Media Nusa Creative
Anggota IKAPI (162/JTI/2015)
Bukit Cemara Tidar H5 No. 34 - Malang
Telp : 0341 - 563 149 / 08223 2121 888
Email : mnc.publishing.malang@gmail.com
Website : www.mncpublishing.com

9 786020 839479

Trija Fayeldi, M.Si
Tatik Retno Murniasih, S.Si., M.Pd

DASAR-DASAR PEMROGRAMAN KOMPUTER
DENGAN MENGGUNAKAN MATLAB

MNC PUBLISHING
FUTURE BOOKS WITH PASSION

DASAR-DASAR
PEMROGRAMAN KOMPUTER DENGAN MENGGUNAKAN MATLAB

Trija Fayeldi, M.Si
Tatik Retno Murniasih, S.Si., M.Pd

Dasar-Dasar Pemrograman Komputer Dengan Menggunakan MATLAB

Penyusun:

Trija Fayeldi, M.Si

Tatik Retno Murniasih, S.Si, M.Pd

Penyunting :

Amak Yunus E., M. Kom

Dasar-Dasar Pemrograman Komputer Dengan Menggunakan MATLAB

Penulis :
Trija Fayeldi, M.Si
Tatik Retno Murniasih, S.Si., M.Pd

Penyunting :
Amak Yunus E., M. Kom

Desain Cover :
Tim MNC Publishing

Cetakan I, 2016

Diterbitkan Oleh :

Media Nusa Creative
Anggota IKAPI 162/JTI/2015
Bukit Cemara Tidar H5 No. 34 Malang
Telp : 0341 - 563 149 / 08223 2121 888
Email : mnc.publishing.malang@gmail.com
Website : www.mncpublishing.com

ISBN : 978-602-0839-47-9

Kata Pengantar

Pemrograman komputer telah menjadi salah satu kemampuan yang dituntut untuk dikuasai oleh para mahasiswa dari beragam bidang ilmu, termasuk pula dalam bidang matematika. Pemrograman komputer tidak sekedar mampu menuliskan beragam syntax dari suatu bahasa pemrograman, tetapi yang lebih mendasar adalah mampu memikirkan algoritma penyelesaian dari suatu permasalahan. Dengan menguasai algoritma penyelesaiannya, suatu permasalahan dapat dipecahkan atau diimplementasikan ke dalam berbagai bahasa pemrograman yang ada. Buku ini akan mengupas masalah pemrograman komputer mulai dari dasar, yaitu algoritma. Kemudian, algrotima-algoritma yang telah dibahas ini akan diimplementasikan pada suatu bahasa pemrograman. Dalam hal ini, bahasa pemrograman yang dipilih adalah Matlab dengan alasan bahwa Matlab merupakan bahasa pemrograman yang paling umum digunakan pada matematika.

Buku ini merupakan awal dari suatu pekerjaan besar, yaitu membuat sebuah buku mengenai dasar-dasar pemrograman komputer yang cukup memadai bagi keperluan mahasiswa matematika dalam menempuh studi mereka. Beragam kritik dan saran dari pembaca tentu akan menjadi masukan bagi penulis.

Malang, 2016

Penyusun

Daftar Isi

Tujuan Pembelajaran dan Kompetensi

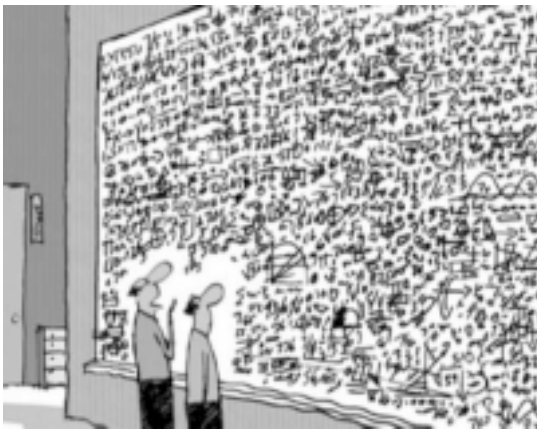
I. Tujuan Pembelajaran (*Learning Objective*)

1. Mampu menerapkan algoritma untuk menyelesaikan suatu permasalahan sederhana.
2. Mampu menyusun dan membaca algoritma atau flowchart.
3. Mampu menterjemahkan algoritma atau flowchart ke dalam kode-kode program.
4. Mampu membaca dan mengartikan kode-kode program.
5. Mampu melacak kesalahan kode-kode program.

II. Kompetensi (*Learning Outcomes*)

1. Mampu memahami pengertian sistem komputer.
2. Memahami dan menguasai penggunaan konsep dasar algoritma.
3. Memahami dan menguasai penggunaan konsep dasar pemrograman.
4. Mampu membuat rancangan aplikasi program sederhana.
5. Mampu berpikir kritis dan logis.

Bab I



Sumber: https://gurukalehuru.files.wordpress.

Algoritma dan Komputer

1.1 Pengertian Algoritma

http:\\i.ytimg.com.jpg

Gambar 1.1
Ilustrasi menentukan bilangan terbesar dari tiga bilangan berbeda.

Misalkan teman Anda menyebutkan tiga bilangan berbeda kepada Anda. Dapatkan Anda menemukan bilangan yang terbesar di antara ketiga bilangan tersebut? Anda pasti dapat langsung menemukan bilangan yang terbesar, bukan? Sekarang, apabila teman Anda tadi bertanya bagaimana *langkah-langkah* Anda menemukan bilangan terbesar tadi, dapatkah Anda menjelaskannya dengan urut? Anda mungkin akan kebingungan untuk menjelaskannya. Di sinilah peranan kemampuan Anda dalam membuat algoritma diperlukan. Apakah algoritma itu?

1.2 Kriteria Algoritma

http:\\lostislamichistory.com.

Gambar 1.2
Al-Khuwarizmi sang pencetus algoritma.

Algoritma merupakan rangkaian instruksi yang dijalankan secara terurut untuk menyelesaikan suatu permasalahan. Algoritma diperkirakan berasal dari kata Al-Khuwarizmi, yaitu seorang ilmuwan yang menulis Kitab *Al Jabar Wal-Muqabala*. Kriteria-kriteria dari suatu algoritma yang baik antara lain sebagai berikut.

1. Input, yaitu memiliki masukan.
2. Output, yaitu memiliki keluaran.
3. Definiteness, yaitu memiliki instruksi yang jelas dan tidak ambigu .
4. Finiteness, yaitu memiliki titik henti.
5. Efectiveness, yaitu efektif dalam pelaksanaan.

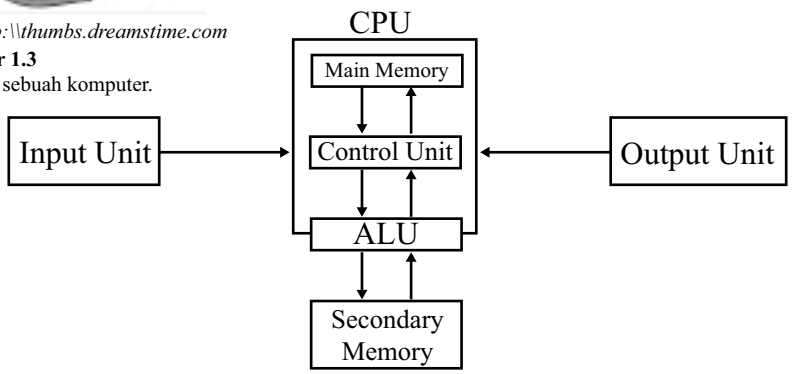
6. Generality, yaitu langkah-langkah algoritma yang diberikan berlaku untuk semua himpunan input yang diberikan, tidak hanya untuk himpunan input tertentu.

1.3 Komputer

http:\\thumbs.dreamstime.com

Gambar 1.3
Ilustrasi sebuah komputer.

Pada mata kuliah Dasar Pemrograman Komputer ini akan berkaitan erat dengan penggunaan komputer. Oleh karena itu, perlu kiranya Anda mengetahui diagram blok dari sebuah komputer terlebih dahulu. Diagram blok dari suatu sistem komputer dapat digambarkan seperti berikut.



Gambar 1.4
Diagram blok dari suatu sistem komputer.

Uraian dari setiap bagian adalah sebagai berikut.

http:\\www.russalpc.co.za

Gambar 1.4
Contoh input unit.

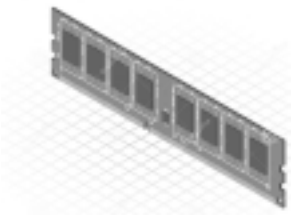
1. **Input Unit**
   Input unit (piranti masukan) berfungsi untuk memasukkan data dari pengguna ke memori komputer, contohnya keyboard, mouse, dan card reader.

http:\\www.thumbs.dreamstime.com

Gambar 1.5
Contoh output unit.

2. Output Unit

Output unit (piranti keluaran) berfungsi untuk mengeluarkan data dari memori komputer ke pengguna, contohnya monitor dan printer.



http:\\www.thumbs.dreamstime.com

Gambar 1.6
Contoh main memory.

3. Main Memory

Main memory, disebut juga RAM (Random Accsess Memory) merupakan wadah untuk menampung semua data dan masukan yang akan diolah selanjutnya oleh komputer. Main memory bersifat temporer, artinya data-data yang tersimpan di memory tersebut akan hilang jika komputer dimatikan.

4. ALU

ALU (Arithmetic and Logic Unit) berfungsi melakukan semua operasi aljabar dan logika pada komputer. ALU merupakan otak utama dari suatu komputer.

5. Control Unit

Control unit berfungsi mengatur seluruh aliran data yang ada di komputer. Control Unit dan ALU disebut juga sebagai CPU (Central Processing Unit).

http:\\www.clker.com

Gambar 1.7
Contoh secondary memory.

6. Secondary Memory

Secondary Memory berfungsi untuk menyimpan semua data yang diinginkan agar tidak hilang saat komputer dimatikan.

1.4 Program dan Pemrograman

Algoritma baru akan efektif jika dijalankan oleh sebuah pemroses (*processor*). Pemroses itu dapat berupa manusia, mesin, dan komputer. Pemroses akan membaca setiap instruksi di dalam algoritma lalu mengerjakan instruksi tersebut. Sebuah pemroses harus memenuhi syarat-syarat berikut.

1. Mengerti setiap langkah di dalam algoritma.
2. Mengerjakan operasi yang bersesuaian dengan langkah tersebut.

Pada kuliah ini, pemroses yang dimaksud adalah sebuah komputer. Agar komputer dapat memahami algoritma yang diberikan maka algoritma harus ditulis dalam bahasa yang dipahami oleh komputer. Bahasa komputer yang digunakan untuk menulis program dinamakan bahasa pemrograman.

Orang yang membuat suatu program komputer dinamakan programmer, dan kegiatan programmer dalam menulis program disebut pemrograman atau *coding*.

1.5 Belajar Memprogram dan BelajarBahasa Pemrograman

Belajar memprogram jelas berbeda dengan belajar bahasapemrograman. Belajar memprogram berarti mempelajari metodologi pemecahan masalah, kemudian menuliskannya dalam bentuk algoritma. Adapun belajar bahasa pemrograman berarti belajar menggunakan suatu bahasa pemrograman, termasuk aturan sintaks dan instruksinya. Berdasarkan tujuan aplikasinya, bahasa pemrograman dapat dibagi menjadi dua kelompok, yaitu sebagai berikut.

http:\\i.stack.imgur.com

Gambar 1.8
Ilustrasi seorang programmer.

1. Bahasa pemrograman bertujuan umum. Bahasa ini dapat digunakan untuk berbagai tujuan, misalnya Pascal, Visual Basic, dan Visual C++.
2. Bahasa pemrograman bertujuan khusus. Bahasa ini lebih spesifik digunakan untuk tujuan tertentu, misalnya Matlab dan SQL.

Adapun Bahasa pemrograman menurut tingkatannya, dapat dibagi menjadi tiga, yaitu sebagai berikut.

1. Bahasa mesin, yaitu bahasa yang dijalankan oleh komputer itu sendiri. Bahasa mesin ini tersusun atas rangkaian bilangan biner, yaitu bilangan 0 dan 1.
2. Bahasa asembli, bahasa ini memiliki tingkatan yang sedikit di atas bahasa mesin. Bahasa ini menggu nakan kata-kata sederhana, seperti MOV, ADD, atau STR. Agar dapat dimengerti oleh komputer, bahasa asembli perlu diterjemahkan men jadi bahasa mesin. Penerjemahnya dinamakan *assembler*.
3. Bahasa tingkat tinggi, bahasa ini menggunakan instruksi berupa kata-kata yang mirip dengan bahasa sehari-hari. Bahasa ini diterjemah-kan ke dalam bahasa mesin dengan menggunakan *compiler*. Contoh bahasa tingkat tinggi antara lain Pascal, Fortran, dan Matlab.

1.6 Notasi Algoritmik

Notasi algoritmik adalah rancangan urutan langkahpencapaian solusi dalam bentuk deskriptif. Notasi algoritmik dapat ditulis dalam tiga cara berikut.

1. Runtunan kalimat deskriptif;
2. Bagan alir/Flowchart;
3. Pseudo-Code.

1. Kalimat Deskriptif

Misalkan Anda akan membuat notasi algoritmik menghitung luas lingkaran berdasarkan jari-jarinya dengan menggunakan kalimat deskriptif seperti berikut.

Contoh 1.1

Program hitung_luas

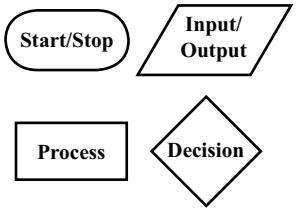
Algoritma:

1. Masukkan r

2. kuadratkan r

3. kalikan dengan 3,14

Pada praktiknya, cara yang lebih sering digunakan untuk mendeskripsikan suatu algoritma adalah flowchart dan pseudo-code.

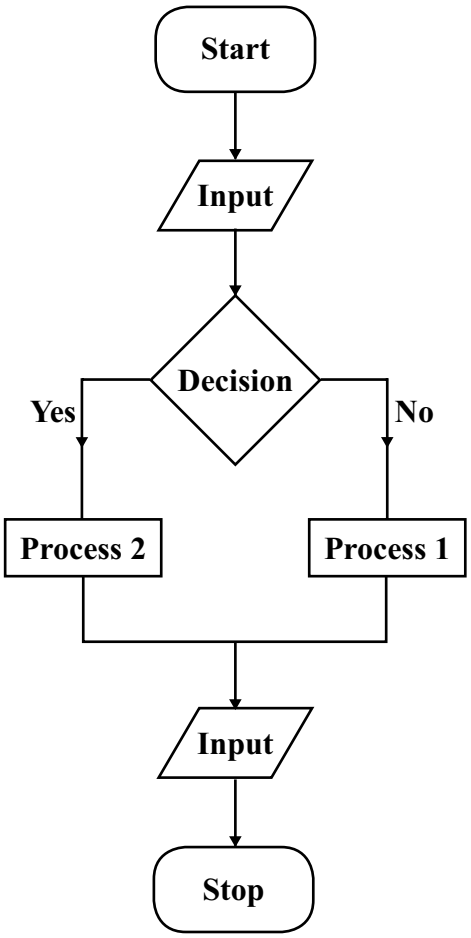


Gambar 1.9
Beberapa simbol flowchart.

2. Flowchart

Flowcharting adalah rangkaian gambar yang menunjukkan aliran proses data/algoritma yang dibuat. Beberapa simbol flowchart yang sering digunakan antara lain dapat dilihat pada gambar di samping.

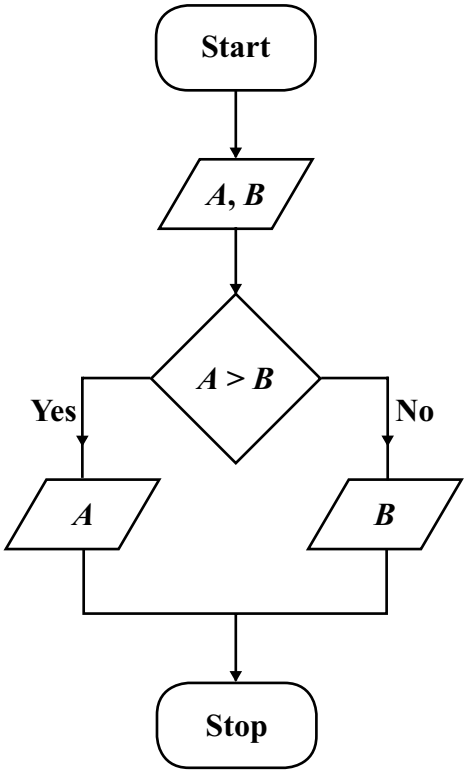
Suatu flowchart terdiri atas gabungan dari beberapa simbol yang dihubungkan dengan anak panah. Gambar berikut menunjukkan contoh flowchart.



Gambar 1.10
Contoh flowchart.

Contoh 1.2

Gambar berikut memperlihatkan flowchart untuk mencari bilangan yang terbesar di antara dua bilangan berbeda.



Penjelasan dari flowchart tersebut adalah sebagai berikut. Diberikan dua bilangan bulat berbeda *A* dan *B* sebagai masukan. Bilangan terbesar di antara keduanya dapat dicari dengan cara membandingan kedua bilangan tersebut. Jika bilangan *A* lebih besar daripada *B* maka bilangan yang terbesar adalah *A*, begitu pula sebaliknya.

1.7 Pseudocode

Pseudocode adalah notasi algoritmik yang lebih menyerupai bahasa pemrograman tingkat tinggi, misalnya Pascal. Agar suatu algoritma dalam bentuk pseudocode mudah dipahami, maka algoritma tersebut perlu ditulis dalam struktur tertentu. Struktur dari suatu algoritma adalah sebagai berikut.

1. Header

Header memuat nama dan informasi mengenai algoritma yang ditulis.

2. Deklarasi

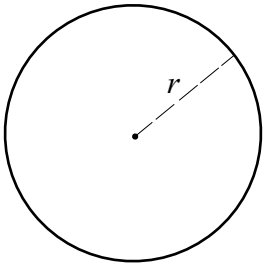
Deklarasi memuat definisi berbagai variabel dan tipe data yang digunakan dalam algoritma tersebut.

3. Deskripsi

Deskripsi memuat langkah-langkah penyelesaian masalah dengan menggunakan algoritma tersebut.

Berikut ini merupakan contoh algoritma menghitung luas lingkaran dengan menggunakan pseudocode.

Contoh 1.3



Algoritma Luas Lingkaran

{Menghitung luas lingkaran dengan masukan jari-jari}

Deklarasi

real r, L

const π

Deskripsi

read(r)

$L \leftarrow \pi \times r \times r$

write(L)

1.8 Algoritma Tracing

Algoritma tracing atau pelacakan algoritma adalah proses menjalankan suatu algoritma secara terurut langkah demi langkah. Tujuan dari algoritma tracing adalah untuk memeriksa aliran logika dan mencari kesalahan dari algoritma tersebut. Algoritma tracing dilakukan dengan mencoba berbagai kemungkinan masukan dari algoritma tersebut. Algoritma tracing tidak bertujuan untuk membuktikan kebenaran dari suatu algoritma, melainkan hanya memastikan bahwa tidak ditemukan kesalahan aliran logika pada algoritma tersebut.

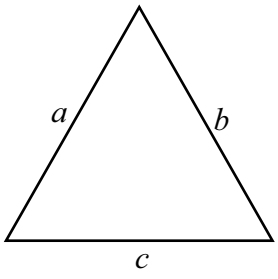
Latihan Bab I

1. Sebutkan bagian-bagian yang merupakan struktur dari suatu algoritma.

2. Apakah perbedaan antara input unit dan output unit?

3. Apakah perbedaan antara main memory dan secondary memory?

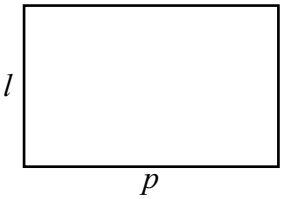
4. Buatlah flowchart untuk menghitung luas segitiga dengan masukan panjang sisi *a*, *b*, dan *c* dengan menggunakan Teorema Heron, yaitu



$L = \sqrt{s(s-a)(s-b)(s-b)}$

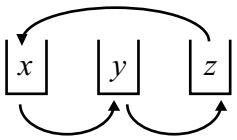
$s = \frac{1}{2}$keliling

5. Buatlah flowchart untuk menghitung keliling dari suatu persegi panjang dengan masukan panjang dan lebar persegi panjang tersebut.



6. Buatlah flowchart untuk membaca dua bilangan *x* dan *y*, kemudian pertukarkanlah isinya dengan menggunakan variabel tambahan.

7. Buatlah flowchart untuk membaca dua bilangan *x* dan *y*, kemudian pertukarkanlah isinya TANPA menggunakan variabel tambahan.



8. Buatlah flowchart untuk membaca tiga bilangan *x*, *y*, dan *z*. Kemudian, pertukarkan isinya dengan aturan isi *x* pindah ke *y*; isi *y* pindah ke *z*; dan isi *z* pindah ke *x* dengan mengguna kan satu variabel tambahan.

9. Buatlah flowchart untuk membaca tiga bilangan *x*, *y*, dan *z*. Kemudian, pertukarkan isinya dengan aturan isi *x* pindah ke *y*; isi *y* pindah ke *z*; dan isi *z* pindah ke *x* TANPA menggunakan variabel tambahan.

10. Buatlah flowchart untuk menghitung akar-akar real dari suatu persamaan kuadrat dengan masukan berupa koefisien dari persamaan kuadrat tersebut. Jika diperkirakan akan menghasilkan akar yang bukan bilangan real maka pengguna harus memberi masukan baru.

Bab II

Sumber: http://thumbs.dreamstime.com.

Tipe Data dan Variabel

Secara umum, terdapat dua macam tipe data pada suatu bahasa pemrograman, yaitu tipe dasar dan tipe bentukan.

2.1 Tipe Dasar

http://wp.pierce.ctc.edu.

Gambar 2.1
Bilangan Pi merupakan contoh data bertipe konstanta.

Tipe data yang termasuk pada tipe dasar antara lain sebagai berikut.

1. Integer, yaitu bilangan bulat misalnya −3, 0, dan 8.
2. Real, yaitu bilangan real misalnya −9, 12, 2 × 10, dan 0, 12.
3. Konstanta, yaitu bilangan yang nilainya selalu tetap di dalam suatu algoritma misalnya π =3, 14.
4. Karakter, yaitu semua huruf dan simbol yang tidak dioperaskan secara matematis misalnya a, C, dan ?.
5. Boolean, yaitu tipe data yang hanya bernilai benar atau salah.

2.2 Tipe Bentukan

http://macotar.blogspot.com.

Gambar 2.2
Ilustrasi sebuah larik dua dimensi (matriks).

Tipe bentukan terdiri atas berbagai tipe berikut.

1. Larik, yaitu suatu tipe untuk menampung data sejenis. Dalam matematika, larik ini dapat dianalogikan sebagai sebuah vektor.
2. String, yaitu deretan karakter yang membentuk sebuah kata atau kalimat, misalnya nama dan alamat.

3. Record, yaitu tipe data untuk menampung elemen data yang tipenya tidak sama dengan tujuan untuk mewakili suatu objek, misalnya record data mahasiswa yang terdiri atas NIM, nama, dan umur.

2.3 Variabel

Variabel adalah identitas yang mewakili suatu elemen data, misalnya *x*, *y*, nama. Aturan dalam pemberian nama variabel antara lain sebagai berikut.

1. Harus dimulai dengan abjad, tidak diperkenankan dimulai dengan angka atau simbol.
2. Tidak boleh ada spasi.
3. Tidak menggunakan titik dua, titik, koma, dan sejenisnya.
4. Nama variabel berkaitan dengan elemen data.
5. Nama variabel pendek saja.

2.4 Pemberian dan PembacaanNilai

Terdapat dua cara yang dapat digunakan untuk memberi nilai pada suatu variabel, yaitu assignment dan pembacaan. Berikut ini merupakan beberapa contohnya.

1. Assignment

Pemberian nilai dengan assignment memiliki bentuk *variabel* ← *nilai*. Berikut beberapa contohnya.

1. $x \leftarrow 2$
2. jarak ← 8
3. nama ← trija
4. $x \leftarrow 5$
   $y \leftarrow 3$
   $z \leftarrow x + y$

2. Pembacaan

Pemberian nilai dengan pembacaan memiliki bentuk *read(variabel)*, contohnya read(x) dan read(nama).

Untuk menampilkan nilai dari suatu variabel digunakan perintah write(variabel). Berikut beberapa contohnya.

1. write(”Nama Anda adalah ”,nama);
2. write(x);
3. write(”Jumlahnya adalah ”, $x + y$).

2.5 Ekspresi

Ekspresi adalah transformasi data atau variabel dalam bentuk persamaan. Berikut ini beberapa bentuk ekspresi.

1. Ekspresi Aritmetika

Ekspresi aritmetika yaitu ekspresi yang memuat operator aritmetika. Berikut beberapa contoh ekspresi aritmetika.

a. $x \leftarrow (y + 5) * t$
b. total ← gajipokok + lembur
c. luas ← 3, 14**r***r*

2. Ekspresi Relasional

Ekspresi relasional, yaitu ekspresi yang memuat operator relasional. Berikut beberapa contoh ekspresi relasional.

a. $x > y$
b. $a < b$

3. Ekspresi Logika

Ekspresi ini memuat operator logika, yaitu AND dan OR. Berikut beberapa contoh ekspresi logika.

a. $x \leftarrow A \,\&\&\, B$
b. $n \leftarrow a \,||\, b$

4. Ekspresi String

Ekspresi string yaitu ekspresi yang memuat operator string. Berikut beberapa contoh ekspresi string.

a. nama ← ”Trija”
b. email ← “trija@gmail.com”

Latihan Bab II

1. Tulislah algoritma dari latihan-latihan yang terdapat pada Bab I.

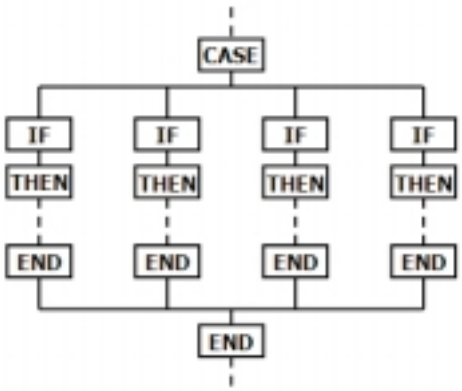
2. Buatlah algoritma harga total suatu barang, yaitu harga barang ditambah dengan pajaknya. Pajak barang tersebut adalah 25% dari harga barangnya.

3. Buatlah algoritma yang menampil kan NIM, nama, alamat, dan nomor handphone seorang mahasiswa.

4. Buatlah algoritma untuk menampil kan persentase keuntungan penjua lan suatu barang dengan masukan harga beli dan harga jual barang tersebut.

5. Buatlah algoritma untuk menampil kan harga suatu barang setelah didis kon dengan masukan harga awal barang dan besaran diskon yang di berikan.

Bab III



Sumber: http://upload.wikimedia.org.png.

Instruksi Pemilihan

3.1 Pengertian Instruksi Pemilihan

Instruksi pemilihan adalah suatu instruksi yang digunakan untuk memilih salah satu aksi bergantung pada terpenuhi atau tidaknya suatu syarat. Syarat yang diperiksa pada umumnya berupa ekspresi boolean, yaitu suatu ekspresi yang hanya bernilai benar atau salah,namun tidak keduanya, Tabel 3.1 memperlihatkan beragam notasi yang digunakan pada instruksi pemilihan.

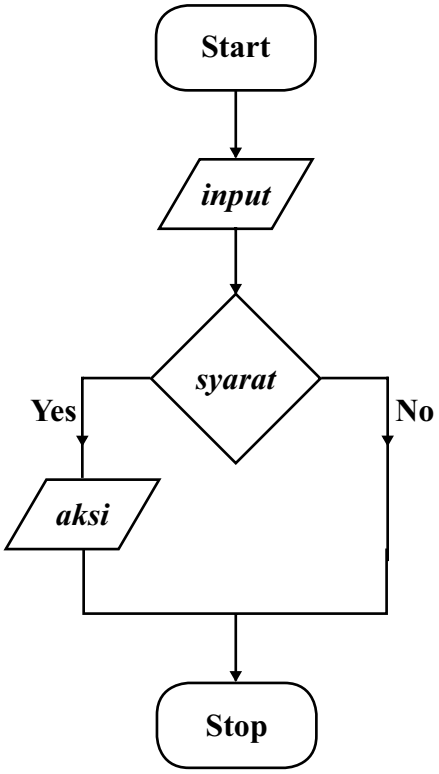
Tabel 3.1 Beragam Notasi yang Digunakan Pada Instruksi Pemilihan

Notasi	Makna
<	kurang dari
<=	kurang dari atau sama dengan
>	lebih dari
>=	lebih dari atau sama dengan
=	sama dengan
<>	tidak sama dengan

Instruksi pemilihan yang akan Anda pelajari pada bagian ini antara lain instruksi **if ... then ... else endif** dan instruksi **case**.

3.2 Bentuk IF... THEN... ENDIF

Bentuk ini digunakan apabila terdapat satu syarat. Aksi tertentu akan dilakukan apabila syarat tersebut terpenuhi. Apabila syarat tidak terpenuhi, maka tidak ada aksi yang dilakukan. Contoh flowchart dari bentuk tersebut dapat dilihat pada Gambar 3.1 berikut.



Deskripsi dari bentuk tersebut adalah sebagai berikut.

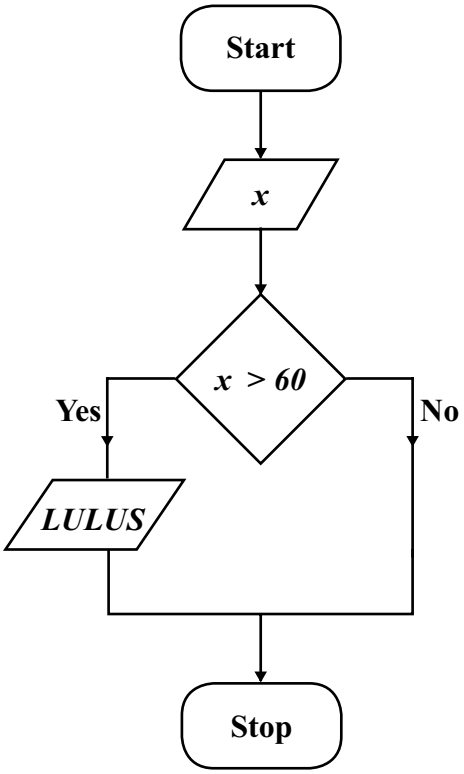
IF (syarat)
    THEN (aksi)
ENDIF

Contoh 3.1

Algoritma berikut akan menampilkan kata LULUS jika nilai mahasiswa lebih dari 60.

Algoritma Lulus
real : x
read(x)
if x > 60
then write('LULUS')
endif

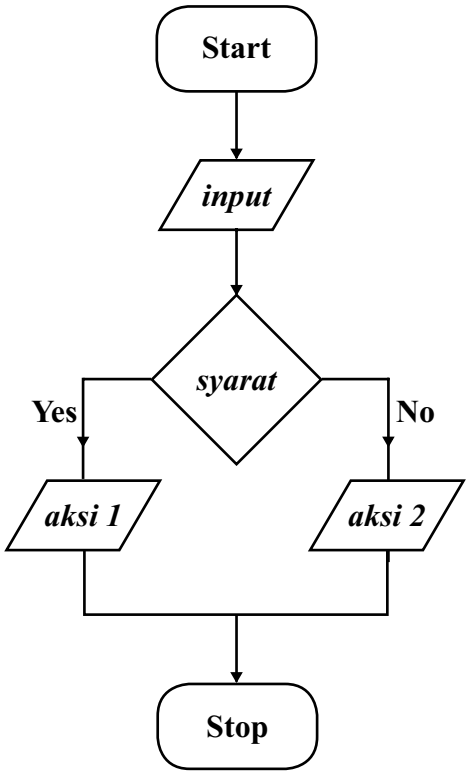
Flowchart dari permasalahan tersebut adalah sebagai berikut.



3.3 Bentuk IF... THEN... ELSE ... ENDIF

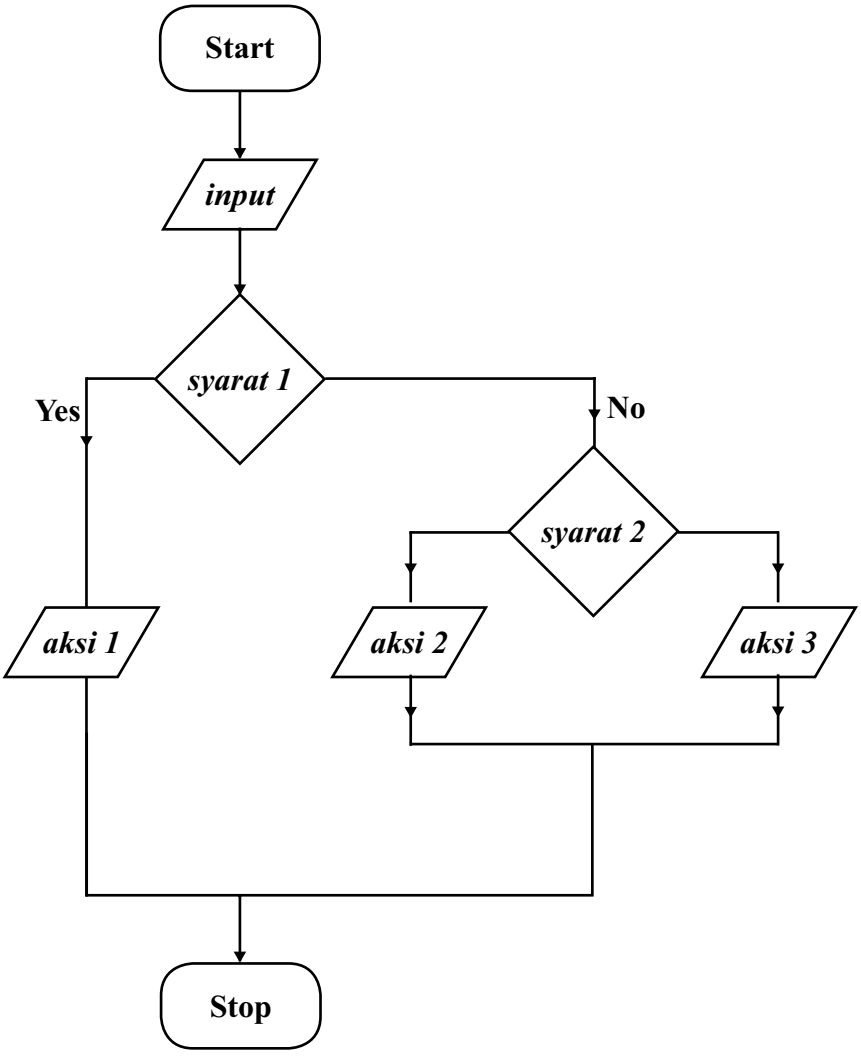
Bentuk pemilihan IF ... THEN ... ELSE ... ENDIF ini digunakan apabila terdapat satu syarat. Aksi tertentu akan dilakukan apabila syarat tersebut terpenuhi. Apabila syarat tidak terpenuhi, maka ada aksi lain yang dilakukan. Deskripsi dari bentuk tersebut adalah sebagai berikut.

IF (Syarat)
    THEN (Aksi 1)
    ELSE (Aksi 2)
ENDIF

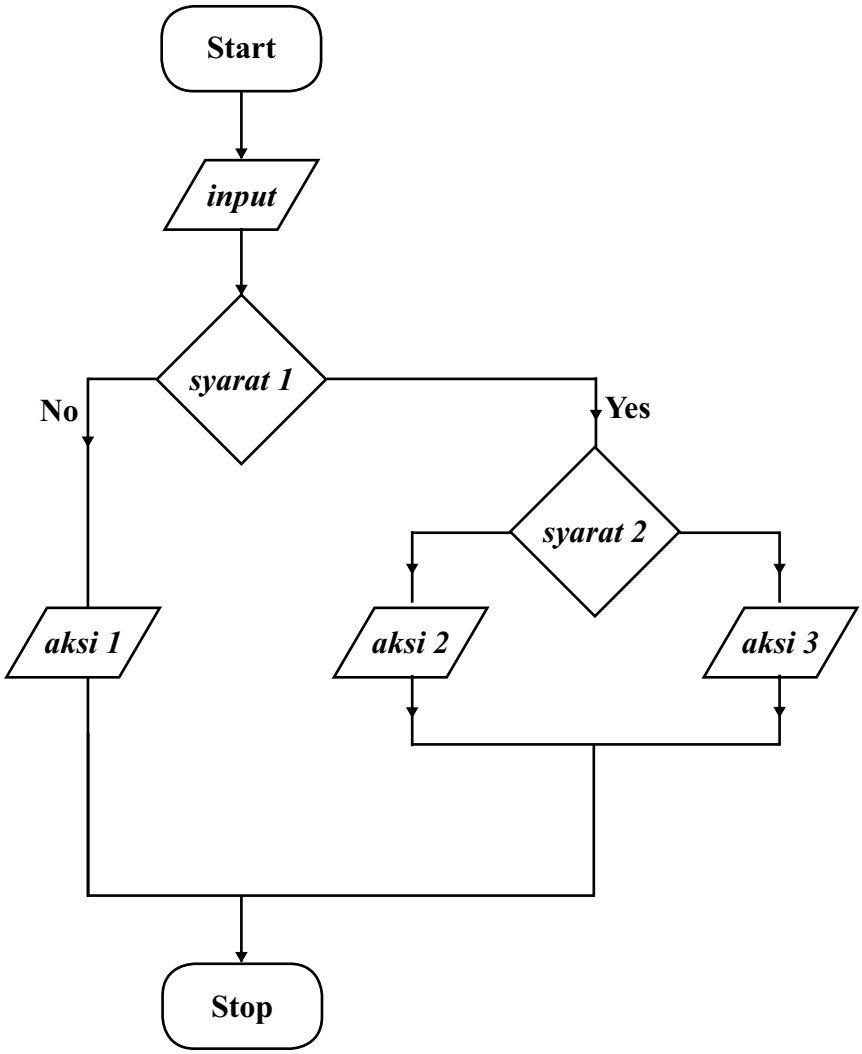


3.4 Bentuk Bersusun

Selain kedua bentuk tadi, Anda dapat pula memodifikasi instruksi pemilihan ke dalam bentuk-bentuk lain sesuai dengan permasalahan yang dihadapi. Dapatkah Anda mendeskripsikan pseudocode dari flowchart berikut?



Setelah itu, tuliskan pula pseudocode dari flowchart berikut.



3.5 Instruksi Case

Instruksi case digunakan sebagai instruksi pemilihan jika aksi yang akan dilakukan bergantung pada nilai variabel yang bersesuaian. Bentuk instruksi case adalah sebagai berikut.

case (VARIABEL)
nilai 1: aksi 1
nilai 2: aksi 2
.
.
.
default: aksi n
endcase

Sifat dari instruksi case antara lain sebagai berikut.

1. Terdapat n aksi.
2. Setiap aksi hanya dilakukan jika nilai variabel yang dimasukkan memenuhi syarat.
3. Apabila tidak ada satupun nilai variabel yang cocok maka aksi yang dijalankan adalah default.

Berikut adalah contoh penggunaan case pada algoritma penghitungan upah.

Contoh 3.1

Algoritma upah
char: golongan;
integer: upah;
read(golongan);
case (golongan)
'A': upah ← Rp1000
'B': upah ← Rp2000
'C': upah ← Rp3000
'D': upah ← Rp4000
'E': upah ← Rp5000
default: upah ← 0;
endcase
write(upah);

Latihan Bab III

1. Tulislah algoritma yang membaca sebuah bilangan bulat, lalu menulis kan pesan GENAP jika bilangan tersebut genap atau GANJIL jika bilangan tersebut ganjil.

2. Buatlah algoritma untuk membaca dua bilangan bulat, lalu menentu-kan bilangan yang terbesar.

3. Tuliskan algoritma untuk membaca tiga bilangan bulat lalu menentukan bilangan yang terbesar.

4. Karyawan PT Kanjuruhan digaji berdasarkan jumlah jam kerjanya selama seminggu. Upah per jam Rp2000,00. Bila jumlah jam kerja lebih besar daripada 48 jam maka sisanya dianggap lembur. Upah lembur Rp3000, 00 per jam. Tuliskan algoritma untuk menentu kan upah mingguan karyawan dengan masukan jumlah jam kerja mereka.

5. Suatu tahun disebut tahun kabisat jika memenuhi salah satu syarat berikut: habis dibagi 4 tetapi tidak habis dibagi 100; atau habis dibagi 400.

6. Buatlah algoritma untuk menentu kan apakah suatu bilangan bulat itu positif, negatif, atau nol.

7. Buatlah algoritma untuk menentu kan wujud air (padat, cair, gas) pada suhu tertentu dengan masukan suhu air itu.
8. Buatlah algoritma untuk menentu kan kuadran dari suatu titik dengan masukan koordinat titik tersebut.

Bab IV

Sumber: http://thatsmaths.com

Instruksi Pengulangan

4.1 Pengertian Instruksi Pengulangan

Instruksi pengulangan adalah suatu instruksi untuk mengulangi pelaksanaan sederetan instruksi lain berulang kali sesuai dengan syarat yang ditetapkan. Instruksi perulangan ada tiga jenis, yaitu for, while..do,dan repeat...until.

4.2 Perulangan FOR

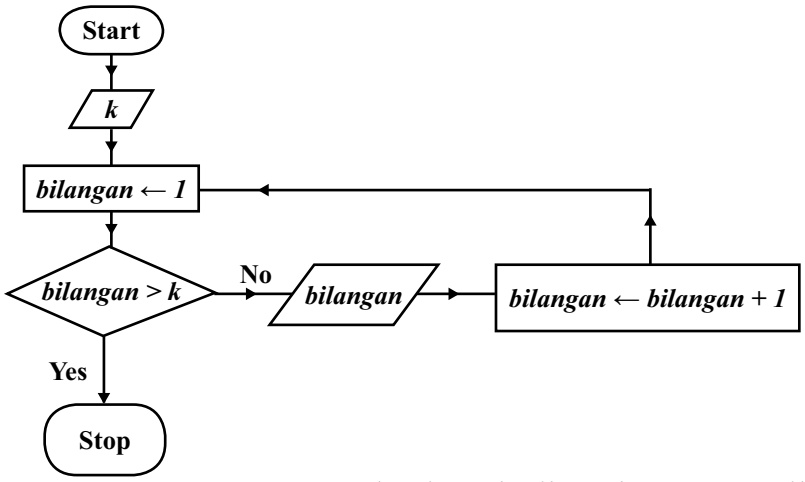
Apabila dituliskan deskripsinya, maka skema perulagan for akan berbentuk seperti berikut.

```
for (var = awal to akhir step n)
    aksi
endfor
```

Algoritma berikut akan menampilkan bilangan asli 1 sampai dengan *k* dengan kenaikan sebesar 1.

```
Algoritma tampil_bilangan
Deklarasi
    integer: k,bilangan;
Deskripsi
    read(k);
    for (bilangan = 1 to k step 1)
        write(bilangan)
    endfor
```

Flowchart dari contoh tersebut dapat dilihat pada gambar berikut.



Algoritma berikut akan menampilkan bilangan asli *k* sampai dengan 1 dengan penurunan sebesar 1.

Algoritma tampil_bilangan

Deklarasi

integer: k,bilangan;

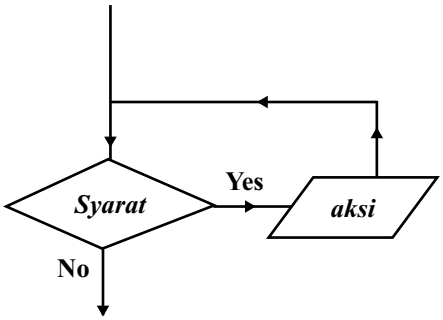
Deskripsi

read(k);
for (bilangan = k to 1 step -1)
write(bilangan)
endfor

Sekarang, coba Anda gambarkan flowchart dari deskripsi tersebut.

4.3 Perulangan WHILE ... DO

Flowchart dari perulangan while..do adalah sebagai berikut.



Apabila dituliskan dalam bentuk deskripsi maka akan menjadi seperti berikut.

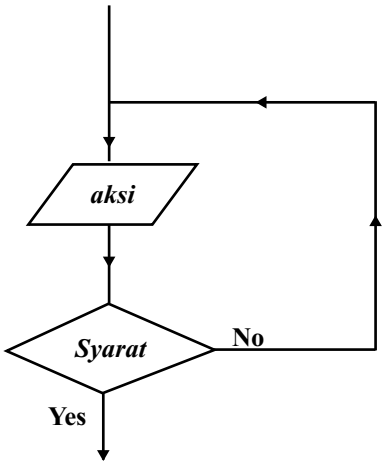
```
while (SYARAT) do
    AKSI
endwhile
```

Berikut adalah contoh algoritma dengan menggunakan while...do.

```
Algoritma tampil_bilangan
Deklarasi
integer: k,bilangan;
Deskripsi
    read(k);
    bilangan <- 1;
    while (bilangan <= k) do
        write(bilangan)
        bilangan <- bilangan + 1;
    endwhile
```

4.4 Perulangan REPEAT ... UNTIL

Perulangan repeat...until dapat digambarkan dalam bentuk flowchart berikut.



Apabila dituliskan dalam bentuk deskripsi maka akan menjadi seperti berikut.

repeat
    AKSI
until (SYARAT)

Berikut ini adalah contoh menggunakan repeat ... until.

Algoritma tampil_bilangan
Deklarasi
integer: k,bilangan;

Deskripsi

read(k);
bilangan < –1;
repeat
write(bilangan);
bilangan <- bilangan + 1;
until (bilangan > k)

Latihan Bab IV

1. Buatlah algoritma untuk menampilkan tulisan HELLO WORLD sebanyak 5 kali.

2. Buatlah algoritma untuk membaca integer tak negatif *n* dan menghitung faktorialnya.

3. Buatlah algoritma untuk menghitung 1 + 2 + 3 +···+ 100.

4. Buatlah algoritma untuk menghitung 1 + 3 + 5 +···+ 99.

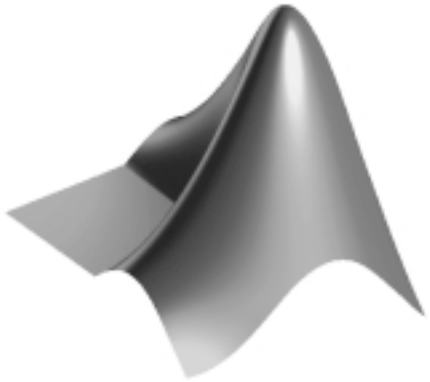
5. Buatlah algoritma untuk menghitung $1 \times 2 \times 4 \times 7 \times 11 \times \cdots \times n$, dengan $n < 100$.

6. Diketahui deret Fibonacci $S_1 = 1$, $S_2 = 1$, $Sn = S_{n-1} + S_{n-2}$, untuk $n \geq 3$. Buat algoritma untuk membaca nilai n, kemudian menghitung nilai S_n.

7. Buatlah algoritma dari syair lagu ANAK AYAM.

8. Buatlah algoritma dari sebuah perpangkatan.
   $a^n = a \times a \times a \times a \ldots \times a$

Bab V



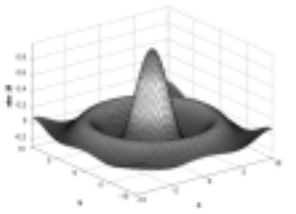
Sumber: http://www.mathworks.com.

Mengenal Matlab

5.1 Pendahuluan

Matlab merupakan sebuah singkatan dari Matrix Laboratory. Matlab dikenalkan untuk kali pertama oleh University of New Mexico dan University of Stanford. Matlab awalnya hanya digunakan untuk keperluan analisis numerik, aljabar linear, dan matriks. Akan tetapi, saat ini kemampuan dan fitur yang dimiliki oleh Matlab sudah jauh lebih lengkap dengan ditambahkannya beragam toolbox. Beberapa manfaat yang didapatkan dari Matlab antara lain sebagai berikut.

1. Perhitungan matematika.
2. Komputasi numerik.
3. Simulasi dan pemodelan.
4. Visualisasi dan analisis data.
5. Pembuatan grafik untuk keperluan sains dan teknik.
6. Pengembangan aplikasi berbasis General User Interface.



upload.wikimedia.org

Gambar 5.1
Contoh grafik hasil olahan Matlab.

Beberapa hal penting yang harus Anda perhatikan dalam penulisan instruksi pada Matlab, yaitu sebagai berikut.

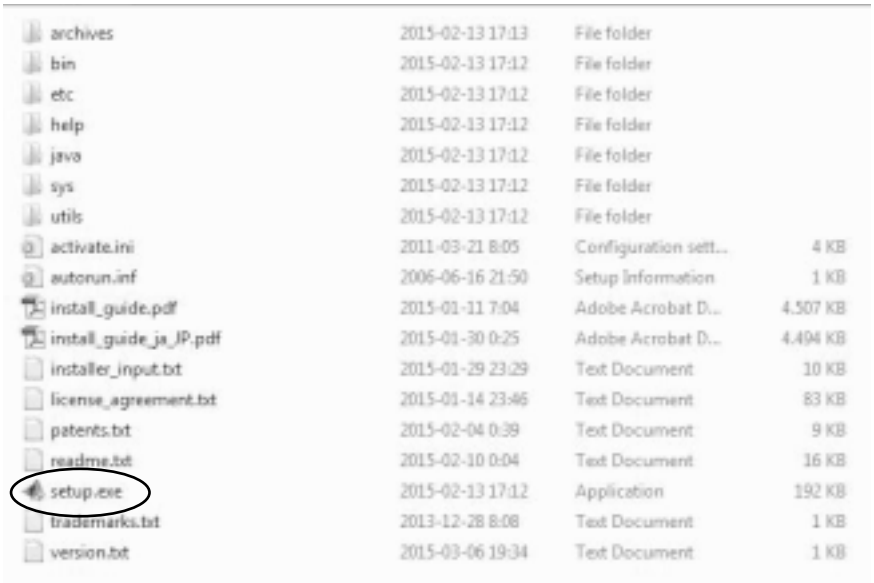
1. Variabel bersifat case sensitive, artinya Matlab akan membedakan adanya huruf besar dan kecil dalam penamaan variabel tersebut.
2. Panjang nama variabel maksimum 31 karakter.

3. Penamaan variabel harus selalu diawali dengan huruf, tidak boleh dengan bilangan, ataupun simbol.

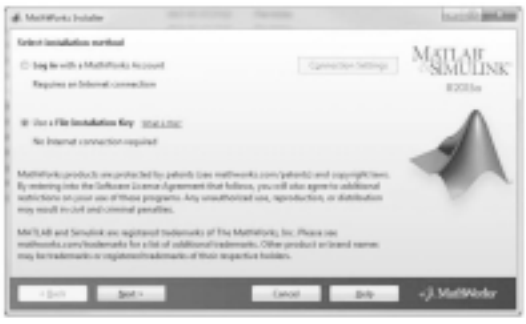
5.2 Cara Instalasi Matlab

Berikut ini akan dijelaskan cara melakukan instalasi Matlab pada sistem operasi windows.

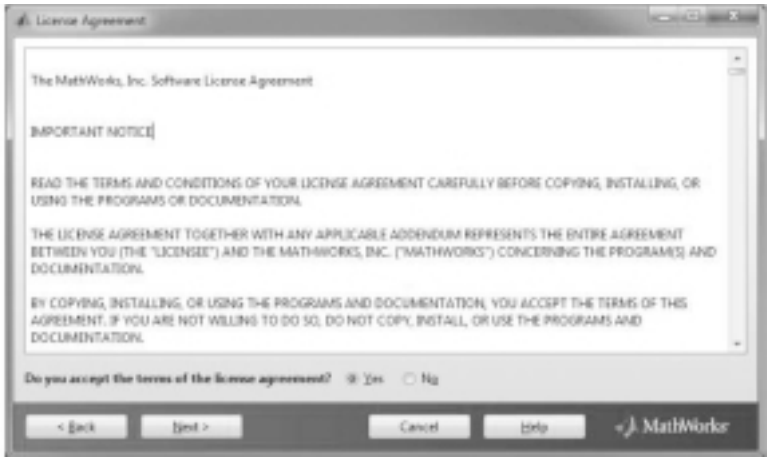
1. Mulai instalasi dengan mengklik file **setup.exe**.



2. Selanjutnya akan tampak tampilan berikut.



3. Anda dapat memilih install dengan menggunakan akun Mathworks atau dengan file installation key. Pada contoh ini digunakan file installation key.

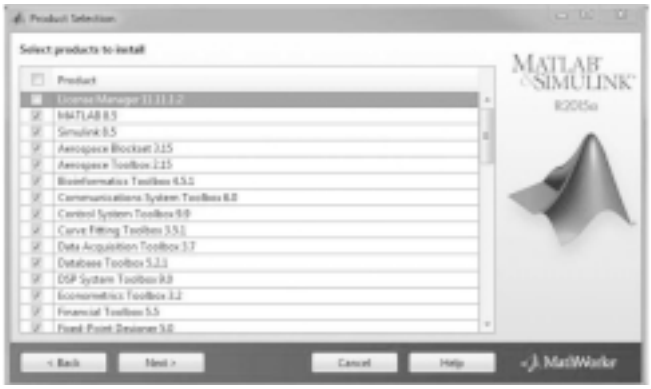


4. Akan muncul pertanyaan *Do you accept the terms of the license agreement*? Pilih **Yes**.

5. Masukkan file installation key.

6. Setujui folder instalasi.

7. Pilih produk yang akan diinstalasi.



8. Masukkan path file license.

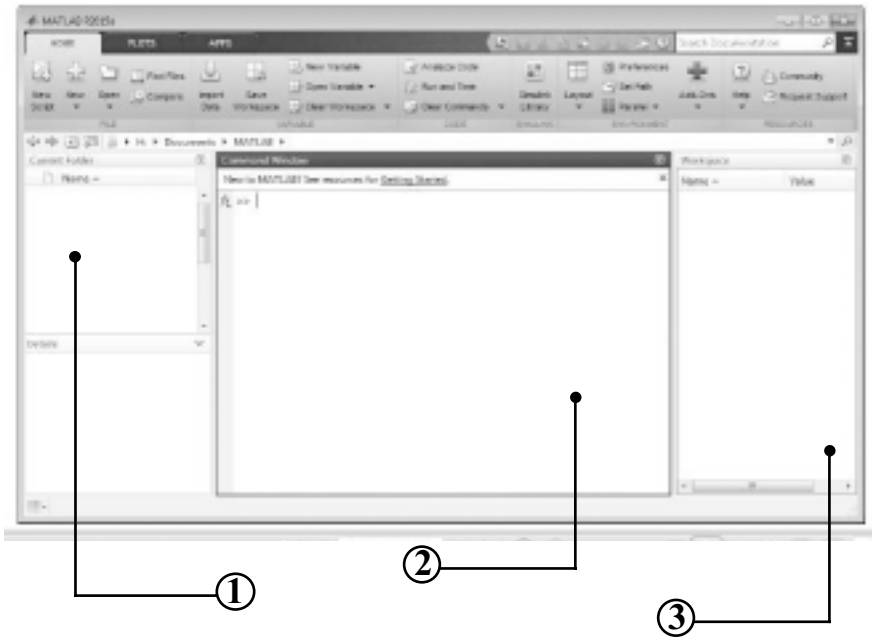
9. Klik **Install**.

10. Proses instalasi dimulai. Tunggulah beberapa saat hingga proses selesai.

11. Instalasi selesai.

5.3 Desktop Dasar Matlab

Saat Anda menjalankan Matlab, akan tampak tampilan dasar berikut.



Tampilan desktop terdiri atas beberapa panel berikut.

1. **Current Folder**
   Panel ini berfungsi untuk mengakses file Anda.
2. **Command Window**
   Panel ini sebagai tempat untuk menuliskan perintah, ditandai dengan simbol >>.
3. **Workspace**
   Panel ini untuk mengeksplorasi data yang Anda buat atau impor dari file.

Sekarang, Anda akan mencoba untuk membuat sebuah variabel di Matlab.

```
>> a = 1
```

Matlab akan membuat sebuah variabel a di workspace dan menampilkan hasilnya di Command Window.

```
a =
     1
```

Sekarang, coba Anda buat beberapa variabel baru berikut.

```
>> b = 2
b =
     2

>> c = a+b
c =
     3

>> d = cos(a)
d =
     0.5403
```

Apabila Anda tidak membuat variabel outputnya, maka Matlab akan menggunakan variabel bernama ans, singkatan dari answer, untuk menyimpan hasil perhitungan. Perhatikan contoh berikut.

```
>> sin(a)
ans =
    0.8415
```

Jika Anda mengakhiri pernyataan dengan menggunakan titik koma, Matlab akan menghitung hasilnya, tetapi tidak akan menampilkan hasilnya di layar.

```
>> e = a*b;
```

Anda dapat memanggil perintah yang telah lalu dengan cara menekan panah ke atas ↑ atau panah ke bawah ↓.

5.4 Variabel Pada Matlab

Berbeda dengan Pascal, Matlab tidak memiliki struktur yang harus diikuti seperti pada struktur penulisan algoritma. Matlab tidak perlu menuliskan header dan deklarasi variabel di awal program. Untuk lebih jelasnya, perhatikan contoh berikut.

```
>> x = ‘a‘
x =
a

>> y = 100;

>> z = ‘makanan‘
z =
makanan
```

Dari tiga contoh tadi, Anda lihat bahwa ada variabel yang dimunculkan kembali di layar, yaitu x dan z, ada pula yang tidak dimunculkan di layar, yaitu y. Hal ini terjadi karena pada variabel y diakhiri dengan tanda titik koma, sedangkan pada x dan z tidak diakhiri dengan tanda titik koma. Berikut ini disajikan tabel beberapa perbedaan dasar antara struktur algoritma dan struktur pemrograman Matlab.

Tabel 5.1 Perbedaan Struktur Penulisan

Algoritma	Matlab
ada header	tidak perlu header
ada deklarasi	tidak perlu deklarasi
read(x)	x = input(‘...‘)
write(x)	disp(x)

5.5 M-file Pada Matlab

Program-program yang ada kebanyakan tidak hanya terdiri atas beberapa baris saja. Banyak sekali program yang terdiri atas puluhan, bahkan ratusan baris. Untuk itu, pengetikkan program perlu dilakukan di sebuah tempat khusus di Matlab yang bernama M-files. Untuk membuka M-files, klik ikon yang terdapat di sudut kiri jendela Matlab.

Coba Anda ketikkan program berikut di M-files, kemudian jalankan program tersebut.

```
clc;
clear all;
r = input('Masukkan jari-jari lingkaran: ');
L = pi*r^2;
disp('Luas lingkaran adalah ');
disp(L)
```

Anda akan melihat tampilan berikut di layar Command Window.

```
Masukkan jari-jari lingkaran:
```

Misalkan jari-jari lingkaran yang Anda masukkan adalah 6 satuan, ketikkan angka 6, kemudian tekan enter sehingga tampak tampilan berikut.

```
Masukkan jari-jari lingkaran: 6
Luas lingkaran adalah
    113.0973
```

Pada program di atas, terdapat perintah clc dan clear all. Perintah clc adalah perintah untuk membersihkan layar Command Window, sedangkan clear all adalah perintah untuk membersihkan seluruh variabel yang ada sebelum program dijalankan.

Latihan Bab V

Buatlah program Matlab dari soal-soal berikut.

1. Buatlah program untuk menghitung nilai rata-rata dari tiga nilai ujian seorang mahasiswa.

2. Buatlah program untuk menghi tung luas segitiga dengan masukan panjang sisi a, b, dan c menggunakkan Teorema Heron.

3. Buatlah program untuk menghitung keliling suatu persegi panjang dengan masukan panjang dan lebar persegi panjang tersebut.

4. Buatlah program untuk membaca dua bilangan *x* dan *y*, kemudian pertukarkanlah isinya TANPA menggunakan variabel tambahan.

5. Buatlah program untuk membaca tiga bilangan *x*, *y*, dan *z*. Kemudian, pertukarkan isinya dengan aturan isi *x* pindah ke *y*; isi *y* pindah ke *z*; dan isi *z* pindah ke *x* dengan menggunakan satu variabel tambahan.

Bab VI

Sumber: http://thumbs.dreamtime.com.

Perintah disp dan fprintf

Di bab sebelumnya, Anda telah mengenal cara untuk menuliskan variabel beserta isinya pada Matlab. Sekarang, Anda akan mengenal beragam cara untuk menampilkan output pada Matlab, yaitu menampilkan variabel beserta isinya atau hanya sekadar isi dari variabel tersebut. Secara umum, terdapat tiga cara untuk menampilkan output pada Matlab, yaitu sebagai berikut.

1. Menuliskan nama variabel tanpa diakhiri dengan tanda titik koma.
2. Menggunakan perintah disp.
3. Menggunakan perintah fprintf.

Berikut ini adalah penjelasan setiap cara yang dikemukakan di atas.

6.1 Menuliskan Variabel Tanpa Diakhiri Dengan Tanda Titik Koma

Misalkan Anda mendeklarasikan sebuah variabel dengan isi berikut.

```
>> angka = 1;
```

Panggilah kembali variabel angka.

```
>> angka
angka =
     1
```

Terlihat bahwa nama variabel, yaitu *angka* akan muncul kembali diikuti dengan isi variabel itu, yaitu 1.

6.2 Menggunakaan Perintah disp

Perintah disp(x) akan mengeluarkan nilai dari suatu variabel *x* tanpa mencetak nama variabel tersebut. Sebagai contoh, coba Anda ketikkan perintah-perintah berikut pada Command Window.

```
>> x = 2;
>> y = 'Hello World';
>> disp(x);
     2
>> disp(y);
Hello World
```

Pada perintah di atas, penulisan isi dari variabel x, yaitu 2 tidak perlu menggunakan tanda petik awal dan akhir, berbeda dengan penulisan isi dari variabel y, yaitu 'Hello World'. Mengapa demikian, karena isi dari variabel x bukan string, sedangkan isi dari variabel y merupakan string.

Pada kedua contoh tadi, Anda hanya memanggil satu variabel saja. Apabila nilai variabel yang akan ditampilkan lebih dari satu, maka perlu dilakukan modifikasi pada perintah tadi. Sebagai contoh, ketikkan program berikut.

```
>> nama = 'Tedi';
>> umur = 18;
>> tampil1 = [nama,' saat ini berumur ',num2str(umur),' tahun.'];
tampil1 =
Tedi saat ini berumur 18 tahun.
>> disp(tampil1);
Tedi saat ini berumur 18 tahun.
>> tampil2 = sprintf('%s saat ini berumur %d tahun.',nama,umur);
tampil2 =
Tedi saat ini berumur 18 tahun.
>> disp(tampil2);
Tedi saat ini berumur 18 tahun.

>> fprintf('%s saat ini berumur %d tahun.',nama,umur)
Tedi saat ini berumur 18 tahun.
```

Program di atas akan menampilkan output yang sama, yaitu kalimat *Tedi saat ini berumur 18 tahun*. Pada pembahasan ini, Anda akan lebih difokuskan pada bentuk fprintf. Pada program di atas, Anda menemukan perintah %s dan %d, bukan? Untuk lebih memahami penggunaan perintah-perintah tersebut perhatikanlah Tabel 6.1 berikut.

Perintah	Fungsi
%d	cetak sebagai bilangan bulat
%f	cetak sebagai bilangan desimal
%s	cetak sebagai string
%c	cetak sebagai karakter
\n	membuat baris baru
\%	membuat tanda %

Sekarang, cobalah Anda ketikkan perintah-perintah berikut pada command window dan amati perbedaannya.

```
>> x = 2;
>> fprintf('x adalah %d',x);
x adalah 2
>> fprintf('x adalah %f',x);
x adalah 2.000000
>> fprintf('x adalah %.2f',x);
x adalah 2.00
>> fprintf('x adalah %s',x);
>> fprintf('x adalah %c',x);
>> fprintf('x adalah %d',x);fprintf(' selesai');
x adalah 2 selesai
>> fprintf('x adalah %d\n',x);fprintf('selesai'
);
x adalah 2
selesai
```

Dari contoh di atas, terlihat bahwa perintah %.2f akan memunculkan bilangan dengan dua angka di belakang koma. Berdasarkan hal itu, jika Anda akan memunculkan sebuah bilangan dengan 5 angka di belakang koma maka perintahnya adalah %.5f.

Contoh 6.1

Buatlah program berikut, kemudian beri nama rataan.m

```
%program menghitung rata-rata
clc;
clear all;
disp('========================')
disp('PROGRAM MENGHITUNG RATA-RATA')
disp('========================')
n1 = input('Masukkan nilai 1: ');
n2 = input('Masukkan nilai 2: ');
n3 = input('Masukkan nilai 3: ');
rata = (n1+n2+n3)/3;
fprintf('Nilai rata-rata mahasiswa adalah %.3f',rata)
```

Cobalah Anda jalankan program tersebut, kemudian beri nilai-nilai masukan n1 = 2, n2 = 5, dan n3 = 7. Anda akan melihat tampilan berikut.

```
==========================
PROGRAM MENGHITUNG RATA-RATA
==========================
Masukkan nilai 1: 2
Masukkan nilai 2: 5
Masukkan nilai 3: 7
Nilai rata-rata mahasiswa adalah 4.667
>>
```

Untuk lebih memantapkan pemahaman Anda, coba kerjakan program berikut, kemudian beri nama luas_segitiga.

Contoh 6.2

```
%program menghitung luas segitiga
clc;
clear all;
disp('================================')
disp('PROGRAM MENGHITUNG LUAS SEGITIGA')
disp('================================')
a = input('Masukkan panjang sisi a: ');
b = input('Masukkan panjang sisi b: ');
c = input('Masukkan panjang sisi c: ');
disp('================================')
k = a+b+c;
s = 0.5*k;
L = sqrt(s*(s-a)*(s-b)*(s-c));
fprintf('Luas %.2f dan keliling %.2f',L,k);
```

Contoh 6.3

Program berikut digunakan untuk menemukan solusi dari sistem persamaan linear dua variabel $a_1x + b_1y = c_1$ dan $a_2x + b_2y = c_2$ dengan menggunakan aturan Cramer.

```
clc;
clear all;
disp('================================================')
disp('      SOLUSI SPLDV DENGAN METODE CRAMER')
disp('================================================')
a1 = input('Masukkan a1: ');
b1 = input('Masukkan b1: ');
c1 = input('Masukkan c1: ');
disp('------------------------------------------------')
a2 = input('Masukkan a2: ');
b2 = input('Masukkan b2: ');
c2 = input('Masukkan c2: ');
disp('------------------------------------------------')
x = (c1*b2 - b1*c2)/(a1*b2 - b1*a2);
y = (a1*c2 - c1*a2)/(a1*b2 - b1*a2);
fprintf(' ')
fprintf('Solusinya adalah x = %.2f dan y = %.2f\n',x,y)
disp('================================================')
```

Apabila program tersebut dijalankan maka akan tampak tampilan seperti berikut.

```
==================================
SOLUSI SPLDV DENGAN METODE CRAMER
==================================
Masukkan a1: 2
Masukkan b1: 3
Masukkan c1: 8
-------------------------------------------
Masukkan a2: 4
Masukkan b2: -1
Masukkan c2: 2
-------------------------------------------
 Solusinya adalah x = 1.00 dan y = 2.00
==================================
>>
```

Latihan Bab VI

1. Buatlah program untuk menghitung volume sebuah bola dengan masukan berupa jari-jari bola tersebut.

2. Buatlah program untuk menghitung akar-akar dari persamaan kuadrat $ax^2 + bx + c = 0$ dengan masukan a, b, dan c.

3. Sebuah mobil melaju dengan kecepatan v km/jam. Buatlah program untuk menghitung jarak s yang ditempuh mobil tersebut selama t jam dengan masukan v dan t. Jarak dihitung dengan menggunakan rumus $s = v \times t$.

4. Buatlah program untuk menghitung jarak s yang ditempuh oleh sebuah peluru yang ditembakkan dengan kecepatan awal v_0 dan sudut sebesar α (dalam radian) dengan rumus jarak = $(2\ v_0^2 \sin\alpha \cos\alpha)/g$, g adalah gaya gravitasi sebesar 9,8 m/dt^2.

5. Jelaskan perbedaan antara perintah disp dan fprintf.

Bab VII

Sumber: http://timoelliott.com

Pengambilan Keputusan

Anda sudah mempelajari proses pengambilan keputusan dan pengulangan di bab sebelumnya. Sekarang, Anda akan mempelajari perintah-perintah Matlab yang berkaitan dengan pengambilan keputusan dan pengulangan. Akan tetapi, sebelumnya Anda perlu mengetahui tentang operator relasional dan operator logika terlebih dahulu.

7.1 Operator Relasional

Operator relasional adalah semua operator yang berfungsi untuk melakukan pembandingan. Beberapa operator relasional dapat dilihat pada tabel berikut.

Tabel 7.1 Beragam Operator Operasional

Operator	Keterangan	Contoh
<	Kurang dari	x < 5
<=	Kurang dari atau sama dengan	x <= 5
>	Lebih dari	x > 5
>=	Lebih dari atau sama dengan	x >= 5
==	Sama dengan	x == 5
~=	Tidak sama dengan	x ~= 5

Berikut ini disajikan beberapa contoh penggunaan operator relasional pada Matlab.

```
>> x = 5

x =

    5

>> x < 5

ans =

    0

>> x <= 5

ans =

    1

>> x > 5

ans =

    0
```

```
>> x >= 5

ans =

     1

>> x == 5

ans =

     1

>> x ~= 5
ans =

     0
>>
```

Makna dari angka 1 dan 0 adalah angka 1 bermakna ekspresi relasional tersebut benar, sedangkan angka 0 bermakna bahwa ekspresi relasional tersebut salah.

7.2 Operator Logika

Operator logika berfungsi untuk menggabungkan dua ekspresi relasional atau untuk membalik nilai logika dari suatu ekspresi relasional. Beberapa operator logika dapat dilihat pada tabel berikut.

Tabel 7.2 Beragam Operator Logika

Operator	Operator	Contoh
&	dan	p & q
|	atau	p | q
~	negasi	~p

Cara menentukan nilai kebenaran dari operator-operator tersebut dapat dilihat pada tabel berikut.

Tabel 7.3 Nilai Kebenaran Operator Logika

p	q	p & q	p | q	~p
benar	benar	benar	benar	salah
benar	salah	salah	benar	salah
salah	benar	salah	benar	benar
salah	salah	salah	salah	benar

Dari Tabel 7.3 tersebut, terlihat bahwa ekspresi p & q hanya akan bernilai benar jika p, q keduanya benar. Adapun p | q akan bernilai benar jika paling tidak salah satu di antara p, q bernilai benar. Berikut ini beberapa contoh penggunaan ekspresi logika pada Matlab.

```
>> x = 5;
>> x > 1 & x < 10

ans =

    1

>> x > 1 & x > 10

ans =

    0
>>
```

Pada contoh di atas, x kita beri nilai 5. Jelas x > 1 dan x < 10, sehingga pernyataan x > 1 & x < 10 akan bernilai benar, dalam hal ini Matlab akan mengeluarkan output 1. Adapun pernyataan x > 1 & x > 10 akan bernilai salah, sehingga Matlab akan mengeluarkan output 0. Sekarang, perhatikan contoh berikut.

```
x > 1 | x < 10

ans =

    1
```

```
>> x < 1 | x < 10

ans =

    1
```

Pada kedua contoh tersebut, kedua pernyataan x > 1 | x < 10 dan pernyataan x < 1 | x < 10 akan bernilai benar karena salah satu bentuk selalu bernilai benar, yaitu bentuk x < 10, sehingga Matlab memberikan output nilai 1. Adapun untuk penggunaan operator ~ dapat dilihat pada contoh berikut.

```
>> ~(x == 5)

ans =

    0

>> ~(x < 1)

ans =

    1

>>
```

Pada contoh itu, ~(x == 5) akan bernilai salah atau 0 karena x = 5 sehingga bentuk ~(x = 5) sama artinya dengan x ≠ 5. Adapun ~(x < 1) akan bernilai benar atau 1 karena ~(x < 1) sama artinya dengan x > 1.

7.3 Pernyataan IF

Untuk menangani pengambilan keputusan, Matlab menyediakan struktur IF dalam bentuk pertama berikut.

if *ekspresi*
    pernyataan
end

atau dapat pula dalam bentuk kedua berikut

if *ekspresi*
    pernyataan 1
else
    pernyataan 2
end

Pernyataan di sini dapat berupa lebih dari satu aksi. Pada bentuk pertama, pernyataan hanya akan dijalankan apabila ekspresi bernilai benar.

Adapun pada bentuk kedua, *pernyataan 1* hanya akan dijalankan apabila ekspresi bernilai benar, sedangkan pernyataan 2 akan dijalankan apabila ekspresi bernilai salah. Penggunaan if dapat Anda lihat pada contoh berikut.

```
>> x = 10;
>> if x >= 2
disp('oke');
end
oke
>>
```

Pada contoh di atas, tulisan oke akan muncul di layar karena ekspresi, yaitu x > = bernilai benar karena x = 10. Sekarang, cobalah contoh berikut.

```
>> x = –10;
>> if x >= 0
disp('Oke')
end
>>
```

Pada contoh tersebut, kata oke tidak muncul karena ekspresi x >= 0 bernilai salah karena x = –10.

7.4 Pernyataan If .. elseif ... else

Bentuk if ... elseif ... else dapat ditulis-kan sebagai berikut.

if *ekspresi1*
    pernyataan1
elseif *ekspresi2*
    pernyataan2
elseif *ekspresi3*
    pernyataan3
...
else
    pernyataann
end

Tanda ... menyatakan bahwa Anda boleh membuat lebih dari satu buah elseif. Pada bentuk tersebut, pernyata-an1 hanya akan dijalankan jika ekspre-si1 bernilai benar; pernyataan2 hanya akan dijalankan jika ekspresi2 bernilai benar dan ekspresi1 bernilai salah; dan pernyataann hanya akan dijalankan apa-bila semua ekspresi salah. Pada contoh berikut, akan diperlihatkan penggunaan bentuk elseif.

Contoh 7.1

Buatlah program untuk menentukan nilai huruf dengan masukan nilai angka yang bersesuaian. berdasarkan tabel berikut.

Kriteria	Nilai huruf
nilai $\geq$ 90	A
70 $\leq$ nilai < 90	B
60 $\leq$ nilai < 70	C
50 $\leq$ nilai < 60	D
nilai $\leq$ 50	E

Programnya adalah sebagai berikut.

```
nilai = input('Masukkan nilai (0 - 100): ');
if nilai >= 90
   hasil = 'A';
elseif nilai  >= 70
   hasil = 'B';
elseif nilai >= 60
   hasil = 'C';
elseif nilai >= 50
   hasil = 'D';
else
   hasil = 'E';
end
fprintf('Nilai huruf Anda adalah %c\n',hasil)
```

Apabila program tersebut dijalankan akan tampak seperti berikut.

```
>> skor_huruf
Masukkan nilai (0 - 100): 90
Nilai huruf Anda adalah A
>> skor_huruf
Masukkan nilai (0 - 100): 78
Nilai huruf Anda adalah B
>> skor_huruf
Masukkan nilai (0 - 100): 66
Nilai huruf Anda adalah C
>> skor_huruf
Masukkan nilai (0 - 100): 55
Nilai huruf Anda adalah D
>> skor_huruf43
??? Undefined function or variable 'skor_huruf43'.

>> skor_huruf
Masukkan nilai (0 - 100): 43
Nilai huruf Anda adalah E
>>
```

7.5 Pernyataan Switch

Pernyataan switch adalah bentuk lain dari pernyataan if, sehingga switch pun berfungsi untuk mengambil keputusan. Bentuk dari pernyataan switch adalah sebagai berikut.

```
switch ekspresi
    case {kasus11, kasus12, ...}
        pernyataan1
    case kasus2
        pernyataan2
    ....
    otherwise
        pernyataann
end
```

Nilai ekpresi akan dicocokan dengan nilai kebenaran pada kasus. Pencocokan nilai tersebut dilakukan secara bertingkat mulai dari kasus yang paling atas. Jika nilai ekspresi cocok dengan kasus1 maka hanya pernyataan1 yang akan dijalankan. Jika ternyata ekspresi tidak cocok dengan kasus1 maka pencocokan akan dilanjutkan ke kasus2, dan seterusnya. Apabila ternyata tidak ada satu kasuspun yang cocok maka bagian **otherwise** akan dijalankan. Pada bentuk switch, Anda dapat menuliskan satu buah kasus atau lebih untuk ekspresi yang sama. Apabila Anda memiliki lebih dari satu kasus untuk ekspresi yang sama maka gunakan tanda kurung kurawal {} di awal dan akhir kasus, serta gunakan tanda koma untuk memisahkan kasus.

Penggunaan switch dapat dilihat pada contoh berikut.

Contoh 7.2

```
st = input('Masukkan salah satu dari empat penjuru mata angin: ','s');
switch lower(st)
   case {'utara', 'north'}
      disp('Utara/North')
   case {'selatan','south'}
      disp('Selatan/South')
   case {'barat','west'}
      disp('Barat/West')
   case {'timur','east'}
      disp('Timur/East')
   otherwise
      disp('Arah mata angin salah')
end
```

Pada program tersebut, perintah lower(st) berfungsi untuk mengonversikan semua huruf kapital menjadi huruf kecil. Dengan menggunakan huruf kapital ataupun huruf kecil. Pada program ini berlaku hal-hal berikut.

1. Jika input diisi utara atau north maka pernyataan disp('Utara/North') akan dijalankan.

2. Jika input diisi selatan atau south maka pernyataan disp('Selatan/South') akan dijalankan.

3. Jika input diisi barat atau west maka pernyataan disp('Barat/West') akan dijalankan.

4. Jika input diisi timur atau east maka pernyataan disp('Timur/East') akan dijalankan.

Latihan Bab VII

1. Tentukan hasil dari x >= 8 & y < 5 jika x = 3 dan y = 7.

2. Tentukan hasil dari x >= 8 | y < 5 jika x = 3 dan y = 7.

3. Buatlah program dengan masukan sebuah bilangan bulat. Program dapat menampilkan tulisan positif jika x > 0, nol jika x = 0, dan negatif jika x < 0.

4. Buatlah sebuah program dengan masukan sebuah bilangan. Apabila bilangan tersebut genap maka keluarkan output GENAP, begitu pula sebaliknya apabila bilangan itu ganjil.

5. Tuliskan algoritma untuk membaca tiga bilangan bulat lalu menentukan bilangan yang terbesar.

6. Karyawan PT Kanjuruhan digaji berdasarkan jumlah jam kerjanya selama seminggu. Upah per jam Rp2000, 00. Bila jumlah jam kerja lebih besar daripada 48 jam maka sisanya dianggap lembur. Upah lembur Rp3000, 00 per jam. Tuliskan algoritma untuk menentukan upah mingguan karyawan dengan masukan jumlah jam kerja mereka.

7. Suatu tahun disebut tahun kabisat jika memenuhi salah satu syarat berikut: habis dibagi 4 tetapi tidak habis dibagi 100; atau habis dibagi 400.

8. Buatlah algoritma untuk menentukan wujud air (padat, cair, gas) pada suhu tertentu dengan masukan suhu air itu.

9. Buatlah algoritma untuk menentukan kuadran dari suatu titik dengan masukan koordinat titik tersebut.

10. Gunakan struktur CASE untuk mencetak nama bulan sesuai dengan angka bulan yang dimasukkan.

Bab VII

Sumber: http://comps.canstockphoto.com.

Pengulangan

8.1 Pernyataan while

Pernyataan for dan while merupakan perintah yang berfungsi untuk menangani suatu pengulangan. Pada bahasan pertama, Anda akan mempelajari penggunaan perintah while dalam pengulangan terlebih dahulu, baru setelah itu Anda akan mempelajari penggunaan perintah for.

Pernyataan while merupakan perintah yang berguna untuk menangani suatu pengulangan. Bentuk umum dari pernyataan while dapat dituliskan sebagai berikut.

while *ekspresi*
    pernyataan
end

Diagram alir bentuk pernyataan while dapat digambarkan sebagai berikut.

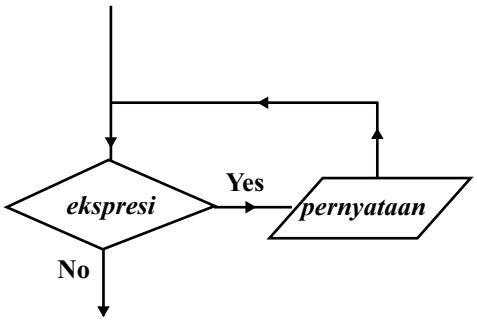


Diagram tersebut menunjukkan bahwa pengulangan terhadap bagian *pernyataan* dilakukan selama ekspresi pada while bernilai benar. Gambar tersebut juga menunjukkan bahwa ada kemungkinan bagian *pernyataan* tidak dijalankan sama sekali apabila ekspresi pada while bernilai salah.

Pada program berikut, akan ditunjukkan cara menampilkan tulisan *hello world* sebanyak 10 kali.

Contoh 8.1

```
clc;
clear all;

hitung = 1;

while hitung <= 10
   disp('Hello World')
   hitung = hitung + 1;
end
```

Pada program tersebut, terdapat variabel yang bernama hitung. Fungsi dari variabel ini adalah untuk menghitung banyaknya kemunculan tulisan *hello world* tersebut. Perintah hitung = hitung + 1 berfungsi untuk menaikkan isi variabel hitung setiap kali pengulangan dijalankan.

Apabila program tersebut dijalankan maka tampilannya akan tampak seperti berikut.

```
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
>>
```

Apabila Anda ingin menampilkan isi dari variabel hitung, Anda dapat dapat menggunakan contoh berikut.

Contoh 8.2

```
clc;
clear all;

hitung = 1

while hitung <= 10
   hitung = hitung + 1
end
```

Apabila program tersebut dijalankan maka hasilnya adalah sebagai berikut.

```
hitung =

    1


hitung =

    2


hitung =

    3


hitung =

    4


hitung =

    5
```

```
hitung =

     6

hitung =

     7

hitung =

     8

hitung =

     9

hitung =

    10

hitung =

    11

>>
```

Pada penggunaan perintah while, harus ada pernyataan yang akan membuat ekspresi while bernilai salah untuk menghentikan perulangan. Pada akhir program di atas, nilai akhir dari variabel hitung adalah 11. Nilai inilah yang akan membuat ekspresi while hitung <=10 akan bernilai salah sehingga pengulangan akan berhenti.

8.2 Pernyataan for

Selain perintah while, Matlab juga menyediakan perintah for untuk melakukan operasi pengulangan. Bentuk pernyataan for dapat ditulis sebagai berikut.

```
for variabel = ekspresi
    pernyataan
end
```

Program berikut akan memunculkan 15 bilangan asli pertama.

Contoh 8.3

```
clc;
clear all;

for r = 1:10
   disp(r)
end
```

Apabila program tersebut dijalankan maka tampilannya akan menjadi seperti berikut.

```
     1

     2

     3

     4

     5

     6

     7

     8

     9

    10

>>
```

Ekspresi for r = 1:10 menyatakan bahwa pada setiap iterasi, nilai r akan bergerak mulai dari 1 hingga 10.

Contoh berikutnya adalah program untuk melakukan penjumlahan *n* bilangan asli pertama.

Contoh 8.4

```
clc;
clear all;

n = input('Masukkan bilangan asli n: ');
jumlah = 0;
for r = 1:n
   jumlah = jumlah + r;
end
fprintf('Jumlah bilangan dari 1 hingga %d adalah %d',n,jumlah)
```

Apabila program tersebut dijalankan maka tampilannya akan sebagai berikut.

```
Masukkan bilangan asli n: 3
Jumlah bilangan dari 1 hingga 3 adalah 6
```

Pada program tersebut, nilai variabel jumlah akan dimulai dari 0. Kemudian, seiring dengan berjalannya iterasi, nilai variabel jumlah akan berubah menjadi jumlah sebelumnya ditambah dengan r, ditulis jumlah = jumlah + r.

8.3 Pernyataan break

Pernyataan break berfungsi untuk mengakhiri eksekusi dari suatu pernyataan for atau while. Perhatikan contoh berikut.

Contoh 8.5

```
n = input('Masukkan bilangan asli n: ');
jumlah = 0;
for r = 1:n
   jumlah = jumlah + r;
   if r == 5
      break
   end
end
fprintf('Jumlah bilangan dari 1 hingga %d
adalah %d',r,jumlah)
```

Apabila program tersebut dijalankan, maka akan tampak seperti berikut.

```
Masukkan bilangan asli n: 3
Jumlah bilangan dari 1 hingga 3 adalah 6
Masukkan bilangan asli n: 8
Jumlah bilangan dari 1 hingga 5 adalah 15
Masukkan bilangan asli n: 12
Jumlah bilangan dari 1 hingga 5 adalah 15
>>
```

Pada tampilan program di atas, proses iterasi maksimal yang dapat dilakukan hanyalah hingga r = 5, walaupun input yang dimasukkan lebih dari 5.

Latihan Bab VIII

1. Buatlah program untuk menampilkan bilangan ganjil mulai dari 1 hingga 15.

2. Buatlah program untuk menampilkan bilangan mulai dari −4, 25 hingga 2, 8 dengan kenaikan 0, 25

3. Buatlah program untuk menampilkan bilangan 1 hingga 10 dan tentukan jenis bilangan-bilangan tersebut apakah genap atau ganjil.

4. Buatlah algoritma untuk menghitung 1+2+3+···+100. Kemudian, tampakkan di layar tulisan berupa 1+2+3+···+ 100 = 5050 dengan memanfaatkan instruksi pengulangan dan pemilihan.

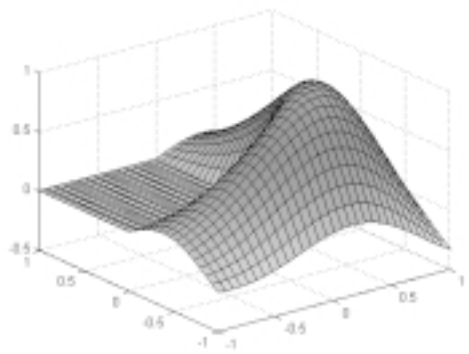
5. Buatlah algoritma untuk menghitung 1+3+5+···+ 99.

6. Buatlah algoritma untuk membaca integer tak negatif n dan menghitung faktorialnya. Kemudian, tampilkan tulisan misalnya 3! = 1 × 2 × 3=6.

7. Buatlah algoritma untuk menghitung 1 ×2 ×4 ×7 × 11 ×⋯× *n*. Kemudian, tampilkan tulisan misalnya 1 × 2 × 4 = 8.

8. Diketahui deret Fibonacci
   $S_1 = 1, S_2 = 1, S_n = S_{n-1} + S_{n-2}$
   untuk $n \geq 3$. Buat algoritma untuk membaca nilai n, kemudian menghitung nilai Sn dan tampilkan barisan Fibonaccinya.

Bab IX



Sumber: http://www.mathworks.com.

Larik dan Grafik Pada Matlab

9.1 Mengenal Larik

Sebuah larik dapat menampung sejumlah data yang sejenis. Dalam matematika, larik dapat disamakan dengan vektor ataupun matriks. Vektor merupakan larik satu dimensi. Vektor kolom adalah vektor dengan satu kolom dan vektor baris adalah vektor dengan satu baris. Adapun matriks adalah larik yang berdimensi dua. Cara menyatakan larik misalnya sebagai berikut.

```
>> A = [1;2;3;4;5]

A =

    1
    2
    3
    4
    5

>> B = [1 2 3 4 5]

B =

    1    2    3    4    5

>>
```

Pada contoh tersebut, A adalah sebuah vektor kolom karena hanya terdiri atas sebuah kolom, sedangkan B merupakan vektor baris karena hanya terdiri atas satu baris. Tanda [ ] digunakan untuk menyatakan larik, sedangkan tanda titik koma digunakan sebagai pemisah antarelemen larik.

9.2 Transpos Pada Larik

Pada Matlab, terdapat operator yang dinamakan dengan transpos. Transpos akan menukar posisi elemen larik dari baris menjadi kolom. Perhatikan contoh berikut.

```
>> A

A =

    1
    2
    3
    4
    5

>> A'
ans =

    1    2    3    4    5
>>
```

Pada contoh tersebut, A adalah sebuah vektor kolom. Ketika ditranspos, A akan berubah dari vektor kolom menjadi vektor baris. Begitu pula sebaliknya dengan vektor baris akan berubah mejadi vektor kolom seperti berikut.

```
>> B = [1 2 3 4 5]

B =

    1    2    3    4    5

>> B'

ans =

    1
    2
    3
    4
    5

>>
```

Vektor merupakan contoh dari larik satu dimensi, sedangan matriks merupakan contoh dari larik 2 dimensi. Cara membentuk matriks pada Matlab dapat dilihat pada contoh berikut.

Misalkan A adalah matriks dengan:

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Perintah Matlab untuk menuliskan matriks tersebut adalah

```
>> A = [1 2; 3 4]

A =

    1    2
    3    4

>>
```

9.3 Operasi Pada Larik

Seperti halnya matriks, larik pun dapat dioperasikan secara matematis. Perhatikan contoh-contoh berikut.

Contoh 9.1

```
>> A = [1 2; 3 4]

A =

    1    2
    3    4
```

```
>> B = [5 6; 7 8]

B =

    5    6
    7    8

>> C=A+B

C =

    6    8
   10   12

>>
```

Operasi di atas merupakan contoh operasi penjumlahan pada larik. Berikut adalah contoh operasi pengurangan pada larik.

Contoh 9.2

```
D = A - C

D =

   -5   -6
   -7   -8
```

Contoh 9.3

```
>> E = A*B

E =

    19    22
    43    50
```

Contoh di atas merupakan contoh perkalian dua larik. Operasi lain yang dapat dilakukan adalah operasi invers matriks persegi. Berikut ini contohnya.

Contoh 9.4

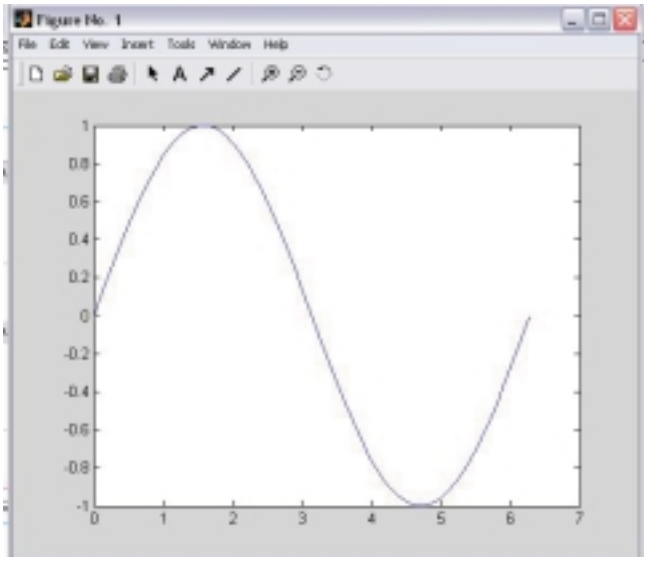
```
>> inv(A)

ans =

   -2.0000    1.0000
    1.5000   -0.5000

>>
```

9.4 Bekerja Dengan Grafik

Menyajikan grafik dua dimensi yang menyatakan hubungan nilai dalam sumbu x dan sumbu y dapat dilaksanakan dengan mudah dengan menggunakan fungsi bernama plot. Contoh penggunaan plot misalnya untuk menggambarkan fungsi sinus.

Contoh 9.5

```
>> x = linspace(0,2*pi);
>> y = sin(x);
>> plot(x,y);
>>
```

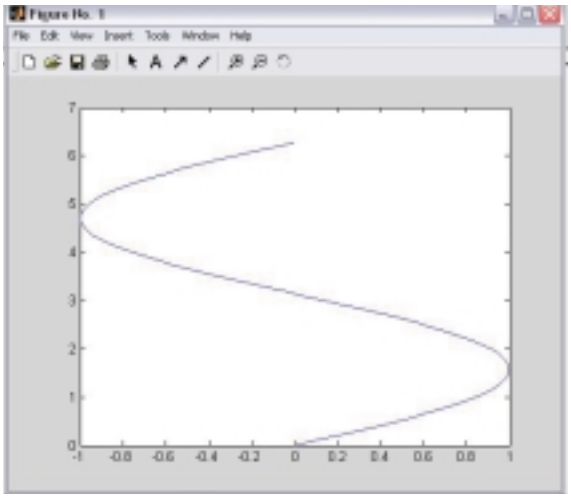


Pada contoh di atas, x berisi 100 nilai antara 0 sampai dwengan 2*pi. Adapun y berisi 100 nilai sinus yang didasarkan pada nilai vektor x.

Anda juga dapat mencoba untuk menukarkan posisi x dan y pada plot seperti berikut.

Contoh 9.6

```
>> x = linspace(0,2*pi);
>> y = sin(x);
>> plot(y,x);
>>
```



Tiga buah fungsi yang berguna untuk memberikan judul untuk grafik adalah xlabel, ylabel, dan title. Fungsi dari setiap perintah tersebut adalah sebagai berikut.

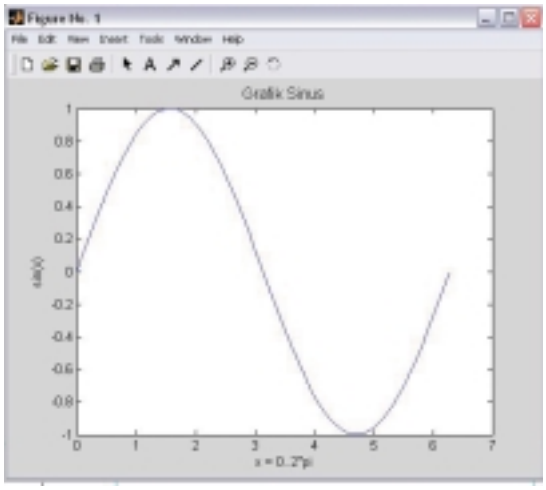
1. xlabel berfungsi menambahkan judul pada sumbu x
2. ylabel berfungsi menambahkan judul pada sumbu y.

3. title berfungsi menambahkan judul grafik.

Sebagai contoh, perhatikan program berikut.

Contoh 9.7

```
>> clear all;
>> x = linspace(0,2*pi);
>> y = sin(x);
>> plot(x,y);
>> xlabel('x = 0..2*pi');
>> ylabel('sin(x)');
>> title('Grafik Sinus','Fontsize',12);
>>
```



Latihan Bab IX

1. Bagaimana perintah untuk mendapatkan vektor baris berikut? [10 30 31 22 45 71 15]

2. Buatlah grafik dengan plot untuk menggambarkan fungsi tangen pada $0 \leq x \leq 1$. Berikan label pada sumbu x dan sumbu y.

3. Buatlah grafik dengan plot untuk menggambarkan fungsi cos pada $0 \leq x \leq 2\pi$. Berikan label pada sumbu x dan sumbu y.

4. Buatlah grafik dengan plot untuk menggambarkan fungsi sin(2x) pada $0 \leq x \leq 2\pi$. Berikan label pada sumbu x dan sumbu y.

5. Buatlah grafik dengan plot untuk menggambarkan fungsi 0,8cos(x) pada $0 \leq x \leq 2\pi$. Berikan label pada sumbu x dan sumbu y.

Daftar Pustaka

Attaway, S. 2009. *MATLAB A Practical Introduction to Programming and Problem Solving*. Burlington: Butterworth-Heinemann.

Chapman, S. J. 2008. *MATLAB Programming for Engineer*. Toronto: Thomson-Learning.

Davis, T.A. 2011. *MATLAB Primer*. Boca Raton: CRC Press.

McMahon, D. 2007. *MATLAB Demystified A Self-Teaching Guide*. New York: The McGraw-Hill Companies.

The MathWorks. 2009. *Image Processing Toolbox 6 User's Guide*. Natick: The MathWorks, Inc.