7 FUNGSI PENGATURCARAAN BERMODULAR

FUNGSI -
Kumpulan jujukan suruhan[2] yg diberikan nama. Contoh fungsi[2] yg telah dipelajari:

main()
printf()
scanf()
getchar()
putchar()
puts()

} Fungsi[2] piawai perpustakaan yg ditulis oleh pembekal pengkompil spt Borland & Microsoft

2 kategori fungsi :
1. Fungsi piawai perpustakaan C, spt di atas.
2. Fungsi takrifan pengguna, yg direkabentuk & ditulis sendiri oleh pengaturcara.

7.1 Mengapa Perlu Fungsi ?

- ***Struktur mudah difahami***
  Aturcara menjadi lebih mudah dibaca dan difahami kerana kumpulan suruhan[2] yg melakukan kerja tertentu dikumpulkan dlm beberapa bahagian yg dipanggil **fungsi**. Ia juga memudahkan proses penyahpepijat.

- ***Jimat masa/ruang & Guna semula***
  Jujukan suruhan yg berulang[2] dlm sesuatu aturcara boleh dijadikan fungsi. Fungsi ini hanya perlu ditulis sekali shj & boleh digunakan berulang[2] dgn memanggil nama fungsi tersebut. Contohnya fungsi printf yg panjangnya beratus baris hanya ditulis sekali oleh pembekal pengkompil, tetapi boleh digunakan berulang[2] dlm aturcara kita.

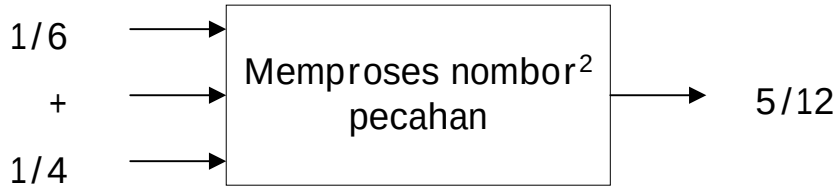
- ***Menggalakkan aturcara bermodular***
  Masalah yg besar dipecahkan kpd masalah yg kecil (pendekatan atas-bawah) & skop masalah yg kecil tersebut diselesaikan satu-persatu.

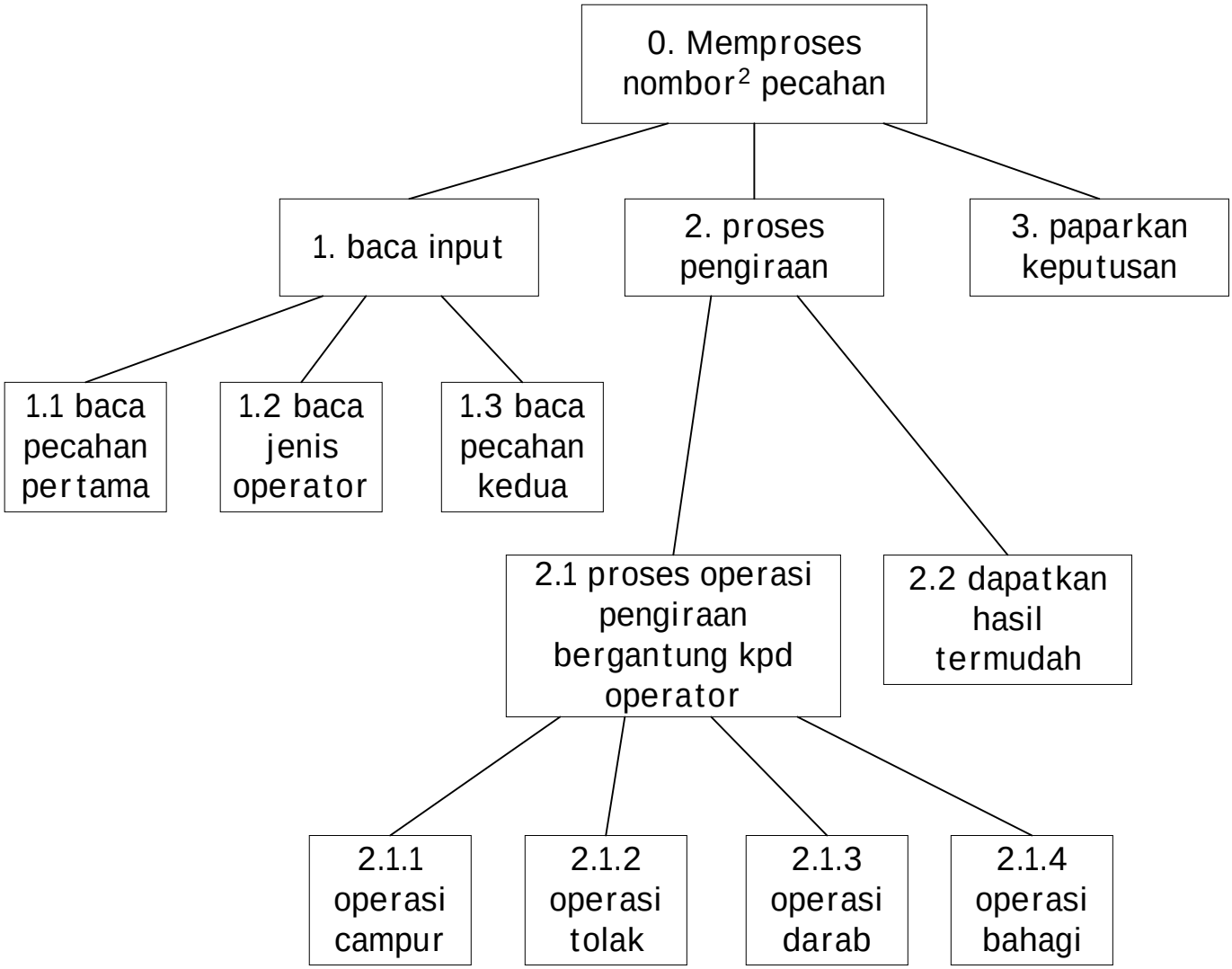
7.1 Mengapa Perlu Fungsi ? (samb....)

- Contoh PM menggunakan pendekatan atas-bawah

Anda diminta untuk menyediakan satu aturcara yang boleh mengira nombor^2 pecahan. Operasi pengiraan adalah operasi campur, tolak, darab dan bahagi. Hasil dari pengiraan tersebut mestilah dalam bentuk pecahan termudah.

Perhatikan contoh satu input & output kpd proses tersebut.





7.2 Cara Menulis Fungsi

Bentuk asas sesuatu fungsi:

```
jenis_pulangan  nama_fungsi (parameter_masukan)
{
        badan_fungsi;
        return(nilai);
}
```

- ***jenis_pulangan*** :
  Jenis (unsigned/signed char, int, float, long) output/pulangan yg dihasilkan oleh fungsi tersebut. Jika fungsi tiada output/ pulangan jenisnya mestilah **void**. Jika *jenis_pulangan* ditinggalkan, jenis **int** adalah dianggap.

- ***nama_fungsi*** :
  Nama fungsi. Nama ini akan digunakan utk memanggil/ melaksanakan fungsi tersebut. Nama fungsi mestilah mematuhi syarat[2] yg sama dgn pencam.

- ***parameter_masukan*** :
  Pembolehubah[2] yg menjadi masukan/input kpd fungsi tersebut bagi melakukan sesuatu kerja. Jika ada lebih dari satu parameter, mereka mesti dipisahkan oleh koma (,). **Jenis, bilangan** dan **aturan** bagi *parameter_masukan* mestilah sama semasa fungsi ditakrifkan dan digunakan.

- ***badan_fungsi*** :
  Terdiri dari satu atau lebih kenyataan[2] C termasuk pengisytiharan pembolehubah[2] yg melakukan sesuatu kerja. Cara menulisnya sama dgn cara menulis badan fungsi main().

- ***return (nilai)*** :
  Suruhan ini akan menghasilkan output/pulangan dari fungsi tersebut. *nilai* ialah sebarang nilai dari pembolehubah/pemalar tetapi jenisnya mestilah sama dgn *jenis_pulangan* yg diberikan di atas.
  Bagi fungsi jenis **void**, suruhan return() tidak ditulis.

7.2 Cara Menulis Fungsi (samb....)

Contoh 1:

a)
```
int campur (int x, int y)
{
    return(x+y);
}
```

b)
```
void cetak_mesej (void)
{
    printf("Hi\n");
}
```

c)
```
float kira_luas (float l, float p)
{
   float A;
   A = p*l;
    return(A);
}
```

Contoh 2:

Tulis satu aturcara utk mengira & memaparkan luas satu segiempat, satu bulatan atau satu segitiga atas pilihan pengguna. Saiz sisi^2 diberikan oleh pengguna melalui papan kekunci. Anggap saiz positif shj yg dimasukkan.
Tulis tiga fungsi utk mencari luas segi4, bulatan & segi3.
Pembolehubah2 jenis float.

Penyelesaian:

i) Analisis:

Input:

media : papan kekunci

data : pilih operasi, panjang & lebar @ jejari @ tapak & tinggi

Output:

media : skrin

data : luas segiempat @ luas bulatan @ luas segitiga

Proses:

luas segiempat = panjang x lebar

luas bulatan = π x jejari2

luas segitiga = (tapak x tinggi)/2

7.2 Cara Menulis Fungsi (samb....)

Penyelesaian ....:

ii) Kod pseudo I:

1. cetak menu pilihan operasi
2. baca pilihan
3. jika pilihan cari luas segiempat
   3.1 baca nilai panjang & lebar
   3.2 luas segiempat = panjang x lebar
4. jika pilihan cari luas bulatan
   4.1 baca nilai jejari
   4.2 luas bulatan = π x jejari^2
5. jika pilihan cari luas segitiga
   5.1 baca nilai tapak & tinggi
   5.2 luas segitiga = (tapak x tinggi)/2
6. cetak luas

ii) Kod pseudo II:

```
cetak menu pilihan operasi
baca pilihan
if pilihan = 1 then
        begin
          baca nilai panjang & lebar
          luas segiempat = panjang x lebar
        end
  else if pilihan = 2 then
        begin
          baca nilai tapak & tinggi
          luas segitiga = (tapak x tinggi)/2
        end
   else if pilihan = 3 then
        begin
          baca nilai jejari
          luas bulatan = π x jejari^2
        end
  else if pilihan luas segitiga then
        begin
          baca nilai tapak & tinggi
          luas segitiga = (tapak x tinggi)/2
        end
  else cetak amaran
  endif
  cetak luas
```

7.2 Cara Menulis Fungsi (samb....)

Penyelesaian ....:

iii) Aturcara:

fungsi luas_segi4

```
#include <stdio.h>

float luas_segi4 (void)
{
   float pjg, lbr;

   printf("masukkan saiz: panjang lebar\n");
   scanf("%f %f", &pjg, &lbr);
   return (pjg*lbr);
}
```

fungsi luas_segi3

```
float luas_segi3 (float tapak, float tinggi)
{
   float luas;

   luas =(tapak*tinggi)/2.0;
   return ( luas );
}
```

fungsi luas_bulat

```
float luas_bulat (float jejari)
{
   return ( 3.1415*jejari*jejari );
}
```

fungsi menu

```
void menu (void)
{
   printf("1: Luas segi4\n");   printf("2: Luas segi3\n");
   printf("3: Luas bulatan\n"); printf("pilih 1, 2 @ 3 >");
}
```

fungsi main

```
void main()
{
     float luas, j, tpk, tg;    int pilih;

     menu();   scanf ("%d", &pilih);
     if (pilih == 1)
        luas = luas_segi4 ();
     else if (pilih == 2) {
        printf("masukkan saiz: tapak lebar\n");
        scanf("%f %f", &tpk, &tg);
        luas = luas_segi3 (tpk, tg);
     } else if (pilih == 3 ) {
        printf("Saiz jejari ?: ");
        scanf("%f", &j);  luas = luas_bulat (j);
     } else   printf("pilih 1, 2 @ 3 shj\n");

     printf ("luas ialah %f\n", luas);
}
```

panggilan^2 fungsi

7.3 Prototaip Fungsi

- Jika rujuk semula penyelesaian mencari luas di ms 6, fungsi^2 yg dipanggil diletakkan sebelum fungsi yg memanggil, kes ini tidak memerlukan prototaip fungsi. Tapi jika sebaliknya kita memerlukan prototaip fungsi.

- Perhatikan 2 kes di bawah:

prototaip fungsi

tidak memerlukan prototaip fungsi

```
void fungsi1 (...)
{
     :
     :
}

void fungsi2 (...)
{
     :
   fungsi1 (...);
     :
}

void fungsi3 (...)
{
     :
     :
}


void main()
{
     fungsi1 (...);
     fungsi2 (...);
     fungsi3 (...);
}
```

memerlukan prototaip fungsi

```
void fungsi1 (...);
void fungsi2 (...);
void fungsi3 (...);

void main()
{
     fungsi1 (...);
     fungsi2 (...);
     fungsi3 (...);
}


void fungsi2 (...)
{
     :
   fungsi1 (...);
     :
}

void fungsi1 (...)
{
     :
     :
}

void fungsi3 (...)
{
     :
     :
}
```

- Syarat menulis prototaip fungsi
  * menyerupai baris pertama takrifan fungsi & ditambah semikolon.
  *nama fungsi, bilangan, susunan & jenis parameter mesti sama
  *nama parameter boleh ditinggalkan, jika tidak ditinggalkan boleh berbeza @ sama

7.3 Prototaip Fungsi (samb....)

Aturcara penyelesaian mencari luas di ms 6 dgn prototaip fungsi.

```
#include <stdio.h>

float luas_segi4 (void);
float luas_segi3 (float, float);
float luas_bulat (float jejari);
void menu (void);

void main()
{
     float luas, j, tpk, tg;    int pilih;

     menu();   scanf ("%d", &pilih);
     if (pilih == 1)
        luas = luas_segi4 ();
     else if (pilih == 2) {
        printf("masukkan saiz: tapak lebar\n");
        scanf("%f %f", &tpk, &tg);
        luas = luas_segi3 (tpk, tg);
     } else if (pilih == 3 ) {
        printf("Saiz jejari ?: ");
        scanf("%f", &j);  luas = luas_bulat (j);
     } else   printf("pilih 1, 2 @ 3 shj\n");
     printf ("luas ialah %f\n", luas);
}

float luas_segi4 (void)
{
   float pjg, lbr;
   printf("masukkan saiz: panjang lebar\n");
   scanf("%f %f", &pjg, &lbr);
   return (pjg*lbr);
}

float luas_segi3 (float tapak, float tinggi)
{
   float luas;
   luas =(tapak*tinggi)/2.0;
   return ( luas );
}

float luas_bulat (float jejari)
{
   return ( 3.1415*jejari*jejari );
}

void menu (void)
{
   printf("1: Luas segi4\n");   printf("2: Luas segi3\n");
   printf("3: Luas bulatan\n"); printf("pilih 1, 2 @ 3 >");
}
```

7.4 Parameter[2] dlm. Fungsi

- **Paramater formal** - nama & jenis parameter yang ditakrif semasa fungsi diisytiharkan.

Parameter formal

```
float luas_bulat (float jejari)
{
   return ( 3.1415*jejari*jejari );
}
```

- **Parameter sebenar** - nilai sebenar yang diberikan oleh fungsi pemanggil kpd parameter formal semasa fungsi itu dipanggil/ digunakan.

```
void main()
{
     :
     :
  scanf("%f", j);
  luas=luas_bulat ( j );
     :
     :
}
```

Pemindahan parameter

```
float luas_bulat (float jejari)
{
   return ( 3.1415*jejari*jejari);
}
```

Parameter sebenar

7.5 Panggilan & Pulangan Fungsi

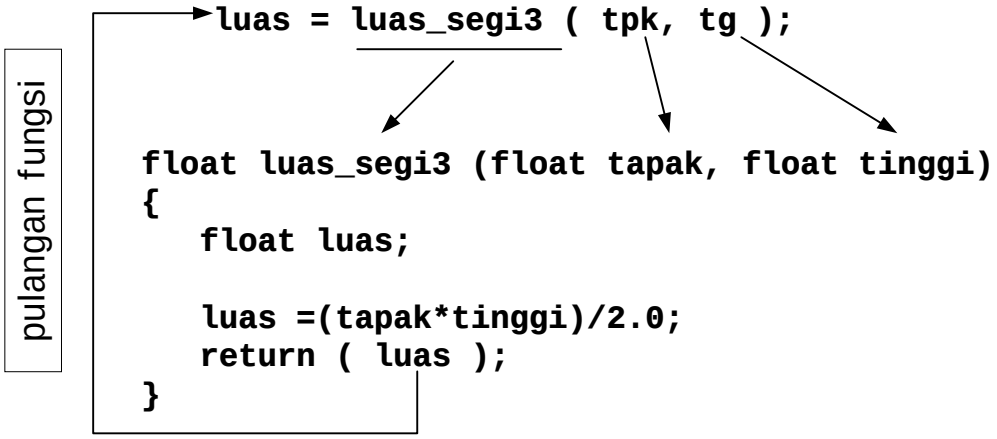
- Untuk melaksankan fungsi, ia perlu dipanggil menggunakan format tertentu.
  Fungsi boleh dipanggil oleh mana[2] fungsi luar atau fungsi yg sama (dinamakan rekursif).

- 3 perkara yg perlu diperhatikan utk memanggil fungsi:

Perkara	Syarat
1. nama fungsi	*nama mesti sama dgn nama fungsi
2. Data yg dihantar : parameter formal lwn. parameter sebenar	*jenis & bilangan mesti sama *mesti dalam turutan yg sama
3. pulangan fungsi	*jenis mesti sama

7.4 Panggilan & Pulangan Fungsi (samb....)

- Contoh panggilan fungsi **luas_segi3**.



- Satu fungsi tidak semestinya memulangkan nilai atau menghantar data. Contoh panggilan fungsi **menu**. yg tidak memulangkan nilai & tidak menghantar apa-apa data spt berikut:

```
menu ( );
```

```
void menu (void)
{
   printf("1: Luas segi4\n");   printf("2: Luas segi3\n");
   printf("3: Luas bulatan\n"); printf("pilih 1, 2 @ 3 >");
}
```

contoh[2] panggilan fungsi yg sah:

```
printf( "Luas segi3 = %f", luas_segi3(tpk,tg));
luassegi3 = luas_segi3(14.1,10.5);
luasbulat = luas_bulat(2.2 + j);
luassegi4_10 = luas_segi4() + 10.0;
```

contoh[2] kenyataan **return** yg sah:

```
return 0;      return luas;        return (luas);
return ( 3.1415*jejari*jejari );
```

- Bila satu fungsi dipanggil kenyataan di dlm badan fungsi akan dilaksanakan satu persatu sehingga tamat badan fungsi atau kenyataan return dilaksanakan. Contoh di bawah **fungsi1** kenyataan1 & 2 akan dilaksanakan sebelum kembali ke fungsi yg memanggil,ttp **fungsi2** hanya kenyataan1 shj dilaksanakan.

```
void fungsi1(void)
{
     kenyataan1;
     kenyataan2;
}
```

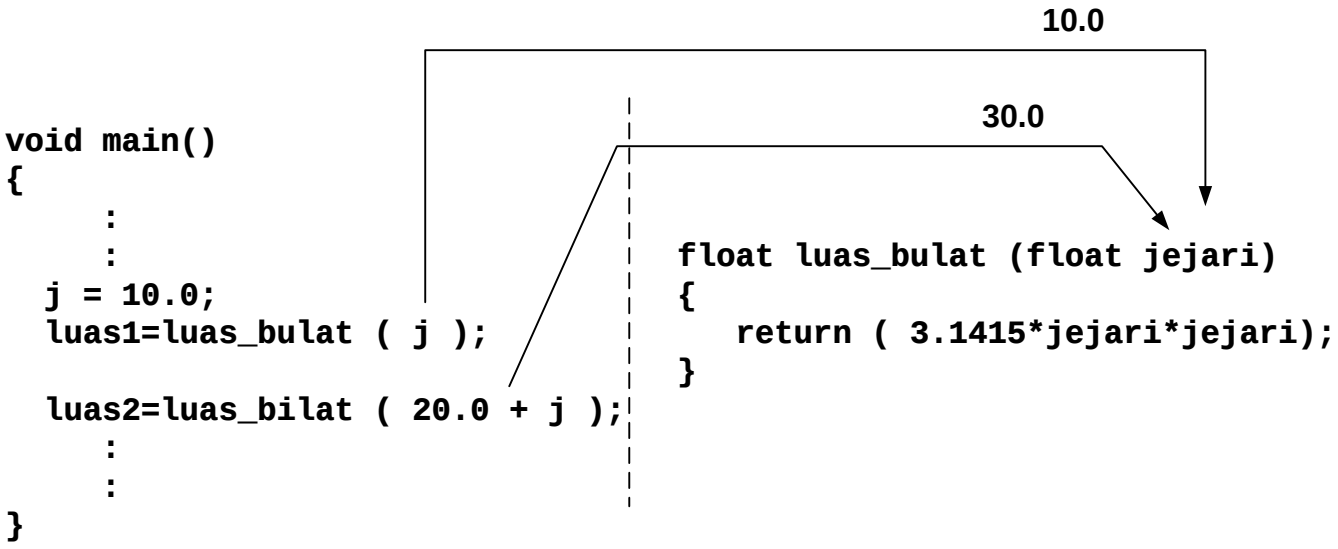
```
int fungsi2(void)
{
     kenyataan1;
     return 0;
     kenyataan2;
}
```

7.5 Penghantaran Nilai kpd Fungsi

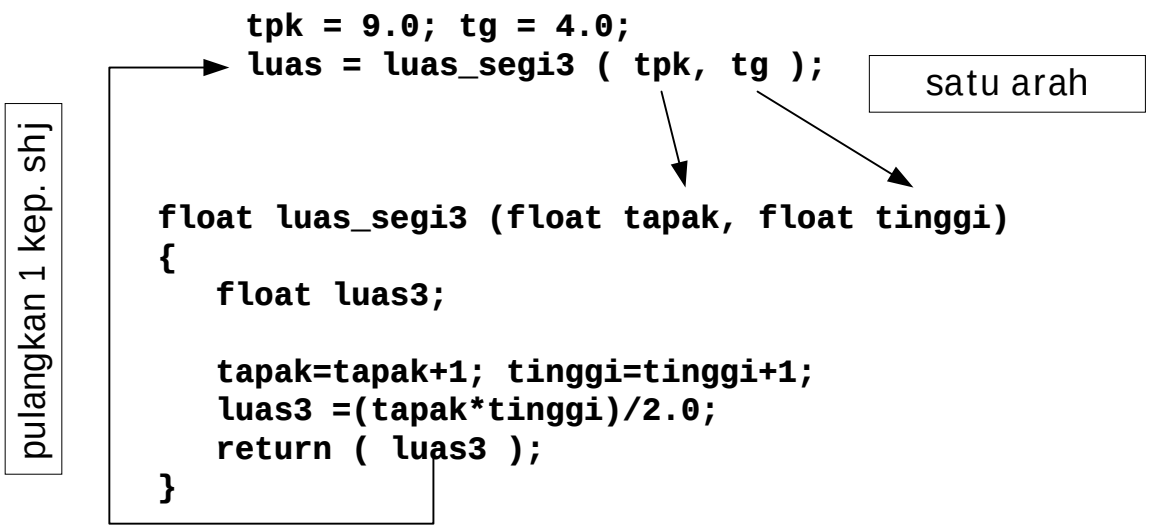
- Semasa fungsi dipanggil, penghantaran nilai oleh pemanggil boleh dibuat dlm 2 kaedah:
  - penghantaran nilai sebenar
  - pnghantaran alamat nilai

7.5.1 Penghantaran Nilai Sebenar

- Nilai sebenar akan diumpukan kpd parameter formal pada fungsi yang memanggil. Contohnya:



  - Penghantaran nilai sebenar, membolehkan satu @ beberapa nilai dihantar kpd sesuatu fungsi dlm bentuk satu arah. Jika kita perlukan fungsi yg dipanggil memberi nilai kpd fungsi yg memanggil, penyataan return boleh digunakan. Masalah pernyataan return hanya boleh pulangkan satu keputusan shj.
  Perhatikan contoh fungsi **luas_segi3** berikut:

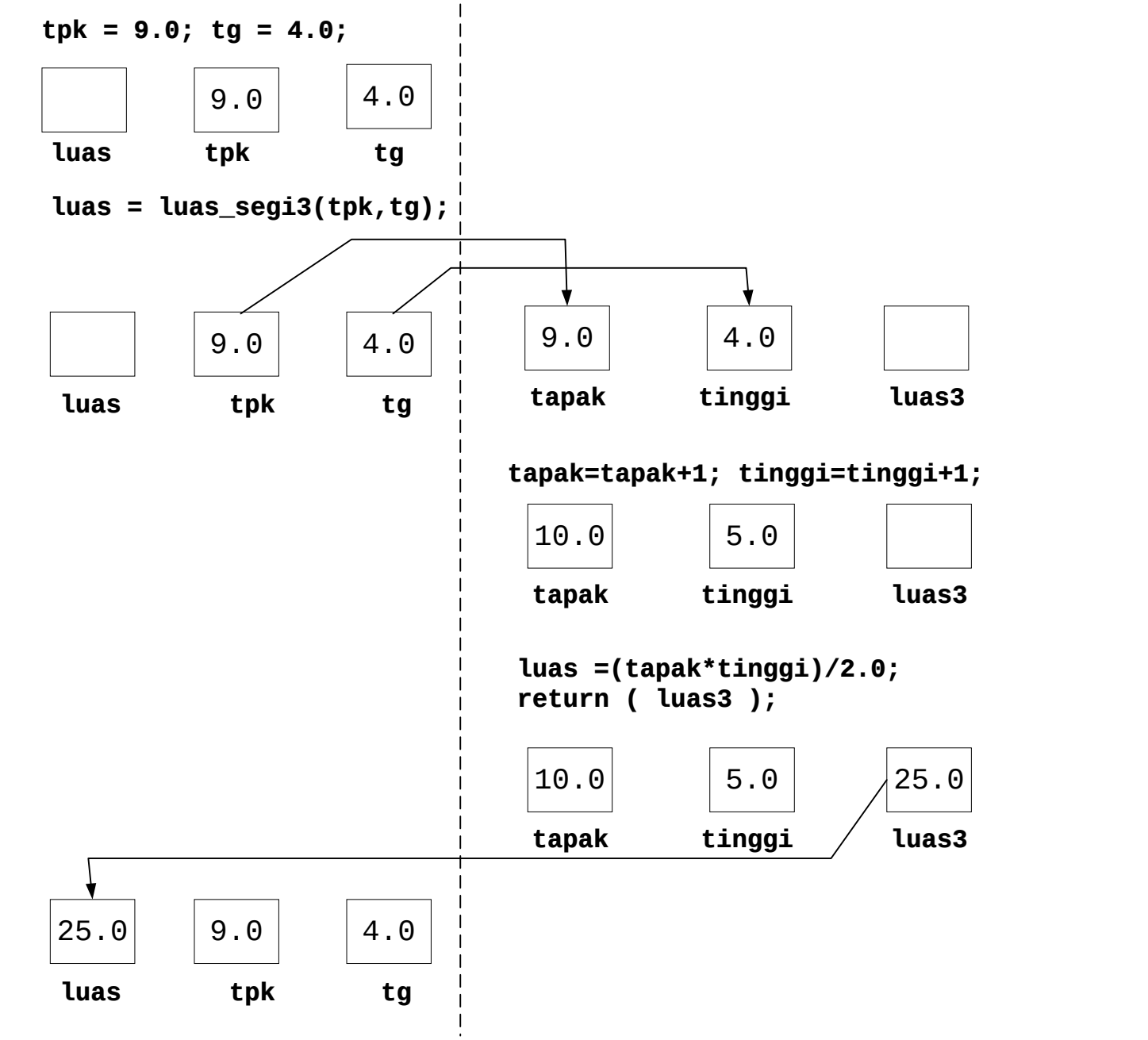


7.5.1 Penghantaran Nilai Sebenar (samb....)

- Apa yg berlaku dlm contoh panggilan fungsi **luas_segi3** ialah:

```
void main()
{
     :
     :
tpk = 9.0; tg = 4.0;

luas = luas_segi3(tpk,tg);

     :
     :
}
```

```
float luas_segi3(float tapak,float tinggi)
{
   float luas3;

   tapak=tapak+1; tinggi=tinggi+1;
   luas3 =(tapak*tinggi)/2.0;
   return ( luas3 );
}
```



7.5.2 Penghantaran Alamat Nilai

- Alamat di mana nilai disimpan di RAM akan dihantar kpd parameter formal pada fungsi yang memanggil. Alamat sesuatu pembolehubah boleh didapati dgn:

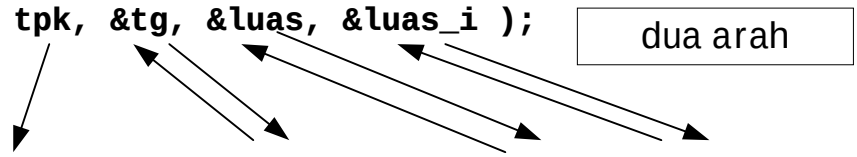
  &nama_pembolehubah

- Fungsi yg menerima alamat tersebut, boleh menggunakan alamat tersebut utk mendapatkan nilai (dengan menuding) yg terkandung dlm alamat yg dihantar. Nilai pembolehubah yg distorkan di dlm alamat boleh dirujuk dgn:

  ***nama_pembolehubah**

- Penghantaran alamat nilai, membolehkan satu @ beberapa nilai dihantar kpd sesuatu fungsi dlm bentuk dua arah. Jika kita perlukan fungsi yg dipanggil memberi nilai kpd fungsi yg memanggil, penyataan return tidak perlu digunakan dan lebih dari satu nilai boleh diberikan.
  Perhatikan contoh fungsi **luas_segi3** berikut:

```
luas = 0.0; luas_i = 0;
tpk = 9.0; tg = 4.0;
luas_segi3 ( tpk, &tg, &luas, &luas_i );
```




```
void luas_segi3 (float tapak, float *tinggi, float *l_f, int *l_i)
{
   tapak=tapak+1; *tinggi=*tinggi+1;

   *l_f =(tapak*(*tinggi))/2.0;
   *l_i = (int)(*l_f/2); /* bhg 2 & tukar kpd no. bulat */
}
```

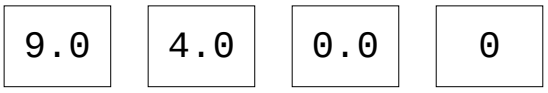
- perhatihan apa yg berlaku dlm fungsi ini di sebelah.

7.5.2 Penghantaran Alamat Nilai (samb....)

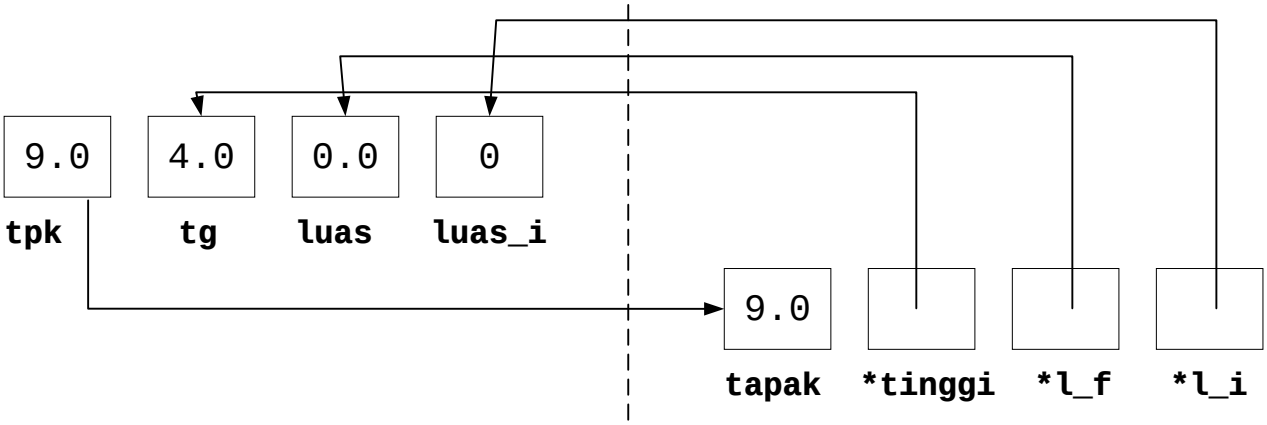
```
void main()
{
     :
luas = 0.0; luas_i = 0;
tpk = 9.0; tg = 4.0;

luas_segi3 ( tpk, &tg,
        &luas, &luas_i );

          :
}
```

```
void luas_segi3 (float tapak, float
       *tinggi, float *l_f, int *l_i)
{
   tapak=tapak+1; *tinggi=*tinggi+1;

   *l_f =(tapak*(*tinggi))/2.0;
   *l_i = (int)(*l_f/2);
    /* bhg 2 & tukar kpd no. bulat */
}
```

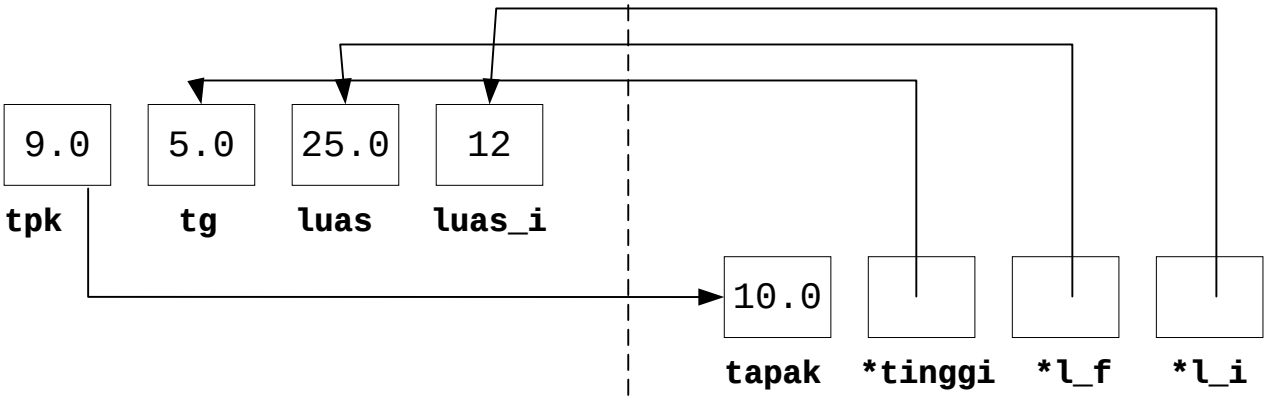
```
luas = 0.0; luas_i = 0;
tpk = 9.0; tg = 4.0;
```




```
luas_segi3 ( tpk, &tg,
        &luas, &luas_i );
```




```
tapak=tapak+1; *tinggi=*tinggi+1;
*l_f =(tapak*(*tinggi))/2.0;
*l_i = (int)(*l_f/2);
```



7.6 Skop Pembolehubah

- Pembolehubah yang diisytiharkan di dlm sesuatu fungsi dinamakan **pembolehubah setempat/lokal**.

- Pembolehubah lokal :
  1. boleh digunakan/dicapai oleh fungsi tersebut shj. Fungi^2 lain termasuk *main* tidak boleh mencapai pembolehubah^2 lokal tersebut
  2. hanya wujud semasa fungsi itu aktif/dipanggil

- Pembolehubah yang diisytiharkan di luar fungsi dinamakan **pembolehubah sejagat/global**.

- Pembolehubah global boleh digunakan/dicapai oleh semua fungsi kecuali jika pembolehubah global tersebut mempunyai nama yang sama dgn pembolehubah lokal.

Contoh 1: Pertimbangkan aturcara berikut.

```
#include <stdio.h>

float luas= 0;

float luas_segi4 (void)
{
   float pjg, lbr, luas;

   luas = pjg*lbr;
   return (luas);
}

float luas_bulat (float jejari)
{
   float luas;

   luas = 3.1415*jejari*jejari;
   return(luas);
}

void main()
{
     float j;

     :
     luas_bulat (j);
     :
}
```

7.6 Skop Pembolehubah (samb....)

- Pembolehubah **pjg, lbr** & **luas** adalah lokal kpd **luas_segi4().** Hanya boleh dicapai oleh fungsi **luas_segi4()** shj.
  Pembolehubah **luas** adalah lokal kpd **luas_bulat()** hanya boleh dicapai oleh fungsi **luas_bulat()** shj.
  Pembolehubah **j** adalah lokal kpd **main()** hanya boleh dicapai oleh fungsi **main()** shj.

- Pembolehubah **luas** di luar fungsi adalah pembolehubah global. Ia boleh dicapai oleh semua fungsi^2 kecuali fungsi^2 **luas_segi4** & **luas_bulat** kerana pembolehubah lokal mempunyai nama yang sama.

7.7 Skop Prototaip Fungsi

- Di ms 7, pada kes yg ke 2 prototaip fungsi perlu digunakan kerana fungsi memanggil diketak sebelum fungsi dipaggil. Seperti skop pembolehubah, skop prototaip fungsi juga boleh diisytihar secara sejagat dan setempat.

- Prinsipnya juga sama dgn skop pembolehubah. Protataip sejagat diisytihar di luar fungsi dan protataip setempat diisytiharkan di dlm fungsi. Skop panggilan protataip sejagat, boleh dipanggil dimana^2 fungsi di dlm aturcara. Skop panggilan protataip setempat, hanya boleh dipanggil dlm fungsi yg mengisytiharkannya.

- Perhatikan rangka aturcara mencari luas di ms sebelah:
  Dlm fungsi **main** prototaip fungsi **luas_segi3** diisytiharkan setempat & dlm fungsi **luas_segi4** prototaip fungsi **luas_bulat** juga diisytiharkan setempat, fungsi^2 ini hanya boleh dipanggil didlm fungi ini shj.
  Protataip fungsi **menu & luas_segi4** diisytihar sejagat, fungsi^2 boleh dipanggil oleh semua fungsi dlm aturcara tersebut.

7.7 Skop Prototaip Fungsi (samb...)

Contoh rangka aturcara mencari luas dgn prototaip fungsi yg pelbagai skop.

```
#include <stdio.h>

void menu (void);
float luas_segi4 (void);


void main()
{
     float luas_segi3 (float, float);
          :
     menu();
     luas = luas_segi4 ();
     luas = luas_segi3 (tpk, tg);
          :
}

float luas_segi4 (void)
{
     float luas_bulat (float jejari);
     :
     luas = luas_bulat (j);
     :

}

float luas_segi3 (float tapak, float tinggi)
{
     :
}

float luas_bulat (float jejari)
{
     menu();
     :
}

void menu (void)
{
     :
}
```

prototaip fungsi diisytiharkan sejagat

prototaip fungsi diisytiharkan setempat

Latihan:

1. Berdasarkan aturcara dibawah, jejakkan aturcara tersebut & berikan hasil output yg akan dicetak di skrin.

```
#include <stdio.h>

void fungsi_1(void);
int  fungsi_2(void);
void fungsi_3(int *a, int b);
int  fungsi_4(int a);
void fungsi_5(int *a, int *b,
              int *c);
int ikan = 10, ayam = 5;

void main ()
{
   int itek = 11,
   kambing = 4, kucing = 2;

   fungsi_1();
   itek++;
   ikan = ++kambing + 1;
   ikan++; ayam++;
   itek = itek + fungsi_2();
   fungsi_3(&kambing, itek);
   fungsi_5(&itek, &kambing,
            &kucing);
   ayam = ayam++ + ikan;
   ikan = ikan + itek++ -
          ++kucing;

   printf("\n Dalam fungsi
    main nilai itek=%d
    kambing=%d kucing=%d",
    itek, kambing, kucing);
   printf("\n Dalam fungsi
    main nilai global ikan=%d
    ayam = %d", ikan, ayam);
}

void fungsi_1(void)
{  int ayam = 0;
   ikan +=2;  ayam +=2;
   printf("\n Dalam fungsi_1
   nilai ikan=%d ayam=%d",
   ikan, ayam);
}
```

```
int  fungsi_2(void)
{  int kucing;
   kucing = ayam * 2;
   printf("\n Dalam fungsi_2
    nilai kucing = %d",
    kucing);
    return kucing;
}

void fungsi_3(int *badak, int
                gajah)
{  *badak = 2 * gajah;
   gajah  = 2 + (*badak);
   printf("\n Dalam fungsi_3
    nilai badak=%d gajah
    = %d", *badak, gajah);
}

int  fungsi_4(int keldai)
{  int kancil;
   kancil = ++keldai +
            ikan + ayam;
  printf("\n Dalam fungsi_4
   nilai keldai=%d kancil=
   %d",keldai, kancil);
   return (kancil);
}

void fungsi_5(int *rama2, int
          *lebah, int *kupu2)
{
   int kuda=6;
   *lebah = *lebah -
            fungsi_4(kuda);
   *rama2 = *lebah - *rama2 +
            *kupu2;
   *kupu2 = kuda +
            (*kupu2)++;
   printf("\n Dalam fungsi_5
   nilai rama2=%d lebah=%d
   kupu2 = %d", *rama2,
   *lebah, *kupu2);
}
```