Bahasa Pemrograman

Arrummaisha Adrifina, ST., MMSI., MSc.
Universitas Gunadarma

14 Januari 2013
Materi Ajar STIMIK - MURA

Outline

Prinsip Bahasa Pemrograman

Bahasa adalah suatu sistem untuk berkomunikasi. Bahasa tertulis menggunakan simbol (yaitu huruf) untuk membentuk kata. Dalam ilmu komputer, bahasa manusia disebut bahasa alamiah (natural languages), dimana komputer tidak bisa memahaminya, sehingga diperlukan suatu bahasa komputer.

Prinsip Bahasa Pemrograman

- Bahasa pemrograman vs bahasa alami
  - Memfasilitasi komunikasi antar manusia
  - Bahasa pemrograman juga memfasilitasi komunikasi manusia dengan mesin
  - Bahasa pemrograman hanya pada domain komputasional
- Perancang bahasa memiliki vocabulary dasar tentang:
  - Struktur bahasa
  - Arti

Prinsip Bahasa Pemrograman

- Prinsip perancangan bahasa : (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.

Generasi Bahasa Pemrograman

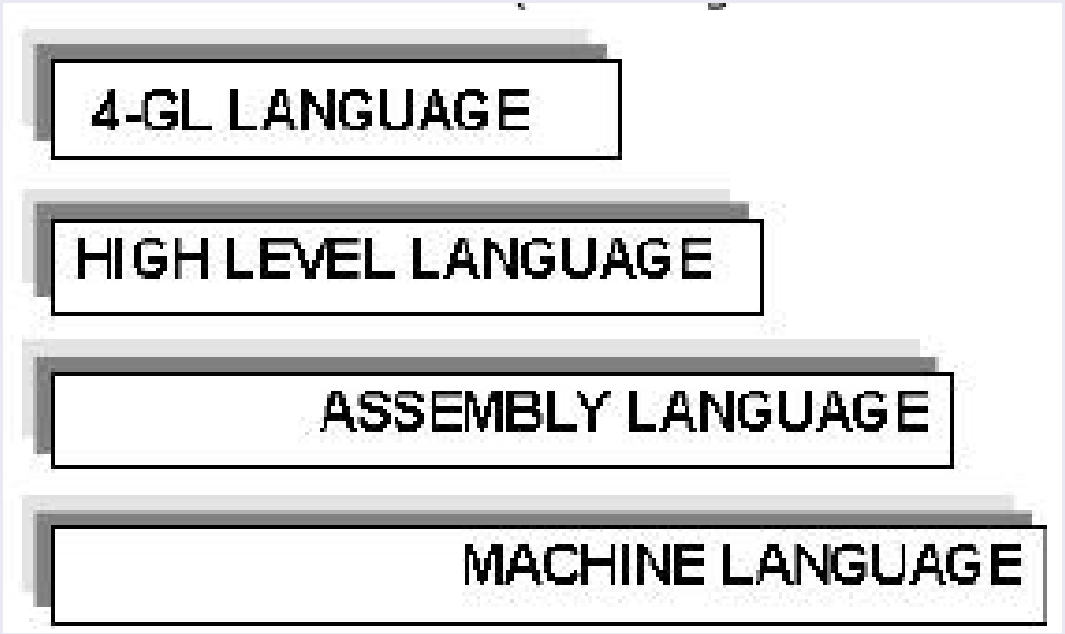


Figure: Generasi Bahasa

Paradigma Pemrograman

- Paradigma pemrograman adalah bentuk pemecahan masalah mengikuti aliran atau "genre" tertentu dari program dan bahasa.
- Empat paradigma pemrograman pada tiga dekade terakhir:
  - Imperative programming
  - Object-oriented programming
  - Functional programming
  - Logic programming
- Beberapa bahasa dirancang mendukung lebih dari satu paradigma. Contoh: C++ (imperative dan OOP), Prolog (functional, logic), dll.

Imperative Programming

- Paradigma paling tua, didasari oleh model komputasi klasik "von Neumann-Eckert".
- Program dan variabel disimpan bersama.
- Program terdiri dari instruksi yang membentuk perhitungan, asignment, input, output, dan kontrol.
- Contoh: Cobol, Fortran, C, Ada, Perl

Object Oriented Programming

- Program adalah kumpulan dari obyek yang saling berinteraksi satu sama lain.
- Program membungkus (encapsulate) data dan fungsi atau prosedur menjadi suatu obyek (class).
- Meliputi mekanisme obyek, pewarisan, dan passing parameter.
- Contoh: Smalltalk, C++, Java, C#

Functional Programming

- Memodelkan masalah komputasi sebagai suatu fungsi matematika, yang mempunyai input (domain) dan hasil atau output (range).
- Tidak dapat menggunakan mekanisme assignment yang tidak dapat diterima secara matematika, misalnya: $x = x + 1$
- Fungsi mengkombinasikan kondisi dan rekursif.
- Contoh: Lisp (List Programming), Scheme, Haskell.

Logic Programming

- Disebut juga Declarative Programming
- Memodelkan masalah menggunakan bahasa deklaratif, yang terdiri dari fakta dan aturan.
- Kadang disebut juga sebagai rule-based languages.
- Contoh: Prolog (Programming in Logic).

Desain Bahasa Pemrograman

- Menciptakan bahasa sehingga pemrogram dapat memecahkan persoalan yang kompleks.
- Kendala yang harus diperhatikan:
  - Architecture
  - Technical Setting
  - Standards

Kendala Desain Pemrograman

- **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.

Tujuan Desain Pemrograman

- **Simplicity and Readability**, program harus mudah ditulis, dan mudah dibaca oleh programmer umumnya.
- **Clarity about Binding**, memiliki batasan definisi dan waktu yang jelas, misalnya reserved words, ukuran memori suatu tipe data, run time, dsb.
- **Reliability**, program akan melakukan hal yang sama ketika memperoleh input data yang sama.
- **Support**, mudah diakses, dipelajari, dan di-install oleh siapa saja.
- **Efficient**.

Compiler and Virtual Machines

- Bahasa program dianalisis dan selanjutnya diterjemahkan ke dalam bentuk yang dapat dipahami mesin, salah satu dari:
  - Dijalankan oleh komputer - "real machine" $\rightarrow$ compiling
  - Dijalankan oleh interpreter - software yang mensimulasikan "virtual machine" dan menjalankan dalam "real machine" $\rightarrow$ interpreting

Compiler

Compiler Adalah suatu program yang menterjemahkan bahasa program (source code) ke dalam bahasa objek (object code). Compiler menggabungkan keseluruhan bahasa program dikumpulkan kemudian disusun kembali.

Tahapan Kompilasi

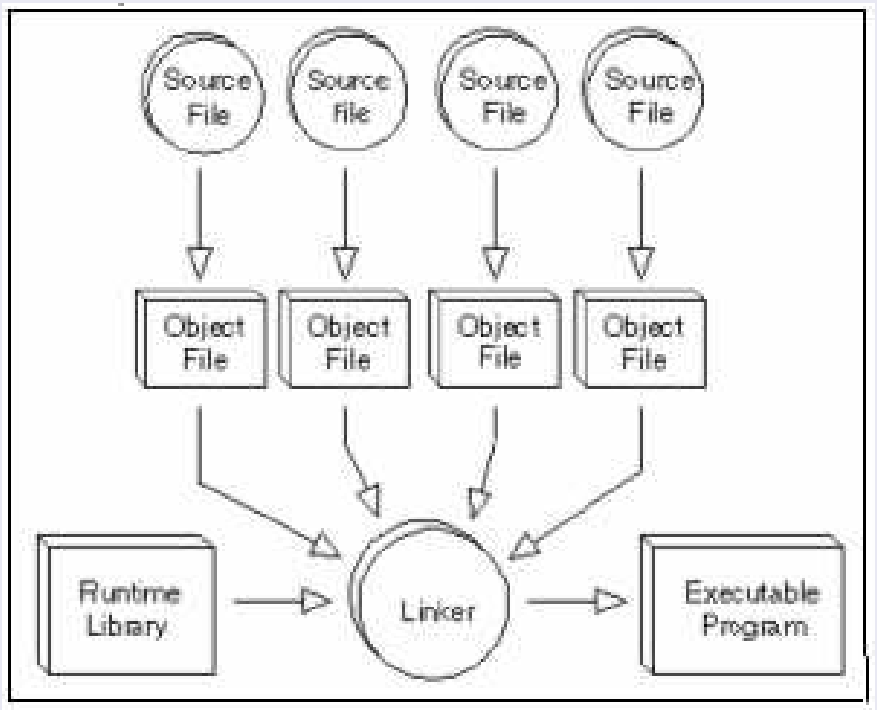


Figure: Proses Kompilasi Program Komputer

Interpreter

Interpreter menganalisis dan mengeksekusi setiap baris dari program tanpa melihat program secara keseluruhan.

Keuntungan dari Interpreter :

1. Proses eksekusi lebih cepat tanpa melalui tahap kompilasi
2. Cocok untuk program berskala besar.

Struktur Level Data

Variable

Variabel adalah suatu nama yang dapat diasosiasikan dengan sebuah nilai yang dapat kita manipulasi.

Konstanta

- Konstanta adalah suatu nilai yang tidak berubah yang diberi nama.
- Contoh : phi (Π), adalah konstanta yang digunakan sebagai perbandingan keliling lingkaran terhadap diameternya
- Sebuah konstanta boleh diberi nilai yang tidak akan diubah di dalam program

Tipe Data

Kategori Data

Data secara umum dapat dikategorikan:

- Tipe data sederhana atau data sederhana
  1. Tunggal
     - Integer
     - Real
     - Boolean
     - Char
  2. Majemuk
     - String

Kategori Data

- Struktur Data
  1. Sederhana: Array dan Record
  2. 2. Majemuk terdiri atas L
     - Linier: Linier Linked List, Stack, Queue
     - Non Linier: Binary Tree, Binary Search Tree, General Tree, Tree, Graf

Integer

- Integer adalah suatu tipe bilangan bulat (bisa menampung nilai negatif, positif, dan nol).
- Dipakai dalam kebanyakan operasi matematika dan loop
- Bahkan beberapa prosesor tidak memiliki kemampuan perhitungan bilangan real sehingga semua perhitungan numerik dilakukan dengan integer

Representasi Integer dalam Komputer

Integer memiliki representasi yang sederhana dalam komputer. Komputer memandang integer sebagai nilai dari serangkaian bilangan biner. Namun komputer tidak memproses satu bit demi satu bit, tapi per blok bit yang umumnya terdiri dari 8 bit (dikenal sebagai 1 byte atau binary eight).

Tipe Variabel

- byte integer 8 bit
- Bilangan integer 8 bit artinya diperlukan memori sebesar 8 bit untuk menyimpan tipe tersebut
- 0000 0000 - 1111 1111
- 5 = 0000 0101

Operasi Terhadap Integer

- kali, bagi, tambah, kurang, dan mod.
- Operator unary : minus (-) & plus
- Operasi perbandingan integer : >, <, <=, >=, <>, dan =

Karakter

- Sebuah variabel bertipe karakter hanya boleh diisi dengan satu simbol saja, seperti ini :

```
program assign_char;
var c:char;
begin
c:= 'A'; (* c berisi huruf A *)
end.
```

Representasi Karakter dalam Komputer

- Sebuah karakter adalah sebuah bilangan integer 8 bit yang memiliki nilai dari 0 sampai 255
- Setiap nilai ini dipetakan dalam suatu simbol, misalnya nilai 65 adalah nilai untuk simbol A, 66 adalah nilai simbol B, dst
- Memiliki pemetaan standar disebut dengan karakter ASCII (American Standard Code For Information Interchange),

Konversi char ke ASCII dan Sebaliknya

```
writeln(ord('A'));
// akan mencetak angka 65.
```

```
writeln(chr(65));
// akan mencetak huruf A.
```

Boolean

- Boolean adalah suatu tipe data yang hanya memiliki nilai true (benar) dan false (salah).
- Sangat diperlukan dalam kondisi perulangan dan kondisional (menggunakan if).
- Ekspresi yang menghasilkan boolean bisa berupa ekspresi dengan tipe-tipe yang terdiri dari tipe boolean, bisa juga berupa ekspresi dari tipe lain.

Boolean

- Contoh ekspresi yang menghasilkan boolean adalah $6 > 5$, karena bilangan 6 memang lebih besar dari 5 maka nilai ekspresi tersebut adalah true, sedangkan $6 < 2$ adalah ekspresi yang nilainya false.

Operator Boolean

- **OR**
  Akan mengasilkan **TRUE** jika **SALAH SATU** operandnya bernilai **TRUE**
- **AND**
  Akan mengasilkan **TRUE** jika **KEDUA** operandnya **TRUE**
- **XOR**
  Akan menghasilkan **TRUE** jika operandnya memiliki nilai boolean yang **BERBEDA**
- **NOT**
  Akan menghasilkan nilai boolean **KEBALIKAN** dari nilai yang diberikan

Representasi Boolean dalam Komputer

- Nilai boolean disimpan sebagai angka 0 untuk false dan 1 untuk true.

Real

Real adalah tipe yang dapat menampung bilangan real. Tipe ini bisa menampung bilangan dengan suatu nilai di belakang koma dengan presisi tertentu (lihat bagian representasi real)

Operasi Terhadap Real

- Operasi yang bisa dilakukan terhadap real meliputi: tambah, kali, minus (sama seperti integer), dan pembagian (memakai simbol / yang menghasilkan bilangan real)
- Operasi MOD & DIV tidak terdefinisi untuk real.
- Operasi perbandingan selain sama dengan(=) dan tidak sama dengan (<>) bisa dilakukan terhadap real (>, <, >=, <=, <>).
- Operasi sama dengan dan tidak sama dengan sebenarnya bisa dilakukan (tidak dibatasi oleh bahasa Pascal), namun dalam kasus tertentu hasilnya mungkin tidak sesuai dengan yang diharapkan.

Kompatibilitas Tipe

Sebuah variabel bertipe string tentu tidak bisa dijumlahkan dengan variabel bertipe integer, karena tipe string tidak kompatibel dengan tipe integer. Kedua tipe itu sangat berbeda, sehingga masalah kompatibilitas tipe itu sudah jelas. Namun dalam kasus tertentu, kompatibilitas tipe mungkin tidak terlalu jelas dan harus diperhatikan dengan baik.

Contohnya, variabel bertipe real tidak bisa langsung diassign pada variabel bertipe integer, karena variabel bertipe real mungkin mengandung pecahan, tapi hal yang sebaliknya (meng-assign integer pada real) bisa dilakukan

Ekspresi

- Ekspresi adalah pernyataan yang menghasilkan nilai dengan tipe tertentu, contoh ekspresi yang paling sederhana adalah operasi aritmatika seperti $5 + 2$ (ekspresi yang menghasilkan nilai bilangan bulat).
- Sebuah angka atau nilai juga merupakan sebuah ekspresi (5 adalah ekspresi)
- Hasil operasi terhadap nilai juga merupakan ekspresi ($5 + 8$ adalah ekspresi)
- Sebuah variabel adalah sebuah ekspresi

Assignment

Assignment adalah pemberian nilai kepada variabel. Assignment memberikan nilai pada ruas kiri sesuai dengan hasil nilai di ruas kanan (berupa sebuah ekspresi).

```
program assign_variabel;
var a: integer;
begin
a := 2;
b := a * 5;
end.
```

Representasi Tipe

- Komputer hanya bisa memproses angka, sehingga semua tipe data dalam komputer akan diproses dalam bentuk bilangan integer.
- Bahkan kata-kata yang muncul dalam komputer juga diproses sebagai bilangan.
- Pengetahuan mengenai representasi tipe merupakan penting dalam pemrograman. Dengan mengetahui representasi tipe, pemrogram bisa mengetahui batasan dari setiap tipe yang ada, sehingga dapat memilih tipe yang tepat ketika membuat program.

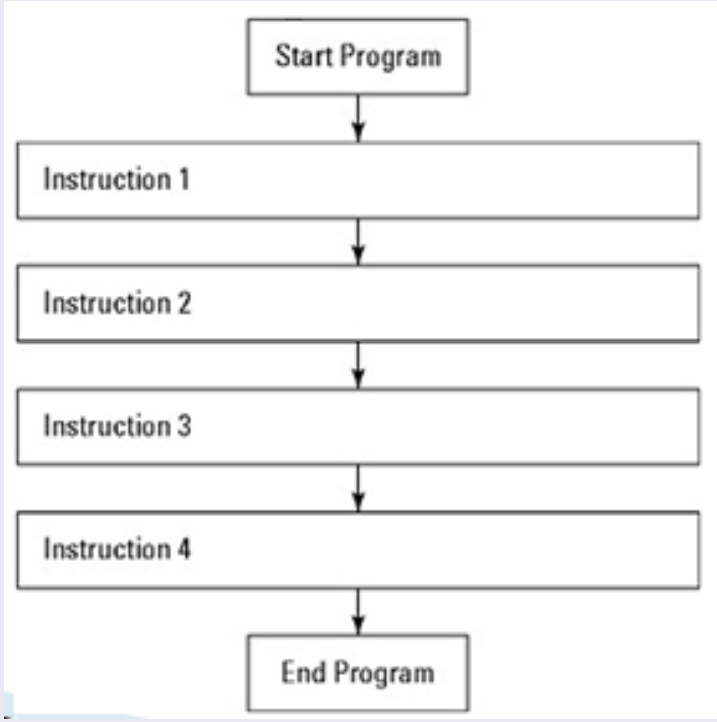
Struktur Level Program

Komponen Pemrograman

- Sequence (urutan) :
  - Eksekusi statement/instruksi secara terurut
- Selection (seleksi) :
  - Eksekusi salah satu statement bergantung pada kondisi tertentu
- Repetition (pengulangan) :
  - Eksekusi sebuah statement hingga mencapai kondisi tertentu

Sequential Instructions

Sequential berarti program melakukan instruksi-instruksi yang ada secara berurutan. Mulai dari instruksi pertama hingga instruksi terakhir.



Sequential Instructions

Berikut ini adalah contoh program yang melakukan perintah yang ada secara berurutan.

Contoh Program:

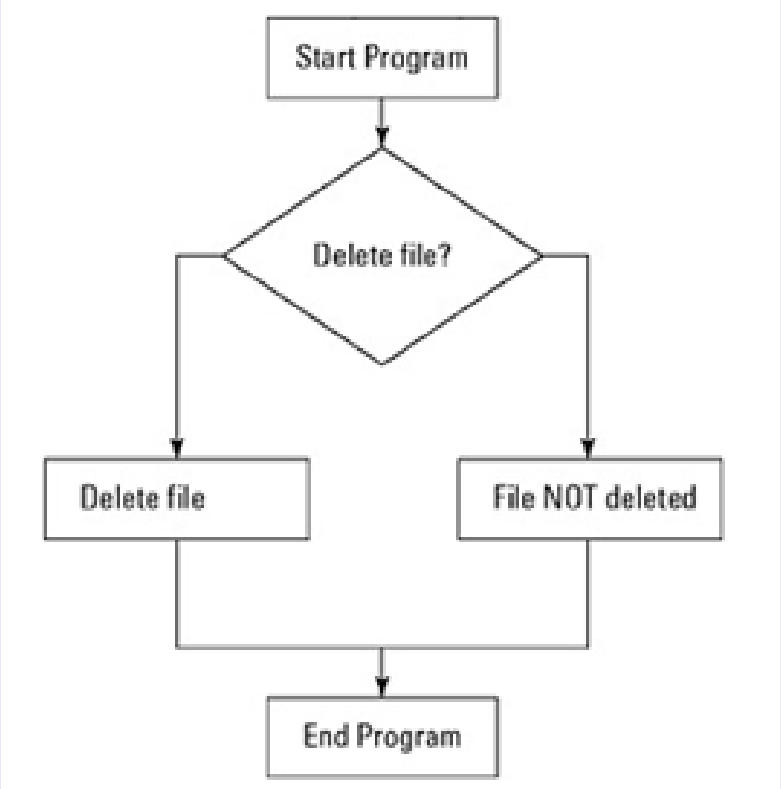
```
Writeln('I am a useless program');
Writeln('Because this is all I can do');
```

Hasil:

I am a useless program
Because this is all I can do

Branching Instructions

Program dapat melakukan respon atau pilihan hasil terhadap proses input yang dilakukan user



Branching Instructions

Branching instructions memungkinkan program menjalankan instruksi dengan bergantung kepada nilai tertentu.

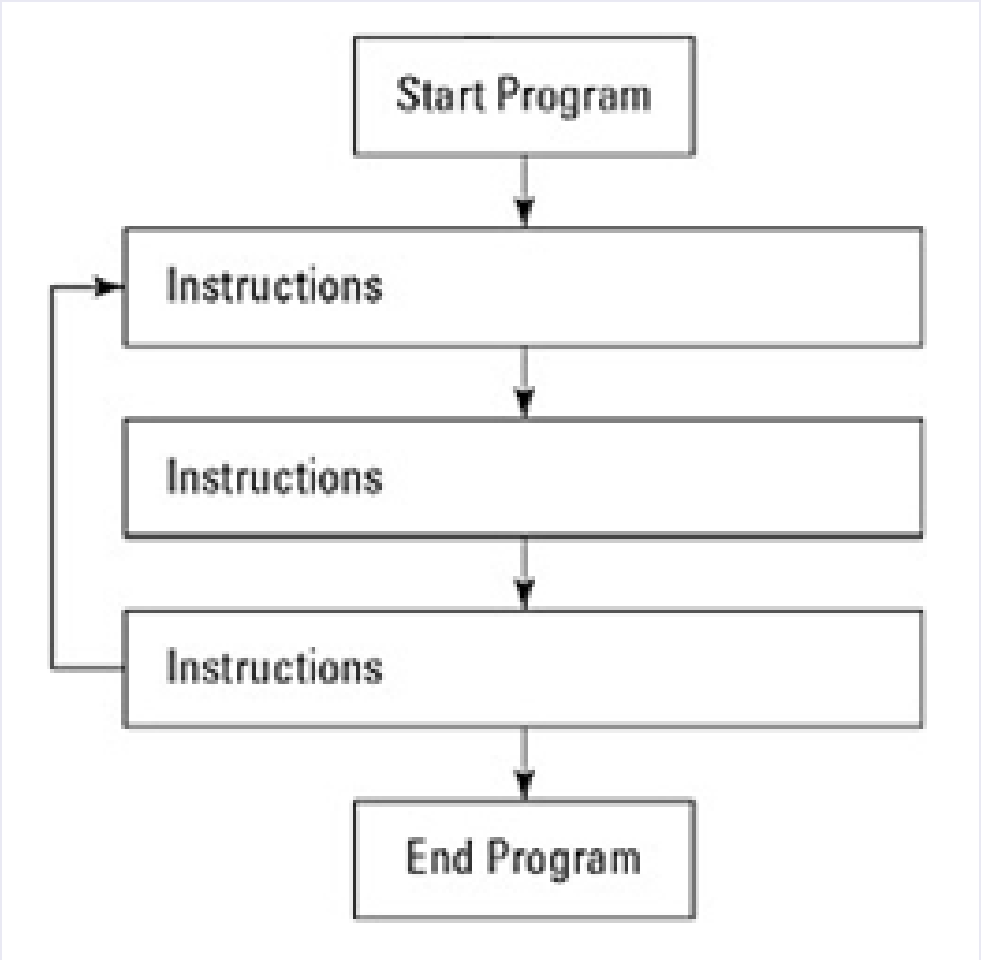
```
Write(‘ Do you want to delete the file? (Y or N) ’);
Read(answer);

IF answer = ‘Y’
THEN writeln (‘Deleting file’ )
ELSE writeln (‘File NOT deleted’);
```

Looping Instructions

- Setiap perintah pada program hanya akan dijalankan SATU kali
- Looping Instruction memungkinkan program menjalankan perintah secara berulang
- Looping instructions memungkinkan computer untuk mengulang perintah

Looping Instructions



Looping Instructions

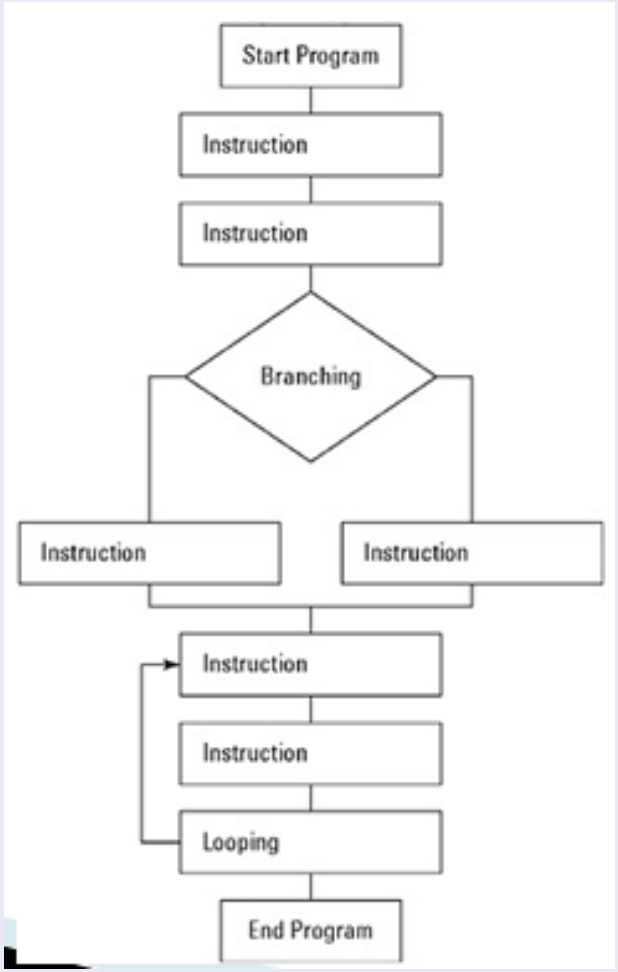
Contoh Program

```
FOR i = 1 TO 5 DO
Write(i);
```

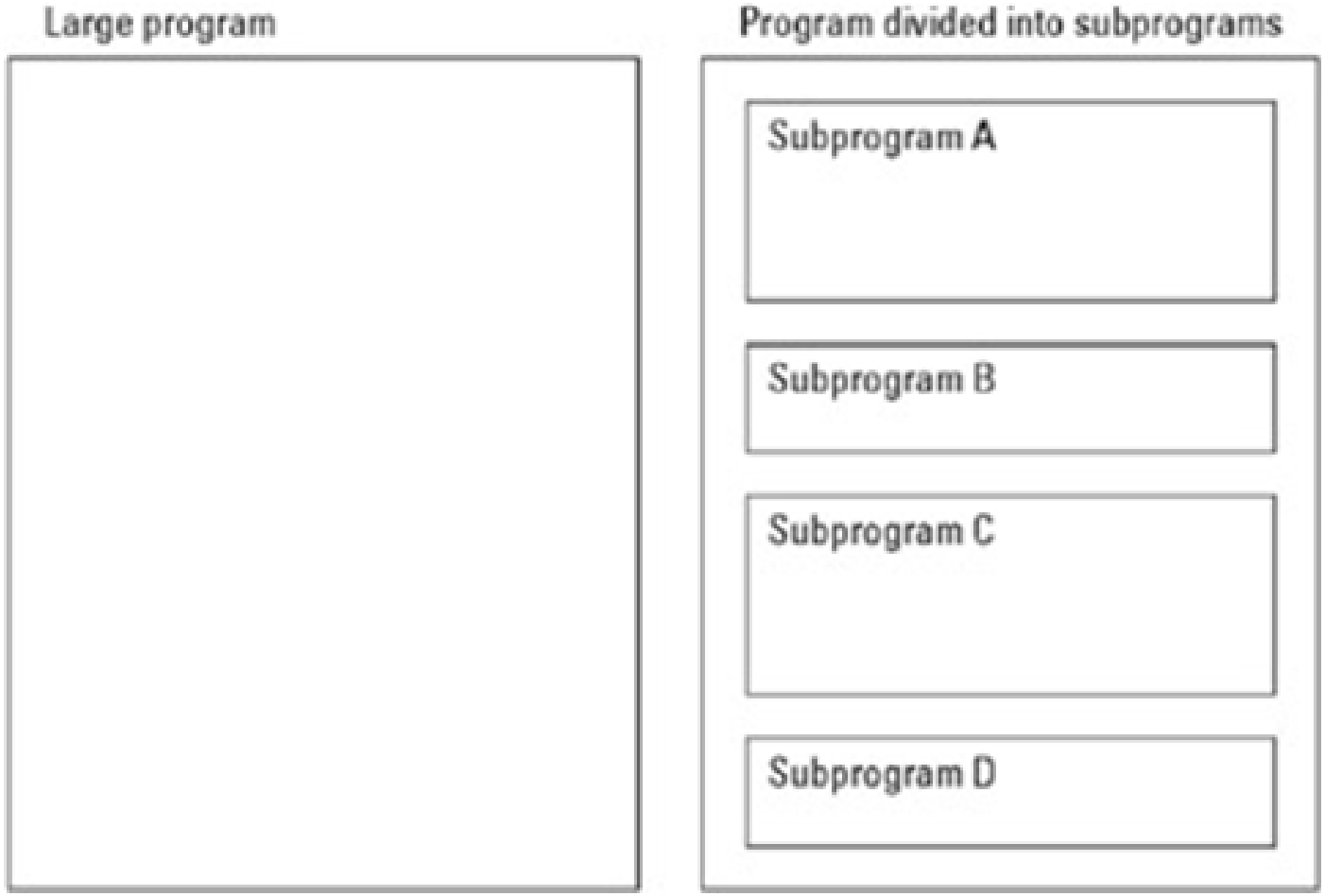
Hasil:

1
2
3
4
5

The Building Blocks of Programming



Subprogram


Large program
Program divided into subprograms
Subprogram A
Subprogram B
Subprogram C
Subprogram D

Pengenalan C++

- C sebagai bahasa pendahulunya C++
- Bahasa pemrograman tingkat menengah
- Pencipta C adalah Brian W. Kernighan dan Dennis M. Ritchie pada tahun 1972.
- C merupakan bahasa pemrograman terstruktur yang membagi program ke dalam sejumlah blok (sub program).

Pengenalan C++

- C++ diciptakan satu dekade setelah C.
- C++ diciptakan oleh Bjarne Stroustroup dari Laboratorium Bell, AT&T pada tahun 1983.
- Pada awalnya C++ diberi nama "A better C". Nama C++ sendiri diberinama oleh Rick Mascitti. Adapun tanda ++ berasal dari operator increment pada bahasa C.
- Tahun 1990, C++ menjadi bahasa berorientasi objek.

Compiler C++

- Microsoft
  - Microsoft C/C++
  - Visual C++
- Borland International
  - Turbo C++
  - Borland C++

Kerangka Program C++

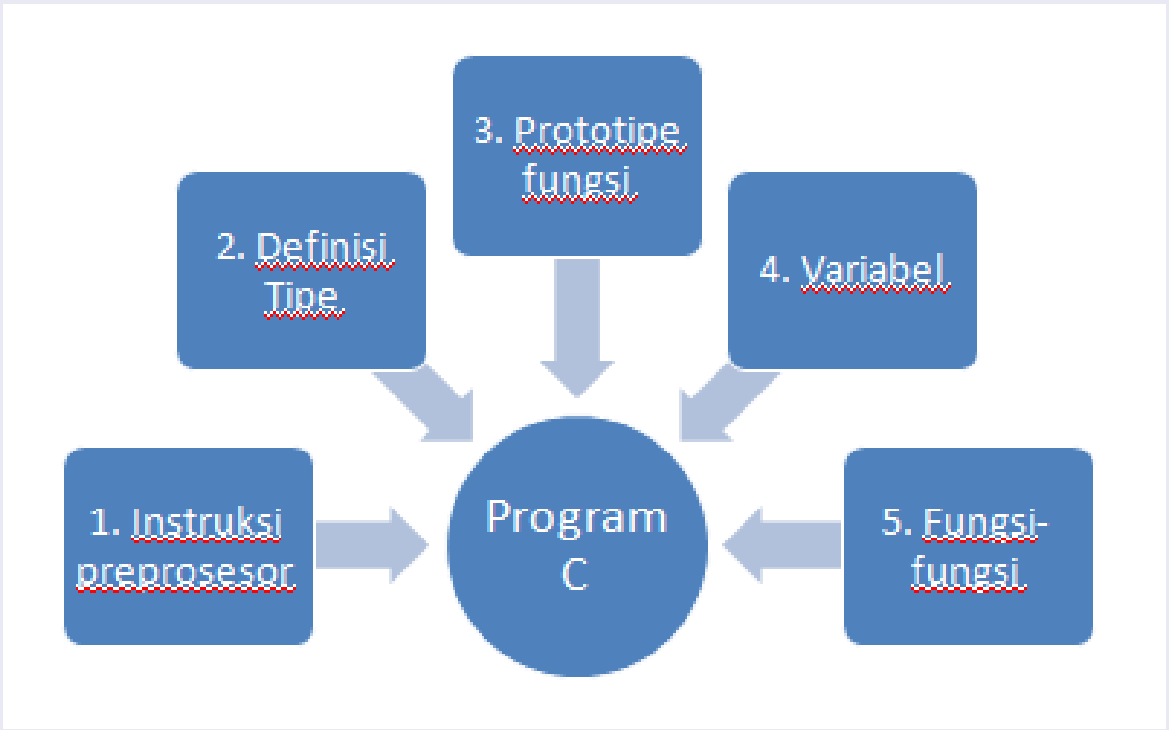


Figure: Kerangka Program C++

Struktur bahasa C++

```
<<<<Header>>>> <<<<deklarasi fungsi>>>>

void main()
{
//deklarasi variable & konstanta

//proses
}
```

ELEMEN DASAR C++

- Identifier (Pengenal)
  suatu nama yang biasa dipakai dalam pemrograman untuk menyatakan variabel, konstanta, tipe data, dan fungsi

Aturan untuk penulisan identifier sama dengan aturan dalam C, antara lain:

1. Tidak boleh dimulai dengan karakter non huruf
    1. Tidak boleh ada spasi
        1. tidak boleh menggunakan karakter-karakter ~ ! @ # $ % ^ & * ( ) + ‘ - = { } [ ] : ” ; ’ < > ? , . / |
        2. Tidak boleh menggunakan reserved words yang ada pada C++

ELEMEN DASAR C++ (Cont'd)

- Tipe Data

Tipe data bilangan bulat:

1. char
    1. int (integer)
    2. short (short integer)
    3. long (long integer)

- Tipe data bilangan bulat:
    1. Float (real)
    2. Double (real double)
    3. long double

Prepocessor : #define

- Digunakan untuk mendefinisikan konstanta atau makro
- Formula :

  #define <macro> <replacement name>
- Contoh:

  #define TRUE 1
  #define FALSE !TRUE
  #define PI 3.14

Preprocessor : #include

- Dipakai untuk membaca file yang diantaranya berisi deklarasi fungsi dan konstanta
- File yang dibaca adalah file berekstensi .h (istilahnya file header)
- C menyediakan beberapa file header siap pakai, contoh stdio.h, stlib.h, dll

Contoh Penggunaan Directive

- #include <stdio.h>, artinya adalah pada memerintahkan kompiler untuk membaca file stdio.h pada saat kompilasi
- Bentuk umum directive :
  - #include <nama_file_header>

Pola Deklarasi Fungsi

```
tipe_data nama_fungsi (parameter)
{ //letakkan variabel lokal di sini
//letakkan pernyataan C di sini
}
```

Penulisan Komentar pada C++

- Ada dua cara untuk menuliskan komentar :
  - **// komentar baris**
    akan mengabaikan apapun mulai dari tanda(//) sampai akhir baris.
  - **/* komentar blok */**
    akan mengabaikan apapun yang berada diantara tanda /* dan *

Input dan Output

- Dalam ANSI C, operasi input dan output dilakukan dengan menggunakan fungsi-fungsi header file stdio.h misalkan : printf, scanf, putc, dsb.
- Untuk input dan output ke file digunakan fread, fwrite, fputc, dsb.
- Dalam C++ menggunakan iostream.h, stsrtrea.h, fstream.h dan constrea.h

Contoh Input Output pada C++

```
#include <iostream.h>
void main()
{ int x;
cout <<“Masukkan sebuah bilangan: “<<end1;
cin >> x;
cout <<“Bilangan yang dimasukkan adalah “ << x <<
end1;
}
```

Deklarasi Variable

- Variable dalam C++ harus dideklarasikan terlebih dahulu sebelum digunakan
- Cara pendeklarasian variable pada C++:
  <jenis Variable> <nama_variable>;
- Contoh:
  int a; float mynumber; int a, b ,c;

Deklarasi Variable (cont'd)

- Inisialisasi Variable
  Contoh : int a = 0;

Operator

- Aritmetic Operators (=, -, *, /, %)
- Compound assignation operators (+=, -=, *=, /=, %=, >>=, <<=, &=, ^=, |=) Contoh : value += increase; equivalen dengan value = value + increase;
- Increase (++) dan decrease (–)
- Relational Operators (==, !=, >, <, >=, <=)
- Logic Operators (!, &&, ||)
- Conditional operator (?)

ARRAY

- Di dalam C++ tidak ada tipe variabel untuk memasukkan sejumlah karakter string. Untuk itu digunakan array dari tipe char
- Contoh, array berikut ini (atau karakter string):
  char jenny [20];
  dapat menampung karakter sampai 20 karakter

ARRAY

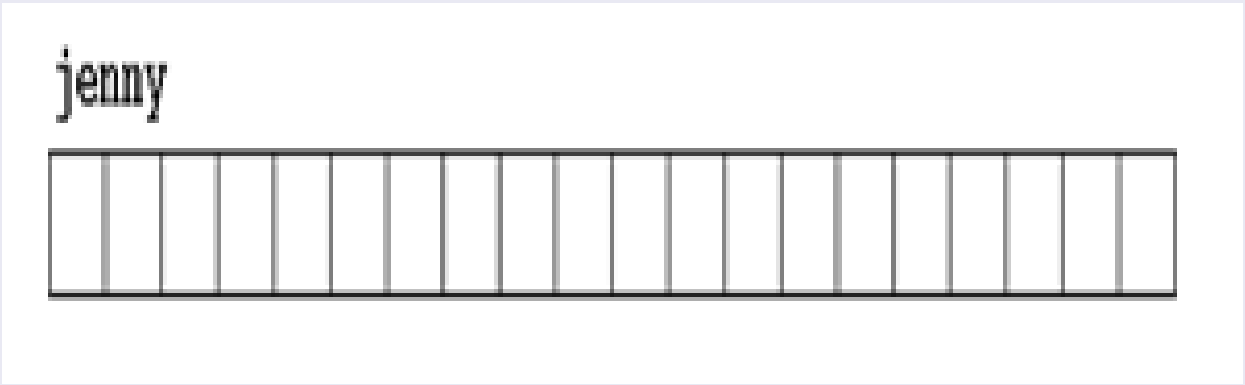


Figure: Array

Kondisi

- if
- if else
- switch

Perulangan

- for
- while
- do while

if

Bentuk Umum

```
if (condition)
    action;
```

if

```
#include <iostream>
using nonespace std

int main()
{
int x;
cout<<"Masukkan angka: ";
cin>>x;
if (x==100)
cout<<"x adalah 100";
return 0;
}
```

if else

Bentuk Umum

```
if (condition)
        action if true;
else (condition)
        action if false;
```

if else

```
#include <iostream>
using nonespace std

int main()
{
int x;
cout<<"Masukkan angka: ";
cin>>x;
if (x==100)
cout<<"x adalah 100";
else
cout<<"x bukan 100"<<"\n";
return 0;
}
```

for

Bentuk Umum

```
for(initialization; condition; update)
    do something;
```

for

```
for (i=1; i<100;i++) {
    cout << "Checking " << i << endl;
    if ( i%2 )
        cout << i << " is odd" << endl;
    else
        cout << i << " is even" <<
    endl;
}
```

while

Bentuk Umum

```
while (condition)
    do something;
```

while

```
lettergrade = 'A';
cutoff = 90;
while (grade < cutoff){
lettergrade = lettergrade + 1;
cutoff = cutoff - 10;
}
if (lettergrade > 'F')
lettergrade = 'F';
```

do while

Bentuk Umum

```
do
        somestuff;
while(condition);
```

do while

```
i=1;
do
cout << “i is “ << i++ << endl;
while (i <=10);
```

Perulangan

- while
- for
- do while

Pointer

- Pointer : variabel yang berisi alamat memori
  Bentuk Umum : Type *variable name; Type adalah tipe dasar pointer Variable name adalah nama variabel pointer * adalah operator memori untuk mengembalikan nilai variabel pada alamatnya yang ditentukan oleh operand.
- Contoh program pointer
  //Program :pointer1. cpp #include <iostream.h>
  // cetak p dan *p
  Void main(void)
  { int v = 7, *p;
  p = &v;
  cout << "Nilai v = " << v << " dan *p = " << *p << "
  \nAlamatnya = " << p << 'n';}
- Hasil: Nilai v = 7 dan *p = 7 Alamatnya = efffb24

Terima Kasih

Konsep Pemrograman Berbasis Objek

- Teknik membuat suatu program berdasarkan objek

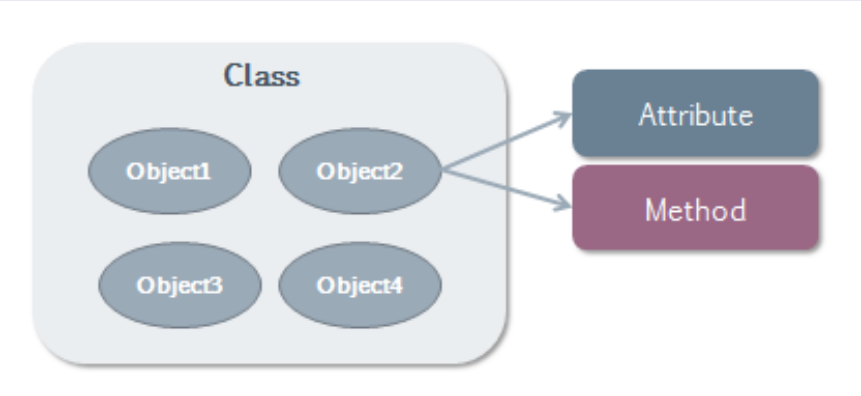


Figure: Konsep PBO

Konsep Pemrograman Berbasis Objek

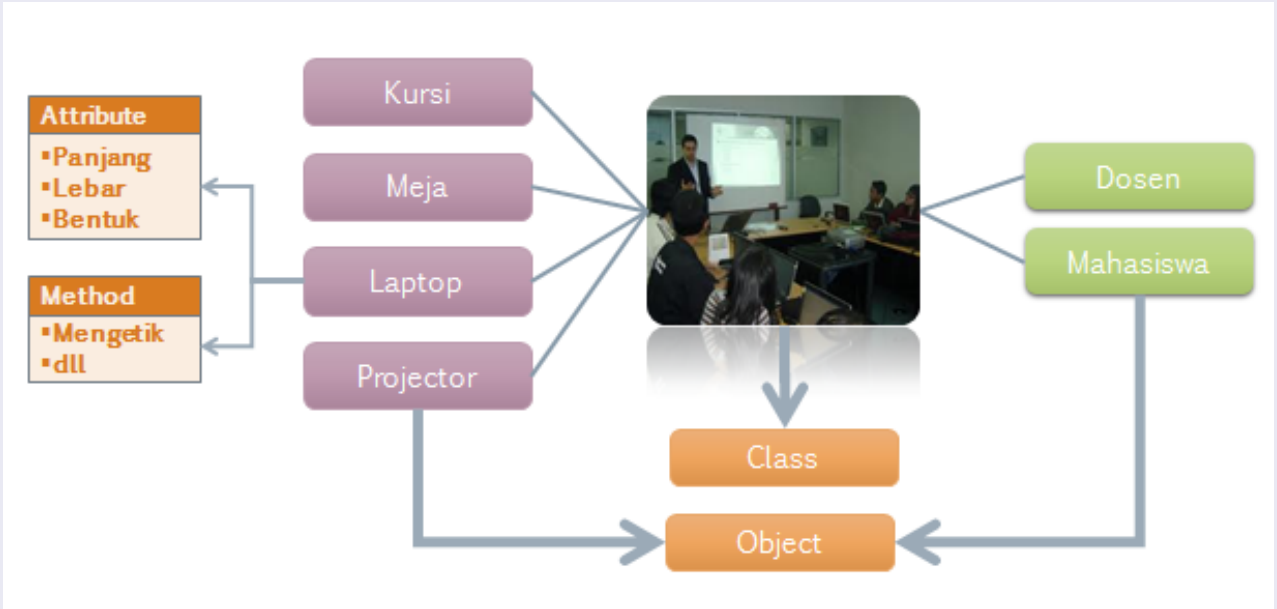


Figure: Konsep PBO

Bahasa Pemrograman Berbasis Objek

- Java
- C++
- Pascal
- Visual Basic.NET
- Ruby
- Python
- PHP
- C#
- Delphi
- Perl
- Adobe Flash AS 3.0

Java

Sejarah Java

- 1991 : Sun Microsystems → Project (Green)→Kendali Cerdas Star 7
- Dipimpin oleh Patrick Naughton James Gosling
- Memiliki kendala:
  - Membutuhkan bahasa untuk perangkat dengan memori kecil
  - Membutuhkan bahasa yang dapat berjalan pada berbagai processor

Figure: Sejarah Java

Perkembangan Teknologi Java

- Java Card
  Untuk peralatan elektronik dengan memori terbatas Misal: Smart Card
- J2ME (Java 2 Platform, Micro Edition)
  Untuk handphone, PDA
- J2SE (Java 2 Platform, Standard Edition)
  Untuk penerapan teknologi pada komputer desktop
- J2EE (Java 2 Platform, Enterprise Edition)
  Untuk penerapan teknologi pada komputer server

Tahapan Pemrograman Java

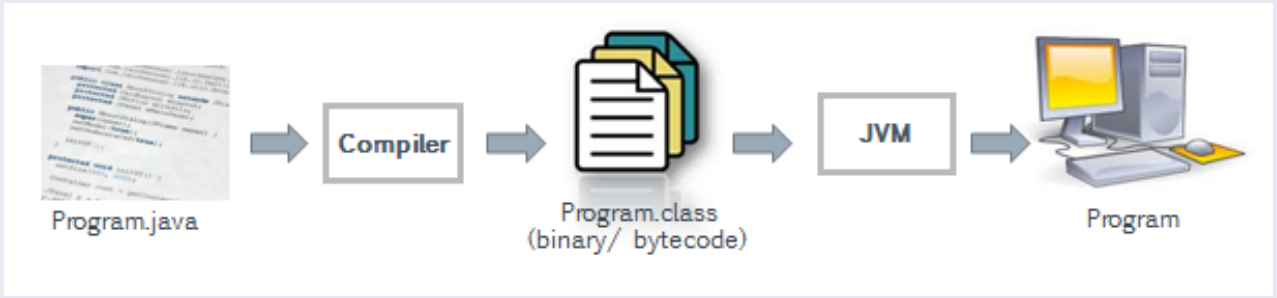


Figure: Tahapan Pemrograman Java

Perangkat Lunak Pendukung

- JDK (Java Development Kit)
- Notepad
- Notepad++
- Edit Plus
- Aplikasi IDE (Integrated Development Environment)
  - Netbeans
  - Eclipse
  - Jbuilder
  - JCreator
  - Intelli J IDEA

Setting Path JAVA

- Langkah-langkah untuk menambahkan variabel JAVA_HOME ke dalam environment variable pada Windows 7 32bit/64 bit, sebagai berikut :
  1. klik kanan My Computer, kemudian klik Properties
  2. Setelah muncul window System Properties
  3. Pilih advanced sytem settings (pada menu sebelah kiri)
  4. Pilih tab Advanced klik Environment Variables, maka akan muncul window Environment Variables
  5. Untuk menambahkan JAVA_HOME, klik New
  6. Isikan JAVA_HOME pada Variable name
  7. Isikan alamat dari direktori dimana JDK terinstall pada Variable value, sebagai contoh C:\Program Files\Java\jdk1.6.0_10
  8. klik OK

Setting Path JAVA

PATH

masih pada Environment Variables

1. Cari dan pilih variabel PATH pada tabel System Variables yang telah terkonfigurasi sebelumnya pada komputer
2. Edit, kemudian tambahkan C:\Program Files\Java\jdk1.6.0_10\bin pada Variable value
3. Klik OK

Dasar Pemrograman Java

Token

Token adalah elemen terkecil di program yang masih memiliki arti. Ada 5 token dalam bahasa java yaitu.

- Identifier
- Keyword
- Literal
- Tipe data
- Operator
- Separator

Identifier

Identifier adalah token yang merepresentasikan nama sesuatu.

- Variabel
- Konstanta
- Attribute
- Method
- Kelas
- Package
- Interface
- Nama file
- Dan lain-lain.

Keyword

abstract	continue	for	new	switch
boolean	default	goto	null	synchronized
break	do	if	package	this
byte	double	implements	private	threadsafe
byvalue	else	import	protected	throw
case	extends	instanceof	public	throws
catch	false	int	return	transient
car	final	interface	short	true
class	finally	long	static	try
const	float	native	super	void
				while

Literal

Literal adalah nilai variabel/attribute atau nilai konstanta atau nilai objek data. Ada tiga besaran literal dalam java yaitu angka, karakter, dan string.

Bentuk umum :

```
TipeData namaVar = ungkapan_atau_nilai;
TipeData namaVar1, namaVar2, ...; [modifier] static final
TipeData NAMAKONSTANTA = nilai;
```

Tipe Data

Tipe Data Primitif	Jangkauan	Ukuran (bit)
byte	-128 s/d 127	8
short	-32767 s/d 32767	16
int	-2147483648 s/d 2147483647	32
long	-9223372036854775808 s/d 9223372036854775807	64
char	sebuah Unicode	16
float	3.4e-038 s/d 3.4e+038	32
double	1.7e-308 s/d 1.7e+308	54
boolean	false = 0 atau true = 1	8

Operator

Prioritas	Kelompok Operator	Keterngan
1	. [] ()	Sekaligus
2	++var, --var, ~, instanceof	preincrement, predecrement, unary, instance dari kelas ...
3	(type) (casting)	
4	!	Not
5	*, /, %	perkalian, pembagian, modulus
6	+, -	penjumlahan, pengurangn
7	<<, >>, >>>	geser untuk bil biner
8	<, >, <=, >=	pembandingan
9	==, !=	kesamaan, ketidaksamaan
10	&	and
11	^	exclusive or
12	|	unconditional or
13	&&	conditional and
14	||	conditional or
15	? :	shorthand untuk if..then...else...
16	=, +=, -=, *=, /=, %=, ^=	operator penugasan
17	&=, |=, <<=, >>=, >>>=	operator penugasan
18	var++, var--	postincrement, postdecrement

Separator

Notasi	Nama	Deskripsi
(...)	kurung	mengelompokkan parameter method.
{...}	kurung kurawal	mengelompokkan nilai-nilai suatu array, mendefinisikan blok kode kelas ataupun kode method.
[...]	kurung siku	mendeklarasikan tipe array
;	titik koma	mengakhiri pernyataan, merangkai pernyataan-pernyataan di dalam for.
,	koma	memisahkan identifier-identifier di bagian deklarsi variable.
.	titik	memisahkan nama-nama package, memisahkan kelas dari objek, dan objek dari method.

Struktur Program

```
Komentar program

Deklarasi package dan import

// Kelas pertama
class NamaKelas {
  <pernyataan>
}

// Kelas lainnya
class NamaKelasLain {
  <pernyataan>
}
```

Struktur Program

- Dapat ditulis dalam satu file .java:
  - Berisi satu kelas
  - Berisi beberapa kelas
- Ditulis dalam beberapa file .java:
  - Satu file satu kelas
  - Satu file banyak kelas
- Hanya boleh ada satu fungsi utama pada setiap program objek.

Kompilasi dan Eksekusi Java

- Kompilasi
  javac <namafile.java>
  contoh:
  javac HelloWorld.java
- Eksekusi
  java <namafile hasil kompilasi>
  contoh:
  java HelloWorld

Keyword Break

- Penggunaan keyword break:
  - Keluar dari kendali percabangan switch,
  - Keluar dari kendali perulangan.

  Percabangan/perulangan akan diakhiri, kemudian eksekusi dilanjutkan ke pernyataan setelah blok percabangan/perulangan tersebut.

Contoh Keyword Break

```
public class ContohBreak {
    public static void main(String args[]) {
        int i = 0;
        do {
            i++;
            System.out.println(i);
            if (i==5) break;
        } while (i <= 9);
    }
}
```

Continue

Penggunaan keyword ini untuk segera lompat ke perulangan berikutnya. Pernyataan-pernyataan setelah keyword continue dalam blok perulangan saat itu berarti diabaikan.

Continue

```
public class contohContinue {
    public static void main(String args[]) {
        int i=0;
        do {
            i++;
            if (i==3) continue;
            System.out.println( iterasi ke : +i);
            if (i==5) break; }
        while(i <= 9);
    }
}
```

Return

Keyword ini digunakan untuk keluar dari suatu method

```
int abs(int x) {
    if (x >= 0)
        return x;
    else
        return(-x)
    .....
    }
}
```

Kelas

Kelas digunakan untuk membuat objek, dan berperan sebagai tipe data dari objek. Kelas merupakan sarana pengkapsulan kumpulan data dan kumpulan method yang mengoperasikan kumpulan data tersebut.

Anatomi Kelas

(modifier1) class NamaKelas (modifier2) {
    classbody
}

Classbody terdiri dari attribute, constructor, dan method.

Kelas

Ada 10 keyword yang digunakan sebagai modifier1 dan dikelompokkan menjadi :

1. Modifier akses (public, protected, default, private)
2. Modifier final
3. Modifier static
4. Modifier abstract
5. Modifier synchronized
6. Modifier native
7. Modifier storage (transient, volatile)

Atribute

Deklarasi diletakkan di dalam classbody (di luar method). Bentuk umum deklarasi attribute :

[modifier] tipedata namavariabel; [public] [static] final tipedata NAMA_KONSTANTA = nilai;

Contoh :

```
public class CircleClass {
    public static final double PI = 3.14159265358979323846;
    public double x, y, r;
    // dan seterusnya
}
```

Method

Method merupakan tingkah laku dari suatu objek atau kelas, jika bersifat static berarti tingkah laku semua objek dalam kelas tersebut. Bentuk umum method :

```
[modifier] tipe_return_value namaMethod(tipe parameter) {
    methodbody;
}  [modifier] tipe_return_value main(String args[]) {

    methodbody
}
```

Method

Ada tiga sintaks pemanggilan suatu method :

- namaMethod([nilaiParamater]);
- namaObjek.namaMethod([nilaiParamater]);
- namaKelas.namaMethod([nilaiParamater]);

Prolog

Sejarah Prolog

- Prolog (Programming in Logic).
- Dikembangkan oleh Alain Colmenraurer dan P.Roussel di Universitas Marseilles Perancis, tahun 1972.
- Prolog populer di Eropa untuk aplikasi artificial intelligence, sedangkan di Amerika peneliti mengembangkan aplikasi yang sama, yaitu LISP.

Perbedaan Prolog dengan Bahasa Lainnya

Bahasa Pemrograman Umum

- Diperlukan algoritma/prosedur untuk memecahkan masalah (procedural languange)
- Program menjalankan prosedur yang sama berulang-ulang dengan data masukan yang berbeda-beda.
- Prosedur dan pengendalian program ditentukan oleh programmer dan perhitungan dilakukan sesuai dengan prosedur yang telah dibuat.

Perbedaan Prolog dengan Bahasa Lainnya

Bahasa Pemrograman Prolog

- Object oriented languange atau declarative languange.
- Tidak terdapat prosedur, tetapi hanya kumpulan data-data objek (fakta) yang akan diolah, dan relasi antar objek tersebut membentuk aturan yang diperlukan untuk mencari suatu jawaban
- Programmer menentukan tujuan (goal), dan komputer menentukan bagaimana cara mencapai tujuan tersebut serta mencari jawabannya.
- Dilakukan pembuktian terhadap cocok-tidaknya tujuan dengan data-data yang telah ada dan relasinya.

Perbedaan Prolog dengan Bahasa Lainnya

Bahasa Pemrograman Prolog

- Prolog ideal untuk memecahkan masalah yang tidak terstruktur, dan prosedur pemecahannya tidak diketahui, khususnya untuk memecahkan masalah non numerik.
- Prolog bekerja seperti pikiran manusia, proses pemecahan masalah bergerak di dalam ruang masalah menuju suatu tujuan (jawaban tertentu).
- Contoh : Pembuatan program catur dengan Prolog

Aplikasi Prolog

- Sistem Pakar (Expert System)
  Program menggunakan teknik pengambilan kesimpulan dari data-data yang didapat, layaknya seorang ahli. Contoh dalam mendiagnosa penyakit
- Pengolahan Bahasa Alami (Natural Languange Processing)
  Program dibuat agar pemakai dapat berkomunikasi dengan komputer dalam bahasa manusia sehari-hari, layaknya penterjemah.

Aplikasi Prolog

- Robotik
  Prolog digunakan untuk mengolah data masukanyang berasal dari sensor dan mengambil keputusan untuk menentukan gerakan yang harus dilakukan.
- Pengenalan Pola (Pattern Recognition)
  Banyak digunakan dalam image processing, dimana komputer dapat membedakan suatu objek dengan objek yang lain.
- Belajar (Learning)
  Program belajar dari kesalahan yang pernah dilakukan, dari pengamataqn atau dari hal-hal yang pernah diminta untuk dilakukan.

Fakta dan Relasi

Prolog terdiri dari kumpulan data-data objek yang merupakan suatu fakta. Fakta dibedakan 2 macam :

- Menunjukkan relasi.
- Menunjukkan milik/sifat.

Penulisannya diakhiri dengan tanda titik “.”

Fakta dan Relasi

Fakta	Prolog
Slamet adalah ayah Amin	ayah(slamet, amin)
Anita adalah seorang wanita	wanita(anita)
Angga suka renang dan tenis	suka(angga, renang) dan suka(angga, tenis)
Jeruk berwarna jingga	jingga(jeruk)

Aturan ("Rules")

- Aturan adalah suatu pernyataan yang menunjukkan bagaimana fakta-fakta berinteraksi satu dengan yang lain untuk membentuk suatu kesimpulan.
- Sebuah aturan dinyakatakan sebagai suatu kalimat bersyarat. Kata "if" adalah kata yang dikenal Prolog untuk menyatakan kalimat bersyarat atau disimbolkan dengan ":-".

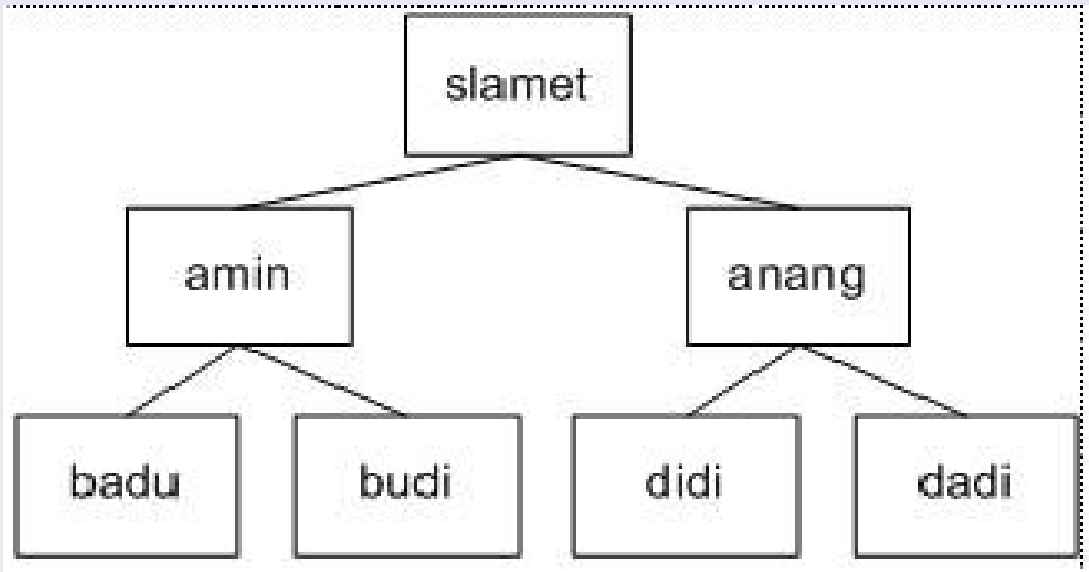
Aturan ("Rules")

Fakta dan Aturan	Prolog
F: Tino suka apel	suka(tino, apel)
A: Yuli suka sesuatu yang disukai tino	suka(yuli, sesuatu):- suka(tino, sesuatu)

Aturan ("Rules")

- Setiap aturan terdiri dari kesimpulan(kepala) dan tubuh.
- Tubuh dapat terdiri dari 1 atau lebih pernyataan atau aturan yang lain, disebut subgoal dan dihubungkan dengan logika "and".
- Aturan memiliki sifat then/if conditional "Kepala(head) benar jika tubuh (body) benar".

Aturan ("Rules")



Pertanyaan (“Query”)

- Setelah memberikan data-data berupa fakta dan aturan, selanjutnya kita dapat mengajukan pertanyaan berdasarkan fakta dan aturan yang ada.
- Penulisannya diawali simbol “?-“ dan diakhiri tanda “.”.

Predikat ("Predicate")

- Predikat adalah nama simbolik untuk relasi.
- Contoh : ayah(slamet,amin).
  Ayah $\rightarrow$ Predikat
  slamet dan amin$\rightarrow$ argumen.
  1. Sebuah predikat dapat tidak memiliki atau memiliki argumen dengan jumlah bebas.
  2. Jumlah argumen suatu predikat disebut aritas (arity)
     - ayah(nama) ..... aritas-nya 1
     - ayah(nama1,nama2) ..... aritasnya 2

Syarat Penulisan Nama Predikat

- Harus diawali dengan huruf kecil dan dapat diikuti dengan huruf, bilangan atau garis bawah.
- Panjang nama predikat maksimum 250 karakter.
- Tidak diperbolehkan menggunakan spasi, tanda minus, tanda bintang dan garis miring.

Variabel

- Varibel adalah besaran yang nilainya dapat berubah-ubah.
- Tata cara penulisan variabel :
  1. Nama varibel harus diawali huruf besar atau garis bawah(_)
  2. Nama variabel dapat terdiri dari huruf, bilangan, atau simbol dan merupakan kesatuan dengan panjang maksimum 250 karakter.
  3. Nama variabel hendaknya mengandung makna yang berkaitan dengan data yang dinyatakannya.

LISP

Bahasa LISP

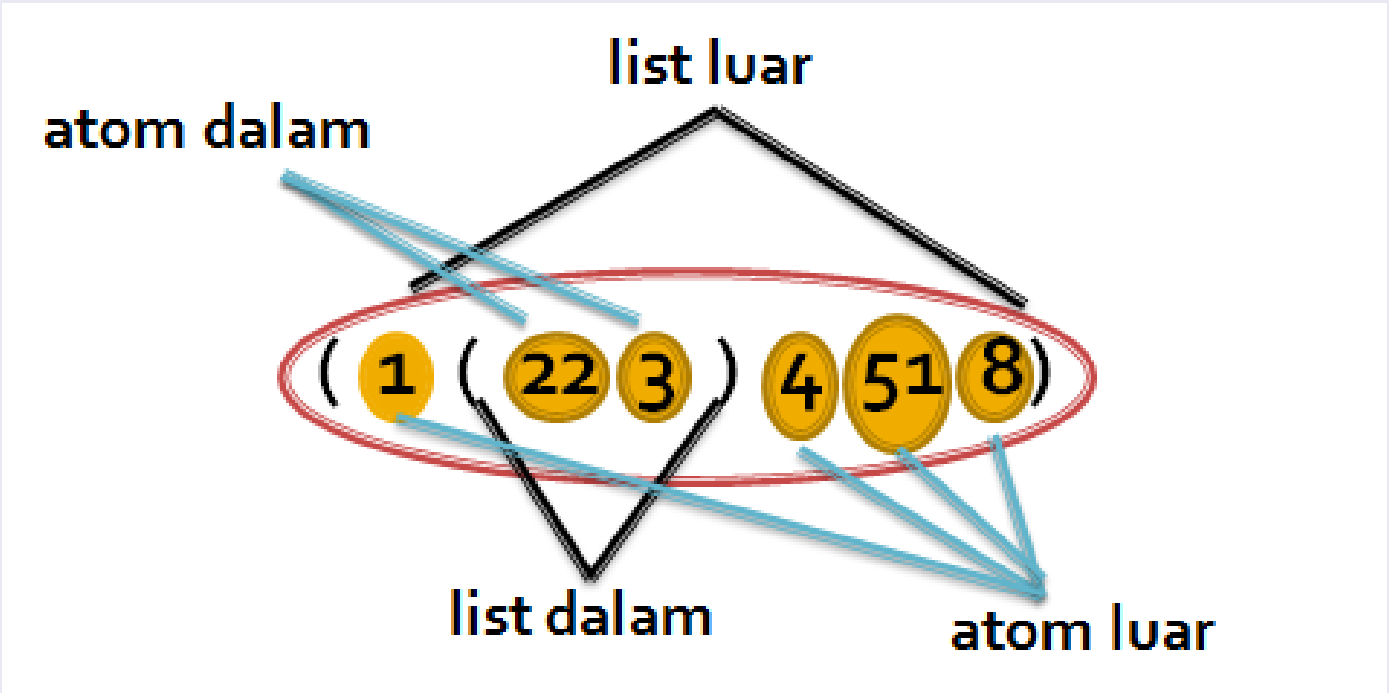
- Lisp (List Programming) adalah bahasa pemrograman fungsional yang asli. Saat ini mempunyai dua varian, yaitu:
  - Common Lisp (Steele, 1990)
  - Scheme (Kelsey, 1998) - www.drscheme.org
- Program ditulis sebagai fungsi.

Bahasa LISP

- Bahasa untuk komputasi simbolik, nilai direpresentasikan dengan ekspresi simbolik
- banyak digunakan di wilayah kecerdasan buatan (robotika, sistem cerdas)
- Biasa dieksekusi dibawah kendali interpreter
- Ekspresi terdiri dari atom atau list.
  Atom $\rightarrow$ string dan karakter (huruf, angka) Contoh : A 68000
  List$\rightarrow$ urutan dari atom atau list, dipisahkan dengan spasi, ditutup dengan tanda kurung
  Contoh : (PLUS A B) ((DAGING AYAM) (SAWI KANGKUNG BAYAM) AIR))

Ekspresi

- List : Segala sesuatu yang dimulai dan diakhiri tanda kurung (. . . . .)
- Atom : Elemen yang berada di dalam list



Ekspresi

- Ekspresi disusun dalam notasi Cambridge-prefix
- Contoh: (+2 2)
- Operasi Aritmatika:
  - (+) → 0
  - (+5) → 5
  - (+54321) → 15
  - (*) → 1
  - (*5) → 5
  - (*12345) → 120
- Contoh lain : (+(*54)(-62)

Ekspresi

- Untuk Percabangan : (if <exp> <then-expr> <else-expr>)
- Contoh : (if (> x y)(-x y)(- y x))