IPB University
Bogor Indonesia

KOM120C -- BAHASA PEMROGRAMAN

Paradigma Pemrograman

- Pengantar Pemrograman
- Paradigma Pemrograman
- Pemrograman C++

Tim Pengajar Bahasa Pemrograman IPB University

BAHASA PEMROGRAMAN

- Bahasa pemrograman adalah suatu sistem notasi untuk menuliskan tugas komputasi yang harus dilaksanakan oleh mesin, dan bentuknya dapat dibaca serta dipahami oleh manusia.
- Beberapa bahasa komputer dirancang untuk memfasilitasi operasi-operasi tertentu, misalnya komputasi numerik, manipulasi teks, I/O, etc.
- Pada umumnya, bahasa pemrograman komputer biasanya dirancang dengan menggunakan paradigma pemrograman tertentu. Artinya mengikuti aliran atau genre tertentu.

PRINSIP PERANCANGAN BAHASA PEMROGRAMAN

Prinsip perancangan bahasa pemrograman:
(1) Sintaks, (2) Nama dan Tipe, (3) Semantik.

- Sintaks menjelaskan bagaimana struktur program yang benar. Struktur bahasa pemrograman modern didefinisikan menggunakan bahasa formal yang disebut context-free-grammar.
- Nama dan Tipe menunjukkan bagaimana aturan penamaan entitas (variabel, fungsi, class, parameter, dsb).
- Semantik, arti dari program. Ketika program dijalankan, efek tiap instruksi didefinisikan oleh semantik dari bahasa.

DESAIN BAHASA PEMROGRAMAN

Desain bahasa pemrograman memperhatikan 3 hal:

- Architecture. Bahasa pemrograman dirancang untuk komputer: *well-match* atau tidak dengan arsitektur komputer yang ada?
- Technical Setting, memperhatikan sistem operasi, IDE (Integrated Development Environment), network, dan referensi lingkungan lainnya.
- Standards: ANSI (American National Standards Institute), atau ISO (International Standards Organization). Contoh: ISO Pascal (1990), ANSI/ISO C++ (2003), dsb.

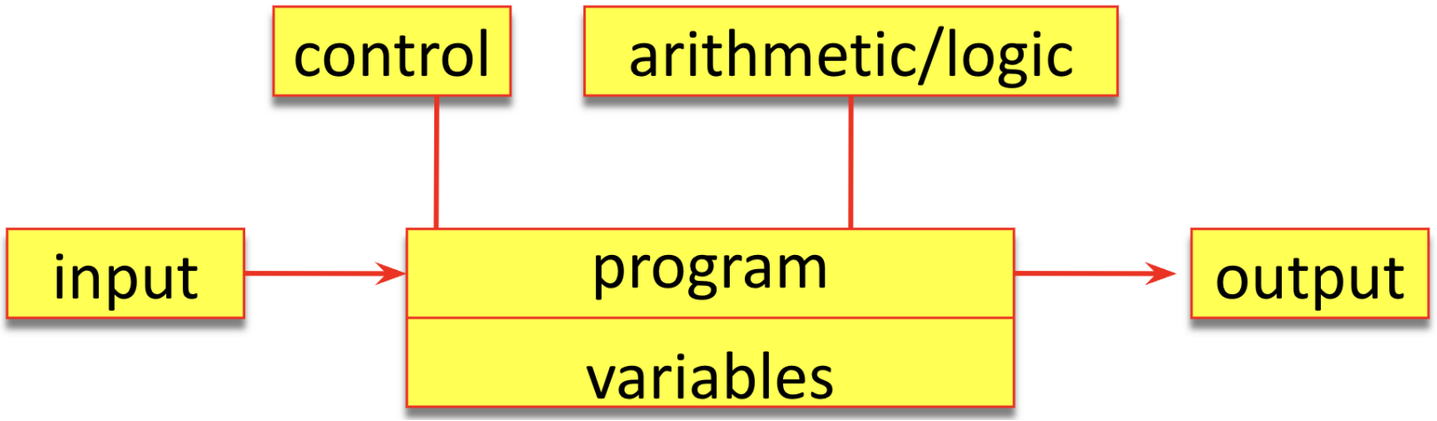
PARADIGMA BAHASA PEMROGRAMAN

Paradigma pemrograman adalah bentuk pemecahan masalah mengikuti aliran atau genre tertentu dari program dan bahasa.

Imperative/ Algorithmic	Declarative		Object-Oriented
	Functional Programming	Logic Programming	
Algor Cobol PL/1 Ada C Pascal Modula-3	Lisp Haskell ML Miranda APL	Prolog	SmallTalk Simula C++ Java

Imperative/Algorithmic

- Paradigma paling tua, didasari oleh model komputasi klasik von Neumann-Eckert : INPUT → PROSES → OUTPUT.
- Program dan variabel disimpan bersama.
- Program terdiri dari instruksi yang membentuk perhitungan, *asignment*, input, output, dan kontrol.



Serba Fungsi

Mulai dikembangkan tahun 1960an, dimotivasi oleh peneliti bidang artificial intelligence, symbolic computation, theorem proving, rule-based system, dan NLP.

LISP

List programming

Functional Programming

Struktur

Memodelkan masalah komputasi sebagai suatu fungsi matematika, yang mempunyai input (domain) dan hasil atau output (range).

Buat program menentukan bilangan terkecil dari tiga bilangan.

```
> (min 6 4 6)
4
```

```
#lang scheme
(define (min a b c)
  ( if (and (< a b) (< a c))
       a
       ( if (< b c) b c )
  )
)
```

Logic Programming

Buat program menghitung banyaknya elemen list yang nilainya kurang dari suatu nilai tertentu

```
?- hitung(10, [1,2,15,7,25,10], X).
X=3
```

```
hitung(A,[H|T], X) :-
    H<A, !, hitung(A,T,Y), X is X+1.
hitung(A,[_|T], X) :- hitung(A, T, X).
```

- **Pemrograman deklaratif**, mendeklarasikan tujuan komputasi, bukan menyusun algoritme secara detil. Disebut juga **rule-based programming**.
- LP dirancang untuk mendeskripsikan properti dari suatu obyek. Hubungan antar obyek dinyatakan dengan aturan if-then (jika-maka).

- LP memiliki mekanisme built-in untuk menarik kesimpulan (*inference*) berdasarkan deskripsi properti obyek tersebut.
- Contoh Aplikasi:
  - Artificial intelligence, misalnya MYCIN
  - Database information retrieval, misalnya SQL

Pemrograman Berorientasi Objek (PBO)

Metode pemrograman yang berorientasikan kepada objek, dimana semua data dan fungsi dalam metode ini didefinisikan ke dalam kelas-kelas atau objek-objek agar bisa saling bekerjasama dalam memecahkan masalah.

Bandingkan dengan pemrograman terstruktur (*algorithmic*) seperti yang dilakukan sebelumnya dalam KOM120B !

Dibutuhkan pemahaman yang dalam tentang objek dalam pemecahan persoalan → termasuk pemrograman tingkat tinggi

Pengantar ke OOP

Pemrograman C++

Baca di sini:
https://cplusplus.com/doc/turorial

C++ as a Better C

Program sederhana C++

```
#include < iostream >
int main()
{
    int a;
    std::cin >> a;
    int b;
    std::cin >> b;
    int sum=a+b;
    std::cout << sum << std::endl;
    return 0;
}
```

Standard input/output stream library

Standard input stream

Namespace

Standard output stream

Inserts a new-line character and flushes the stream

Bandingkan dengan C !

Data Type Scope

Bebas mendeklarasikan variable.
Perhatikan ruang lingkup dari variable yang didefinisikan (local, global, segment).

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


int main() {
    // ...
    for (int i=0; i<5; i++)
    {
        // ...
    }
    return 0;
}
```

Variabel i hanya dikenal di area ini

Fundamental Data Types

Group	Type names*	Notes on size / precision
Character types	**char**	Exactly one byte in size. At least 8 bits.
	char16_t	Not smaller than `char`. At least 16 bits.
	char32_t	Not smaller than `char16_t`. At least 32 bits.
	wchar_t	Can represent the largest supported character set.
Integer types (signed)	**signed char**	Same size as `char`. At least 8 bits.
	signed **short** *int*	Not smaller than `char`. At least 16 bits.
	signed **int**	Not smaller than `short`. At least 16 bits.
	signed **long** *int*	Not smaller than `int`. At least 32 bits.
	signed **long long** *int*	Not smaller than `long`. At least 64 bits.
Integer types (unsigned)	**unsigned char**	(same size as their signed counterparts)
	unsigned short *int*	
	unsigned *int*	
	unsigned long *int*	
	unsigned long long *int*	
Floating-point types	**float**	
	double	Precision not less than `float`
	long double	Precision not less than `double`
Boolean type	**bool**	
Void type	**void**	no storage
Null pointer	**decltype(nullptr)**	

Function Overloading

Perbedaan fungsi dilihat dari prototype-nya (return type, function name, argument).
Memungkinkan membuat fungsi dengan nama yang sama.

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


int kuadrat(int x) { return x*x; }
double kuadrat(double x) { return x*x; }


int main() {
   cout << kuadrat(7) << endl
        << kuadrat(7.5) << endl;
   return 0;
}
```

Function Template

To pass data type as a parameter.
Memungkinkan membuat sebuah fungsi yang berlaku untuk beberapa tipe data berbeda.

```
#include <iostream>
using namespace std;
template <typename T>
T maksimum(T n1, T n2, T n3) {
   T besar = n1;
   if (n2>besar) besar=n2;
   if (n3>besar) besar=n3;
   return besar;
}
int main() {
   int i1, i2, i3;
   cin >> i1 >> i2 >> i3;
   cout << maksimum(i1,i2,i3) << endl << endl;
   double d1, d2, d3;
   cin >> d1 >> d2 >> d3;
   cout << maksimum(d1,d2,d3) << endl;
   return 0;
}
```

Latihan 1 : Gunakan C++

Buat program membaca n bilangan bulat dan menghitung rata-rata dari bilangan-bilangan yang berada di posisi kelipatan k. Jika n = 0 atau tidak ada data pada posisi tersebut, maka rata-rata bernilai 0.
Batasan: 0<n<2M, 1<k<(n-1)

Contoh Input:

```
7 2
10 20 30 40 50 60 70
```

Contoh Output:

```
40.00
```

Latihan 2 : Gunakan C++

Diketahui 2 deretan bilangan yang sudah terurut ascending, masing-masing diakhiri dengan -9. Buat program mencetak seluruh bilangan yang ada secara terurut ascending juga. [TIDAK BOLEH ADA PROSES SORTING ARRAY]

Contoh Input:

```
2 5 9 -9
4 8 10 15 20 -9
```

Contoh Output:

```
2 4 5 8 9 10 15 20
```