nilai kondisinya benar (*true*), dan pernyataan kedua dijalankan jika kondisinya salah (*false*). Agar lebih jelas, perhatikan kode program berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int belanja;
7     double diskon;
8
9     cout<<"Total Belanja : ";
10    cin>>belanja;
11
12    if(belanja >= 100000){
13        diskon = 0.1;
14        cout<<"Selamat, Kamu mendapatkan diskon "<<(diskon*100)<<% sebesar : Rp. "<<(belanja*diskon)<<endl;
15        belanja = belanja - (belanja*diskon);
16    }else{
17        cout<<"Maaf! Kamu belum mendapatkan diskon"<<endl;
18    }
19
20    cout<<"Total belanja yang harus kamu bayar adalah Rp. "<<belanja<<endl;
21    cout<<"Terima Kasih Sudah Berbelanja";
22
23    return 0;
24 }
```

Gambar 1.40 Percabangan IF – ELSE

Sumber: Kusmadi (2022)

*Output* percobaan 1:

Total Belanja : 110000

Selamat, kamu mendapatkan diskon 10% sebesar : Rp 11000

Total belanja yang harus kamu bayar adalah Rp 99000

Terima Kasih Sudah Berbelanja

*Output* percobaan 2:

Total Belanja : 80000

Maaf! Kamu belum mendapatkan diskon

Total belanja yang harus kamu bayar adalah Rp 80000

Terima Kasih Sudah Berbelanja

Contoh lain penggunaan IF – ELSE untuk konfirmasi *username* dan *password* seperti pada kode program berikut.

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     string username = "admin";
7     string password = "12345";
8
9     cout<<"Masukkan Username : ";
10    cin>>username;
11
12    cout<<"Masukkan Password : ";
13    cin>>password;
14
15 if(username == "admin" && password == "12345"){
16     cout<<"Selamat Datang "<<username<<endl;
17 }else {
18     cout<<"Maaf!\nUsername dan atau Password kamu salah!"<<endl;
19 }
20
21 return 0;
22 }
```

Gambar 1.41 Percabangan IF – ELSE

Sumber: Kusmadi (2022)

*Output* percobaan 1:

Masukkan *Username*: admin

Masukkan *Password*: 12345

Selamat Datang Admin

*Output* percobaan 2:

Masukkan *Username*: guest

Masukkan *Password*: guest

Maaf!

*Username* dan atau *Password* kamu salah!

### 3) Percabangan IF – ELSE – IF

IF – ELSE – IF adalah percabangan dengan lebih dari dua blok pilihan. Agar lebih jelas perhatikan kode program berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai;
7     char lagi;
8
9     label:
10    cout<<"Masukkan nilai : ";
11    cin>>nilai;
12
13    if(nilai >= 90){
14        cout<<"Selamat! Kamu lulus memuaskan"<<endl;
15    }else if(nilai >= 75){
16        cout<<"Selamat! Kamu lulus";
17    }else{
18        cout<<"Maaf! Kamu belum lulus";
19    }
20
21    cout<<"Coba Lagi (y/t): ";
22    cin>>lagi;
23
24    if(lagi == 'y'){
25        goto label;
26    }
27
28    return 0;
29 }
```

Gambar 1.42 Percabangan IF – ELSE - IF

Sumber: Kusmadi (2022)

*Output:*

```
Masukkan nilai: 76
Selamat! Kamu lulus
Coba Lagi (y/t): y
Masukkan nilai: 74
Maaf! Kamu belum lulus
Coba Lagi (y/t): y
Masukkan nilai: 90
Selamat! Kamu lulus memuaskan
Coba Lagi (y/t): t
```

Pada contoh penggunaan perintah IF apabila perintah yang ada di dalamnya hanya satu baris saja maka kalian diperbolehkan untuk tidak menggunakan tanda kurung kurawal { } seperti contoh pada kode program berikut.

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai;
7     char lagi;
8
9     label:
10    cout<<"Masukkan nilai : ";
11    cin>>nilai;
12
13    if(nilai >= 90)
14        cout<<"Selamat! Kamu lulus memuaskan" << endl;
15    else if(nilai >= 75)
16        cout<<"Selamat! Kamu lulus";
17    else
18        cout<<"Maaf! Kamu belum lulus";
19
20    cout<<"Coba Lagi (y/t): ";
21    cin>>lagi;
22
23    if(lagi == 'y')
24        goto label;
25
26    return 0;
27 }
```

Gambar 1.43 Contoh Lain IF – ELSE - IF

Sumber: Kusmadi (2022)

*Output:*

```
Masukkan nilai: 76
Selamat! Kamu lulus
Coba Lagi (y/t): y
Masukkan nilai: 74
Maaf! Kamu belum lulus
Coba Lagi (y/t): y
Masukkan nilai: 90
Selamat! Kamu lulus memuaskan
Coba Lagi (y/t): t
```

#### 4) Percabangan Switch – Case

Percabangan ini merupakan bentuk lain dari percabangan IF – ELSE – IF yang memiliki lebih dari 2 blok kondisi. Agar lebih jelas perhatikan kode program berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai;
7     char lagi, grade;
8
9     label:
10    cout<<"Masukkan nilai : ";
11    cin>>nilai;
12
13    if(nilai >= 90) grade = 'A';
14    else if(nilai >= 80) grade = 'B';
15    else if(nilai >= 75) grade = 'C';
16    else grade = 'D';
17
18    switch(grade){
19        case 'A':
20            cout<<"Selamat! Kamu lulus memuaskan"<<endl;
21            break;
22        case 'B':
23        case 'C':
24            cout<<"Selamat! Kamu lulus";
25            break;
26        default:
27            cout<<"Maaf! Kamu belum lulus";
28            break;
29    }
30
31    cout<<"Coba Lagi (y/t): ";
32    cin>>lagi;
33
34    if(lagi == 'y') goto label;
35
36    return 0;
37 }
```

Gambar 1.44 Percabangan Switch – Case

Sumber: Kusmadi (2022)

*Output:*

```
Masukkan nilai: 89
Selamat! Kamu lulus
Coba Lagi (y/t): y
Masukkan nilai: 90
Selamat! Kamu lulus memuaskan
Coba Lagi (y/t): y
Masukkan nilai: 67
Maaf! Kamu belum lulus
Coba Lagi (y/t): t
```

Contoh lain penggunaan Switch – Case untuk menghasilkan *output* seperti di atas bisa kalian perhatikan kode program berikut.

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai;
7     char lagi;
8
9     label:
10    cout<<"Masukkan nilai : ";
11    cin>>nilai;
12
13    switch(nilai){
14        case 90 ... 100 :
15            cout<<"Selamat! Kamu lulus memuaskan" << endl;
16            break;
17        case 75 ... 89:
18            cout<<"Selamat! Kamu lulus";
19            break;
20        default:
21            cout<<"Maaf! Kamu belum lulus";
22            break;
23    }
24
25    cout<<"Coba Lagi (y/t): ";
26    cin>>lagi;
27
28    if(lagi == 'y') goto label;
29
30    return 0;
31 }
```

Gambar 1.45 Percabangan Switch – Case Rentang Nilai

Sumber: Kusmadi (2022)

*Output:*

```
Masukkan nilai: 89
Selamat! Kamu lulus
Coba Lagi (y/t): y
Masukkan nilai: 90
Selamat! Kamu lulus memuaskan
Coba Lagi (y/t): y
Masukkan nilai: 67
Maaf! Kamu belum lulus
Coba Lagi (y/t): t
```

Kode program pada gambar di atas merupakan contoh Switch – Case yang digunakan untuk membaca rentang nilai yang ditandai dengan titik sebanyak 3 (...) pada bagian Case.

## 5) Percabangan Bersarang (Nested IF)

Dalam Bahasa pemrograman C++, kalian diperkenan untuk membuat percabangan di dalam percabangan. Percabangan jenis ini disebut percabangan bersarang atau Nested IF. Agar lebih jelas perhatikan kode program berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int belanja;
7     char lagi, member='n';
8     double diskon;
9
10    label:
11        cout<<"Total Belanja : ";
12        cin>>belanja;
13        cout<<"Punya Kartu Member (y/t) : ";
14        cin>>member;
15
16    if(belanja >= 100000){
17        diskon = 0.1;
18        if(member == 'y'){
19            diskon = 0.15;
20        }
21        cout<<"Kamu Mendapatkan diskon : Rp. "<<(belanja*diskon)<<endl;
22        belanja = belanja - (belanja*diskon);
23    }
24
25    cout<<"Total belanja yang harus kamu bayar : Rp. "<<belanja<<endl;
26    cout<<"Terima Kasih"<<endl;
27
28    cout<<"Coba Lagi (y/t): ";
29    cin>>lagi;
30
31    if(lagi == 'y') goto label;
32
33    return 0;
34 }
```

Gambar 1.46 Percabangan Bersarang

Sumber: Kusmadi (2022)

*Output:*

Total Belanja : 100000

Punya Kartu Member (y/t) : y

Kamu Mendapatkan diskon : Rp 15000

Total belanja yang harus kamu bayar : Rp 85000

Terima Kasih!

Coba Lagi (y/t): y

Total Belanja : 100000

Punya Kartu Member (y/t) : t

Kamu Mendapatkan diskon : Rp 10000

Total belanja yang harus kamu bayar : Rp 90000

Terima Kasih!

Coba Lagi (y/t): y

Total Belanja : 90000

Punya Kartu Member (y/t) : t

Total belanja yang harus kamu bayar : Rp 90000

Terima Kasih!

Coba Lagi (y/t): t



## Aktivitas Pembelajaran



### Ayo Berlatih

Aktivitas individu untuk memperdalam pemahaman

Imam ingin menghitung berat badan ideal orang laki-laki menggunakan rumus sebagai berikut.

$$\text{Berat badan ideal (kilogram)} = \\ [\text{tinggi badan (sentimeter)} - 100] - [(\text{tinggi badan (sentimeter)} - 100) \times 10 \text{ persen}]$$

Bagaimana cara menghitung berat badan ideal berdasarkan tinggi badan yang sudah diketahui menggunakan bahasa pemrograman C++?



### Ayo Berdiskusi

Diskusi Kelompok dengan anggota 4-5 orang perkelompok

Berdasarkan uraian di atas tentang percabangan, lakukan diskusi bersama teman kelompokmu untuk memperdalam pengetahuan tentang percabangan. Pelajarilah studi kasus berikut ini. Pak Budi sebagai seorang guru mata pelajaran informatika ingin mengolah nilai ulangan yang telah dilakukan hari ini, dan untuk menampilkan predikat dari nilai akademik masing-masing peserta didiknya berdasarkan kriteria sebagai berikut.

- Nilai 90 sampai 100 mendapatkan *grade A* dengan predikat **Sangat Baik**

2. Nilai 80 sampai 89 mendapatkan *grade* **B** dengan predikat **Baik**
3. Nilai 65 sampai 79 mendapatkan *grade* **C** dengan predikat **Cukup**
4. Nilai 0 sampai 64 mendapatkan *grade* **D** dengan predikat **Kurang**

Berdasarkan studi kasus tersebut silahkan kalian buatkan program yang bisa digunakan untuk menampilkan kriteria seperti pada tabel di atas!

### c. Perulangan

Setelah kalian mempelajari berbagai materi di atas, tentunya kalian paham cara mencetak informasi ke layar. Lantas, bagaimana jika kalian diminta mencetak 10 informasi berisi “Belajar Bahasa Pemrograman C++” ?

Cara mudah yang kalian lakukan adalah menulis perintah cout<<”Belajar Bahasa Pemrograman C++”<<endl; sebanyak 10 kali, kan?

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6
7     cout<<"Belajar Bahasa Pemrograman C++"<<endl;
8     cout<<"Belajar Bahasa Pemrograman C++"<<endl;
9     cout<<"Belajar Bahasa Pemrograman C++"<<endl;
10    cout<<"Belajar Bahasa Pemrograman C++"<<endl;
11    cout<<"Belajar Bahasa Pemrograman C++"<<endl;
12    cout<<"Belajar Bahasa Pemrograman C++"<<endl;
13    cout<<"Belajar Bahasa Pemrograman C++"<<endl;
14    cout<<"Belajar Bahasa Pemrograman C++"<<endl;
15    cout<<"Belajar Bahasa Pemrograman C++"<<endl;
16    cout<<"Belajar Bahasa Pemrograman C++"<<endl;
17
18 }
```

Gambar 1.47 Perulangan

Sumber: Kusmadi (2022)

*Output* dari kode program tersebut adalah:

Hal ini bisa dianggap benar karena jumlahnya sedikit. Lalu, bagaimana jika jumlahnya 1000 atau sejuta baris? Apabila cara di atas tetap dilakukan, selain menghabiskan banyak waktu juga tidak efisien.



Program yang baik adalah program yang didalamnya terdapat sedikit baris kode tetapi memiliki fungsi yang efektif dan efisien.

Terdapat 3 jenis perulangan yang dapat dipakai dalam bahasa pemrograman C++.

### 1) Perulangan FOR

Perulangan FOR merupakan jenis perulangan yang memiliki batasan maksimal iterasi. Sebagai gambaran, perhatikan kode program berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai_awal;
7     int nilai_akhir= 10;
8
9     //bentuk perulangan for 1
10    for(nilai_awal = 1; nilai_awal <= nilai_akhir; nilai_awal++){
11        cout<<nilai_awal<<endl;
12    }
13
14    //bentuk perulangan for 2
15    for(nilai_awal = 1; nilai_awal <= nilai_akhir; ){
16        cout<<nilai_awal<<endl;
17        nilai_awal++;
18    }
19
20    //bentuk perulangan for 3
21    nilai_awal = 1;
22    for(; nilai_awal <= nilai_akhir; nilai_awal++){
23        cout<<nilai_awal<<endl;
24    }
25
26    //bentuk perulangan for 4
27    nilai_awal = 1;
28    for(; nilai_awal <= nilai_akhir;){
29        cout<<nilai_awal<<endl;
30        nilai_awal++;
31    }
32 }
```

Gambar 1.48 Perulangan FOR Penambahan Nilai

Sumber: Kusmadi (2022)

Hasil luaran apabila kode program tersebut dijalankan akan mencetak angka 1 sampai 10 sebanyak 4 kali dengan urutan hasil pencetakan seperti di bawah ini.

```
1
2
3
4
5
6
7
8
9
10
```

Penambahan nilai\_awal dengan cara nilai\_awal++ disebut *increment* 1 atau bisa juga ditulis menjadi nilai\_awal = nilai\_awal + 1 atau nilai\_awal += 1.

```
1 #include <iostream>
2
3 using namespace std;
4
5 * int main() {
6     int nilai_awal;
7     int nilai_akhir= 1;
8
9     //bentuk perulangan for 1
10    for(nilai_awal = 10; nilai_awal >= nilai_akhir; nilai_awal--){
11        cout<<nilai_awal<<endl;
12    }
13
14    //bentuk perulangan for 2
15    for(nilai_awal = 10; nilai_awal >= nilai_akhir; ){
16        cout<<nilai_awal<<endl;
17        nilai_awal--;
18    }
19
20    //bentuk perulangan for 3
21    nilai_awal = 10;
22    for(; nilai_awal >= nilai_akhir; nilai_awal--){
23        cout<<nilai_awal<<endl;
24    }
25
26    //bentuk perulangan for 4
27    nilai_awal = 10;
28    for(; nilai_awal >= nilai_akhir;){
29        cout<<nilai_awal<<endl;
30        nilai_awal--;
31    }
32 }
```

Gambar 1.49 Perulangan FOR Pengurangan Nilai

Sumber: Kusmadi (2022)

Hasil luaran apabila kode program tersebut dijalankan akan mencetak angka 10 sampai 1 sebanyak 4 kali dengan urutan hasil pencetakan seperti di bawah ini.

```
10
9
8
7
6
5
4
3
2
1
```

Pengurangan nilai\_awal dengan cara nilai\_awal-- disebut *decrement* 1 atau bisa juga ditulis menjadi nilai\_awal = nilai\_awal - 1 atau nilai\_awal -= 1.

Penulisan *increment* nilai\_awal++ dan *decrement* nilai\_awal-- hanya bisa digunakan jika proses *increment/decrement* menggunakan angka 1. Artinya penulisan nilai\_awal+2 atau nilai\_awal-2 tidak dapat dilakukan. Untuk mengingat kembali, kalian dapat mempelajari konsep *increment* dan *decrement* pada materi perulangan yang telah dijelaskan sebelumnya.

Format penulisan perulangan FOR ada 4, yaitu:

a) **Format 1**

```
for(nilai_awal = 1; nilai_awal <= nilai_akhir; nilai_awal++){
    cout<<nilai_awal<<endl;
}
```

Gambar 1.50 Perulangan FOR Format 1 Penambahan Nilai

Sumber: Kusmadi (2022)

Dalam format ini, perulangan FOR diawali dengan menuliskan kata *for* (huruf kecil) kemudian diikuti oleh variabel nilai\_awal yang memiliki nilai 1, kemudian nilai\_awal dibandingkan dengan nilai\_akhir (nilai\_awal <= nilai\_akhir), nilai\_akhir memiliki nilai 10. Jika hasil perbandingan ini menghasilkan nilai benar (*true*), maka baris perintah yang ada di dalam kode blok ( {}) dikerjakan. Setelah semua baris yang ada dalam kode blok selesai dikerjakan, maka nilai\_awal ditambah 1 dengan perintah nilai\_awal++.

```
//bentuk perulangan for 1
for(nilai_awal = 10; nilai_awal >= nilai_akhir; nilai_awal--){
    cout<<nilai_awal<<endl;
}
```

Gambar 1.51 Perulangan FOR Format 1 Pengurangan Nilai

Sumber: Kusmadi (2022)

Dalam format ini, perulangan FOR diawali dengan menuliskan kata *for* (huruf kecil) kemudian diikuti oleh variabel nilai\_awal yang memiliki nilai 10, kemudian nilai\_awal dibandingkan dengan nilai\_akhir (nilai\_awal >= nilai\_akhir), nilai\_akhir memiliki nilai 1. Jika hasil perbandingan ini menghasilkan nilai benar (*true*), maka baris perintah yang ada di dalam kode blok ( {}) dikerjakan. Setelah semua baris yang ada dalam kode blok selesai dikerjakan, maka nilai\_awal dikurangi 1 dengan perintah nilai\_awal--.

## b) Format 2

```
for(nilai_awal = 1; nilai_awal <= nilai_akhir; ){
    cout<<nilai_awal<<endl;
    nilai_awal++;
}
```

Gambar 1.52 Perulangan FOR Format 2 Penambahan Nilai

Sumber: Kusmadi (2022)

Dalam format ini, perulangan FOR diawali dengan menuliskan kata *for* (huruf kecil) kemudian diikuti oleh variabel nilai\_awal yang memiliki nilai 1, kemudian nilai\_awal dibandingkan dengan nilai\_akhir (nilai\_awal  $\leq$  nilai\_akhir), nilai\_akhir memiliki nilai 10. Jika hasil perbandingan ini menghasilkan nilai benar (*true*), maka baris perintah yang ada di dalam kode blok ( {}) dikerjakan. Setelah semua baris yang ada dalam kode blok selesai dikerjakan, maka nilai\_awal ditambah 1 dengan perintah nilai\_awal++.

Khusus untuk format ini, penulisan nilai\_awal++ ditulis tepat di bawah baris kode yang ada di dalam kode blok.

```
for(nilai_awal = 10; nilai_awal >= nilai_akhir; ){
    cout<<nilai_awal<<endl;
    nilai_awal--;
}
```

Gambar 1.53 Perulangan FOR Format 2 Pengurangan Nilai

Sumber: Kusmadi (2022)

Dalam format ini perulangan FOR diawali dengan menuliskan kata *for* (huruf kecil) kemudian diikuti oleh variabel nilai\_awal yang memiliki nilai 10, kemudian nilai\_awal dibandingkan dengan nilai\_akhir (nilai\_awal  $\geq$  nilai\_akhir), nilai\_akhir memiliki nilai 1. Jika hasil perbandingan ini menghasilkan nilai benar (*true*), maka baris perintah yang ada di dalam kode blok ( {}) dikerjakan. Setelah semua baris yang ada dalam kode blok selesai dikerjakan, maka nilai\_awal dikurangi 1 dengan perintah nilai\_awal--.

Khusus untuk format ini, penulisan nilai\_awal-- ditulis tepat di bawah baris kode yang ada di dalam kode blok.

### c) Format 3

```
nilai_awal = 1;
for(; nilai_awal <= nilai_akhir; nilai_awal++){
    cout<<nilai_awal<<endl;
}
```

Gambar 1.54 Perulangan FOR Format 3 Penambahan Nilai

Sumber: Kusmadi (2022)

Dalam format ini, perulangan FOR di awali dengan menginisialisasi nilai\_awal dengan nilai 1. Kemudian menuliskan kata *for* (huruf kecil) kemudian nilai\_awal dibandingkan dengan nilai\_akhir (nilai\_awal <= nilai\_akhir), nilai\_akhir yang memiliki nilai 10. Jika hasil perbandingan ini menghasilkan nilai benar (*true*), maka baris perintah yang ada di dalam kode blok ( {}) dikerjakan. Setelah semua baris yang ada dalam kode blok selesai dikerjakan, maka nilai\_awal ditambah 1 dengan perintah nilai\_awal++.

```
nilai_awal = 10;
for(; nilai_awal >= nilai_akhir; nilai_awal--){
    cout<<nilai_awal<<endl;
}
```

Gambar 1.55 Perulangan FOR Format 3 Pengurangan Nilai

Sumber: Kusmadi (2022)

Dalam format ini, perulangan FOR di awali dengan menginisialisasi nilai\_awal dengan nilai 10. Kemudian menuliskan kata *for* (huruf kecil) kemudian nilai\_awal dibandingkan dengan nilai\_akhir (nilai\_awal >= nilai\_akhir), nilai\_akhir yang memiliki nilai 1. Jika hasil perbandingan ini menghasilkan nilai benar (*true*), maka baris perintah yang ada di dalam kode blok ( {}) dikerjakan. Setelah semua baris yang ada dalam kode blok selesai dikerjakan, maka nilai\_awal dikurangi 1 dengan perintah nilai\_awal--.

#### d) Format 4

```
nilai_awal = 1;
for(; nilai_awal <= nilai_akhir;){
    cout<<nilai_awal<<endl;
    nilai_awal++;
}
```

Gambar 1.56 Perulangan FOR Format 4 Penambahan Nilai

Sumber: Kusmadi (2022)

Dalam format ini perulangan FOR diawali dengan menginisialisasi nilai\_awal dengan nilai 1. Kemudian menuliskan kata *for* (huruf kecil) kemudian diikuti oleh variabel nilai\_awal yang memiliki nilai 1, kemudian nilai\_awal dibandingkan dengan nilai\_akhir (nilai\_awal <= nilai\_akhir), nilai\_akhir yang memiliki nilai 10. Jika hasil perbandingan ini menghasilkan nilai benar (*true*), maka baris perintah yang ada di dalam kode blok ( {}) dikerjakan. Setelah semua baris yang ada dalam kode blok selesai dikerjakan maka nilai\_awal ditambah 1 dengan perintah nilai\_awal++.

Khusus untuk format ini, penulisan nilai\_awal++ ditulis tepat di bawah baris kode yang ada di dalam kode blok.

```
nilai_awal = 10;
for(; nilai_awal >= nilai_akhir;){
    cout<<nilai_awal<<endl;
    nilai_awal--;
}
```

Gambar 1.57 Perulangan FOR Format 4 Pengurangan Nilai

Sumber: Kusmadi (2022)

Dalam format ini, perulangan FOR diawali dengan menginisialisasi nilai\_awal dengan nilai 10. Kemudian menuliskan kata *for* (huruf kecil) kemudian nilai\_awal dibandingkan dengan nilai\_akhir (nilai\_awal <= nilai\_akhir), nilai\_akhir memiliki nilai 1. Jika hasil perbandingan ini menghasilkan nilai benar (*true*), maka baris perintah yang ada didalam kode blok ( {}) dikerjakan. Setelah semua baris yang ada dalam kode blok selesai dikerjakan, maka nilai\_awal dikurangi 1 dengan perintah nilai\_awal--.

Khusus untuk format ini, penulisan nilai\_awal-- ditulis tepat di bawah baris kode yang ada di dalam kode blok.

Dari keempat format perulangan FOR yang paling umum digunakan adalah format ke-1. Meskipun keempat format dapat dipakai, tetapi kalian diberikan kebebasan untuk menggunakan salah satu dari empat format tersebut. Hal terpenting adalah kalian dapat menyelesaikan permasalahan perulangan dengan menggunakan perulangan FOR.

Contoh:

Rumah kalian kedatangan tamu, misalnya wali kelas berkunjung ke rumah. Sebagai murid yang baik, kalian akan membuat teh manis sebagai suguhan.

Dari kasus di atas, bagaimana cara kalian menerapkannya ke dalam bentuk pemrograman C++?

Agar lebih jelas, perhatikan kode program berikut!

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     string gelas, sendok, air, gula, teh;
7
8     cout<<"MEMBUAT TEH MANIS"<<endl;
9     cout<<"-----"<<endl;
10    cout<<"Siapkan gelas (sudah/belum)";
11    getline(cin,gelas);
12    cout<<"Siapkan sendok (sudah/belum)";
13    getline(cin,sendok);
14    cout<<"Masukkan gula (sudah/belum)";
15    getline(cin,gula);
16    cout<<"Masukkan teh (sudah/belum)";
17    getline(cin,teh);
18    cout<<"Tuangkan air panas (sudah/belum)";
19    getline(cin,air);
20
21 for(int a= 1; a <= 10; a++){
22     cout<<"Diaduk - aduk"<<endl;
23 }
24
25 cout<<"Gula dan teh sudah bercampur dengan air panas"<<endl;
26 cout<<"Teh manis siap dihidangkan"<<endl;
27 }
```

Gambar 1.58 Perulangan FOR Membuat Teh Manis

Sumber: Kusmadi (2022)

*Output:*

MEMBUAT TEH MANIS

-----  
Siapkan gelas (sudah/belum): sudah

Siapkan sendok (sudah/belum): sudah

Masukkan gula (sudah/belum): sudah

Masukkan teh (sudah/belum): sudah

Tuangkan air panas (sudah/belum): sudah

Diaduk-aduk

Diaduk-aduk  
Diaduk-aduk  
Diaduk-aduk  
Diaduk-aduk  
Diaduk-aduk  
Diaduk-aduk  
Diaduk-aduk  
Diaduk-aduk  
Diaduk-aduk  
Gula dan teh sudah bercampur dengan air panas  
Teh manis siap dihidangkan

## 2) Perulangan While

Perulangan While adalah jenis perulangan yang akan terus berjalan selama kondisinya benar. Seperti pada contoh perulangan menggunakan FOR untuk mencetak angka 1 sampai 10, kalian bisa memperhatikan kode program berikut.

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai_awal = 1;
7     int nilai_akhir = 10;
8
9     cout<<"Mencetak Angka 1 sampai 10 dengan menggunakan While"<<endl;
10    while(nilai_awal <= nilai_akhir){
11        cout<<nilai_awal<<endl;
12        nilai_awal++;
13    }
14}
15 }
```

Gambar 1.59 Perulangan While

Sumber: Kusmadi (2022)

*Output:*

Mencetak angka 1 sampai 10 dengan menggunakan While

1  
2  
3  
4  
5  
6

```
7  
8  
9  
10
```

Ingatkah kalian ketika diberi tugas untuk menulis bilangan genap antara angka 1 sampai 20? Saat kalian menulis bilangan genap, kalian akan menulis angka 2, 4, 6, 8, 10, dan seterusnya. Hal itu terjadi karena yang kalian tulis masih lebih kecil dari angka 20 jadi kondisi benar (*true*), dan apabila sudah melebihi angka 20 maka kondisi salah (*false*) sehingga kalian akan berhenti menulis bilangan genap.

Demikian pula dalam perulangan While. Selama kondisi perulangannya benar (*true*), maka perulangan akan terus dikerjakan sampai kondisi perulangan bernilai salah (*false*).

Agar lebih jelas, perhatikan kode program berikut!

```
1 #include <iostream>  
2  
3 using namespace std;  
4  
5 int main() {  
6     int nilai_awal=1;  
7     int nilai_akhir= 20;  
8  
9     cout<<"Mencetak bilangan genap antara "<<nilai_awal<<" sampai "<<nilai_akhir<<endl;  
10    while (nilai_awal <= nilai_akhir){  
11        if(nilai_awal%2==0){  
12            cout<<nilai_awal<<endl;  
13        }  
14        nilai_awal++;  
15    }  
16}  
17 }
```

**Gambar 1.60** Perulangan While

Sumber: Kusmadi (2022)

*Output:*

Mencetak bilangan genap antara 1 sampai 20

```
2  
4  
6  
8  
10  
12  
14  
16  
18  
20
```

### 3) Perulangan Do-While

Perulangan Do-While memiliki kesamaan dengan perulangan While. Perbedaan dari kedua perulangan ini adalah perulangan Do-While akan melakukan iterasi terlebih dahulu sebanyak satu kali di awal.

Agar lebih jelas, perhatikan kode program berikut!

Mencetak angka 1 sampai dengan 10.

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai_awal = 1;
7     int nilai_akhir = 10;
8
9     cout<<"Mencetak Angka 1 sampai 10 dengan menggunakan Do While"<<endl;
10    do{
11        cout<<nilai_awal<<endl;
12        nilai_awal++;
13    }while(nilai_awal <= nilai_akhir);
14
15 }
```

Gambar 1.61 Perulangan Do-While

Sumber: Kusmadi (2022)

*Output:*

Mencetak angka 1 sampai 10 dengan menggunakan Do-While

```
1
2
3
4
5
6
7
8
9
10
```

Contoh kode program berikutnya mencetak bilangan genap antara 1 sampai 20.

```
1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int nilai_awal=1;
7     int nilai_akhir= 20;
8
9     cout<<"Mencetak bilangan genap antara "<<nilai_awal<<" sampai "<<nilai_akhir<<endl;
10    do{
11
12        if(nilai_awal%2==0){
13            cout<<nilai_awal<<endl;
14        }
15        nilai_awal++;
16
17    } while (nilai_awal <= nilai_akhir);
18 }
```

**Gambar 1.62** Perulangan Do-While

Sumber: Kusmadi (2022)

*Output:*

Mencetak bilangan genap antara 1 sampai 20

```
2
4
6
8
10
12
14
16
18
20
```



## Aktivitas Pembelajaran



### Ayo Berlatih

Aktivitas individu untuk memperdalam pemahaman

Udin ingin mencetak tulisan “Semangat Belajar untuk Menggapai Masa Depan” sebanyak 50 kali. Bagaimana cara menyelesaikan masalah tersebut dengan menggunakan Bahasa Pemrograman C++?



### Ayo Berdiskusi

Diskusi Kelompok dengan anggota 4-5 orang perkelompok

Berdasarkan uraian di atas tentang Perulangan, lakukan diskusi bersama teman kelompokmu untuk memperdalam pengetahuan tentang Perulangan. Dengan cara membuat program untuk menampilkan bilangan dengan kelipatan angka 5 pada rentang angka 1 sampai 50.



## Rangkuman

Algoritma adalah prosedur tertulis dan berurutan untuk memecahkan masalah komputer secara terstruktur.

Dalam pemrograman sederhana, algoritma adalah langkah pertama yang harus ditulis sebelum menulis program. Masalah yang dapat diselesaikan dalam pemrograman adalah masalah yang melibatkan perhitungan matematis.

Definisi algoritma dalam pemrograman adalah aliran yang digunakan untuk menghitung atau memecahkan masalah secara sistematis, dan dalam kegiatan pemrograman algoritma biasanya dianggap sebagai logika yang menentukan program yang akan ditulis.

Dalam pengertian lain, algoritma pemrograman adalah serangkaian proses yang diikuti dalam komputasi, terutama dalam program komputer, untuk memecahkan masalah lain.

Diagram alir adalah gambar atau diagram yang berisi satu atau lebih aliran, tetapi hanya dapat diterapkan secara berurutan atau berkelanjutan. *Flowchart* juga memiliki simbol tertentu yang dapat berupa gambar dari setiap aliran yang dihubungkan oleh panah.

Fungsi utama dari algoritma pada dasarnya adalah pemecahan masalah. Namun, algoritma ini memiliki banyak fitur dan kelebihan lainnya. Misalnya, ini membantu menyederhanakan program yang kompleks dan banyak dan memudahkan penulisan program yang dapat memecahkan masalah tertentu.

Bahasa pemrograman adalah bahasa yang digunakan *programmer* untuk berkomunikasi dengan komputer. Sebuah bahasa pemrograman terdiri atas beberapa konstruksi yang membentuk perintah tertentu. Perintah-perintah tersebut kemudian diterjemahkan ke dalam logika yang dapat dipahami oleh komputer.

Bahasa pemrograman ini adalah seperangkat aturan sintaksis dan semantik yang digunakan untuk mendefinisikan program komputer. Contoh: BASIC, C, C++, COBOL, FORTRAN, Ada, Pascal, Java, .NET, Assembly language.

Secara sederhana, pemrograman adalah membuat program pada komputer. Program ini dibuat dalam bentuk *website*, *software*, aplikasi Android, dan lain sebagainya. Pemrograman dimulai dalam beberapa tahap: menulis, menguji, merevisi, mengevaluasi, dan menguji ulang.



## Asesmen

### 1. Tugas Mandiri

- Buatlah sebuah narasi, *flowchart* dan *pseudocode* untuk menampilkan bilangan dengan kelipatan angka 7 dari 1 hingga 100.
- Implementasikan soal pada poin 1a ke dalam bahasa pemrograman C++.

### 2. Tugas Kelompok

Pemilik sebuah warung tradisional mengalami kesulitan dalam pembuatan nota (melakukan pendataan barang yang dibeli oleh pelanggan). Pemilik berencana meminta tolong kepada kalian untuk membuatkan daftar barang beserta harganya kemudian membuat catatan (nota) dengan memasukkan nama pelanggan, barang yang dibeli dan perhitungan total yang harus dibayar serta pemberian diskon sebesar 5% untuk pelanggan yang total belanjanya diatas Rp 90.000,00. Sebagai anak SMK/MAK yang telah mempelajari Algoritma dan Pemrograman, kalian merasa memiliki kewajiban untuk membantu meringankan beban pemilik warung tradisional tersebut.

Contoh hasil program sebagai berikut.

```
Nama Pelanggan : imam
1. Minyak Goreng 2lt      Rp. 35000
2. Gula Pasir 1Kg        Rp. 13000
3. Sabun Cuci            Rp. 18000
4. Tepung Terigu 1Kg     Rp. 15000
Silahkan Pilih Barang : 1
Masukkan Jumlah pembelian untuk Minyak Goreng : 3
Mau Belanja lagi? Y/T :y
1. Minyak Goreng 2lt      Rp. 35000
2. Gula Pasir 1Kg        Rp. 13000
3. Sabun Cuci            Rp. 18000
4. Tepung Terigu 1Kg     Rp. 15000
Silahkan Pilih Barang : 4
Masukkan Jumlah pembelian untuk Tepung Terigu : 1
Mau Belanja lagi? Y/T :t
Nama Pelanggan :imam
Daftar Barang yang dibeli
Minyak Goreng 2lt Rp. 35000 x 3 = Rp.105000
Tepung Terigu 1Kg Rp. 15000 x 1 = Rp.15000
Total Belanja Rp.120000
Diskon      Rp.60000
Total Bayar  Rp.60000
```

```
Nama Pelanggan : udin
1. Minyak Goreng 2lt      Rp. 35000
2. Gula Pasir 1Kg        Rp. 13000
3. Sabun Cuci            Rp. 18000
4. Tepung Terigu 1Kg     Rp. 15000
Silahkan Pilih Barang : 2
Masukkan Jumlah pembelian untuk Gula Pasir : 1
Mau Belanja lagi? Y/T :y
1. Minyak Goreng 2lt      Rp. 35000
2. Gula Pasir 1Kg        Rp. 13000
3. Sabun Cuci            Rp. 18000
4. Tepung Terigu 1Kg     Rp. 15000
Silahkan Pilih Barang : 3
Masukkan Jumlah pembelian untuk Sabun Cuci : 2
Mau Belanja lagi? Y/T :t
Nama Pelanggan :udin
Daftar Barang yang dibeli
Gula Pasir 1Kg      Rp. 13000 x 1 = Rp.13000
Sabun Cuci Rp. 18000 x 2 = Rp.36000
Total Belanja Rp.49000
Diskon      Rp.0
Total Bayar   Rp.49000
```

Gambar 1.63 Contoh Hasil Pembuatan Aplikasi Penjualan

Sumber: Kusmadi (2022)

Tahap awal yang akan kalian lakukan adalah membuat narasi untuk menyelesaikan masalah yang dihadapi pemilik warung tradisional kemudian menuangkan tulisan tersebut dalam bentuk *flowchart* yang pada akhirnya menerapkan ide yang telah tersusun rapi menjadi sebuah aplikasi berbasis teks.



## Refleksi

Untuk mengasah kemampuan belajar kalian, lakukanlah beberapa hal berikut.

1. Agar prestasi belajar materi Algoritma Pemrograman dapat tercapai, buatlah sebuah catatan singkat mengenai Percabangan dan Perulangan!
2. Buatlah rangkuman tentang tipe data, jenis-jenis simbol dalam *flowchart*!
3. Bersama dengan kelompok, jelaskanlah pengertian Algoritma dan Pemrograman beserta contohnya!
4. Setelah belajar materi Algoritma dan Pemrograman, buatlah solusi terhadap hal-hal yang kalian alami setiap hari!



## Pengayaan

1. Secara berkelompok, diskusikan manfaat Algoritma dan Pemrograman dalam kehidupan sehari-hari!
2. Secara berpasangan, buatlah daftar bahasa pemrograman yang paling banyak digunakan oleh developer seluruh dunia dalam membuat aplikasi!
3. Secara berkelompok, kembangkanlah pemahaman kalian tentang bahasa pemrograman dengan mencoba bahasa pemrograman lain seperti Java dan Python!