GuruVirtual.ID

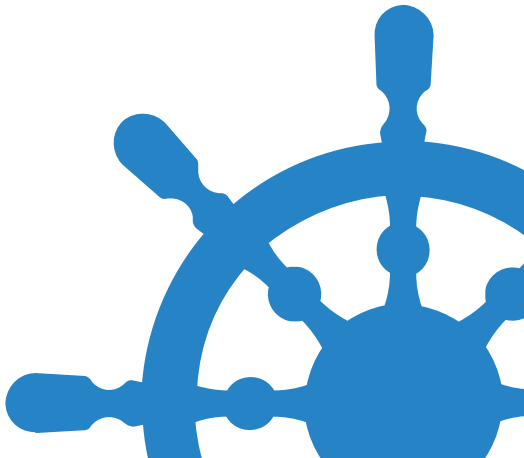
Pengenalan IPO dan Notasi dalam Algoritma

Hartono, S.Pd., M.T.I

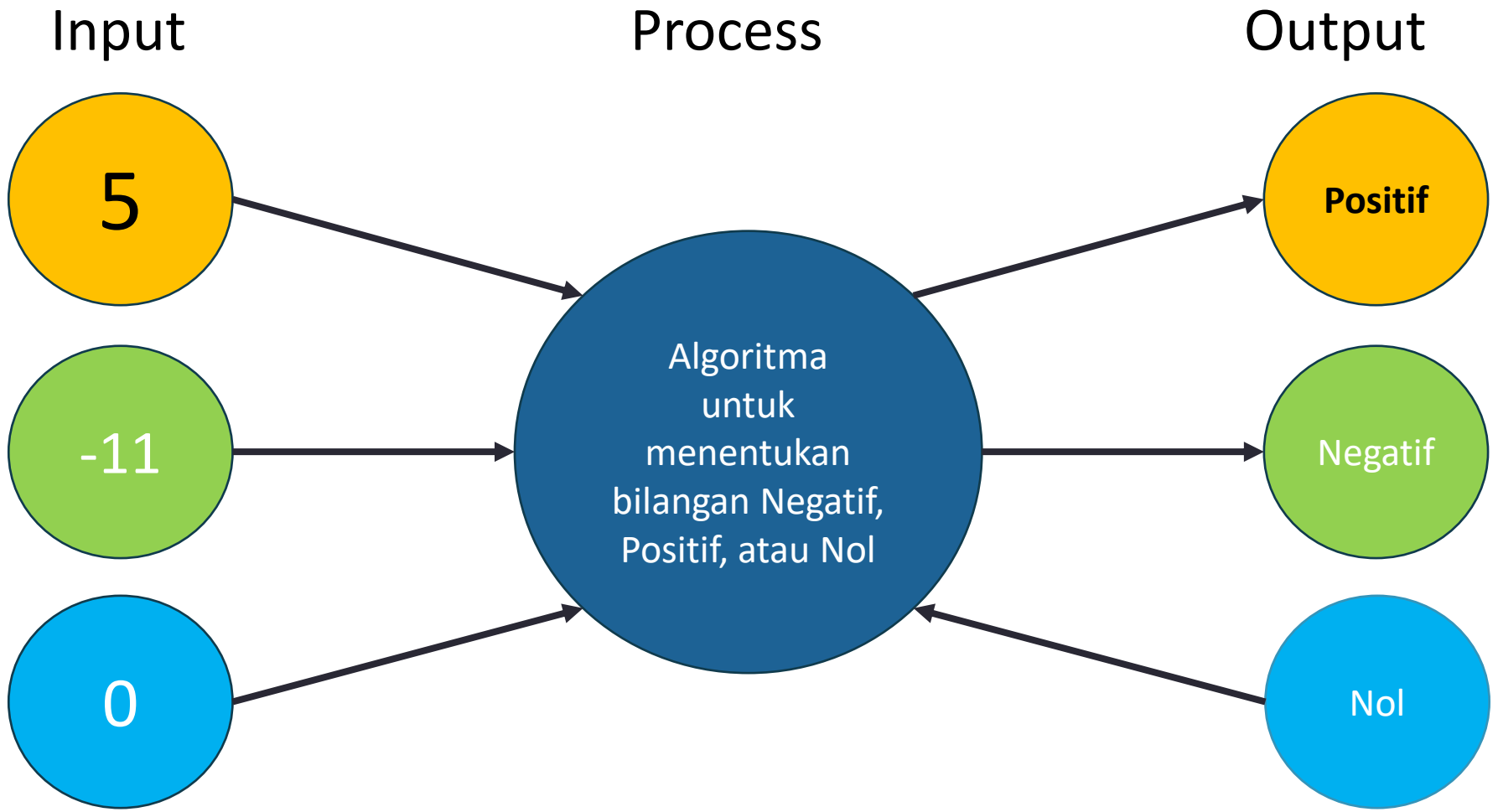
Universitas Muhammadiyah Kotabumi

Konsep Dasar Input, Process, dan Output (IPO)

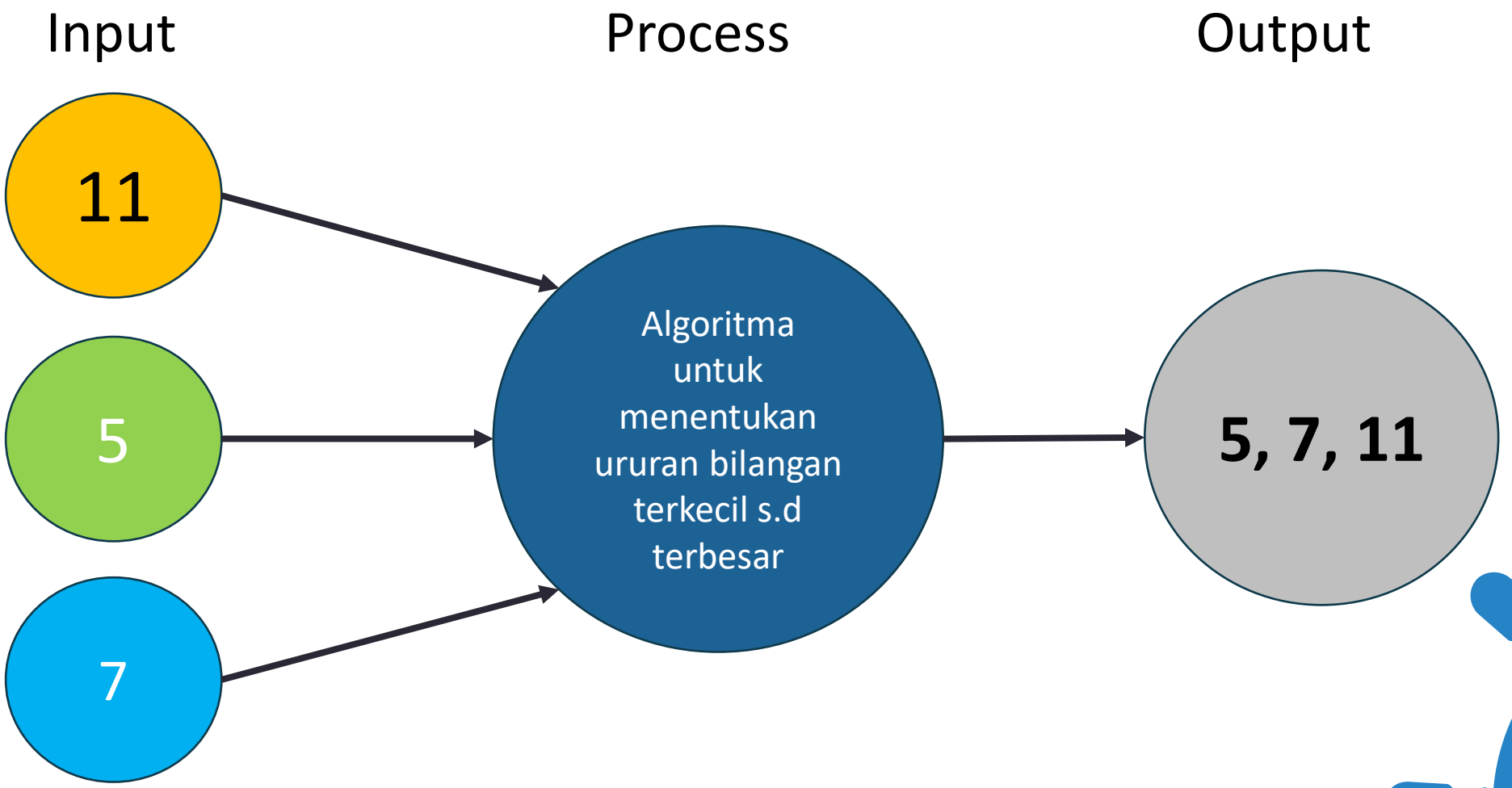
- Konsep input, process, dan output adalah prinsip dasar dalam pemrograman dan pengembangan algoritma.
- Setiap algoritma melibatkan tiga tahap utama: mengambil data masukan (input), melakukan operasi atau pengolahan data (process), dan menghasilkan hasil akhir (output).
- Konsep ini menggambarkan bagaimana algoritma beroperasi untuk memproses informasi.



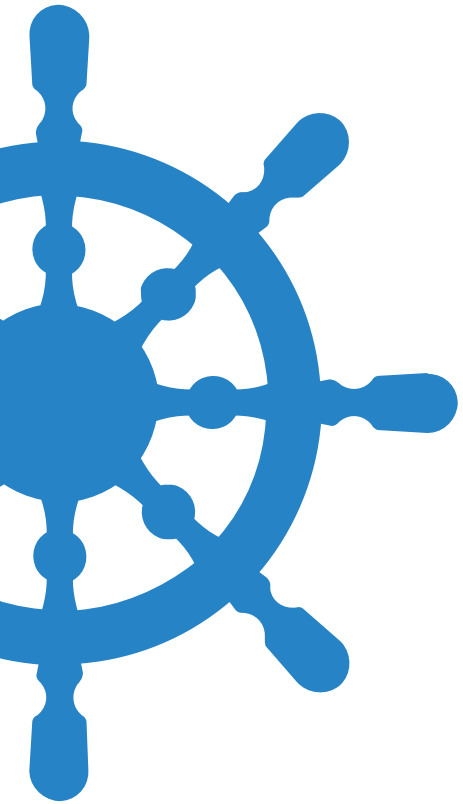
Gambaran IPO (Menentukan Bilangan)



… Gambaran IPO (Menentukan Urutan)



Sifat-Sifat Algoritma



a) Masukan: Sebelum eksekusi algoritma dimulai, masukan diperlukan sebagai input.

b) Hasil Luaran: Output dari algoritma berasal dari metode yang diterapkan sebagai solusi terhadap masalah yang dihadapi.

c) Kejelasan: Langkah-langkah dalam algoritma dinyatakan secara jelas dan tegas untuk memberikan penjelasan atau keputusan yang akurat.

d) Penyelesaian Tuntas: Algoritma harus memiliki titik akhir atau hasil keluaran yang terdefinisi untuk setiap kondisi awal atau input yang diberikan.

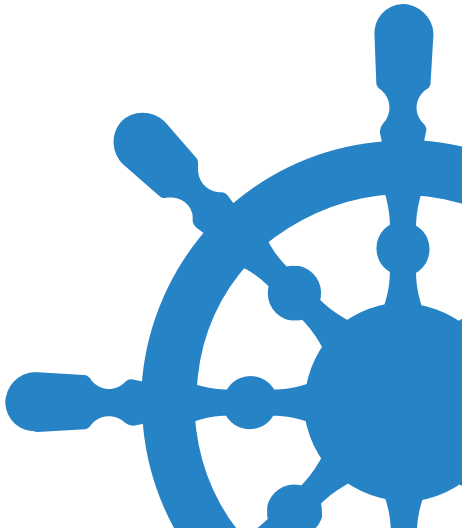
e) Efisiensi: Mencapai kesuksesan dalam menghasilkan hasil yang diinginkan dan memberikan solusi yang memuaskan.

Guru Virtual

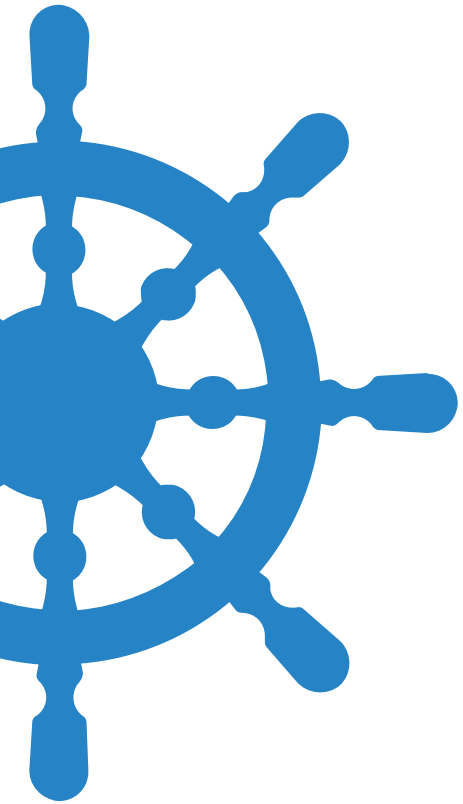
… Sifat-Sifat Algoritma

- Tidak semua urutan Langkah-langkah penyelesaian masalah yang logis atau masuk akal dapat disebut sebagai algoritma.

- Menurut Donald E. Knuth di dalam Art of Komputer Programing [KNU73], algoritma mempunyai lima ciri penting yang meliputi:
  1. Finiteness (keterbatasan)
  2. Efiniteness (kepastian)
  3. Input (masukan)
  4. Output (keluaran)
  5. Effectiveness (efektivitas)



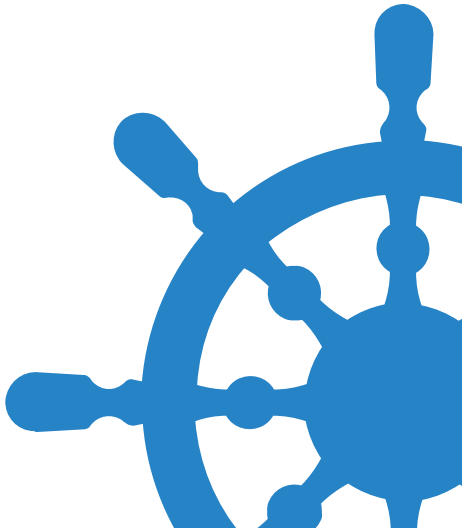
Notasi Algoritma



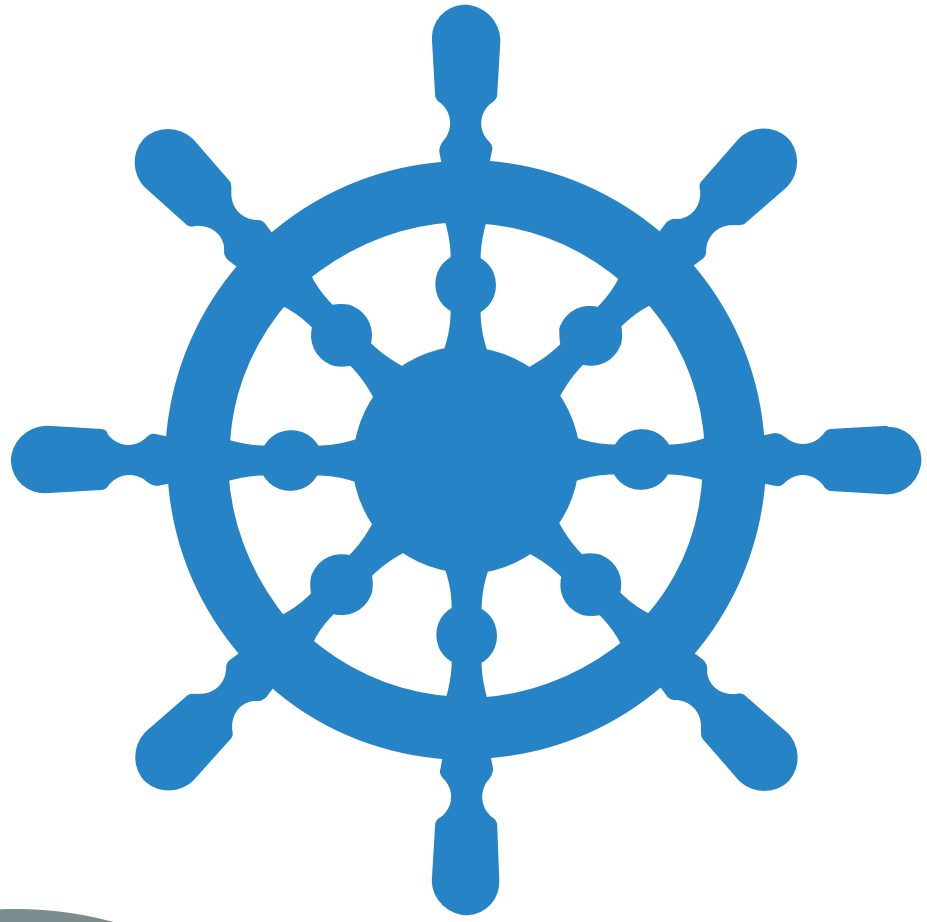
- Pemahaman tentang notasi algoritma menjadi dasar yang esensial bagi siapa pun yang ingin merancang program.
- Notasi algoritma memberikan struktur dasar yang menjadi landasan pembangunan program, menguraikan langkah-langkah yang harus diikuti.
- Penjelasan dalam notasi algoritma tidak tergantung pada tata bahasa pemrograman atau spesifikasi komputer tertentu.

… Notasi Algoritma

- Tidak ada standar pasti dalam cara menulis algoritma; yang paling penting adalah algoritma mudah dimengerti dan dibaca.
- Namun, diperlukan perhatian pada klaritas dalam penulisan algoritma untuk mencegah kesalahan interpretasi.
- Notasi algoritma membantu memastikan pengembangan program berjalan dengan lancar dan efisien.



Guru Virtual
www.guruvirtual.id

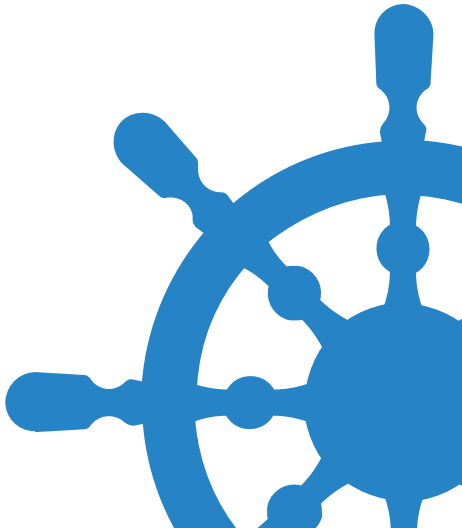


Cara Menuliskan Algoritma

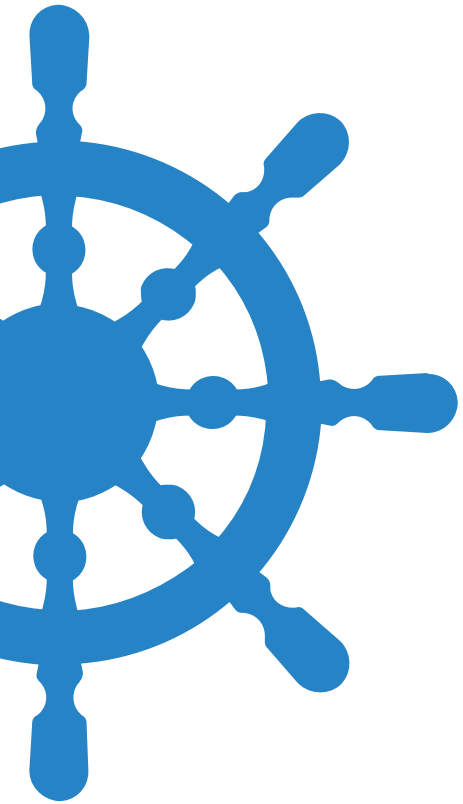
1. Deskriptif
2. Pseudecode
3. Flowchart

1. Notasi Algoritma Deskriptif

1. Notasi algoritma yang menggunakan kalimat deskriptif juga dikenal sebagai notasi alami.
2. Notasi algoritma deskriptif melibatkan penulisan langkah-langkah yang harus dijalankan dalam bentuk kalimat deskriptif yang jelas.
3. Bahasa yang digunakan dalam notasi deskriptif harus mudah dimengerti dan jelas.
4. Notasi deskriptif ini sangat cocok untuk algoritma yang relatif pendek.
5. Namun, untuk algoritma yang panjang, notasi deskriptif mungkin kurang efektif dalam memberikan kejelasan.



… Notasi Algoritma Deskriptif

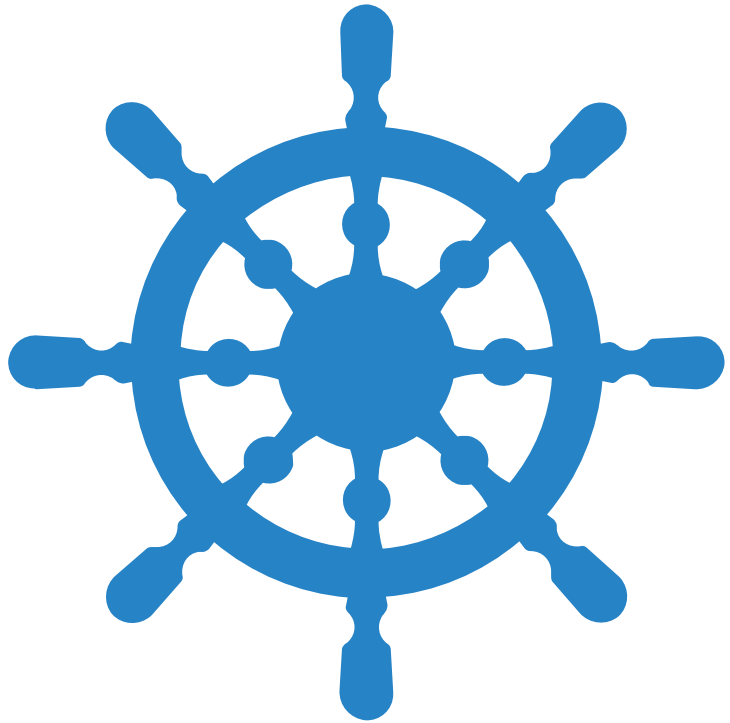


5. Notasi algoritma deskriptif adalah metode untuk menggambarkan langkah-langkah dari suatu algoritma menggunakan kalimat-kalimat deskriptif yang mudah dimengerti.
6. Ini adalah cara untuk menjelaskan urutan tindakan yang harus diambil dalam bahasa manusia tanpa menggunakan sintaks pemrograman khusus.
7. Notasi algoritma deskriptif sangat bermanfaat untuk mengkomunikasikan konsep algoritma kepada rekan kerja atau anggota tim yang mungkin tidak familiar dengan bahasa pemrograman tertentu.

Guru Virtual

Contoh Notasi Algoritma Deskriptif: Menghitung Total Dua Bilangan

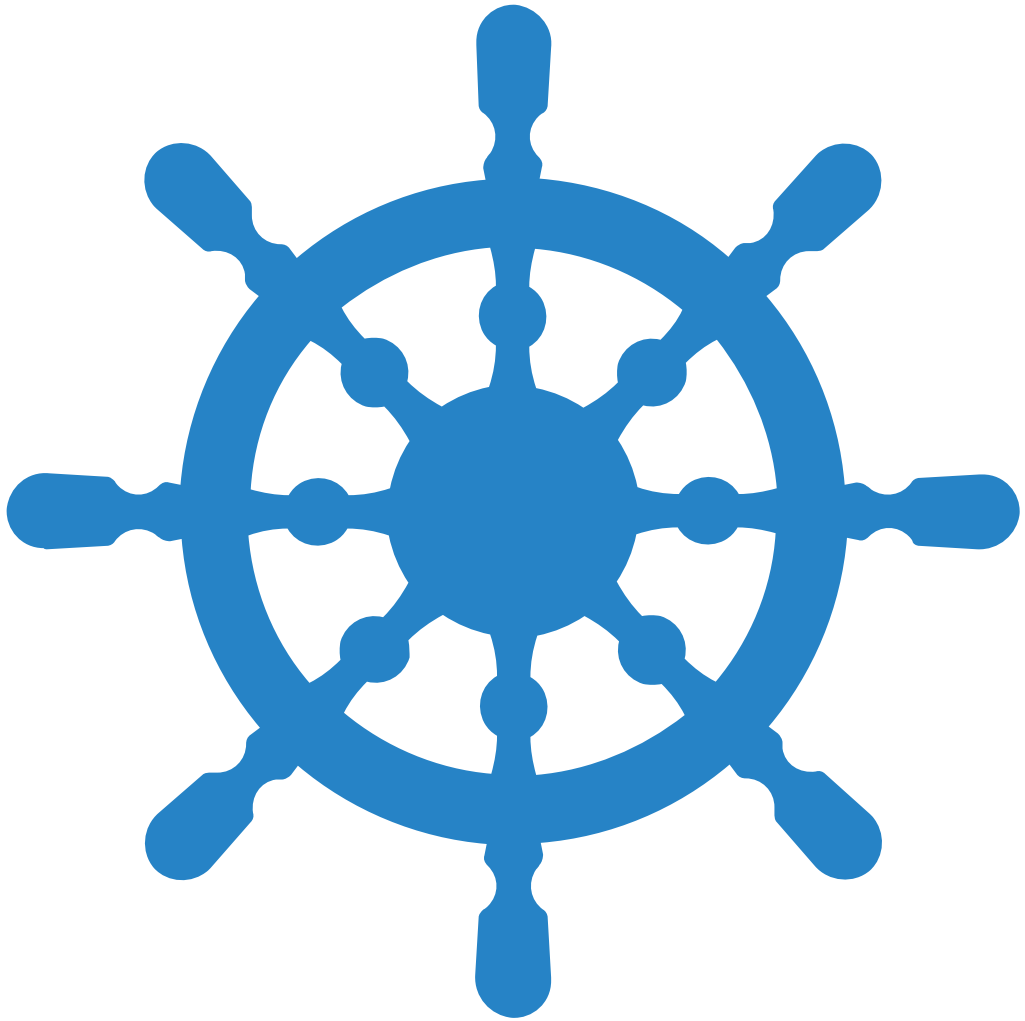
1. Meminta pengguna memasukkan dua bilangan bulat.
2. Menyimpan bilangan pertama sebagai "bilangan1" dan bilangan kedua sebagai "bilangan2".
3. Menghitung total dengan menjumlahkan "bilangan1" dan "bilangan2".
4. Menyimpan hasil penjumlahan sebagai "total".
5. Menampilkan "total" sebagai hasil akhir.



Dalam contoh ini, langkah-langkah algoritma dijelaskan secara deskriptif tanpa mengacu pada bahasa pemrograman tertentu. Ini memungkinkan seseorang untuk memahami algoritma tanpa perlu tahu tentang sintaks pemrograman.

Contoh Notasi Algoritma Deskriptif: Mengirim Surat

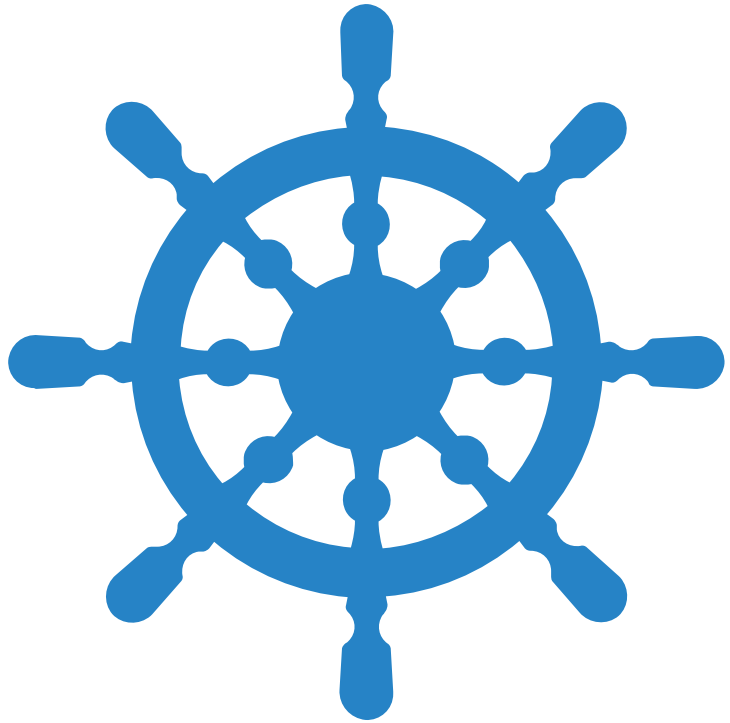
1. Siapkan kertas dan tulis alamat pengirim serta tanggal di atas surat.
2. Tulis alamat penerima dan informasi kontak dengan jelas.
3. Mulai surat dengan salam pembuka, tulis isi pesan, dan akhiri dengan penutup.
4. Tandatangani surat dan lipat dengan rapi.
5. Tulis alamat penerima di depan amplop, sertakan informasi pengirim di sudut kiri atas.
6. Letakkan surat di amplop, pastikan tutup tertutup rapat.
7. Opsional: Sertakan informasi pengirim di belakang amplop.
8. Kirim surat ke kantor pos atau tempat pengiriman, bayar biaya jika diperlukan.



Guru Virtual
www.guruvirtual.id

Keuntungan Notasi Algoritma Deskriptif

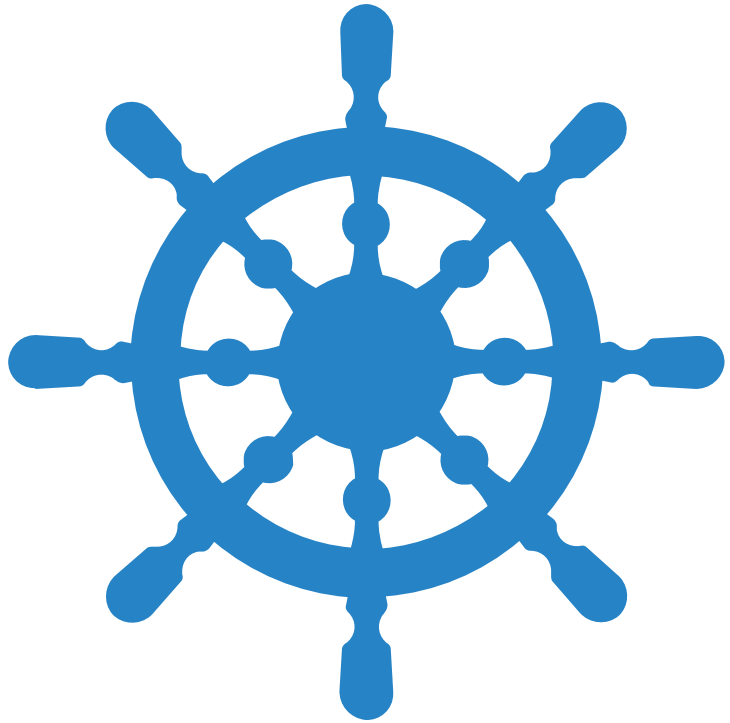
1. Mudah dipahami: Notasi deskriptif memungkinkan orang yang tidak memiliki latar belakang pemrograman untuk mengerti algoritma.
2. Klaritas: Bahasa manusia yang jelas dan sederhana meminimalkan risiko kebingungan atau salah interpretasi.
3. Komunikasi Tim: Ini memfasilitasi komunikasi antara anggota tim yang mungkin memiliki latar belakang dan pengetahuan yang berbeda.
4. Perencanaan: Notasi ini berguna dalam perencanaan sebelum memulai implementasi dalam bahasa pemrograman.



Guru Virtual
www.guruvirtual.id

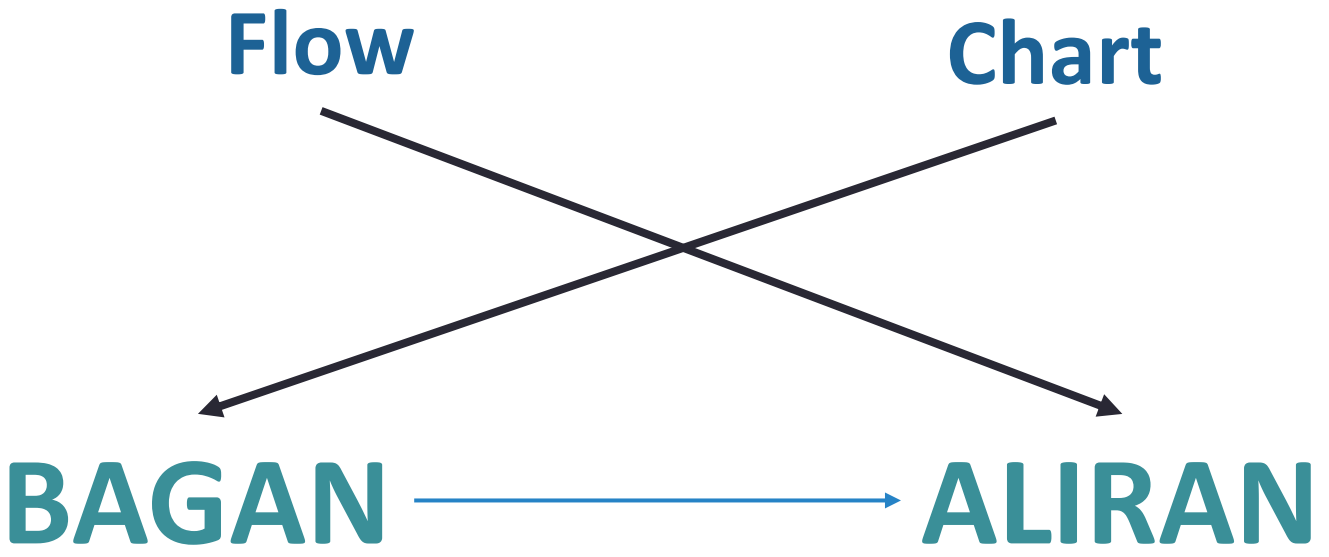
Keuntungan Notasi Algoritma Deskriptif

1. Mudah dipahami: Notasi deskriptif memungkinkan orang yang tidak memiliki latar belakang pemrograman untuk mengerti algoritma.
2. Klaritas: Bahasa manusia yang jelas dan sederhana meminimalkan risiko kebingungan atau salah interpretasi.
3. Komunikasi Tim: Ini memfasilitasi komunikasi antara anggota tim yang mungkin memiliki latar belakang dan pengetahuan yang berbeda.
4. Perencanaan: Notasi ini berguna dalam perencanaan sebelum memulai implementasi dalam bahasa pemrograman.



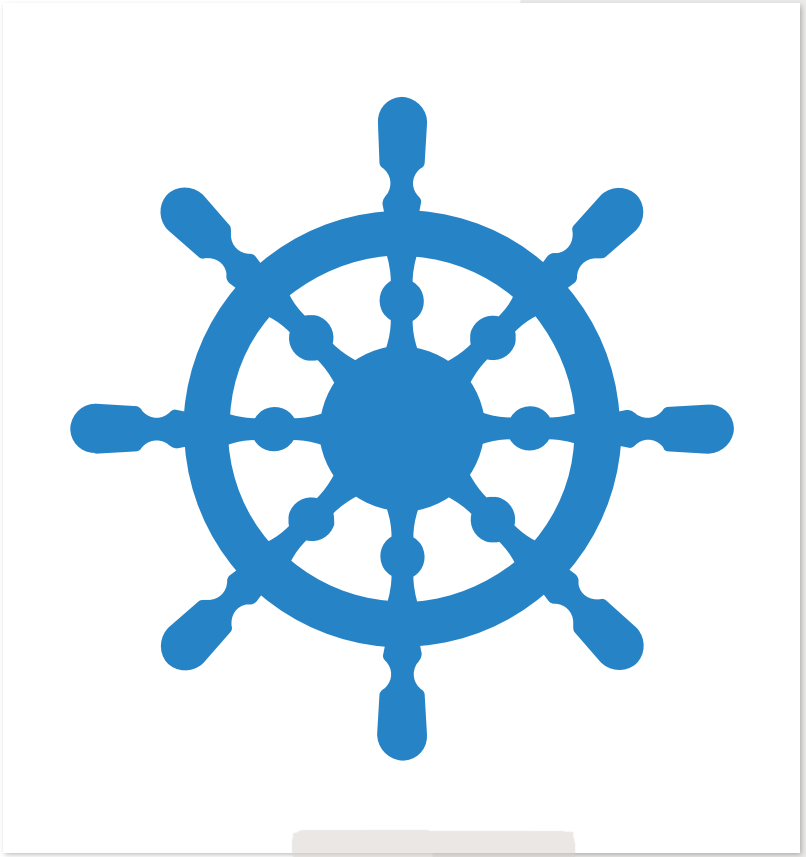
Guru Virtual

2. Notasi Algoritma Flowchart



2. Notasi Algoritma Flowchart

1. **Flowchart** adalah representasi visual atau diagram alir yang digunakan untuk menggambarkan langkah-langkah dan urutan proses suatu algoritma atau program.
2. **Flowchart** menyajikan langkah-langkah dalam bentuk simbol-simbol grafis yang saling terhubung, membantu dalam memvisualisasikan bagaimana informasi mengalir dan bagaimana proses dilakukan.
3. Dalam kaitannya dengan notasi deskriptif, notasi algoritma yang menggunakan flowchart dapat lebih cepat dibaca dan dilihat alur dan hubungannya.



... Simbol-simbol pada Flowchart

1. Setiap elemen flowchart dihubungkan oleh garis aliran bertanda panah;
2. Garis aliran dimulai dari atas symbol dan keluar dari bagian bawah, kecuali symbol keputusan yang alirannya keluar dari bawah atau samping;
3. Aliran bergerak dari atas ke bawah;
4. Proses awal dan akhir menggunakan symbol terminal.

... Simbol-simbol pada Flowchart

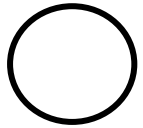
Terminator yang menandakan *Start* (awal) atau *End* (akhir) program.

Flow line yang digunakan untuk menunjukan arah aliran pada program.

Process menunjukan proses yang dilakukan pada masukan.

Input atau output untuk menunjukan masukan dan keluaran.

Preparation digunakan untuk membuat deklarasi nilai awal.

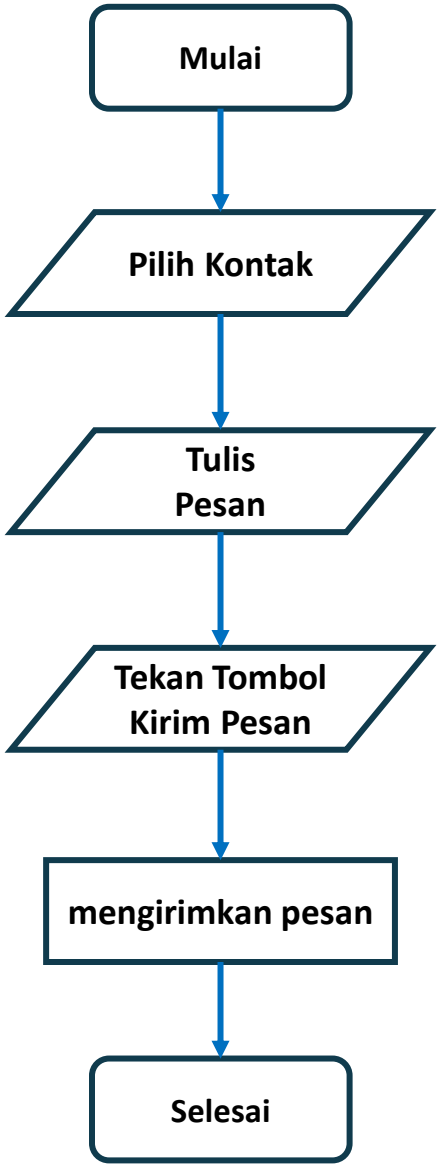


On Page Connector digunakan untuk menghubungkan antar *flowchart*

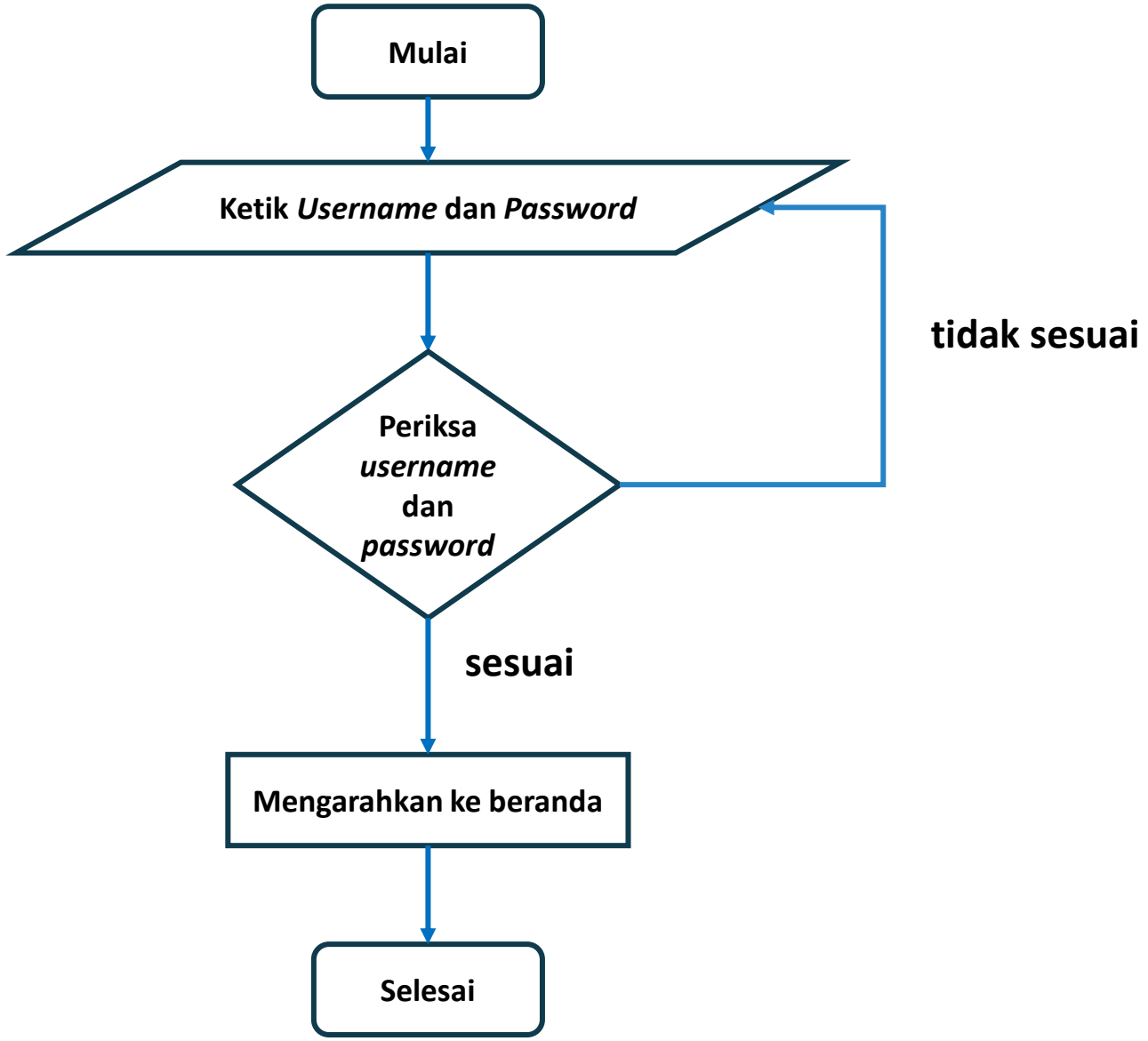
Decision menunjukan keputusan atau kondisi untuk memilih keputusan.

Contoh sederhana
Penggunaan *flowchart*
untuk menunjukan algoritma

Kasus/Aliran:
Mengirim pesan WhatsApp



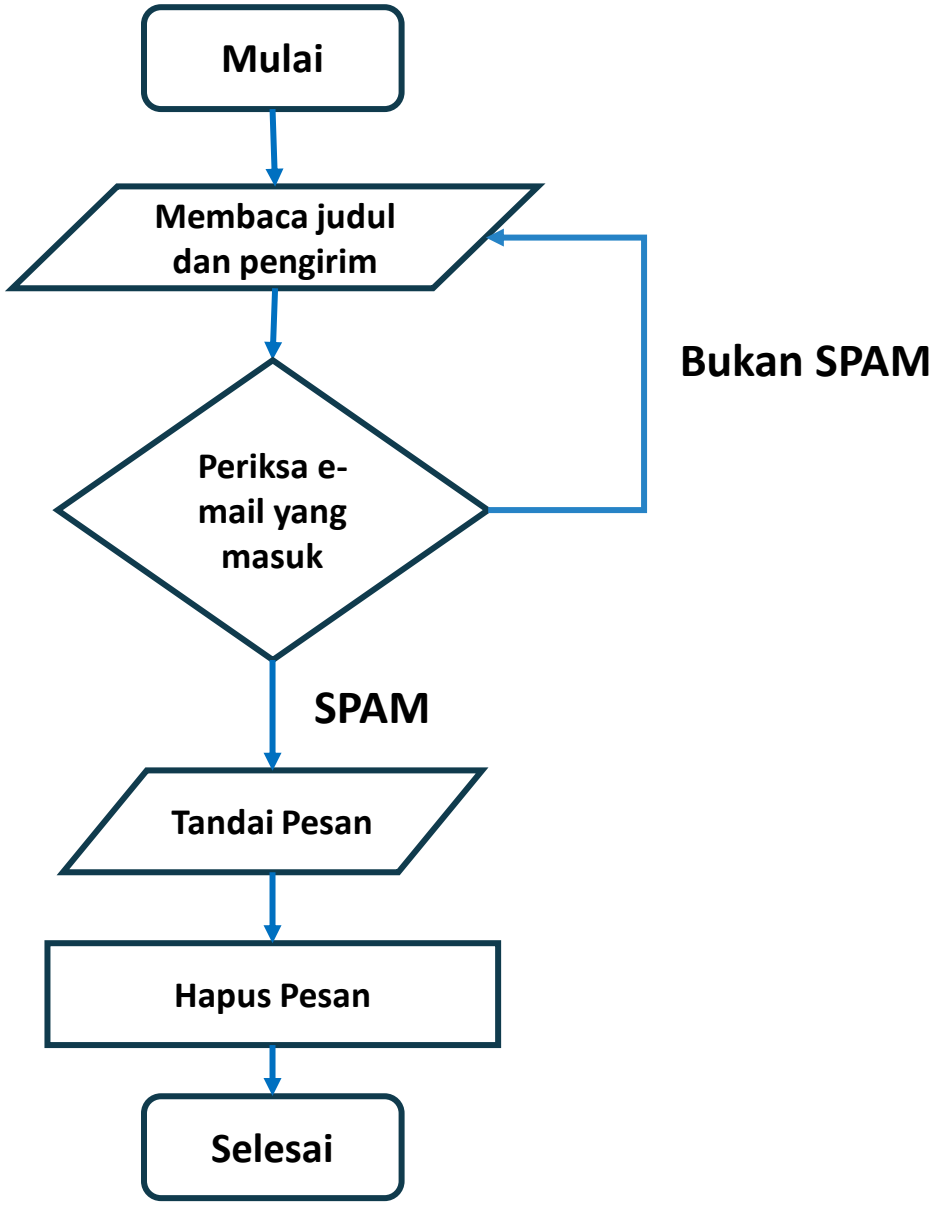
Contoh sederhana Penggunaan *flowchart* untuk menunjukan algoritma



Kasus/Aliran:
Login ke Instagram

Contoh sederhana
Penggunaan *flowchart*
untuk menunjukan algoritma

Kasus/Aliran:
Menghapus e-mail SPAM atau tidak bermanfaat



Kelebihan *Flowchart* (1—3)

1

Visualisasi yang Jelas: Flowchart menyajikan langkah-langkah algoritma atau proses secara visual, memudahkan pemahaman tentang urutan tindakan dan hubungannya.

2

Keterbacaan yang Baik: Simbol-simbol grafis dalam flowchart memudahkan pembacaan dan pemahaman algoritma, bahkan oleh orang yang tidak memiliki latar belakang pemrograman.

3

Analisis Mudah: Flowchart memungkinkan analis dan pengembang untuk dengan cepat menganalisis alur kerja dan identifikasi potensi masalah atau cacat dalam algoritma.

… Kelebihan *Flowchart* (4—6)

1

Perencanaan yang Lebih Baik: Sebelum mengimplementasikan kode, flowchart membantu dalam perencanaan langkah-langkah yang akan diambil dalam pemecahan masalah.

2

Komersialisasi dan Komunikasi: Flowchart memfasilitasi komunikasi antara anggota tim dan pemangku kepentingan dengan memberikan pandangan visual tentang bagaimana suatu program atau proses akan dijalankan.

3

Identifikasi Masalah Awal: Dengan memvisualisasikan langkah-langkah, flowchart membantu dalam mengidentifikasi potensi masalah atau kebingungan sebelum implementasi sebenarnya.

… Kelebihan *Flowchart* (7—10)

1

Pemahaman Keseluruhan: Flowchart membantu dalam melihat gambaran besar algoritma atau proses secara keseluruhan, membantu dalam memahami bagaimana bagian-bagian saling terkait.

2

Rekayasa Balik: Flowchart membantu dalam mengkaji kembali proses atau algoritma yang sudah ada dan melakukan perbaikan atau pengembangan lebih lanjut.

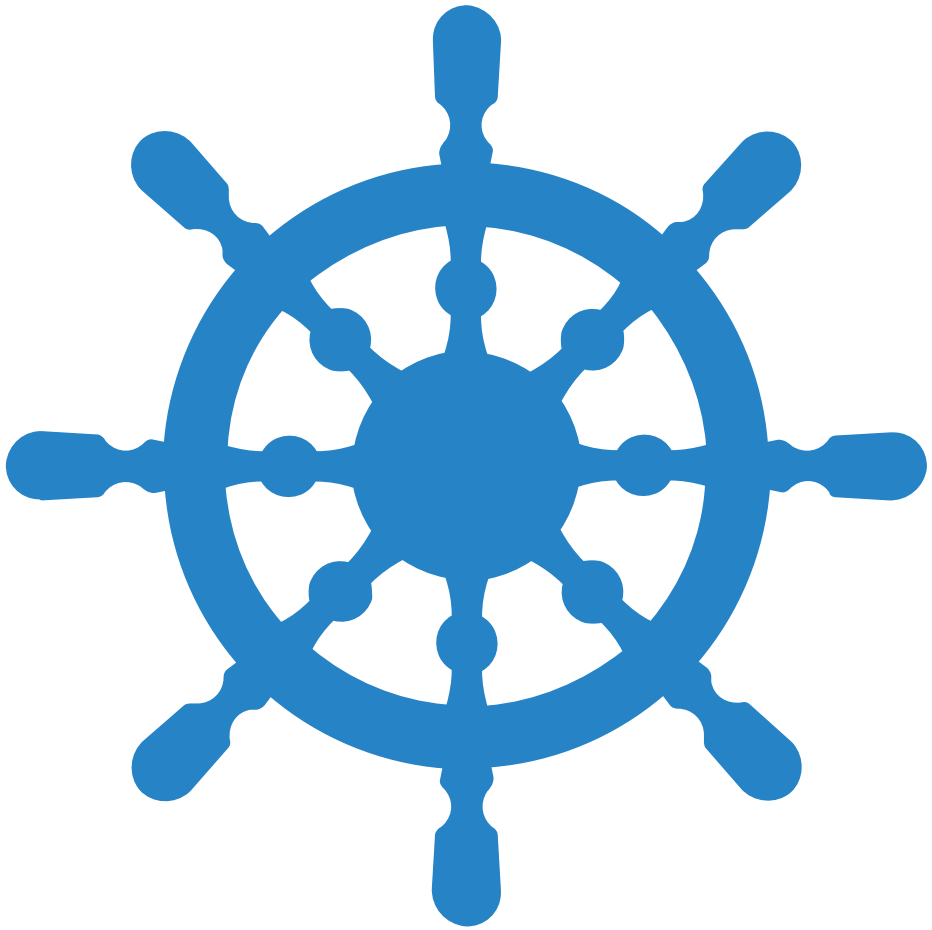
3

Dokumentasi: Flowchart berfungsi sebagai bentuk dokumentasi visual yang memudahkan dalam pemeliharaan, pengembangan, atau perbaikan di masa depan.

4

Pendidikan dan Pembelajaran: Flowchart juga merupakan alat pembelajaran yang baik dalam mengajarkan konsep pemrograman dan pemecahan masalah kepada pemula.

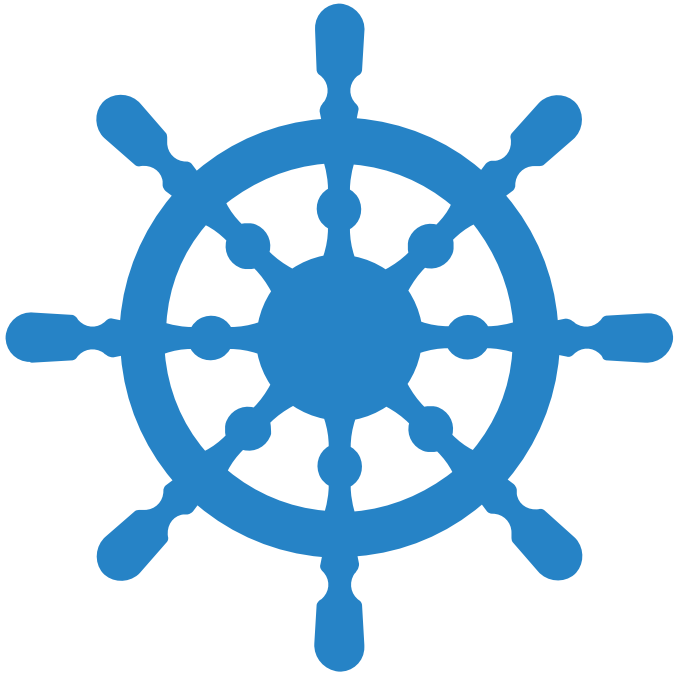
3. Pseudo Code



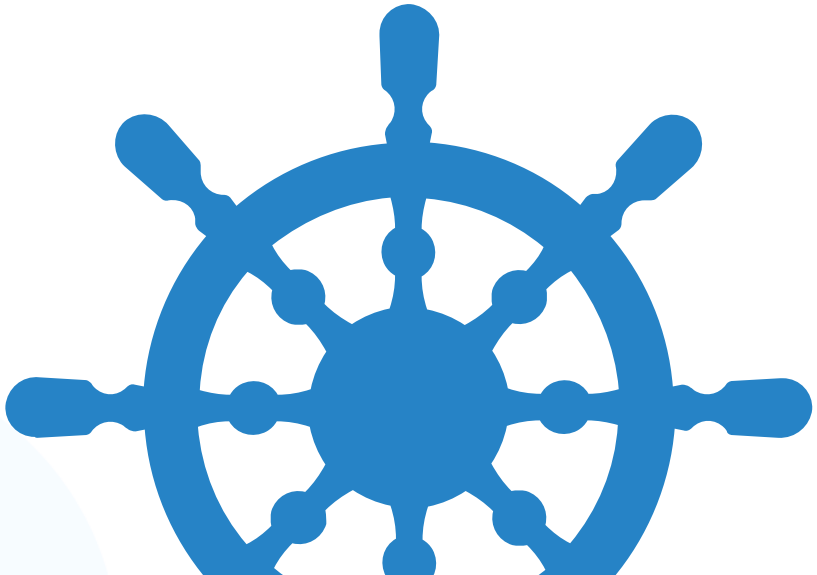
- Pseudocode adalah suatu bentuk deskripsi informal yang mirip dengan bahasa manusia dan digunakan untuk menggambarkan algoritma atau proses secara naratif.
- Ini tidak terikat pada bahasa pemrograman tertentu, tetapi memberikan panduan tentang langkah-langkah yang harus diambil dalam suatu algoritma dengan bahasa yang lebih mudah dimengerti.

… Penggunaan Pseudo Code

1. **Klarifikasi Konsep**: Pseudocode membantu dalam merencanakan langkah-langkah algoritma sebelum menulis kode sebenarnya. Ini membantu dalam menguraikan ide secara jelas.
2. **Kolaborasi Tim**: Pseudocode membantu dalam berkomunikasi antara anggota tim yang mungkin memiliki latar belakang bahasa pemrograman yang berbeda.
3. **Pemecahan Masalah**: Ini memungkinkan pemecahan masalah yang lebih baik sebelum implementasi sebenarnya dimulai.
4. **Penyusunan Logika**: Pseudocode membantu dalam merancang alur kerja dan pengambilan keputusan dalam algoritma.



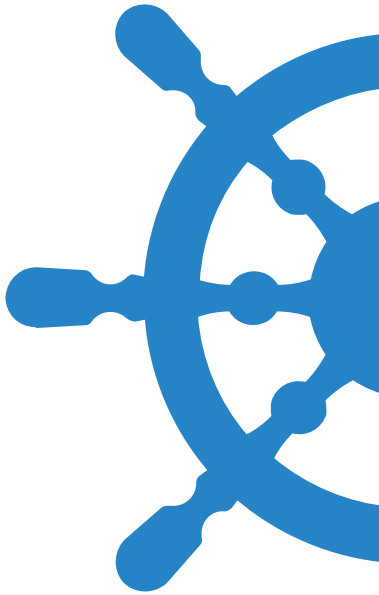
… Penggunaan PseudoCode



5. **Pendidikan**: Ini adalah alat yang baik untuk mengajarkan konsep pemrograman dan pemecahan masalah kepada pemula.
6. **Rekayasa Balik**: Pseudocode digunakan untuk menganalisis kode yang sudah ada dan mengidentifikasi area yang membutuhkan perbaikan atau perubahan.
7. **Desain Algoritma**: Pseudocode membantu dalam mendefinisikan langkah-langkah algoritma dengan bahasa manusia yang lebih akrab.
8. **Dokumentasi**: Pseudocode dapat digunakan sebagai bentuk dokumentasi sederhana yang memberikan gambaran tentang cara kerja algoritma.

Sintaks PseudoCode

5. Pseudocode adalah bentuk notasi pemrograman bahasa tinggi yang digunakan untuk merencanakan dan menggambarkan algoritma tanpa terikat pada sintaks bahasa pemrograman tertentu.
6. Ini lebih fokus pada mengkomunikasikan langkah-langkah logis algoritma dengan cara yang lebih intuitif daripada bahasa pemrograman sebenarnya.



```
class Lesson(BaseModel):
    STATUS_PUBLISHED = 'published'
    STATUS_DRAFT = 'draft'
    CHOICE_STATUS = [
        (STATUS_PUBLISHED, _('Published')),
        (STATUS_DRAFT, _('Draft')),
    ]
    PLATFORM_YOUTUBE = 'youtube'
    CHOICE_VIDEO_PLATFORM = [
        (PLATFORM_YOUTUBE, _('Youtube')),
    ]
    validator = Validators(max_size=(775), filetype='image')
    room = models.ForeignKey(Room, on_delete=models.CASCADE, related_name='lessons', verbose_name=_("Room"))
    title = models.CharField(_("Title"), max_length=256)
    keywords = models.CharField(_("Keywords"), max_length=256, null=True, blank=True)
    subtitle = models.CharField(_("Subtitle"), null=True, blank=True, max_length=256)
    description = RichTextField(null=True, blank=True, verbose_name=_("Description"))
```

Guru Virtual

Sintaks PseudoCode

5. Beberapa elemen seperti **if then else, while do, repeat until, read**, dan **write** memang merupakan konsep umum dalam pemrograman, dan biasanya dapat ditemukan dalam pseudocode karena mewakili struktur dasar dalam pemrograman, tetapi bukan aturan yang ketat.
6. Pseudocode dapat disesuaikan dengan preferensi penulis atau perancang algoritma.
7. Dalam banyak kasus, pseudocode dapat mengandung elemen-elemen yang mirip dengan bahasa pemrograman tertentu, seperti konstruksi kontrol dan operasi input/output, tetapi tidak ada standar yang ketat atau "syntax" resmi untuk pseudocode. Tujuannya adalah memfasilitasi pemahaman algoritma, bukan mengikuti peraturan sintaksis tertentu seperti dalam bahasa pemrograman nyata.

```
class Lesson(BaseModel):
    STATUS_PUBLISHED = 'published'
    STATUS_DRAFT = 'draft'
    CHOICE_STATUS = [
        (STATUS_PUBLISHED, _('Published')),
        (STATUS_DRAFT, _('Draft')),
    ]
    PLATFORM_YOUTUBE = 'youtube'
    CHOICE_VIDEO_PLATFORM = [
        (PLATFORM_YOUTUBE, _('Youtube')),
    ]
    validator = Validators(max_size=(775), filetype='image')
    room = models.ForeignKey(Room, on_delete=models.CASCADE,
    title = models.CharField(_("Title"), max_length=256)
    keywords = models.CharField(_("Keywords"), max_length=25
    subtitle = models.CharField(_("Subtitle"), null=True, bl
    description = RichTextField(null=True, blank=True, verbo
```

Guru Virtual

Contoh PseudoCode

Inisiasi Variabel:

```
N       = 0
total   = 0.0
```

Pengulangan:

```
UNTUK i DARI 1 SAMPAI 10 LANGKAH 2
   CETAK i
END UNTUK
```

Pengkondisional (Conditional):

```
JIKA nilai > 10
   CETAK "Nilai lebih dari 10"
SELAINNYA JIKA nilai = 10
   CETAK "Nilai sama dengan 10"
SELAINNYA
   CETAK "Nilai kurang dari 10"
AKHIR JIKA
```

Fungsi atau Prosedur:

```
FUNGSI tambah(a, b)
   KEMBALIKAN a + b
AKHIR FUNGSI
```

Contoh Lengkap:

```
DEKLARASI variabel n, bilangan, total, rata_rata FLOAT
MINTA "Masukkan jumlah bilangan: " SIMPAN
total = 0.0

UNTUK i DARI 1 SAMPAI n
   MINTA "Masukkan bilangan ke-" + i + ": " SIMPAN bilangan
   total = total + bilangan
END UNTUK

rata_rata = total / n
CETAK "Rata-rata adalah: " + rata_rata
```

Guru Virtual
www.guruvirtual.id

Contoh PseudoCode Menghitung Persegi Panjang

```
DEKLARASI variabel panjang, lebar, luas, keliling INTEGER

MINTA "Masukkan panjang persegi panjang: " SIMPAN panjang
MINTA "Masukkan lebar persegi panjang: " SIMPAN lebar

luas = panjang * lebar
keliling = 2 * (panjang + lebar)

CETAK "Luas persegi panjang: " + luas
CETAK "Keliling persegi panjang: " + keliling
```

Latihan Praktik

Soal 1: Menghitung Nilai Rata-rata
Tuliskan algoritma untuk menghitung rata-rata dari sepuluh bilangan yang dimasukkan oleh pengguna:

1) Gambarlah flowchart yang menjelaskan langkah-langkah algoritma ini
2) Tuliskan pseudocode untuk algoritma tersebut.

Soal 2: Menghitung Total Belanja
Buat algoritma yang menghitung total belanja berdasarkan harga dan jumlah barang yang dibeli.

1) Gambarkan flowchart yang menunjukkan langkah-langkah menghitung total belanja.
2) Tuliskan pseudocode yang menggambarkan algoritma perhitungan total belanja.

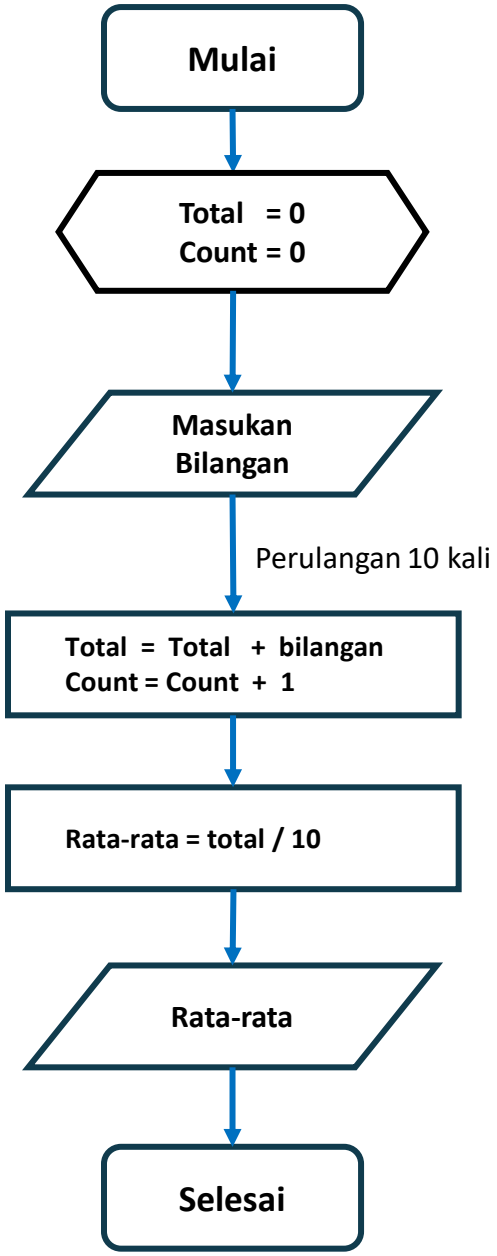
Soal 3: Mencari Bilangan Terbesar
Buat algoritma yang membantu menemukan bilangan terbesar dari tiga bilangan yang dimasukkan.

1) Gambarkan flowchart yang menunjukkan langkah-langkah dalam mencari bilangan terbesar.
2) Tuliskan pseudocode untuk algoritma penentuan bilangan terbesar.

Guru Virtual
www.guruvirtual.id

Jawaban Nomor 1

Flowchart



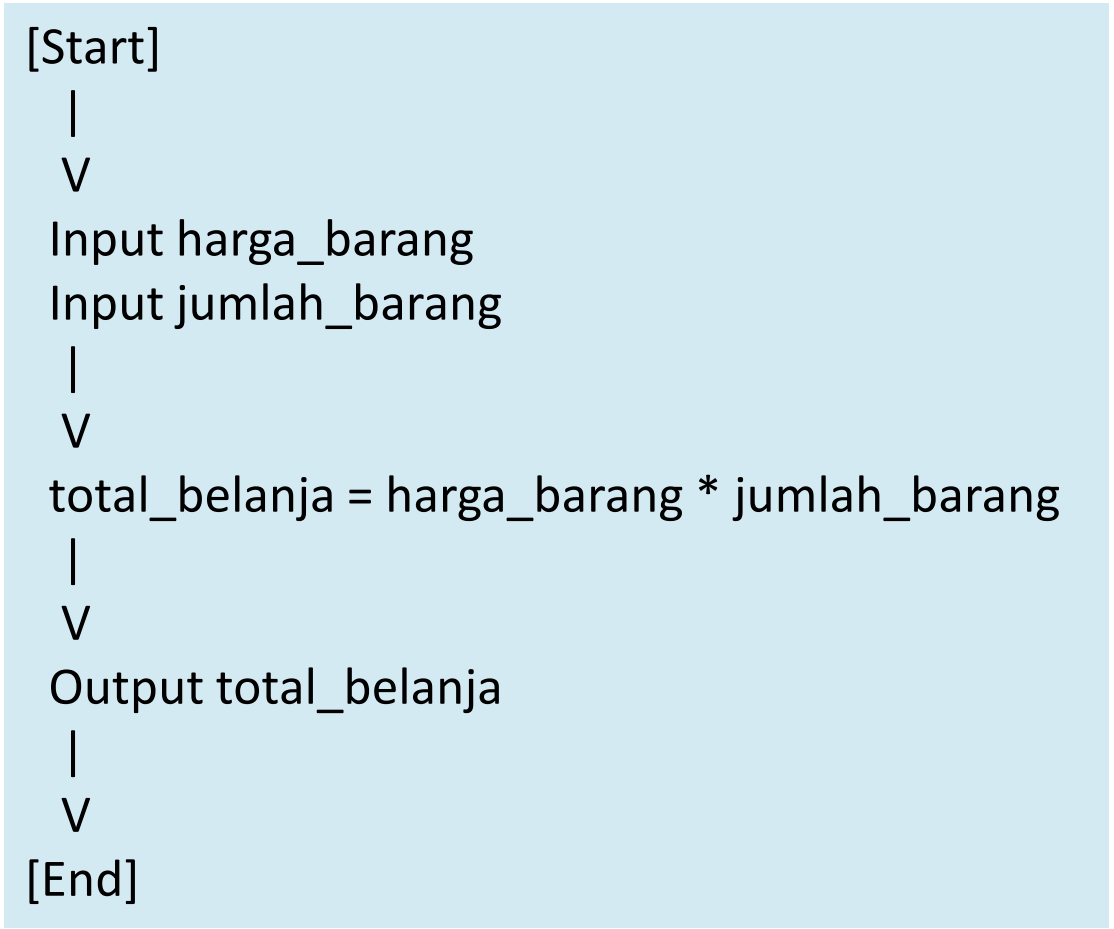
Jawaban Nomor 1

Pseudo-Code

```
DEKLARASI variabel total, count, bilangan, rata_rata INTEGER

total   = 0
count = 0

UNTUK i DARI 1 SAMPAI 10
   MINTA "Masukkan bilangan ke-" + i + ": " SIMPAN bilangan
   total   = total + bilangan
   count = count + 1
END UNTUK

rata_rata = total / 10
CETAK "Rata-rata adalah: " + rata_rata
```

Jawaban Nomor 2

Flowchart



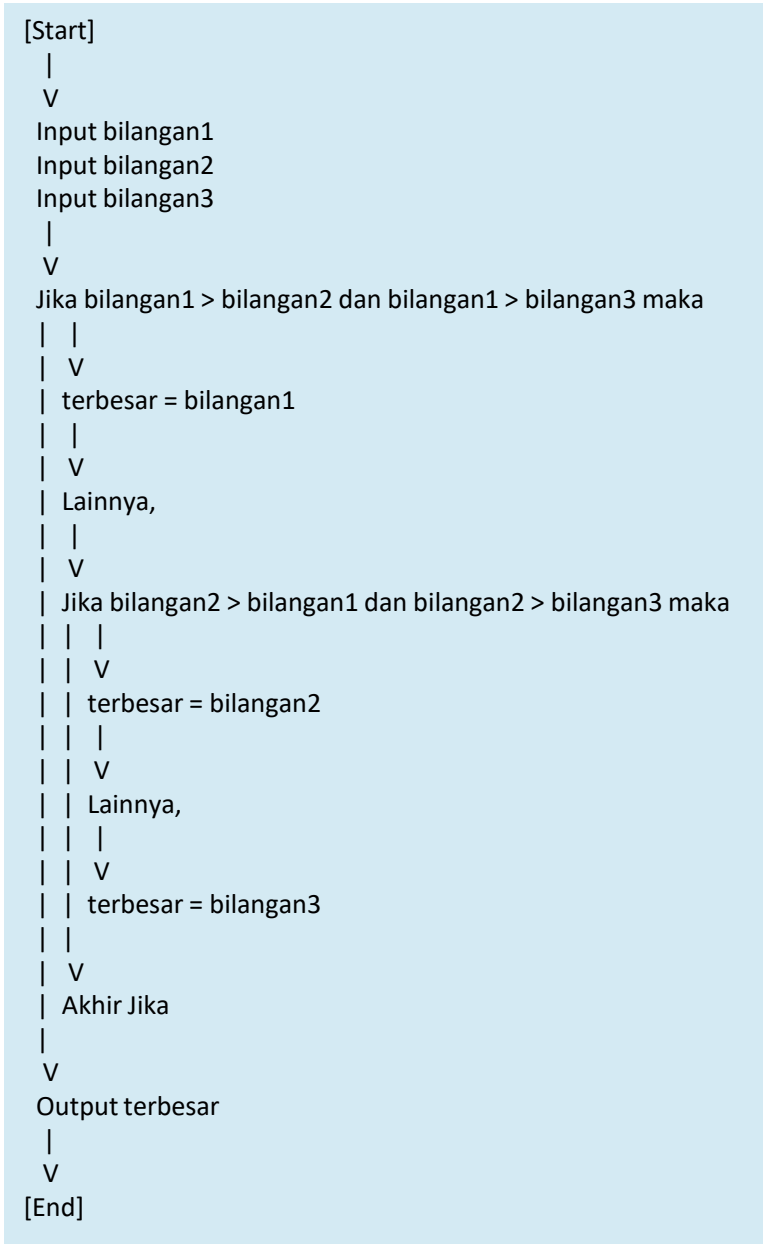
Jawaban Nomor 2

Pseudo-Code

```
DEKLARASI variabel harga_barang, jumlah_barang,
total_belanja INTEGER

MINTA "Masukkan harga barang: " SIMPAN harga_barang
MINTA "Masukkan jumlah barang: " SIMPAN jumlah_barang

total_belanja = harga_barang * jumlah_barang
CETAK "Total belanja: " + total_belanja
```

Jawaban Nomor 3

Flowchart



Jawaban Nomor 3

Pseudo-Code

```
DEKLARASI variabel bilangan1, bilangan2, bilangan3, terbesar INTEGER

MINTA "Masukkan bilangan pertama: " SIMPAN bilangan1
MINTA "Masukkan bilangan kedua: " SIMPAN bilangan2
MINTA "Masukkan bilangan ketiga: " SIMPAN bilangan3

JIKA bilangan1 > bilangan2 DAN bilangan1 > bilangan3 MAKA
   terbesar = bilangan1
LAINNYA, JIKA bilangan2 > bilangan1 DAN bilangan2 > bilangan3 MAKA
   terbesar = bilangan2
LAINNYA
   terbesar = bilangan3
AKHIR JIKA

CETAK "Bilangan terbesar adalah: " + terbesar
```

Guru Virtual
www.guruvirtual.id