Struktur Dasar Program Prosedural (dalam Bahasa C++)

Tim Penyusun Materi PTI-B

KU1072/Pengenalan Teknologi Informasi B
Tahap Tahun Pertama Bersama
Institut Teknologi Bandung

Tujuan

- Subtopik
  - Input – Proses – Output dalam program
  - Deklarasi dan penggunaan variabel, type (dasar dan bentukan), konstanta, ekspresi (aritmatika, relasional, dan logika)
  - Input/output
  - Sekuens
  - Flowchart terkait
  - Contoh kasus
- Outcome
  - Memahami makna dan penggunaan variable, type, konstanta, input/output, dan sekuens.
  - Memahami persoalan yang dapat dikonversi menjadi program sederhana dengan memanfaatkan variable, type, konstanta, ekspresi dasar, input/output, dan sekuens

C++

- **C++** merupakan bahasa pemrograman *general purpose* dan multi paradigma (prosedural, *object oriented*)
- Bahasa pemrograman yang sangat populer dan banyak digunakan
- Dikembangkan oleh Bjarne Stroustrup mulai tahun 1979 di Bell Labs
- Merupakan pengembangan dari Bahasa C (prosedural murni) dengan penambahan konsep, *object-orientation*
- Dalam kuliah ini, hanya akan menggunakan paradigma prosedural
- Merupakan bahasa yang case sensitive → perbedaan huruf besar dan kecil berpengaruh

Masukan – Proses – Luaran


```
Input (A)
Input (B)
```

```
A ← A + B
```

```
Output(A)
Output(B)
```

C++

```
cin >> A;
cin >> B;
```

```
A = A + B;
```

```
cout << A;
cout << B;
```

Apa hasil eksekusinya?

```
#include <iostream>
using namespace std;

int main () {
        //KAMUS
        int A;
        int B;

        //ALGORITMA
        cin >> A;
        cin >> B;

        A = A + B;

        cout << A << endl;
        cout << B << endl;
        return 0;
}
```

Apa hasil eksekusinya?

```
#include <iostream>
using namespace std;

int main () {
        //KAMUS
        string nama;

        //ALGORITMA
        cout << "Tuliskan namamu: " << endl;

        cin >> nama ;

        cout << "Namamu adalah : " << nama << endl;

        return 0;
}
```

Apa hasil eksekusinya?

```
#include <iostream>
using namespace std;

int main () {
        //KAMUS
        int A;
        int B;

        //ALGORITMA
        A = 10;
        B = 5;

        A = A + B;
        B = B - A;

        cout << A << endl;
        cout << B << endl;
        return 0;
}
```

Struktur Dasar Program

```
// Program Test
// Contoh struktur program prosedural dalam C++

#include <iostream>
using namespace std;

int main () {
        //KAMUS
        int A;
        int B;

        //ALGORITMA
        A = 10;
        B = 5;

        A = A + B;
        B = B - A;

        cout << A << endl;
        cout << B << endl;
        return 0;
}
```

Judul Program + spesifikasi, dituliskan dalam komentar

Bagian ini perlu di tambahkan sebagai standard pemrograman C++ di layar Console

KAMUS

ALGORITMA

- `iostream` adalah salah satu *header* file yang ada di C++. *Header* ini digunakan untuk fungsi input dan output yang ada di C++.

  Contoh fungsi input/output: `cin` dan `cout`

- `using namespace std` adalah perintah yang digunakan untuk mendeklarasikan/ memberitahukan kepada *compiler* C++ bahwa kita akan menggunakan semua fungsi/class/file yang terdapat dalam *namespace std*

Kamus:
Tipe Data, Variabel, Konstanta, Ekspresi

Kamus

- Kamus dipakai untuk mendeklarasi nama-nama yang digunakan dalam program
- Deklarasi nama yang didefinisikan pemrogram
  - type
  - variabel
  - konstanta
- Deklarasi BUKAN instruksi
- Contoh deklarasi [variabel]:

PASCAL

```
I           : integer;
JumlahUang  : real;
Titik       : Point;
```

C++

```
int I;
float JumlahUang
Point Titik;
```

Jenis Tipe Data

- Setiap data memiliki jenis yang berbeda-beda
  - Data **umur** seseorang berbeda dengan data **nama**
    - Data Umur dibentuk dari kumpulan angka
    - Data nama dibentuk dari serangkaian huruf
  - Untuk setiap jenis data juga memiliki rentang (range) yang berbeda
    - Data umur rentangnya antara 1 sampai 100 (bila diasumsikan bahwa umur seseorang tidak lebih dari 100).
    - Data nama rentangnya mulai dari 1 sampai 50 (bila di anggap nama tidak ada yang melebihi 50 huruf

Jenis-jenis Tipe Data

- Tipe data **primitif** atau tipe **dasar** (dalam C++)
  - Boolean (bool)
  - Integer (int)
  - Real (float)
  - Character (char)
  - String (string)
- Tipe data **turunan** atau **bentukan**
  - Dibentuk dari gabungan tipe dasar
  - Contoh
    - Tipe DataMahasiswa
      - Dibentuk dari
        - NIM: string
        - Nama: string
        - Umur: integer
        - Kota: string
    - Tipe Array
      - Dibentuk dari kumpulan integer, misalnya 10 data tentang umur

Contoh Tipe Data

Umur → Integer contoh: 25, 44, 35

Kota → String, contoh: “Jakarta”, “Bandung”

Nama → String, contoh: “Budi”, “Ali”

Suhu → Integer atau float, contoh: 37.5 , 100

Luas → Integer atau float, contoh: 400, 43.5

BeratBadan → Integer atau float, contoh: 60.5, 75

NIM → Integer atau string?, contoh: 15812001

Contoh deklarasi tipe bentukan/komposit/struct

```
// Kamus
typedef struct {
    int x;
    int y;
} Point;
typedef struct {
    string NIM;
    string Nama;
    int Umur;
    string Kota;
} DataMahasiswa;
```

Variabel

- Variabel menyimpan nilai ber-”tipe data” sesuai dengan deklarasi
- Variabel :
  - deklarasi (supaya nama dikenal),
  - inisialisasi nilai (siap dimanipulasi)
- Contoh
  - Deklarasi variabel

```
int i;
float A;
```

  - Inisialisasi variabel

```
i  =  100;
```

    - Artinya variabel i di isi dengan nilai 100

```
A = 8.25;
```

    - Artinya variabel A diisi dengan nilai real 8.25
- Operasi terhadap variabel sangat tergantung dari tipe datanya.

Operasi pada nilai suatu tipe data

- Operasi perhitungan akan memerlukan operator seperti **“+”, “-”, “*” dan “/”** (tambah, kurang, kali dan bagi) untuk melakukan kalkulasi
- Operasi “+” pada tipe data bukan numerik memiliki arti yang berbeda
  - Contoh: “Halo “ + “Apa kabar “ → “Halo Apa kabar”
- Tidak semua operator dapat digunakan untuk tipe data numerik.
  - Contoh: “Halo “ * “Apa kabar “ → 

Operasi tipe dasar

- int : * / + - % < > <= >= == !=
- bool : && || ! !=
- float : * / + - < > <= >= !=
- char : == !=

Membuat Nama Variabel yang benar dan "baik"

- Nama variabel harus dimulai dengan huruf dan dapat diikuti dengan huruf lagi dan angka
  - Tidak boleh ada tanda baca
- Dalam nama variabel tidak boleh dipisahkan oleh spasi
- Cari nama variabel yang bisa dimengerti
  - Agar tidak membingungkan
- C++ adalah bahasa yang case sensitive

  → Kesalahan penulisan huruf besar dan kecil menyebabkan error

Contoh yang benar

Total

Jumlah

A

Contoh yang salah

3Roda

Jumlah,total

8

Benar atau salah?

Kar2string ✓

Total45 ✓

-angka ✗

zzzz ✓

SuperDayaGuna ✓

Lum4588abc ✓

Konstanta

- Berbeda dengan Variable, suatu konstanta **tidak boleh diubah** nilainya
- Contoh

  const float PI = 3.1415

  const int nilai = 1000

- Pemakaian yang salah

  PI = 44.5

  nilai = 5000

Keduanya salah karena PI dan nilai sudah ditandai sebagai konstanta dengan nilai 3.14159 dan 1000 jadi nilainya tidak boleh diubah

ALGORITMA

Algoritma

- Adalah bagian program dalam bentuk teks algoritmik yang berisi instruksi atau pemanggilan aksi
- Teks algoritmik tsb. dapat berupa:
  - Perintah dasar: Input/Output, assignment
  - Perintah perintah yang berurutan
  - Analisis kasus (jika-maka)
  - Pengulangan

Perintah-perintah dasar

- Pemberian nilai (assignment) sesuai dengan type ke suatu variabel
- Pembandingan (kesamaan, ketidak-samaan)
- Operasi relasional lain (lebih besar, lebih kecil,….)
- Operasi aritmetika (khusus untuk nilai numerik)

Nilai, Input+Output

- Nilai atau harga: suatu besaran bertype yang telah dikenal
- Pengisian nilai:
  - Pemberian nilai langsung atau disebut sebagai ***assignment***
    - *Contoh:* *A = 10;*
  - Dibaca dari piranti masukan
    - Contoh: cin >> A;

Assignment (=)

- Ruas kiri = Ruas Kanan ;
- Ruas kiri harus variable
- Ruas kanan harus <ekspresi>
- Ekspresi :
  - "rumus perhitungan"
  - Contoh:

Luas = **panjang * lebar ;** ← **Ekspresi**

Ekspresi

- Ekspresi Aritmatika

  A + B

  x + 2 * y

  P – 2 * Q + R/S

- Ekspresi Relasional (pembandingan)

  A < B

  X == Y

  Total >= nilai

- Ekspresi Logika

  A && B

  C || B

Komentar

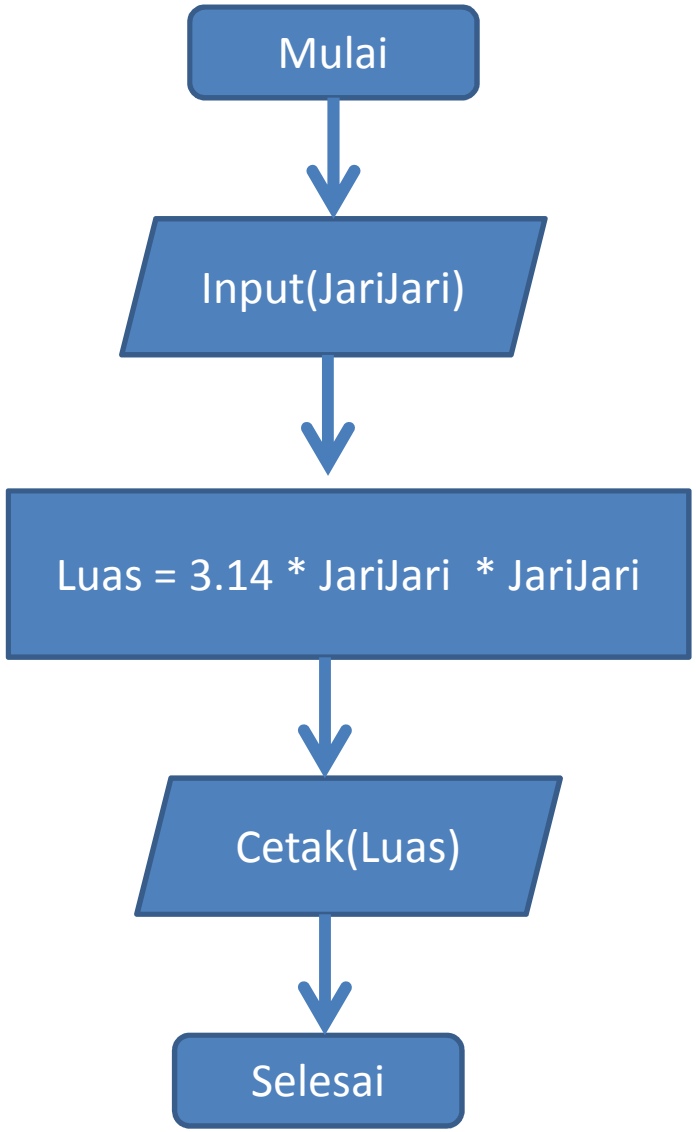
- Dalam bahasa pemrograman komentar adalah bagian program yang tidak dieksekusi
  - Bagian ini hanya digunakan untuk memberikan penjelasan suatu langkah, rumus ataupun bisa hanya berupa keterangan
- Dalam C++, komentar dituliskan sebagai:
  - Antara /* dan */

```
/* ini komentar */
```

  - Diawali dengan //

```
// ini komentar
```

Flowchart Menghitung Luas Lingkaran



Program Hitung Luas Lingkaran

```
// Program HitungLuasLingkaran
// Menghitung luas lingkaran berdasarkan jari-jari
#include <iostream>
using namespace std;

int main()
{
   // KAMUS
   float JariJari;
   float Luas;

   // ALGORITMA
   cin >> JariJari;
   Luas = 3.1415 * JariJari * JariJari;
   cout << Luas << endl;
   return 0;
}
```

Pendefinisian variabel

Algoritma

Program Hitung Luas Lingkaran

```
// Program HitungLuasLingkaran
// Menghitung luas lingkaran berdasarkan jari-jari
#include <iostream>
using namespace std;

int main()
{  // KAMUS
   const float PI = 3.1415;
   float JariJari;
   float Luas;

   // ALGORITMA
   cin >> JariJari;
   Luas = 3.1415 * JariJari * JariJari;
   cout << Luas << endl;
   return 0;
}
```

Pendefinisian konstanta

Pendefinisian variabel

Algoritma

Latihan *(di kertas masing-masing dan gunakan pensil)*

- Buat program Hitung Luas Segitiga
- Buat program menghitung rata-rata dari tinggi badan 5 anak
  - Program akan menerima masukan data tinggi badan untuk 5 orang anak
  - Kemudian program akan menghitung tinggi rata-rata dari ke lima anak tersebut

Program Menghitung Luas Segitiga

```
// Program HitungLuasSegitiga
// Menghitung luas segitiga berdasarkan alas dan tingginya
#include <iostream>
using namespace std;

int main()
{  // KAMUS
   int alas;
   int tinggi;
   int Luas;

   // ALGORITMA
   cin >> alas;
   cin >> tinggi;

   Luas = (alas * tinggi) / 2;

   cout << Luas << endl;
   return 0;
}
```

Program Menghitung Tinggi Rata-Rata

```
// Program TinggiRataRata
// Hitung tinggi rata-rata 5 anak
#include <iostream>
using namespace std;
int main()
{  // KAMUS
   int tinggi1, tinggi2, tinggi3, tinggi4, tinggi5;
   float ratarata;
   // ALGORITMA
   cin >> tinggi1;
   cin >> tinggi2;
   cin >> tinggi3;
   cin >> tinggi4;
   cin >> tinggi5;

   ratarata = (tinggi1 + tinggi2 + tinggi3 +tinggi4 + tinggi5)/5;
   cout << ratarata << endl;
   return 0;
}
```

Definisi Aksi Sekuensial

- Aksi sekuensial
  - sederetan instruksi primitif dan/atau aksi yang akan dilaksanakan (dieksekusi) oleh komputer berdasarkan urutan penulisannya
  - Setiap aksi akan mengubah status dari program
    - Jadi setiap aksi sekuensial harus ada awal dan akhir.
    - atau dengan kata lain suatu program harus dimulai dan suatu ketika harus berakhir
      - Program yang tidak pernah berhenti adalah program yang salah atau error

Penulisan untuk Instruksi Sekuensial

- Instruksi ditulis terurut sesuai penulisan per baris
- Setiap instruksi selalu diakhiri dengan tanda titik koma
  - Jadi dalam satu baru dapat terdiri dari lebih dari instruksi.

Contoh aksi sekuensial

```
/* contoh aksi sekuensial per baris */

int main()
{  /* Kamus */
   int i;
   float x;

   /* Algoritma */
   cin >> i;
   x = 100.75;

   cout << x << endl;
   cout << i * 2 << endl;

   return 0;
}
```

```
/* contoh aksi sekuensial dg titik koma */


int main()
{  /* Kamus */
   int i;
   float x;

   /* Algoritma */
   cin >> i ;  x = 100.75;


   cout << x << endl; cout << i * 2 << endl;


   return 0;
}
```

Contoh aksi sekuensial

```
/* contoh aksi sekuensial per
baris */

int main()
{  /* Kamus */
   int i;
   float x;

   /* Algoritma */
   cin >> i;
   x = 100.75;

   cout << x << endl;
   cout << i * 2 << e

   return 0;
}
```

```
/* contoh aksi sekuensial dg titik koma */


int main()
{  /* Kamus */
   int i;
   float x;

   /* Algoritma */
   cin >> i ;  x = 100.75;


t << i * 2 << endl;
```

Perhatikan bahwa keduanya memliki urutan eksekusi yang sama dan juga hasil eksekusi yang identik.
Perbedaannya hanyalah di cara penulisannya. **Mana yang lebih baik penulisannya?**

- Perhatikan bahwa:
  - ada program yang akan berubah jika urutan baris instruksinya berubah
  - dan ada juga program yang tidak berubah jika urutan baris instruksinya berubah

Pengubahan urutan eksekusi yang tidak merubah hasil eksekusi

```
/* contoh aksi sekuensial per
baris */

int main()
{
   /* Kamus */
   int i;
   float x;

   /* Algoritma */

   cin >> i;
   x = 100.75;

   cout << x << endl;
   cout << i * 2 << endl;
   return 0;
}
```

```
/* contoh aksi sekuensial per
baris */

int main()
{
   /* Kamus */
   float x;
   int i;

   /* Algoritma */

   x = 100.75;
   cin >> i;

   cout << x << endl;
   cout << i * 2 << endl;
   return 0;
}
```

Pengubahan urutan eksekusi yang merubah hasil eksekusi

```
/* contoh aksi sekuensial per
baris */

int main()
{
   /* Kamus */
   int i;
   float x;

   /* Algoritma */

   cin >> i;
   x = 100.75;

   cout << x << endl;
   cout << i * 2 << endl;
   return 0;
}
```

```
/* contoh aksi sekuensial per
baris */

int main()
{
   /* Kamus */
   float x;
   int i;

   /* Algoritma */

   x = 100.75;
   cin >> i;

   cout << i * 2 << endl;
   cout << x << endl;
   return 0;
}
```

Buatlah program untuk menghitung jumlah dari dua buah pecahan

- Spesifikasi program
  - Program menerima masukan pecahan pertama berupa pembilang dan penyebut
  - Kemudian program menerima pecahan kedua
  - Lalu program akan melakukan penjumlahan
  - Kemudian menampilkan hasilnya berupa pembilang dan penyebut hasil penjumlahan

Langkah solusi

- Program menerima masukan pecahan pertama berupa pembilang dan penyebut

```
cin >> pembilang1;
cin >> penyebut1;
```

- Kemudian program menerima pecahan kedua

```
cin >> pembilang2;
cin >> penyebut2;
```

Langkah Solusi 2

- Lalu program akan melakukan penjumlahan

$$\frac{A}{B}+\frac{C}{D}=\frac{AD+BC}{B*D}$$

```
pembilang3 =
    pembilang1 * penyebut2 + pembilang2 * penyebut1;
penyebut3  = penyebut1 * penyebut2;
```

- Kemudian menampilkan hasilnya berupa pembilang dan penyebut hasil penjumlahan

```
cout << pembilang3;
cout << penyebut3;
```

```
// Program JumlahPecahan
// Menghitung pembilang dan penyebut pecahan dari penjumlahan
// dua buah pecahan yang diketahui pembilang dan penyebutnya
#include <iostream>
using namespace std;
int main()
{  // KAMUS
   int pembilang1, pembilang2, pembilang3;
   int penyebut1, penyebut2, penyebut3;
   // ALGORITMA
   cin >> pembilang1;
   cin >> penyebut1;

   cin >> pembilang2;
   cin >> penyebut2;

   pembilang3 =  pembilang1 * penyebut2 + pembilang2 * penyebut1;
   penyebut3  =  penyebut2 * penyebut1;

   cout << pembilang3 << endl;
   cout << penyebut3 << endl;
   return 0;
}
```

Buatlah program untuk menghitung perkalian dari dua buah pecahan

- Spesifikasi program
  - Program menerima masukan pecahan pertama berupa pembilang dan penyebut
  - Kemudian program menerima pecahan kedua
  - Lalu program akan melakukan perkalian
  - Kemudian menampilkan hasilnya berupa pembilang dan penyebut hasil penjumlahan

```
// Program KaliPecahan
// Menghitung pembilang dan penyebut pecahan dari perkalian
// dua buah pecahan yang diketahui pembilang dan penyebutnya
#include <iostream>
using namespace std;
int main()
{  // KAMUS
   int pembilang1, pembilang2, pembilang3;
   int penyebut1, penyebut2, penyebut3;

   // ALGORITMA

   cin >> pembilang1;
   cin >> penyebut1;

   cin >> pembilang2;
   cin >> penyebut2;

   pembilang3 = pembilang1 * pembilang2;
   penyebut3 = penyebut1 * penyebut2;

   cout << pembilang3;
   cout << penyebut3;
   return 0;
}
```

Buatlah program untuk Jarak dari kecepatan dan waktu suatu kendaraan

- Spesifikasi program
  - Program menerima masukan kecepatan
  - Kemudian program menerima masukan waktu
  - Lalu program akan melakukan perhitungan
  - Kemudian menampilkan hasilnya berupa perhitungan jarak

Analisa Masalah (dekomposisi Masalah)

- Program menerima masukan kecepatan
- Kemudian program menerima masukan waktu
- Lalu program akan melakukan perhitungan
- Kemudian menampilkan hasilnya berupa perhitungan jarak

Solusi

- Program menerima masukan kecepatan

  cin >> v;

- Kemudian program menerima masukan waktu

  cin >> t;

- Lalu program akan melakukan perhitungan

  Jarak = v * t;

- Kemudian menampilkan hasilnya berupa perhitungan jarak

  cout << Jarak << endl;

Solusi 1

```
// Program HitungJarak
// Menghitung jarak berdasarkan masukan
// kecepatan (v) dan waktu (t)
#include <iostream>
using namespace std;

int main()
{  // KAMUS
   int jarak, v, t;

   // ALGORITMA

   cin >> v;
   cin >> t;

   jarak = v * t;

   cout << jarak << endl;
   return 0;
}
```

Solusi 2

```
// Program HitungJarak
// Menghitung jarak berdasarkan masukan
// kecepatan (v) dan waktu (t)
#include <iostream>
using namespace std;

int main()
{  // KAMUS
   int jarak, v, t;

   // ALGORITMA
   cout << “kecepatan = “;
   cin >> v;
   cout << “waktu = “;
   cin >> t;

   jarak = v * t;

   cout << “Jarak = “ << jarak << endl;
   return 0;
}
```

Buatlah program yang memeriksa perbedaan dua buah jam

- Spesifikasi program
  - Program menerima masukan jam, menit, dan detik yang pertama
  - Kemudian menerima masukan jam, menit, dan detik yang kedua
  - Kemudian program menghitung selisih waktu
  - Selanjutnya menampilkan hasilnya berupa perhitungan selisihnya

Program yang memeriksa perbedaan dua buah jam

- Program menerima masukan jam, menit dan detik yang pertama

```
cin >> jam1;
cin >> menit1;
cin >> detik1;
```

- Kemudian menerima masukan jam menit dan detik yang kemudian

```
cin >> jam2;
cin >> menit2;
cin >> detik2;
```

- Kemudian program menghitung selisih waktu
  - Konversi dulu ke detik

```
totaldetik1 = jam1 * 3600 + menit1 * 60 + detik1;
totaldetik2 = jam2 * 3600 + menit2 * 60 + detik2;
```

  - Hitung selisihnya

```
detikSelisih = totaldetik2 - totaldetik1;
```

  - Hitung untuk jam, menit dan detik

```
jamHasil   = detikSelisih / 3600;
menitHasil = (detikSelisih % 3600) / 60;
detikHasil = detikSelisih % 60;
```

- Kemudian Cetak hasil
  - Contoh hasil:

```
Selisih = 5 jam 3 menit 4 detik
```

  - Jadi:

```
cout << “Selisih = “ << jamHasil << “ jam” <<
menitHasil << “ menit” << detikHasil << “detik” <<
endl;
```

```
// Program HitungJam
// Menghitung selisih jam
#include <iostream>
using namespace std;

int main()
{   // KAMUS
    int jam1, menit1, detik1;
    int jam2, menit2, detik2;
    int totaldetik1, totaldetik2;
    int detikSelisih;
    int jamHasil, menitHasil, detikHasil;

    // ALGORITMA
    // Masukan jam pertam
    cin >> jam1;
    cin >> menit1;
    cin >> detik1;
    // Masukan jam kedua
    cin >> jam2;
    cin >> menit2;
    cin >> detik2;

    // Perhitungan selisih jam
    totaldetik1 = jam1 * 3600 + menit1 * 60 + detik1;
    totaldetik2 = jam2 * 3600 + menit2 * 60 + detik2;

    detikSelisih = totaldetik2 - totaldetik1;

    jamHasil = detikSelisih / 3600;
    menitHasil = (detikSelisih % 60) / 60;
    detikHasil = detikSelisih % 60;

    // Penulisan selisih jam
    cout << "Selisih = " << jamHasil << " jam "
         << menitHasil << " menit " << detikHasil
         << " detik" << endl;

    return 0;
}
```

Solusi Menggunakan Type Bentukan

- Deklarasi type Jam:

```
typedef struct {
        int JJ;       // Bagian jam
        int MM;    // Bagian menit
        int DD;     // Bagian detik
 } Jam;
```

Program yang memeriksa perbedaan dua buah jam

- Program menerima masukan jam yang pertama (jam1) :

```
cin >> jam1.JJ;
cin >> jam1.MM;
cin >> jam1.DD;
```

- Kemudian menerima masukan jam yang kedua (jam2):

```
cin >> jam2.JJ;
cin >> jam2.MM;
cin >> jam2.DD;
```

- Kemudian program menghitung selisih waktu
  - Konversi dulu ke detik

```
totaldetik1 = jam1.JJ * 3600 + jam1.MM * 60 + jam1.DD;
totaldetik2 = jam2.JJ * 3600 + jam2.MM * 60 + jam2.DD;
```

  - Hitung selisihnya

```
detikSelisih = totaldetik2 - totaldetik1;
```

  - Hitung selisih dalam bentuk jam (jamHasil):

```
jamHasil.JJ   = detikSelisih / 3600;
jamHasil.MM = (detikSelisih % 3600) / 60;
jamHasil.DD = detikSelisih % 60;
```

- Kemudian Cetak hasil
  - Contoh hasil:

```
Selisih = 5 jam 3 menit 4 detik
```

  - Jadi:

```
cout << "Selisih = " << jamHasil.JJ << " jam" <<
jamHasil.MM << " menit" << jamHasil.DD << "detik"
<< endl;
```

```
// Program HitungJam
// Menghitung selisih jam
#include <iostream>
using namespace std;

typedef struct {
        int JJ, MM, DD;
} Jam;

int main()
{   // KAMUS
    Jam jam1, jam2;
    int totaldetik1, totaldetik2;
    int detikSelisih;
    Jam jamHasil;

    // ALGORITMA
    // Masukan jam pertama
    cin >> jam1.JJ;
    cin >> jam1.MM;
    cin >> jam1.DD;
    // Masukan jam kedua
    cin >> jam2.JJ;
    cin >> jam2.MM;
    cin >> jam2.DD;

    // Perhitungan selisih jam
    totaldetik1 = jam1.JJ * 3600 + jam1.MM * 60 + jam1.DD;
    totaldetik2 = jam2.JJ * 3600 + jam2.MM * 60 + jam2.DD;

    detikSelisih = totaldetik2 - totaldetik1;

    jamHasil.JJ = detikSelisih / 3600;
    jamHasil.MM = (detikSelisih % 60) / 60;
    jamHasil.DD = detikSelisih % 60;

    // Penulisan selisih jam
    cout << "Selisih = " << jamHasil.JJ << " jam "
         << jamHasil.MM << " menit " << jamHasil.DD
         << " detik" << endl;

    return 0;
}
```

Terima Kasih