Membuat Algoritma

Yang Baik

Kaidah Membuat Algoritma Yang Benar

Langkah	Penjelasan
1. Definisikan masalah	Pahami apa yang ingin diselesaikan (misal: menghitung rata-rata, mencari nilai maksimum, dsb).
2. Tentukan input	Data apa yang dibutuhkan dari pengguna atau sistem.
3. Tentukan proses	Langkah-langkah logis untuk mengubah input menjadi output.
4. Tentukan output	Hasil akhir yang ingin ditampilkan.
5. Gunakan urutan logis dan jelas	Setiap langkah harus bisa diikuti tanpa ambigu.
6. Hindari asumsi tidak jelas	Jangan melompati langkah atau menulis hal yang tidak bisa diimplementasikan.

Contoh Algortma BenAR

- Ada struktur: deklarasi → deskripsi
- Urutan langkah logis
- Ada input, proses, output
- Bisa langsung diterjemahkan ke kode program
- Independent terhadap Bahasa Pemrograman

```
ALGORITMA Cek_GanjilGenap
DEKLARASI
    bilangan : integer

DESKRIPSI
    1. Tampilkan "Masukkan sebuah bilangan: "
    2. Baca bilangan
    3. Jika bilangan mod 2 = 0 maka
           Tampilkan "Bilangan ini GENAP"
       Jika tidak maka
           Tampilkan "Bilangan ini GANJIL"
    4. Selesai
```

```
ALGORITMA Hitung_RataRata
DEKLARASI
    nilai1, nilai2, nilai3 : integer
    rataRata : real

DESKRIPSI
    1. Tampilkan "Masukkan nilai pertama: "
    2. Baca nilai1
    3. Tampilkan "Masukkan nilai kedua: "
    4. Baca nilai2
    5. Tampilkan "Masukkan nilai ketiga: "
    6. Baca nilai3
    7. rataRata ← (nilai1 + nilai2 + nilai3) / 3
    8. Tampilkan "Nilai rata-rata adalah: ", rataRata
SELESAI
```

Contoh Algoritma Tidak Benar

```
Masukkan angka
Kalau genap tulis genap
Kalau ganjil tulis ganjil
```

Kesalahan:
- Tidak ada logika jelas bagaimana “genap” ditentukan.
- Tidak disebutkan operasi mod 2.
- Tidak ada struktur keputusan formal (“jika… maka…”).

```
ALGORITMA RataRata
1. Masukkan nilai
2. Hitung rata-rata
3. Cetak hasil
```

Jenis Kesalahan	Penjelasan
Terlalu umum	Tidak dijelaskan nilai apa yang dimasukkan dan bagaimana menghitung rata-rata.
Tidak ada variabel	Tidak disebutkan nama variabel yang digunakan untuk menyimpan data.
Langkah tidak operasional	“Hitung rata-rata” tidak dijelaskan caranya.
Tidak dapat diimplementasikan langsung	Programmer tidak bisa menerjemahkan langsung ke kode nyata.

Pseudo Code

3. Pseudo-code (1)

- Pseudo-code menggunakan **bahasa yang hampir menyerupai bahasa pemrograman**.
- Pseudocode adalah notasi yang menyerupai bahasa pemrograman tingkat tinggi.
- Selain itu biasanya pseudo-code menggunakan bahasa yang mudah dipahami secara universal dan juga **lebih ringkas** dari pada algoritma.
- Dalam pseudocode, tidak ada syntax standar yang resmi.
- Karena itu, pseudocode ini dapat kita terapkan dalam berbagai bahasa pemograman.

3. Pseudo-code (2)

- Disarankan untuk menggunakan keyword yang umum digunakan seperti : if, then, else, while, do, repeat, for, dan lainnya
- Keuntungan menggunakan notasi pseudo code adalah kemudahan mengkonversinya lebih tepat yang disebut mentranslasi ke notasi bahasa pemrograman, karena terdapat korespondensi antara setiap pseudo code dengan notasi bahasa pemrograman.

Fortran style pseudo code

```
program bizzbuzz
do i = 1 to 100
    set print_number to true
    if i is divisible by 3
        print "Bizz"
        set print_number to false
    if i is divisible by 5
        print "Buzz"
        set print_number to false
    if print_number, print i
    print a newline
end do
```

Pascal style pseudo code

```
procedure bizzbuzz
for i := 1 to 100 do
    set print_number to true;
    if i is divisible by 3 then
        print "Bizz";
        set print_number to false;
    if i is divisible by 5 then
        print "Buzz";
        set print_number to false;
    if print_number, print i;
    print a newline;
end
```

C style pseudo code:

```
void function bizzbuzz
for (i = 1; i<=100; i++) {
    set print_number to true;
    if i is divisible by 3
        print "Bizz";
        set print_number to false;
    if i is divisible by 5
        print "Buzz";
        set print_number to false;
    if print_number, print i;
    print a newline;
}
```

```
FUNCTION Maksimum(A, n)
    IF n = 1 THEN
        RETURN A[1]
    ELSE
        temp ← Maksimum(A, n - 1)
        IF temp > A[n] THEN
            RETURN temp
        ELSE
            RETURN A[n]
END FUNCTION
```

TUGAS

Menjumlah angka bilangan

- Buatlah flowchart untuk menjumlah angka-angka dari suatu bilangan

- Contoh:
  - Bilangan = 3628
  - Hasil = 3+6+2+8 = 19

Prima

Menentukan suatu bilangan prima atau bukan

Contoh:

Masukkan bilangan : 13
Bilangan prima

Masukkan bilangan : 33
Bukan bilangan prima

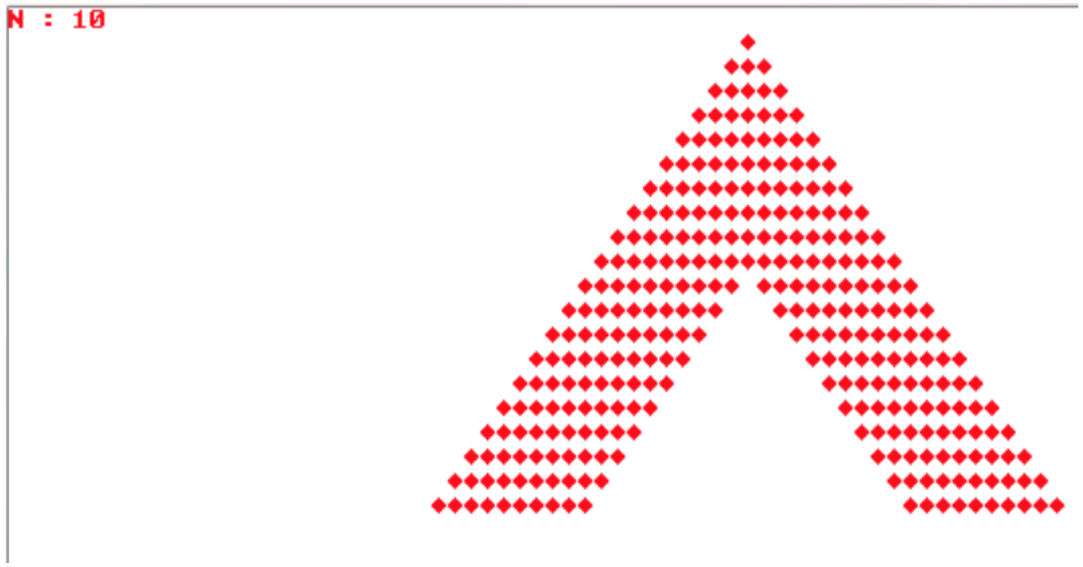
Recursive

- Selesaikan Pangkat (x^n) dengan recursive
  - $2^3 = 2 \times 2 \times 2 = 8$
- Selesaikan Penjumlahan Deret 1 sampai n dengan recursive
  - Deskripsi:
    Hitung total dari 1 + 2 + 3 + … + n
- Selesaikan Membalik String dengan recursive
  - Deskripsi:
    Balikkan urutan huruf dalam string.
    Contoh: “KITA” → “ATIK”

- Bantu membuat penjelasannya yang memudahkan semua teman mudah memahami

Buat Pseudo Code

Buat Pseudo Code



Membalik Bilangan

Menampilkan bilangan dalam urutan terbalik.

Contoh:

Masukkan bilangan : 123

Hasil : 321