

PERTEMUAN 6

MODULARISASI & KOMUNIKASI ANTAR MODUL

POKOK BAHASAN

1. Konsep Pemrograman Modular
2. Komunikasi antar modul
3. Kohesi
4. Kopling

MODULARISASI

- Modularisasi digunakan bila ada suatu permasalahan yang kompleks, sehingga langkah pertama adalah mengidentifikasi tugas utama, setelah itu baru di bagi kedalam tugas yang lebih rinci.
- Proses ini disebut juga dengan Top Down Design

PEMROGRAMAN MODULAR

- Memecahkan algoritma ke dalam algoritma yang lebih kecil/modul.
- Modul yang dibentuk mempunyai kesatuan tugas/fungsi maupun kesatuan proses/prosedur.
- Setiap modul harus mempunyai single entry dan single exit secara beruntun dari atas ke bawah atau dari awal ke akhir modul.
- Memiliki main program dan sub program atau modul

PEMROGRAMAN MODULAR (lanjutan)

Enam langkah dalam modular:

1. Definisi masalah: klasifikasikan dalam input, proses dan output
2. Kelompokkan aktivitas ke dalam modul. Definisikan kegiatan dari modul-modul yang ada
3. Buat bagan susun untuk menjelaskan hirarki dan hubungan antar modul
4. Buat logika dari main program dengan pseudocode. Terlebih dahulu inventarisasi apa saja yang dikerjakan dalam main program
5. Buat logika untuk tiap tiap modul dengan pseudocode
6. Desk checking algoritma: mencek kebenaran algoritma dengan data

JANGKAUAN DATA

- Global Data adalah variabel yang dikenal diseluruh program tersebut, dan dapat diakses dari setiap modul di program tersebut.
- Local Data adalah variabel yang didefinisikan disebuah modul. Variabel ini hanya dikenal di modul dimana variabel tersebut didefinisikan.

SIDE EFFECT

Side effect adalah sebuah bentuk komunikasi antar modul dengan bagian lain dalam program.

- Global Data (Data Global)

Perubahan nilai global data berdampak terhadap nilai data tersebut di semua modul.

- Local Data (Data Lokal)

Perubahan nilai local data hanya berdampak terhadap nilai data pada modul secara lokal.

CONTOH PEMROGRAMAN MODULAR

Program Hitung_luasPP

{menentukan luas persegi panjang berdasarkan data yang diinput}

Deklarasi

integer p, l {global data}

Deskripsi

Baca p

Baca L

Hitung luaspp(p,l)

END

Sub luaspp(integer pj, integer lb)

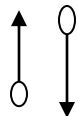
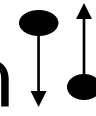
integer luas {local data}

luas = pj * lb

Cetak luas

EndSub

PARAMETER

- Parameter Data 
- Parameter Status flag/boolean 
- Dalam merancang modul sebaiknya lebih banyak menggunakan parameter data
- Hindari menggunakan parameter status sebanyak mungkin

PARAMETER PASSING

- Menyampaikan data dari modul pemanggil ke modul yang dipanggil (subordinate).
- Menyampaikan informasi dari subordinate ke modul pemanggil.
- Informasi/data yang dikirim atau diterima di passing 2 arah dari modul pemanggil ke subordinate maupun sebaliknya.



KOMUNIKASI ANTAR MODUL

- Parameter Aktual

Parameter yang disertakan pada saat prosedur dipanggil untuk dilaksanakan.

Contoh : tukar (a,b); //a dan b adalah parameter aktual

- Parameter Formal

Parameter yang dituliskan pada definisi suatu prosedur atau fungsi.

Contoh : Prosedur tukar(x, y);

KOMUNIKASI ANTAR MODUL (lanjutan)

- Pemanggilan Dengan Nilai (Call By Value)

pemanggilan dengan nilai, nilai dari parameter aktual akan ditulis ke parameter formal. Dengan cara ini nilai parameter aktual tidak bisa berubah, walaupun nilai parameter formal berubah.

- Pemanggilan Dengan Acuan

Pemanggilan dengan reference merupakan upaya untuk melewati alamat dari suatu variabel kedalam fungsi. Cara ini dapat dipakai untuk mengubah isi suatu variabel diluar fungsi dengan melaksanakan pengubahan dilakukan didalam fungsi.

CONTOH KASUS MODULARISASI

Susunlah algoritma untuk mengurutkan beberapa bilangan dengan urutan menaik (ascending) dan tampilkan bilangan hasil sort tersebut

CONTOH KASUS MODULARISASI (lanjutan)

A. Definisi Masalah

Input : banyaknya data, bilangan yang akan diurutkan

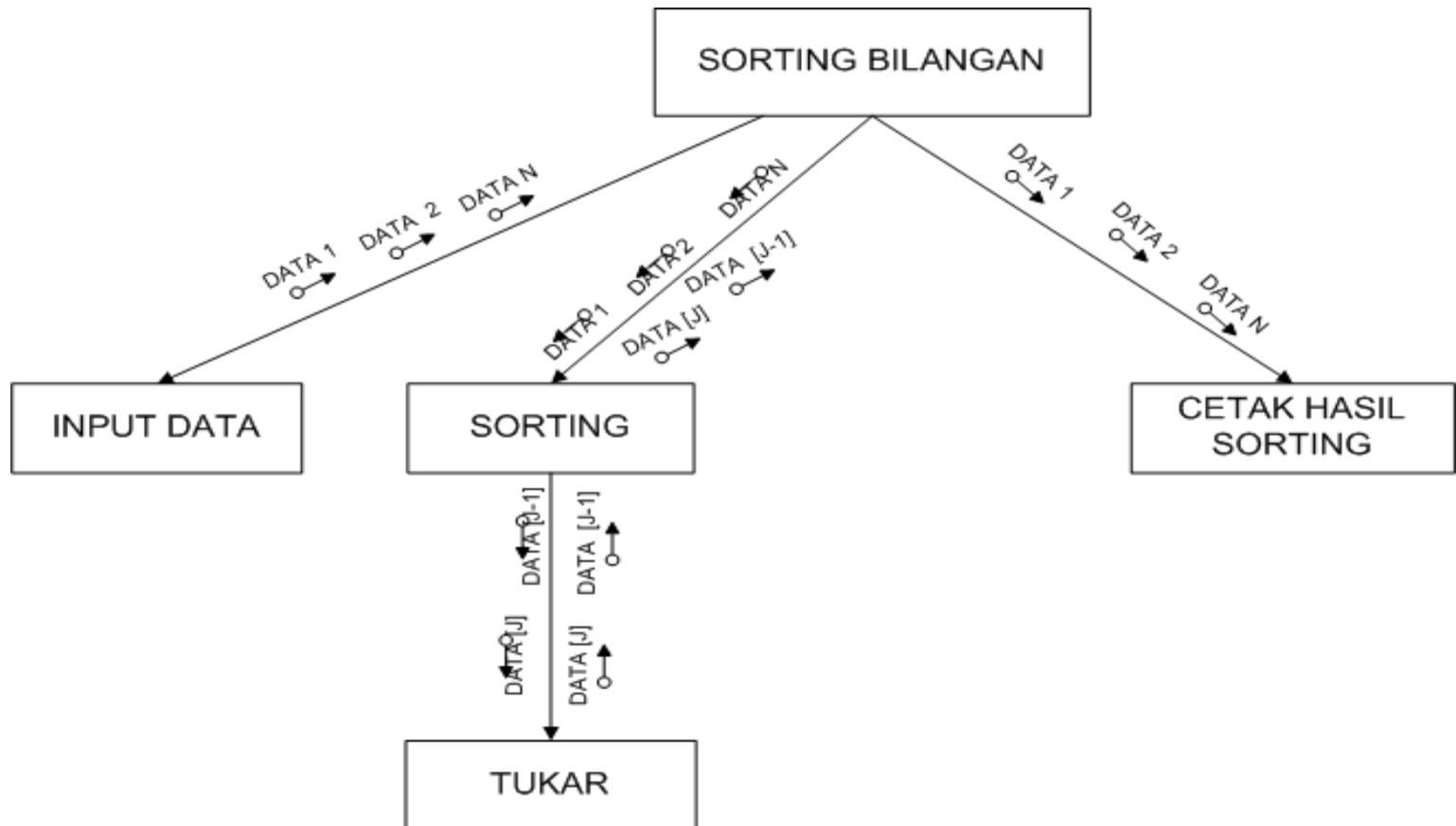
Output : Bilangan yang terurut

Proses : baca data_1, data_2...data_n
sort data bilangan tersebut

A. Outline Solusi

Input	Proses	Output
byk_data	baca data_1, data_2, ...data_n	data_1
data_1	sort data bilangan tersebut	data_2
data_2	cetak hasil sort	data_n
data_n		

Hierarchy Chart



HASIL MODULARISASI

SUB SORTING_DATA

```
FOR I = 1 TO N
    FOR J = N TO J>=I STEP -1
        IF DATA[J] < DATA[J-1] THEN
            TUKAR_DATA(J,J-1)
        ENDIF
    END FOR
END FOR
ENDSUB
```

SUB TUKAR_DATA(int a, int b)

```
INT TEMP
TEMP = DATA[B];
DATA[B] = DATA[A];
DATA[A] = TEMP
ENDSUB
```

SUB INPUT_DATA

```
FOR I = 1 TO N
    BACA DATA[I]
END FOR
ENDSUB
```

SUB CETAK_DATA

```
FOR I = 1 TO N
    BACA DATA[I]
END FOR
ENDSUB
```

PROGRAM SORTING

```
MAIN_SORTING
INT N, J, DATA[10]

    BACA N
    INPUT_DATA
    SORTING_DATA
    CETAK_DATA

END
```


KOHESI & KOPLING

- *Cohesion* dan *Coupling* merupakan konsep dasar dalam perancangan dan rekayasa perangkat lunak.
- Membagi *software*/perangkat lunak menjadi modul-modul yang kecil bukan sekedar memisahkan kumpulan kode dari kumpulan kode lainnya. Tetapi memastikan bahwa modul yang dirancang menganut prinsip "*Loose Coupling, High Cohesion*"

KOHESI

- Kohesi adalah keeratan hubungan elemen-elemen di dalam suatu modul.
- Macam-macam Kohesi
 - Functional
 - Sequential
 - Communicational
 - Procedural
 - Temporal
 - Logical
 - Coincidental

baik/kuat



Jelek/lemah

KOHESI (lanjutan)

- *Perubahan* pada modul dengan tingkat kohesi tinggi tidak terlalu membawa dampak perubahan terhadap modul lain. Sehingga *lebih mudah dalam pemrograman, pengujian dan perawatan*
- Modul dengan tingkat kohesi tinggi, *lebih mudah dipahami dan didokumentasi*
- Pada modul dengan tingkat kohesi tinggi, *informasi lebih mudah disembunyikan*, karena komunikasi antar modul diminimalkan

KOHESI FUNCTIONAL

- Mempunyai satu tugas
- Menghasilkan satu hasil/satu parameter output
- Bisa satu atau lebih parameter input

```
Menghitung_pajak_penjualan
  IF harga>5000 THEN
    pajak = harga * 0.25
  ELSE
    IF harga>4000 THEN
      pajak = harga * 0.25
    ELSE
      IF harga>3000 THEN
        pajak = harga * 0.25
      ELSE
        IF harga>2000 THEN
          pajak = harga * 0.25
        ELSE
          pajak = 0
        ENDIF
      ENDIF
    ENDIF
  ENDIF
END
```

KOHESI SEQUENTIAL

- Mempunyai pekerjaan yang beruntun
- Kegiatan yang dilakukan lebih dari satu
- Hasil dari kegiatan sebelumnya menjadi masukan bagi kegiatan selanjutnya
- Dapat dipecah menjadi Functional

```
Menghitung_penjualan
  IF jml_beli > 300 THEN
    pot = jml_beli * hrg_beli * 15%
  ELSE
    IF jml_beli > 200 THEN
      pot = jml_beli * hrg_beli * 10%
    ELSE
      IF jml_beli > 100 THEN
        pot = jml_beli * hrg_beli * 5%
      ELSE
        pot = 0
      ENDIF
    ENDIF
  ENDIF
  Pembelian = jml_beli * hrg_beli - pot
  Tot_pembelian = Tot_pembelian + Pembelian
END
```

KOHESI KOMUNIKASIONAL

- Kegiatan lebih dari satu
- Menggunakan data yang sama
- Dapat dijadikan Functional
- Contoh:

Sub Proses_perhitungan

$$C = A + B$$

$$D = A - 1$$

$$E = A * B$$

$$F = A / B$$

$$G = A \bmod B$$

EndSub

KOHESI PROCEDURAL

- Satu kegiatan dengan kegiatan lain tidak berhubungan
- Hubungan antara elemen yang satu dengan yang lainnya karena urutan statement
- Dapat dipecahkan menjadi Functional
- Contoh:

Sub Baca_record_mhs_dan_total_usia_mhs

set no_record to 0

set total_usia to 0

baca record_mhs

DO WHILE not EOF

total_usia = usia + total_usia

no_record = no_record + 1

ENDDO

print no_record, total_usia

EndSub

KOHESI TEMPORAL

- Elemen-elemen terlibat dalam berbagai kegiatan yang mempunyai hubungan dalam waktu
- Urutan tidak penting
- Contoh:

Sub Inisialisasi

buka file transaksi

total_transaksi = 0

total_pen = 0

baris = 30

no = 0

hal = 0

EndSub

KOHESI LOGICAL

- Elemen-elemen melakukan kegiatan dengan kategori yang sama
- Parameter masukan menentukan kegiatan yang dilaksanakan
- Tidak semua kegiatan dikerjakan
- Contoh:

Read_all_files(file_code)

 CASE if file_code

 1 : read customer_transaction record

 IF not EOF THEN

 increment cust_trans_count

 2 : read customer_master record

 IF not EOF THEN

 increment cust_master_count

 3 : read product_master record

 IF not EOF THEN

 increment product_master_count

 ENDIF

 ENDCASE

END

KOHESI KOINSIDENTAL

- Elemen-elemen tidak mempunyai hubungan
- Contoh:

Sub File_Processing

Open employee updates file

read employee record

print_page heading

open employee master file

set page_count to one

set error_flag to false

EndSub

KOPLING

- Kopling adalah keeratan hubungan antar modul. Tingkat saling ketergantungan di antara dua modul.
- Faktor yang mempengaruhi kopling:
 1. Jumlah data yang disalurkan
 2. Jumlah kontrol data yang disalurkan
 3. Jumlah elemen data global yang digunakan bersama-sama oleh beberapa modul

JENIS-JENIS KOPLING

- Data
- Stamp
- Kontrol
- External
- Common

baik/lemah



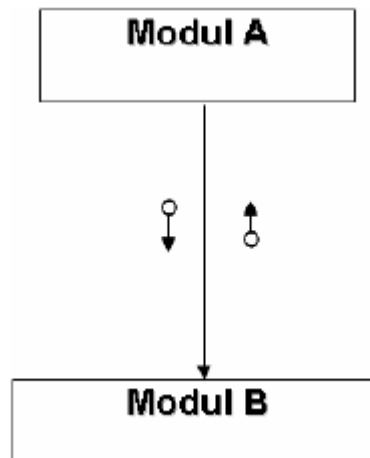
jelek/kuat

Keterangan:

- Makin baik kopling, makin rendah ketergantungan suatu modul terhadap modul lain
- Modul dengan kopling yang baik adalah modul independence

KOPLING DATA

- Komunikasi diantara modul menggunakan data. Diinginkan jumlah data minimal
- Parameter data yang disalurkan semakin sedikit semakin baik



Keterangan:

■ *Parameteranya terdiri dari data*

■ *Jumlah parameter minimal*

CONTOH KOPLING DATA

A. Process_record_pelanggan

...

...

hitung_pajak_penjualan (total_harga, pajak_penjualan)

...

...

END

B. Hitung_pajak_penjualan (long total, pajak)

IF total > 5000 THEN

pajak = total * 0.25

Else If total > 4000 THEN

pajak = total * 0.2

Else

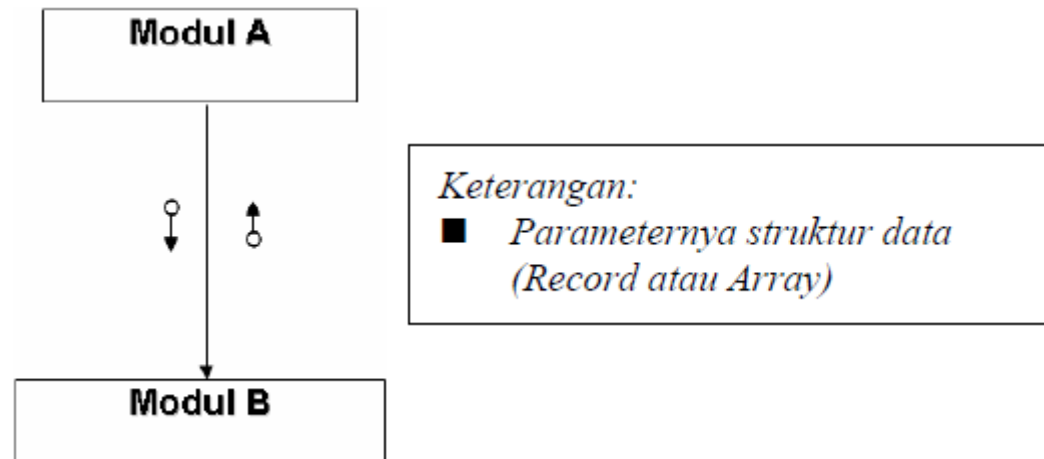
pajak = total * 0.15

ENDIF

END

KOPLING STAMP

- Dua modul melakukan pass struktur data non global yang sama
- Struktur data: record, array
- Timbul bahaya bila modul memeriksa struktur data tetapi hanya menggunakan sebagian



```
Proses_record_transaksi
```

```
....
```

```
Proses_siswa_pria(current_record)
```

```
....
```

```
END
```

```
Proses_siswa_pria(current_record)
```

```
jml_siswa_pria = jml_siswa_pria + 1
```

```
IF usia_siswa > 21 THEN
```

```
    jml_siswa_dewasa = jml_siswa_dewasa + 1
```

```
ENDIF
```

```
....
```

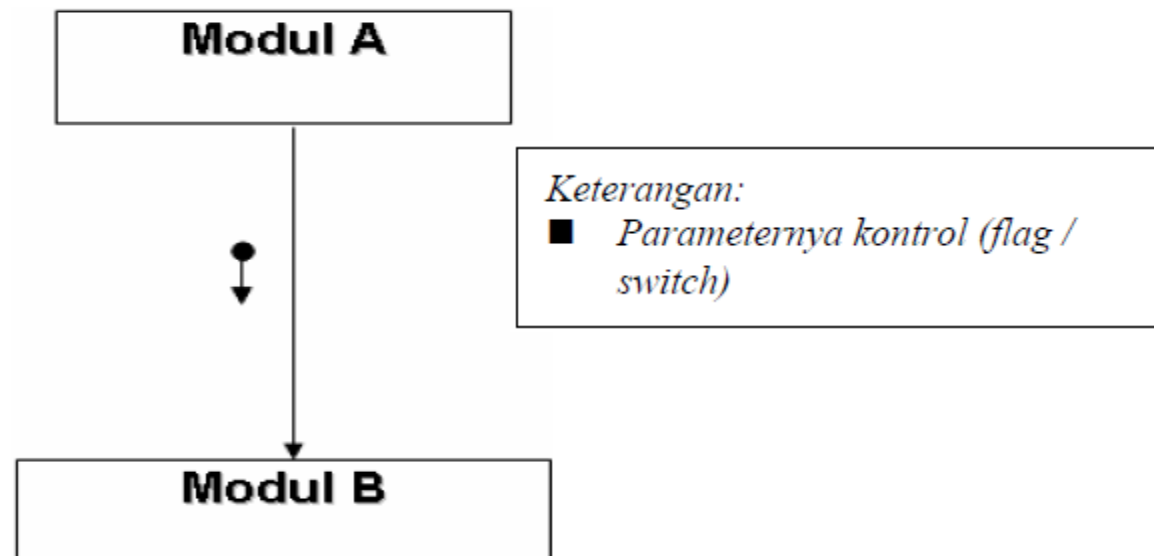
```
....
```

```
END
```

- *current_record hanya berupa penunjuk nomor record sekarang*

KOPLING KONTROL

- Dua modul melakukan passing parameter menggunakan data kontrol (flag/switch)



Proses_input_code

Read input_code

Lakukan_aksi(input_code)

... .

... .

END

Lakukan_aksi(input_code)

CASE of Input_code

1: Read_record_karyawan

2: Cetak_header_halaman

3: Open_master_file_Karyawan

4: jml_halaman = 0

5: error_message

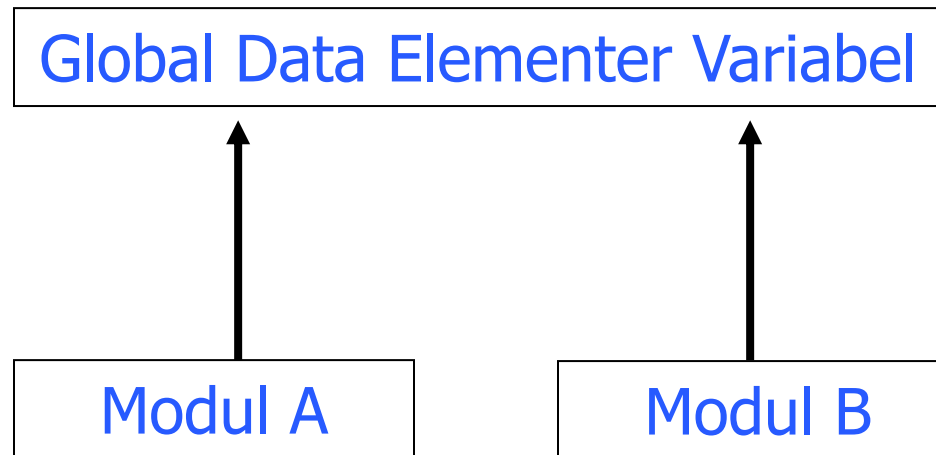
ENDCASE

END

- *input_code berfungsi sebagai switch (berupa switch)*

KOPLING EXTERNAL

- Dua modul atau lebih menggunakan data global yang sama
- Tidak ada parameter yang digunakan dari modul pemanggil ke subordinate dan sebaliknya



Hitung_pajak_Penjualan

```
IF produk = "sampel" THEN
    pajak_pen = 0
ELSE
    IF harga_produk < 50000 THEN
        pajak_pen = hrg_jual * 0.25
        ...
    ENDIF
ENDIF
END
```

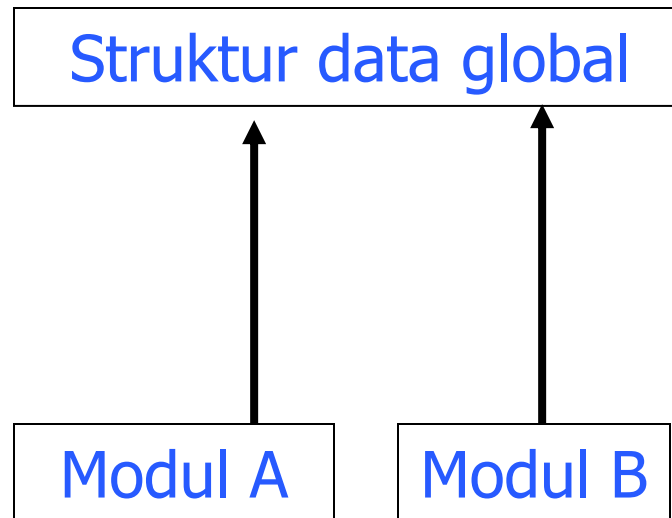
Hitung_nilai_total()

```
...
...
total = nilai_total + pajak_pen
...
...
END
```

- *pajak_pen adalah variabel data global*

KOPLING COMMON

- Dua modul atau lebih menggunakan struktur data global yang sama



```
Baca_record_pelanggan
```

```
    Read record_pelanggan
```

```
    IF EOF THEN
```

```
        EOF_flag = true
```

```
    ENDIF
```

```
END
```

```
Validasi_record_pelanggan()
```

```
    IF noPelanggan NOT numeris THEN
```

```
        error_msg = "invalid nomor pelanggan"
```

```
        Print_laporan_error
```

```
    ENDIF
```

```
    ... .
```

```
    ... .
```

```
END
```

- record_pelanggan adalah struktur data global

LATIHAN 5

- Buatlah pseudocode, plowchart dan program untuk menampilkan menu untuk menghitung luas bangun ruang seperti : persegi panjang, segitiga dan bujur sangkar

TUGAS 5

Buatlah program lengkap dengan pseudocode dan flowchart untuk menampilkan

- Bilangan fibonancii
- Bilangan faktorial
- Angka bilangan bulat dalam bentuk kalimat, contoh : 32,768 ditampilkan Tiga Puluh Dua Ribu Tujuh Ratus Enam Puluh Delapan Rupiah.

TUGAS 5 (lanjutan)

Catatan Tugas :

- Tugas dibuat pada kertas folio bergaris dengan menggunakan bolpoint.
- Tugas dikumpulkan pada saat pertemuan 9. Bagi mahasiswa yang tidak mengumpulkan tugas maka tidak mendapat nilai tugas 5 (tidak ada sistem susulan).

SOAL LATIHAN

1. Fungsi pemanggilan dimana nilai dari parameter aktual akan ditulis ke parameter formal dan nilai parameter aktual tidak bisa berubah, walaupun nilai parameter formal berubah

- a. Pemanggilan dengan acuan
- b. Pemanggilan dengan Nilai
- c. Pemanggilan aktual
- d. Pemanggilan global
- e. Pemanggilan kohesi

2. Kohesi yang kegiatannya lebih dari satu serta menggunakan data yang sama adalah

- a. Kohesi sequential
- b. Kohesi functional
- c. Kohesi komunikasional
- d. Kohesi prosedural
- e. Kohesi temporal

3. Pada perintah `cout<<"Metode";` cout merupakan token

- a.Operator
- b.Identifier
- c.Keyword
- d.Konstanta
- e.Delimiter

4. jika jumlah iterasi pengulangan sudah diketahui adalah syarat dari:

- a.Simple selection
- b.Combined selection
- c.Leading decision loop
- d.Trailing decision loop
- e.Counted loop

5. Hasil dari kegiatan sebelumnya menjadi masukan bagi kegiatan selanjutnya, pengertian dari

- a. Kohesi Functional
- b. Kohesi Komunikasional
- c. Kohesi Procedural
- d. Kohesi Sequential
- e. Kohesi Temporal