



**Laporan Praktikum Algoritma & Pemrograman
Semester Genap 2025/2026**

SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAHINI.

NIM	71251233
Nama Lengkap	Mikael Gratianus Satrio Adi Kuncoro
Minggu ke / Materi	03 / Flowchart dan Pseudocode

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS KRISTEN DUTA WACANA
YOGYAKARTA
2026**

Algoritma

Algoritma adalah rangkaian langkah-langkah logis dan sistematis yang disusun untuk menyelesaikan masalah. Tujuannya sebagai petunjuk atau panduan tentang langkah-langkah penyelesaian masalah yang mudah dipahami dalam mengembangkan program komputer serta mencegah kesalahan logika sejak awal.

Penulisan (Notasi Algoritma)

Ada tiga macam bentuk notasi algoritma antara lain:

1. Uraian deskriptif
2. *Flowchart / Diagram Alir*
3. *Pseudocode*

Uraian Deskriptif

Notasi algoritma dalam bentuk uraian deskriptif, cara penulisan atau penggerjaannya menggunakan bahasa sehari-hari yang disusun secara runtut dan logis untuk menjelaskan langkah-langkah penyelesaian suatu masalah.

Bentuk notasi ini tidak menggunakan simbol khusus dan tidak terikat aturan sintaks seperti bahasa pemrograman. Tujuannya adalah agar langkah-langkah mudah dipahami oleh manusia sebelum diterjemahkan ke dalam bentuk program. Contoh: menyelesaikan permasalahan dalam menghitung luas dan keliling suatu lingkaran. Berikut algoritma menghitung luas dan keliling lingkaran.

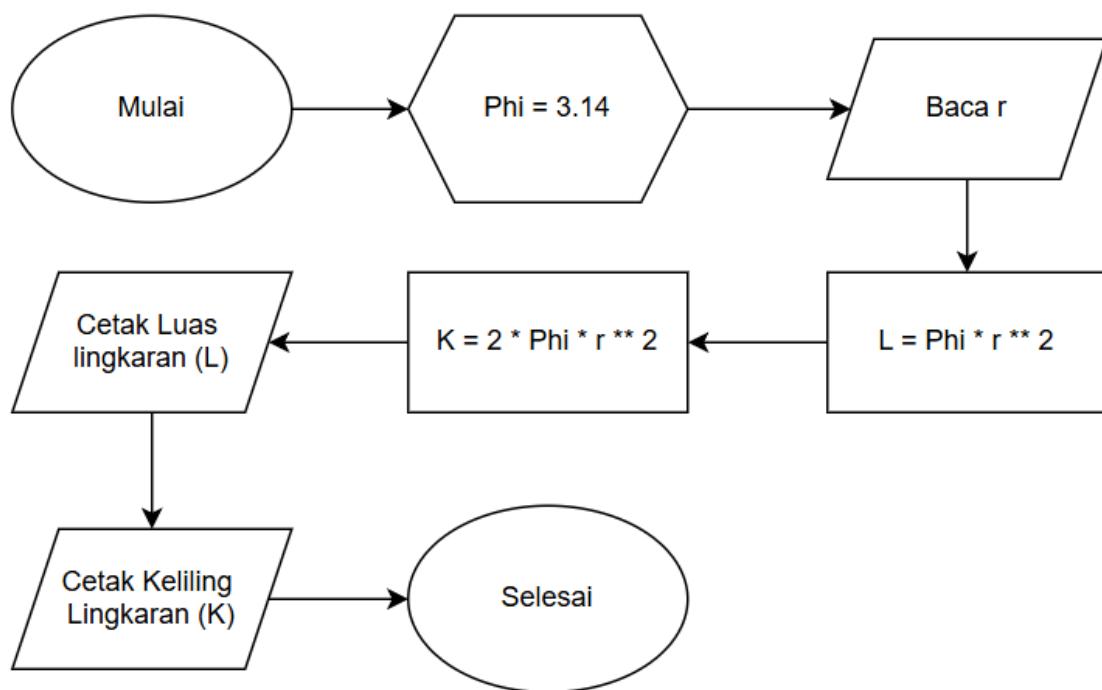
Deskripsi:

1. Masukan jari-jari lingkaran (r).
2. Hitung luas lingkaran dengan rumus $L = \pi * r ** 2$.
3. Hitung keliling lingkaran dengan rumus $K = 2 * \pi * r$.
4. Tampilkan luas lingkaran.
5. Tampilkan keliling lingkaran.

Flowchart / Diagram Alir

flowchart adalah representasi visual dari langkah-langkah penyelesaian suatu masalah yang digambarkan menggunakan simbol-simbol khusus dan dihubungkan dengan panah

untuk menunjukkan alur proses secara sistematis. Flowchart juga membantu mempermudah pemahaman logika karena setiap langkah, seperti proses, input/output, keputusan (percabangan), dan awal atau akhir program ditampilkan dalam bentuk diagram, sehingga lebih mudah dibaca dan dianalisis dibandingkan hanya menggunakan teks. Berikut merupakan contoh *flowchart* menghitung luas dan keliling lingkaran yang algoritmanya dinotasikan dalam bentuk diagram alir (*flowchart*). Contohnya dapat dilihat pada Gambar 1.1.



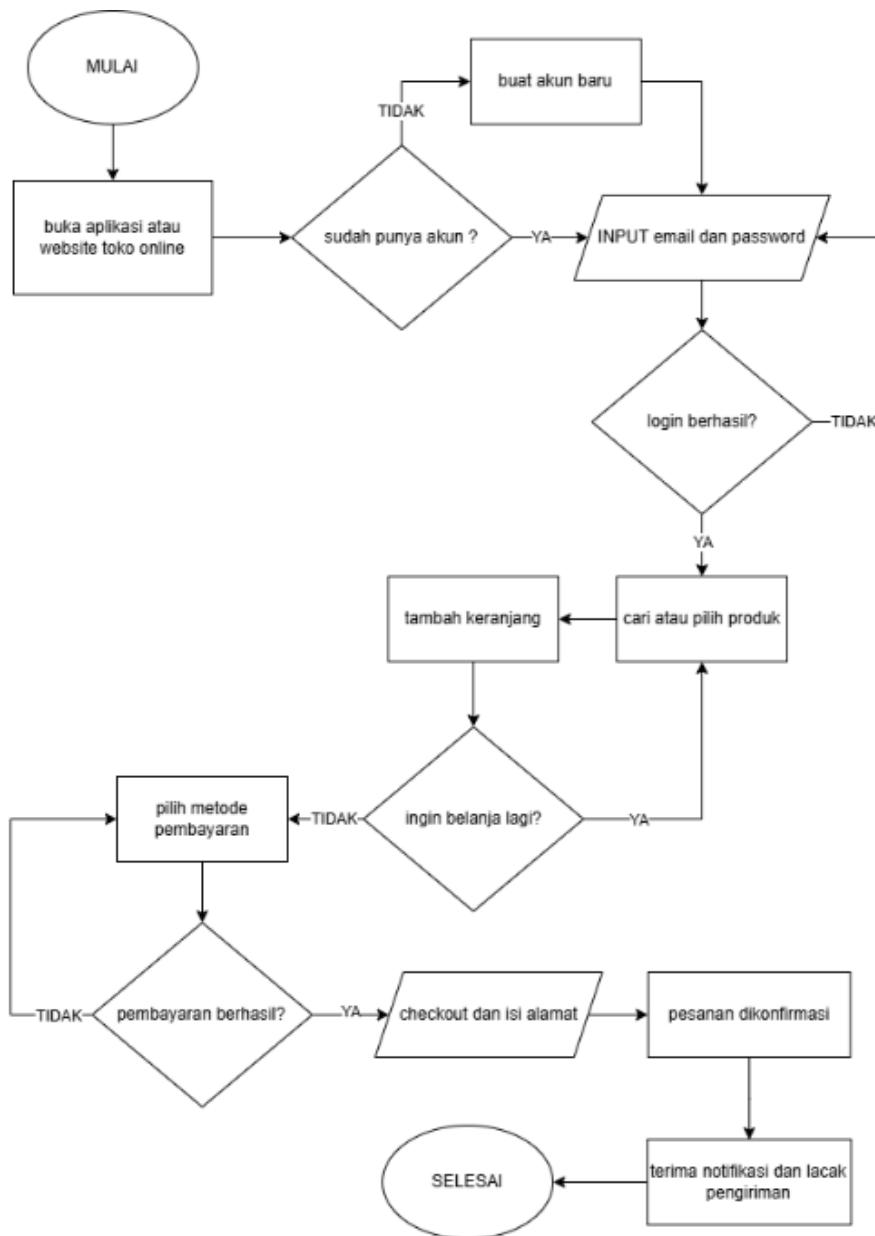
Gambar 1.1: *Flowchart* mencari luas dan keliling lingkaran.

Berikut merupakan contoh lain dari algoritma dalam bentuk *flowchart*, yaitu proses belanja online yang terdapat pada Gambar 1.2.

Alur *flowchart*:

1. Mulai: Buka website/aplikasi
2. Cek akun: Jika belum punya, daftar dulu
3. Login: Jika gagal, ulangi input
4. Cari & pilih produk: Tambah ke keranjang
5. Mau beli lagi?: Jika ya, kembali cari produk

6. Checkout: Isi alamat & pilih pembayaran
7. Pembayaran: Jika gagal, ulangi
8. konfirmasi & Tracking: Selesai



Gambar 1.2: *Flowchart* proses belanja online.

Flowchart menolong analis dan programmer untuk memecahkan masalah kedalam segmen-semen yang lebih kecil dan menolong dalam menganalisis alternatif-alternatif lain dalam pengoperasian. Berikut merupakan kegunaaan nya:

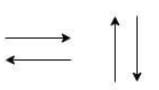
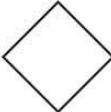
1. Untuk mendesain program.

2. Untuk merepresentasikan program.
3. Membuat proses berpikir jadi lebih terarah.
4. Membantu menyelesaikan masalah secara sistematis.
5. Membuat solusi lebih efisien.

Maka, *flowchart* harus dapat merepresentasikan komponen-komponen dalam bahasa pemrograman.

Notasi *Flowchart*

Pada dasarnya, setiap notasi dalam flowchart memiliki arti dan fungsi yang berbeda-beda. Dengan memahami arti masing-masing simbol, analisis atau pembaca dapat lebih mudah mengikuti logika dan urutan langkah dalam penyelesaian suatu masalah dengan mudah. Berikut adalah contoh-contoh notasi yang sering digunakan dalam proses pembuatan flowchart yang dapat dilihat pada Gambar 1.2.

	Flow Simbol yang digunakan untuk menggabungkan antara simbol yang satu dengan simbol yang lain. Simbol ini disebut juga dengan Connecting Line.
	On-PAGE Reference Simbol untuk keluar - masuk atau penyambungan proses dalam lembar kerja yang sama.
	Off-PAGE Reference Simbol untuk keluar - masuk atau penyambungan proses dalam lembar kerja yang berbeda.
	Terminator Simbol yang menyatakan awal atau akhir suatu program.
	Process Simbol yang menyatakan suatu proses yang dilakukan komputer.
	Decision Simbol yang menunjukkan kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban, yaitu ya dan tidak.
	Input/output Simbol yang menyatakan proses input atau output tanpa tergantung peralatan.
	Manual Operation Simbol yang menyatakan suatu proses yang tidak dilakukan oleh komputer.
	Document Simbol yang menyatakan bahwa input berasal dari dokumen dalam bentuk fisik, atau output yang perlu dicetak.
	Predefine Proses Simbol untuk pelaksanaan suatu bagian (sub-program) atau prosedure.
	Display Simbol yang menyatakan peralatan output yang digunakan.
	Preparation Simbol yang menyatakan penyediaan tempat penyimpanan suatu pengolahan untuk memberikan nilai awal.

Gambar 1.2: Notasi *Flowchart*.

Notasi di atas memiliki jenis dan fungsi yang berbeda-beda. Ada yang berfungsi untuk menghubungkan satu notasi dengan notasi lainnya seperti notasi *flow*, *on-page* dan *off-page reference*. Selain itu ada juga notasi yang berfungsi untuk menunjukkan suatu proses yang

sedang berjalan, dan yang terakhir terdapat notasi yang berfungsi untuk memasukan input dan menampilkan output.

Pseudocode

Pseudocode adalah cara menuliskan langkah-langkah penyelesaian suatu masalah atau algoritma dalam bentuk teks yang menyerupai bahasa pemrograman tetapi tidak terikat pada aturan sintaks tertentu, sehingga lebih mudah dipahami oleh manusia. *Pseudocode* digunakan untuk merancang logika program sebelum diimplementasikan ke dalam bahasa pemrograman seperti Python atau C, dengan tujuan membantu programmer memahami alur proses, mengurangi kesalahan saat coding, dan menyederhanakan komunikasi tentang cara kerja suatu program. Struktur algoritma dibagi ke dalam beberapa bagian, diantaranya:

1. Bagian kepala (header).
2. Bagian Deklarasi (definisi variable).
3. Bagian Deskripsi (rincian langkah).

Contoh:

```
Algoritma Luas_persegi_panjang
{Menghitung sebuah luas persegi panjang
apabila panjang dan lebar persegi panjang
tersebut diberikan}
```

Deklarasi

```
{Definisi nama peubah/variabel}
float panjang, lebar, luas
```

Deskripsi

```
READ (panjang, lebar)    #bisa juga: INPUT
luas <- panjang * lebar
WRITE (Luas)             #bisa juga: OUTPUT
```

Berikut merupakan contoh lain dari suatu algoritma dalam bentuk *pseudocode*, yaitu algoritma menentukan bilangan ganjil dalam urutan angka:

START

INPUT n

FOR i = 1 TO n DO

```
IF i MOD 2! = 0 THEN  
    OUTPUT i  
END IF  
END FOR  
END
```

Penjelasan perbaris:

START menandakan awal dari algoritma. INPUT n meminta pengguna untuk memasukkan sebuah angka sebagai batas akhir urutan. Misalnya pengguna memasukkan angka 10. FOR i = 1 TO n DO memulai perulangan dari i = 1 sampai i = n. Setiap putaran nilai i bertambah 1. Jadi urutannya: 1, 2, 3, 4, 5 ... sampai n. IF i MOD 2! = 0 THEN mengecek apakah angka i adalah bilangan ganjil. Caranya dengan MOD 2 yaitu membagi i dengan 2 lalu mengambil sisa hasilnya. Jika sisa bagi tidak sama dengan 0 maka angka tersebut adalah ganjil. Contoh: 3 MOD 2 = 1 → ganjil berarti benar, 4 MOD 2 = 0 → genap berarti salah. OUTPUT i Jika angka terbukti ganjil maka tampilkan angka tersebut. END IF menandakan akhir dari blok percabangan IF. END FOR menandakan akhir dari blok perulangan FOR. Setelah ini perulangan kembali ke atas sampai i mencapai n. END menandakan akhir dari algoritma.

Notasi *Pseudocode*

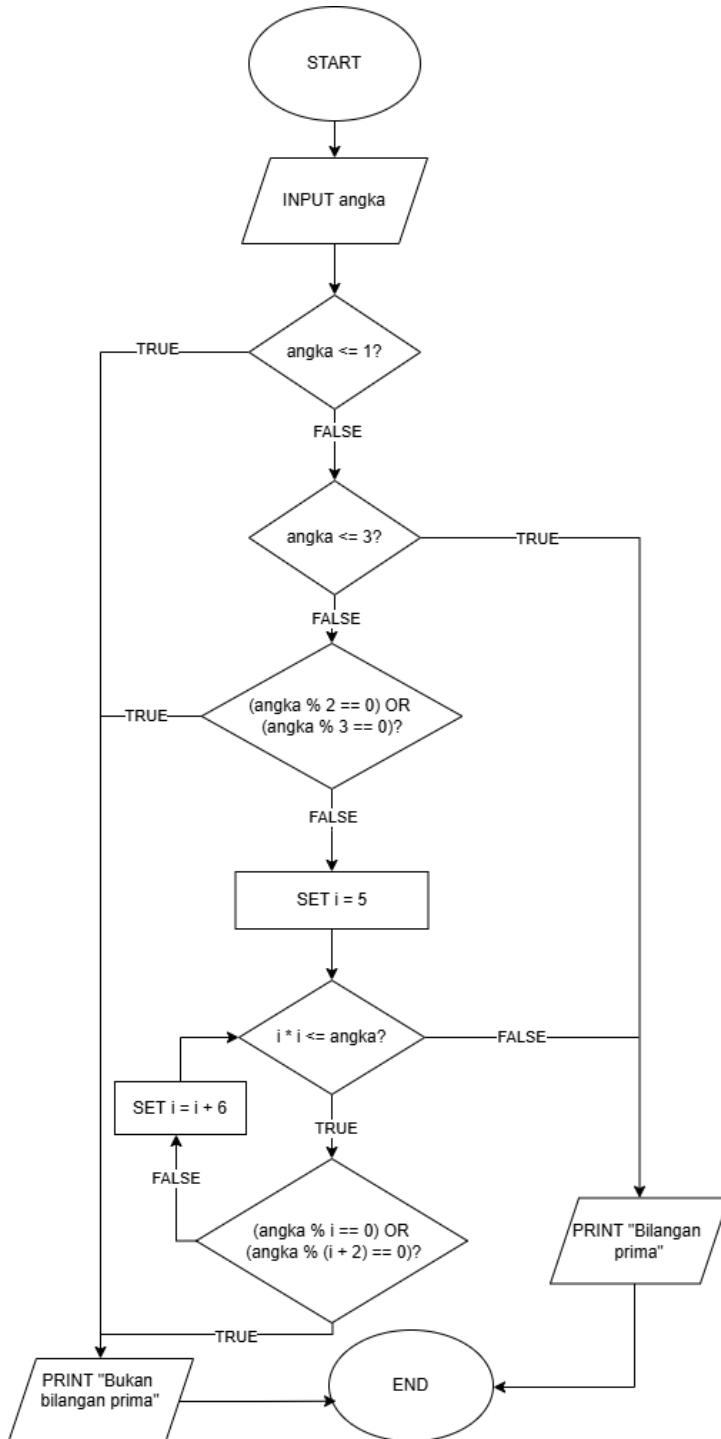
Berikut ini notasi yang sering digunakan dalam *pseudocode*:

1. INPUT -> Digunakan untuk menunjukkan proses memasukan suatu isi variabel.
2. OUTPUT -> Digunakan untuk menunjukkan proses keluaran yang terjadi.
3. WHILE -> Digunakan untuk sebuah perulangan yang memiliki iterasi awali.
4. FOR -> Digunakan untuk sebuah perulangan perhitungan iterasi.
5. REPEAT - UNTIL -> Digunakan untuk sebuah perulangan yang memiliki kondisi akhir.
6. IF – THEN – ELSE -> Digunakan untuk mengambil sebuah keputusan dari beberapa kondisi.

LINK GITHUB

: <https://github.com/Rio-code-07/71251233-Rio-PrakAlpro.git>

SOAL 1



Pseudocode-nya:

START

INPUT bilangan

IF bilangan <= 1 THEN

PRINT "Bukan bilangan prima"

ELSE IF bilangan <= 3 THEN

PRINT "Bilangan prima"

ELSE IF (bilangan % 2 == 0) OR
(bilangan % 3 == 0) THEN

PRINT "Bukan bilangan prima"

ELSE

SET i = 5

WHILE $i * i \leq \text{bilangan}$ DO

IF (bilangan % i == 0) OR
(bilangan % (i + 2) == 0) THEN

PRINT "Bukan bilangan prima"

EXIT

END IF

SET i = i + 6

END WHILE

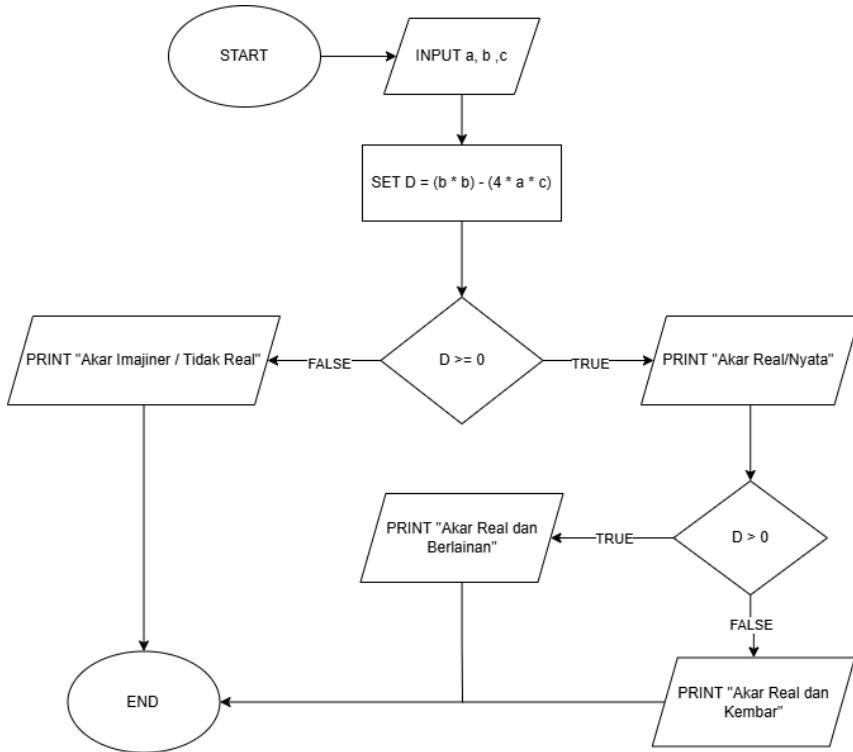
PRINT "Bilangan prima"

END IF

END

Gambar 1.3: Flowchart menentukan bilangan prima atau bukan

SOAL 2



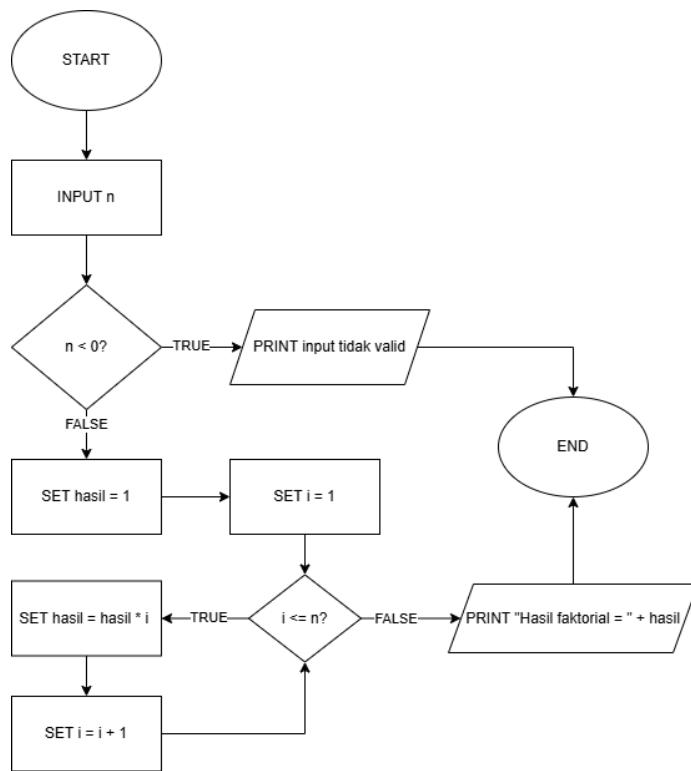
Gambar 1.4: *Flowchart* menentukan jenis akar dari suatu persamaan kuadrat

Pseudocode-nya:

```

START
  INPUT a, b, c
  SET D = (b * b) - (4 * a * c)
  IF D >= 0 THEN
    PRINT "Akar Real/Nyata"
    IF D > 0 THEN
      PRINT "Akar Real dan Berlainan"
    ELSE
      PRINT "Akar Real dan Kembar"
    END IF
  ELSE
    PRINT "Akar Imaginer / Tidak Real"
  END IF
END
  
```

SOAL 3



Gambar 1.5: Flowchart menghitung nilai faktorial dari suatu bilangan

Pseudocode-nya:

```
START
  INPUT n
  IF n < 0 THEN
    PRINT "Input tidak valid" #ini digunakan jika pengguna menginput angka negatif
  ELSE
    SET hasil = 1
    SET i = 1
    WHILE i <= n DO
      SET hasil = hasil * i
      SET i = i + 1
    END WHILE
    PRINT "Hasil faktorial = " + hasil
  END IF
END
```