Modularitas Program IF2030/Algoritma & Struktur Data

Tim Pengajar IF2030

Tujuan

- Mahasiswa memahami kegunaan modul program
- Mahasiswa memahami konsep reusability dalam pembuatan program
- Mahasiswa memahami pembuatan program C dengan beberapa modul program (dalam beberapa file)
- Mahasiswa dapat mengimplementasikan program dengan memakai modul program dalam bahasa C

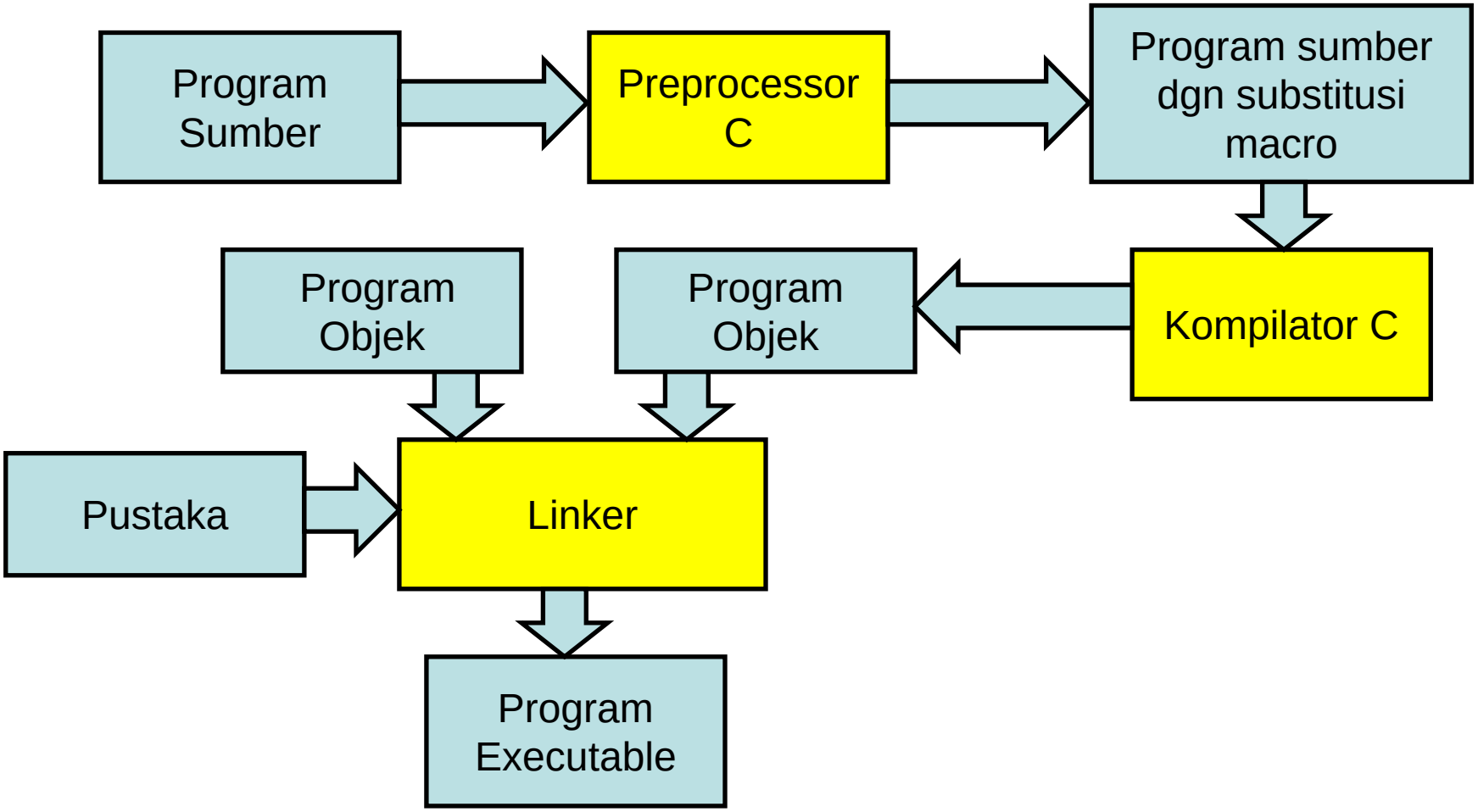
Modularitas Program

- Sebuah program yang “utuh”, seringkali terdiri dari beberapa modul program
- Modul program dapat mewakili :
  - Sekumpulan rutin sejenis
  - ADT (Abstract Data Type) : definisi type dan primitifnya
  - Mesin : definisi state variable dari mesin dan primitifnya

Pembuatan program

- Program terdiri dari :
  - Satu program utama (main program)
  - Beberapa modul yang lain
- Program yang dibagi-bagi menjadi beberapa file seharusnya dapat dikompilasi terpisah :
  - setiap modul membentuk sebuah **object code**
- Pembuatan sebuah **executable code** dilakukan dgn melakukan **link** terhadap sejumlah object code yang sudah dikompilasi :
  - Penghematan waktu dan duplikasi usaha (reusability)

Pemrosesan Program Sumber dalam Bahasa C



Modul Program dalam C(1/3)

- Program utuh terdiri dari 3 kelompok file
- File header dengan nama xxx.h :
  - Untuk setiap type dan primitifnya, ada sebuah file header
  - Contoh : untuk ADT Jam dan ADT Date ada 2 buah file header, yaitu Jam.h dan Date.h
- Fungsi selektor Get dan Set dapat digantikan dengan *macro* berparameter. Misalnya untuk Selektor terhadap Hour(P) , Minute(J), dan Second(J) dituliskan sebagai:

```
#define Hour(J) (J).HH
#define Minute(J) (J).MM
#define Second(J) (J).SS
```

Modul Program dalam C(2/3)

- File body dengan nama xxx.c :
  - Berisi realisasi dari prototype yang didefinisikan dalam file header
  - Akan ada sebuah xxx.c untuk setiap xxx.h
  - Contoh : untuk file header Jam.h dan Date.h akan ada file body Jam.c dan Date.c

Modul Program dalam C(3/3)

- File main (driver) :
  - Berisi program utama dan prosedur/fungsi lain yang hanya dibutuhkan oleh main
  - Misalnya diberi nama main.c
- Program Utuh akan terdiri dari sebuah main.c, sebuah xxx.h dan xxx.c

File Header

```
/* File : xxx.h                                                  */
/* Deskripsi : keterangan isi file header                        */
/* Isi : deklarasi konstanta, type dan prototype                 */
/* File header TIDAK BOLEH mengandung deklarasi variabel!  */

#ifndef xxx_h
#define xxx_h
/* Bagian I : berisi deklarasi konstanta */

/* Bagian II : berisi deklarasi type */

/* Bagian III : berisi deklarasi prototype prosedur & fungsi */
/*              yg merupakan primitif type tsb */
/* Kelompokkan fungsi dan prosedur sesuai standar di kelas */
/* Mis.: konstruktor, selektor, predikat, operator relasional*/
/* operator aritmatika, operator lain, dsb */

#endif
```

File Body

```
/* File : xxx.c                                                    */
/* Deskripsi : keterangan isi file body                            */
/* Isi : realisasi/ kode program dari semua prototype              */
/*       yg didefinisikan pada xxx.h                               */
/* Untuk sebuah mesin akan mengandung deklarasi variabel           */
/* state dari mesin tsb                                            */

#include “xxx.h”

/* Realisasi kode program, sesuai urutan pada xxx.h */

/* Copy dari xxx.h, kemudian edit */
```

File Main Program

```
/* File : main.c                                                    */
/* Deskripsi: program utama & semua nama lokal thd persoalan */

#include “xxx.h”
/* include file lain yg diperlukan */
/* Bagian I : berisi kamus GLOBAL dan prototype */
/* deklarasi semua nama dan prosedur/fungsi global */

/* Bagian II : program utama */
int main() {
/* Kamus lokal terhadap main */

/* Algoritma */

  return 0;
}

/* Bagian III : berisi realisasi kode program yang merupakan */
/* BODY dari semua prototype yg didefinisikan pada file ini  */
/* yaitu pada bagian I, dengan urut-urutan yang sama */
/* Copy prototype, kemudian edit */
```

Contoh

- Diktat “Contoh Program Kecil Bahasa C”
  - Program dalam beberapa modul

Penyimpanan Modul Program

- Untuk setiap modul xxx.h dan xxx.c dibuat main program untuk mentest setiap fungsi/ prosedur yg dibuat
- Main program tsb disebut “driver” atau “teststub”
- Setiap paket yg terdiri dari xxx.h, xxx.c, main.c, dan hasil test disimpan dalam satu direktori

```
/* File : jam.h */
/* Spesifikasi ADT Jam */
#ifndef jam_H
#define jam_H

#include "boolean.h"

/* Notasi Akses : Selektor */
#define Hour(J) (J).HH
#define Minute(J) (J).MM
#define Second(J) (J).SS

/* Type Jam */
typedef struct { int HH;
                 int MM;
                 int SS;
               } Jam;

/*** Konstruktor ***/
Jam MakeJam (int H, int M, int S);
/* Membentuk Jam dari H, M, S yang valid */

/*** Validator jam ***/
int IsJValid (int H, int M, int S);
/* Mengirim true jika H, M, S dapat membentuk Jam yang valid */
```


```
/* Lanjutan jam.h */

/*** Baca & Tulis Jam ***/
void BacaJam (Jam * J);
/* I.S. : J sembarang */
/* F.S. : J terdefinisi dan merupakan Jam valid */
/* Proses : mengulang baca komponen H, M, S sehingga membentuk J yang
   valid */
void TulisJam (Jam J);
/* I.S. : J terdefinisi */
/* F.S. : J ditulis ke layar dengan format HH:MM:SS */
/* Proses : Menulis ke layar */

/*** Konversi terhadap type Jam ***/
long int JamToDetik (Jam J);
/* Konversi Jam menjadi detik */
Jam DetikToJam(long int N);
/* Konversi detik ke Jam */

#endif
```

```
/* File : jam.c */
/* Body ADT Jam */
#include "jam.h"

/*** Konstruktor Jam ***/
Jam MakeJam (int H, int M, int S)
/* Membentuk jam dari H, M, S yang valid */
{  /* Kamus Lokal */
   Jam J1;
   /* Algoritma */
   Hour(J1) = H;
   Minute(J1) = M;
   Second(J1) = S;
   return J1;
}

/*** Validator Jam ***/
int IsJValid (int H, int M, int S)
/* Mengirim true jika H, M, S dapat membentuk Jam yang valid */
{   /* Algoritma */
    return ((H >= 0 && H <= 23) &&
            (M >= 0 && M <= 59) &&
            (S >= 0 && S <= 59));
}
```

```
/* Lanjutan: jam.c */

/*** Baca & Tulis Jam ***/
void BacaJam (Jam * J)
/* I.S. : J sembarang */
/* F.S. : J terdefinisi dan merupakan Jam valid */
/* Proses : mengulang baca komponen H, M, S sehingga membentuk J yang valid
*/
{  /* Kamus Lokal */
   int H, M, S;

   /* Algoritma */
   do {
        printf("Masukkan jam   : "); scanf("%d", &H);
        printf("Masukkan menit : "); scanf("%d", &M);
        printf("Masukkan detik : "); scanf("%d", &S);
   } while(!IsJValid(H,M,S));
   (*J) = MakeJam(H, M, S);
}

void TulisJam (Jam J)
/* I.S. : J terdefinisi */
/* F.S. : J ditulis ke layar dengan format HH:MM:SS */
/* Proses : Menulis ke layar */
{
        printf("%d:%d:%d\n", Hour(J), Minute(J), Second(J));
}
```

```
/* Lanjutan : jam.c */

/*** Konversi terhadap type Jam ***/
long int JamToDetik(Jam J)
/* Konversi Jam menjadi detik */
{   /* Algoritma */
    return (((long int)3600 * (long int)Hour(J))
            +((long int)60 * (long int)Minute(J))
            +(long int)Second(J));
}
Jam DetikToJam(long int N)
/* Konversi detik ke Jam */
{   /* Kamus Lokal */
    long int sisa;
    Jam JOut;

    /* Algoritma */
    setHour(&JOut,(int)(N/(long int)3600));
    sisa = N % (long int)3600;
    setMinute(&JOut,(int)(sisa/(long int)60));
    sisa = sisa % (long int)60;
    setSecond(&JOut,(int)sisa);
    return JOut;
}
```

```
/* File : mjam.c */
/* Driver ADT jam */

#include <stdio.h>
#include "jam.h“

int main()
{    /* KAMUS */
     Jam J1, J2;
     /* ALGORITMA */
     BacaJam(&J1);
     printf("Jam sekarang \n");
     TulisJam(J1);
     printf("Konversi menjadi detik %d\n",JamToDetik(J1));
     printf("Konversi 5000 detik menjadi Jam\n");
     J2 = DetikToJam(5000);
     TulisJam(J2);
     return 0;
}
```

PR

- Kerjakan Modul Pra Praktikum P-01 hingga P-04:
  - Jam (P-01)
  - Point (P-02)
  - Garis (P-03)
  - Tanggal (P-04)

Program Besar dengan Variabel Global

- Untuk program besar, perlu menjaga konsistensi deklarasi objek
- Semua variabel global dalam program didefinisikan dalam suatu file header
- File header di-include dalam semua modul yang memerlukan.
- Perubahan pada suatu variabel cukup dilakukan dengan mengubah file header

File Global.h

```
/* File : global.h                                          */

#ifndef GLOBAL_H
#define GLOBAL_H


/* definisi dan deklarasi variabel-varibel global */

extern int x;

...

#endif
```

File Body Modul Program


```
/* File : Modul1.c                                                    */
/* Contoh body modul yang memanfaatkan global.h */
/* Modul1.c adalah modul yang akan dipakai oleh main program */

#include "Modul1.h"
#include "global.h"

int x; /* x dikenali di modul1 */

/* Kode program di dalam Modul1.c */
```

File Main Program

```
/* File : main.c                                                    */
/* Contoh main program yang memanfaatkan global.h                   */

#include "global.h"

int main() {
   /* Kamus lokal terhadap main program */
   int x;  /* x dikenali di main program */

   /* Algoritma main program */

   return 0;
}
```