Algoritma & Pemrograman #2

by antonius rachmat c, s.kom, m.cs

Langkah Pembuatan Program

Mendefinisikan masalah

- Menurut hukum Murphy (oleh Henry Ledgard):
  - "Semakin cepat menulis program, akan semakin lama kita dapat menyelesaikannya".
- Sering dilupakan programmer
- Hal tersebut berlaku untuk permasalahan yang kompleks.
- Tentukan masalahnya, apa saja yang harus dipecahkan dengan menggunakan komputer, dan apa inputan serta outputnya.

Langkah Pembuatan Program

Menemukan solusi

- Setelah masalah didefinisikan, maka langkah berikutnya adalah menentukan solusi.
- Jika masalah terlalu kompleks, maka ada baiknya masalah tersebut dipecah menjadi modul-modul kecil agar lebih mudah diselesaikan.
- Dengan penggunaan modul tersebut program utama akan menjadi lebih singkat dan mudah dilihat.

Langkah Pembuatan Program

Memilih algoritma

- Pilihlah algoritma yang benar-benar sesuai dan efisien untuk permasalahan tersebut

Menulis program

- Pilihlah bahasa yang mudah dipelajari, mudah digunakan, dan lebih baik lagi jika sudah dikuasai, memiliki tingkat kompatibilitas tinggi dengan perangkat keras dan platform lainnya.

Langkah Pembuatan Program

Menguji program

- Setelah program jadi, silahkan uji program tersebut dengan segala macam kemungkinan yang ada, termasuk error-handlingnya sehingga program tersebut akan benar-benar handal dan layak digunakan.

Menulis dokumentasi

- Menulis dokumentasi sangat penting agar pada suatu saat jika kita akan melakukan perubahan atau membaca source code
- Caranya adalah dengan menuliskan komentar-komentar kecil

Debbuging

- **Syntax errors:** This type of error occurs if you type a command incorrectly,
  - such as misspelling PRINT as PRRINT or if you forget to type a semicolon at the end of each line in a C++ program.
- **Run-time errors:** These errors occur if your program runs into something unexpected, such as if you ask the user to input an age, the user types a negative number, and your program expects a positive number.
- **Logic errors:** These bugs occur when your instructions work but don’t do exactly what you expected, creating unpredictable results.

Langkah Pembuatan Program

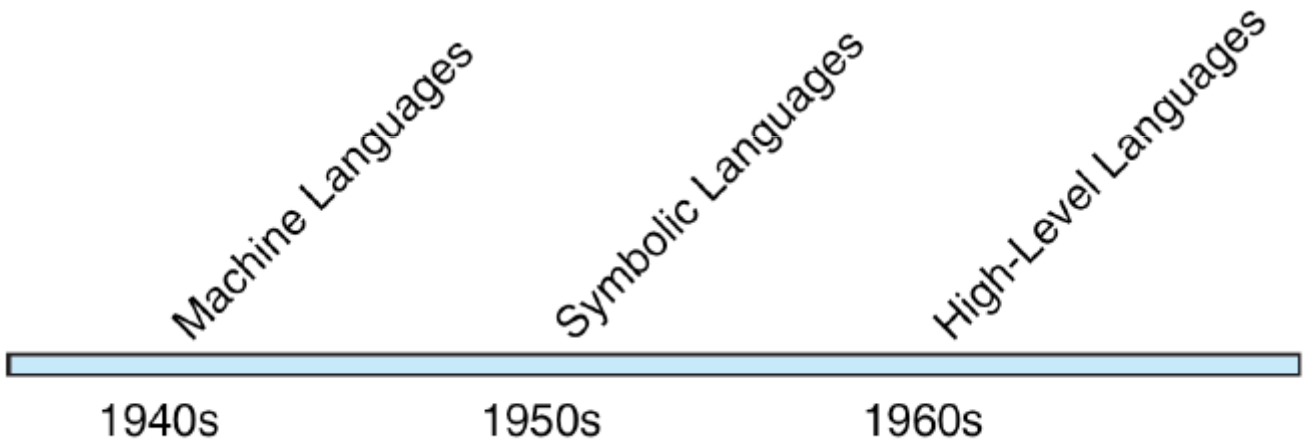
Mendistribusikan aplikasi

- File compression
- Display graphics and play sounds when installing process
- Simplify the copying process

Merawat program

- Program yang sudah jadi perlu dirawat untuk mencegah munculnya bug yang sebelumnya tidak terdeteksi.
- Pengguna membutuhkan fasilitas baru yang dulu tidak ada

Just in time of computer languages

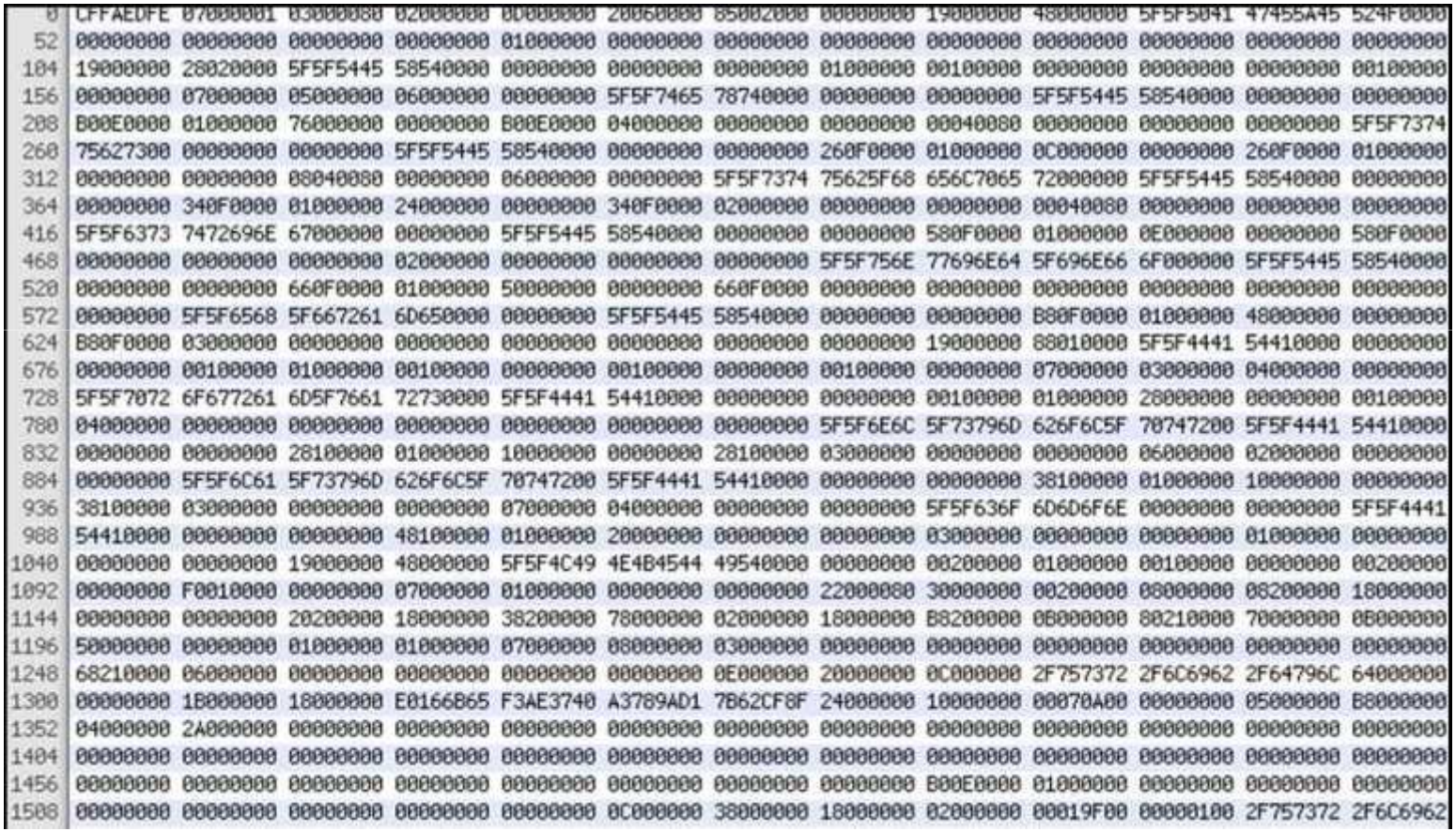


Perkembangan Bahasa Pemrograman

Bahasa Mesin

- Bahasa level terendah
- Isi:
  - kode-kode mesin yg hanya dapat diinterpretasikan langsung oleh mesin komputer
- Berupa kode numerik, biner, dan hexadesimal
- Microcode:
  - sekumpulan instruksi dalam bahasa mesin
- (+) : Eksekusinya cepat
- (-) : Sulit dipelajari manusia

Bahasa Mesin dalam Hexadecimal



0	CFFAEDFE	07000001	03000080	02000000	0D000000	20060000	85002000	00000000	19000000	48000000	5F5F5041	47455A45	524F0000
52	00000000	00000000	00000000	00000000	01000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
104	19000000	28020000	5F5F5445	58540000	00000000	00000000	00000000	01000000	00100000	00000000	00000000	00000000	00100000
156	00000000	07000000	05000000	06000000	00000000	5F5F7465	78740000	00000000	00000000	5F5F5445	58540000	00000000	00000000
208	B00E0000	01000000	76000000	00000000	B00E0000	04000000	00000000	00000000	00040080	00000000	00000000	00000000	5F5F7374
260	75627300	00000000	00000000	5F5F5445	58540000	00000000	00000000	260F0000	01000000	0C000000	00000000	260F0000	01000000
312	00000000	00000000	08040080	00000000	06000000	00000000	5F5F7374	75625F68	656C7065	72000000	5F5F5445	58540000	00000000
364	00000000	340F0000	01000000	24000000	00000000	340F0000	02000000	00000000	00000000	00040080	00000000	00000000	00000000
416	5F5F6373	7472696E	67000000	00000000	5F5F5445	58540000	00000000	00000000	580F0000	01000000	0E000000	00000000	580F0000
468	00000000	00000000	00000000	02000000	00000000	00000000	00000000	5F5F756E	77696E64	5F696E66	6F000000	5F5F5445	58540000
520	00000000	00000000	660F0000	01000000	50000000	00000000	660F0000	00000000	00000000	00000000	00000000	00000000	00000000
572	00000000	5F5F6568	5F667261	6D650000	00000000	5F5F5445	58540000	00000000	00000000	B80F0000	01000000	48000000	00000000
624	B80F0000	03000000	00000000	00000000	00000000	00000000	00000000	00000000	19000000	88010000	5F5F4441	54410000	00000000
676	00000000	00100000	01000000	00100000	00000000	00100000	00000000	00100000	00000000	07000000	03000000	04000000	00000000
728	5F5F7072	6F677261	6D5F7661	72730000	5F5F4441	54410000	00000000	00000000	00100000	01000000	28000000	00000000	00100000
780	04000000	00000000	00000000	00000000	00000000	00000000	00000000	5F5F6E6C	5F73796D	626F6C5F	70747200	5F5F4441	54410000
832	00000000	00000000	28100000	01000000	10000000	00000000	28100000	03000000	00000000	00000000	06000000	02000000	00000000
884	00000000	5F5F6C61	5F73796D	626F6C5F	70747200	5F5F4441	54410000	00000000	00000000	38100000	01000000	10000000	00000000
936	38100000	03000000	00000000	00000000	07000000	04000000	00000000	00000000	5F5F636F	6D6D6F6E	00000000	00000000	5F5F4441
988	54410000	00000000	00000000	48100000	01000000	20000000	00000000	00000000	03000000	00000000	00000000	01000000	00000000
1040	00000000	00000000	19000000	48000000	5F5F4C49	4E4B4544	49540000	00000000	00200000	01000000	00100000	00000000	00200000
1092	00000000	F0010000	00000000	07000000	01000000	00000000	00000000	22000080	30000000	00200000	00000000	08200000	18000000
1144	00000000	00000000	20200000	18000000	38200000	78000000	02000000	18000000	B8200000	06000000	80210000	70000000	0B000000
1196	50000000	00000000	01000000	01000000	07000000	08000000	03000000	00000000	00000000	00000000	00000000	00000000	00000000
1248	68210000	06000000	00000000	00000000	00000000	00000000	0E000000	20000000	0C000000	2F757372	2F6C6962	2F64796C	64000000
1300	00000000	1B000000	18000000	E0166B65	F3AE3740	A3789AD1	7B62CF8F	24000000	10000000	00070A00	00000000	05000000	B8000000
1352	04000000	2A000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
1404	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
1456	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	B00E0000	01000000	00000000	00000000	00000000
1508	00000000	00000000	00000000	00000000	00000000	0C000000	38000000	18000000	02000000	00019F00	00000100	2F757372	2F6C6962

Bahasa Assembly

- Bahasa simbol dari bahasa mesin
- Contoh: ADD, MUL, SUB, DIV

- Macro instruksi:
  - sekumpulan kode dalam bahasa assembly

- (+) : Eksekusi cepat, masih dapat dipelajari daripada bahasa mesin, file kecil
- (-) : Tetap sulit dipelajari, program sangat panjang
- Bisa untuk pembuatan driver, firmware, kernel

Bahasa Assembly

```
 9  Ltmp1:
10      subq    $32, %rsp
11  Ltmp2:
12      movl    %edi, %eax
13      movl    %eax, -4(%rbp)
14      movq    %rsi, -16(%rbp)
15      leaq    L_.str(%rip), %rax
16      movq    %rax, %rdi
17      callq   _puts
18      movl    $0, -24(%rbp)
19      movl    -24(%rbp), %eax
20      movl    %eax, -20(%rbp)
21      movl    -20(%rbp), %eax
22      addq    $32, %rsp
23      popq    %rbp
24      ret
25  Leh_func_end1:
26
27      .section    __TEXT,__cstring,cstring_literals
28  L_.str:
29      .asciz   "Hello World !"
```

Bahasa Tingkat Tinggi

- The 3rd Generation Programming Language

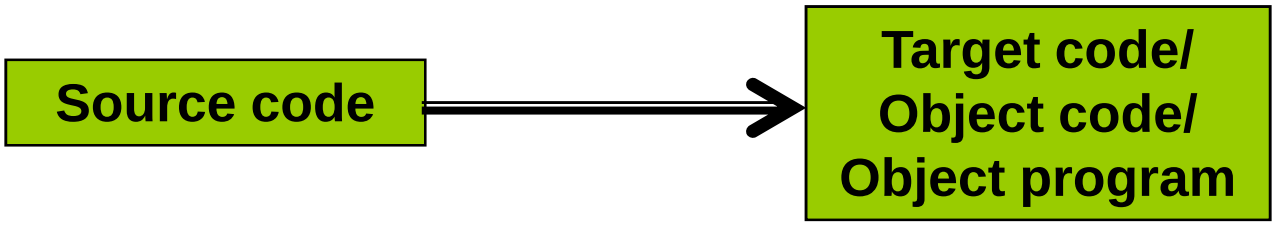
- Lebih dekat dengan bahasa manusia
- Memberi banyak fasilitas kemudahan dalam pembuatan program, mis.: variabel, tipe data, konstanta, struktur kontrol, loop, fungsi, prosedur, dll.
- Contoh: Pascal, Basic, C, Java, PHP

- (+) : Mudah dipelajari, mendekati permasalahan yang akan dipecahkan, kode program pendek
- (-) : Eksekusi lambat

Specific Problem Oriented

- The 4th Generation Programming Language
- Digunakan langsung untuk memecahkan suatu masalah tertentu
- Contoh: SQL untuk database, GUI Programming (Visual Basic.NET, Delphi, Qt)

Translator

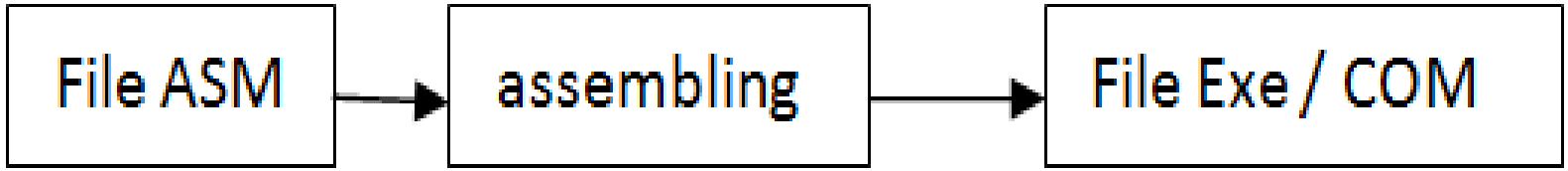


- *Source code*
  - ditulis dengan bahasa pemrograman tertentu
- *Object code*
  - bisa bermacam-macam, tergantung pada *translator*-nya

Macam Translator

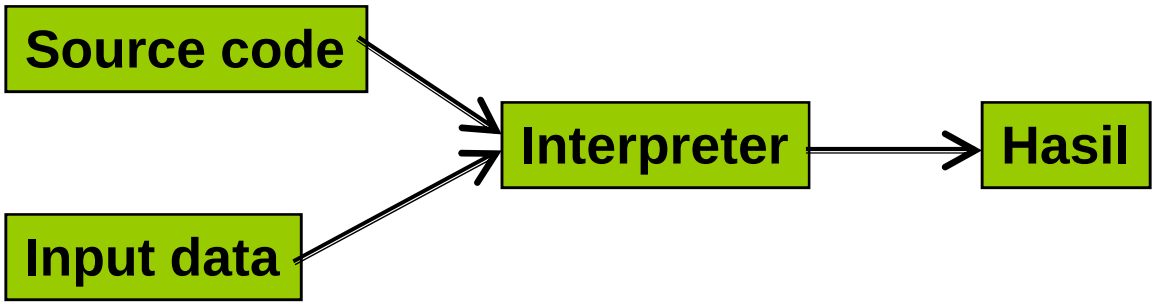
Assembler

- Source code adalah bahasa assembly
- Object code adalah bahasa mesin



Interpreter

- ***Input***
  - *source code* *:* bahasa scripting (PHP, ASP, Basic, dll)
  - masukan program dari *user*
- ***Output***
  - Tidak ada *object code*
  - Translasi internal



Interpreter (2)

- Program tidak harus dianalisis seluruhnya dulu tapi bersamaan dengan jalannya program (saat running)

- (+) :
  - mudah bagi *user*
  - *debugging* cepat

- (-) :
  - eksekusi program lambat
  - tidak langsung menjadi program *executable*

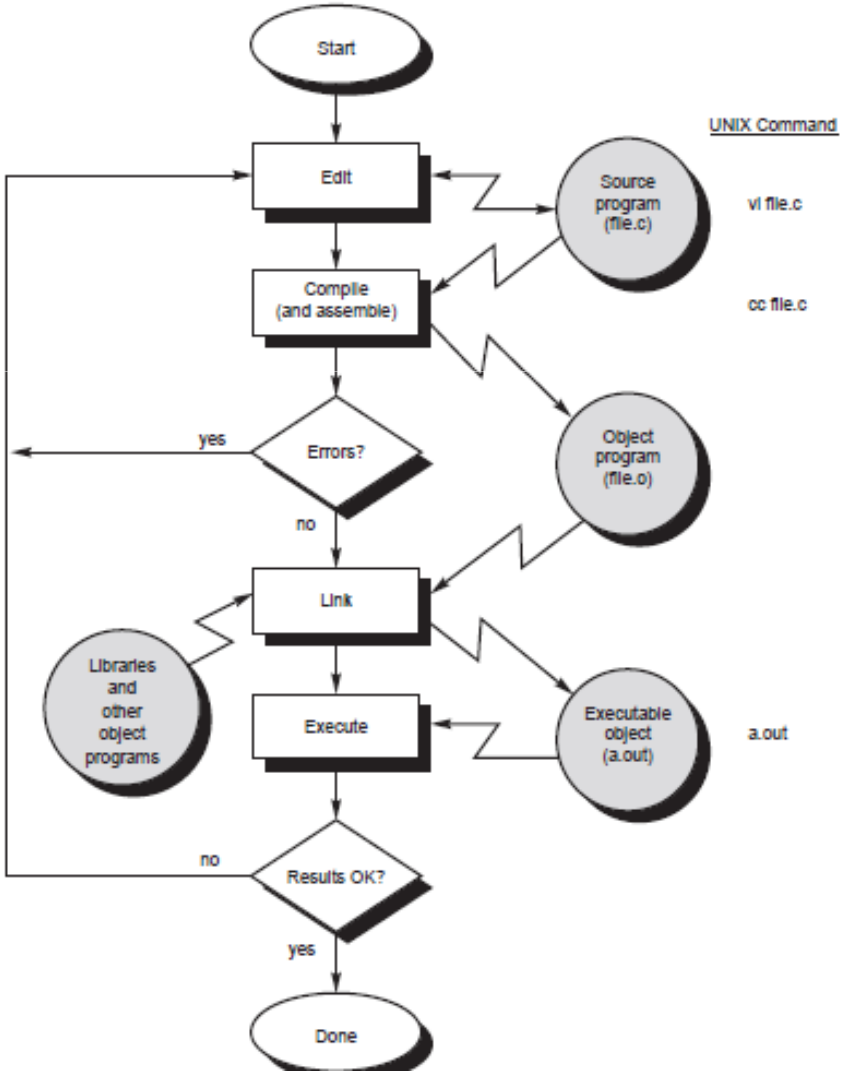
Kompiler

- ***Input***
  - *source code :* bahasa Pascal, C, C++
- ***Output***
  - *object code* : bahasa assembly atau EXE

Kompiler (2)

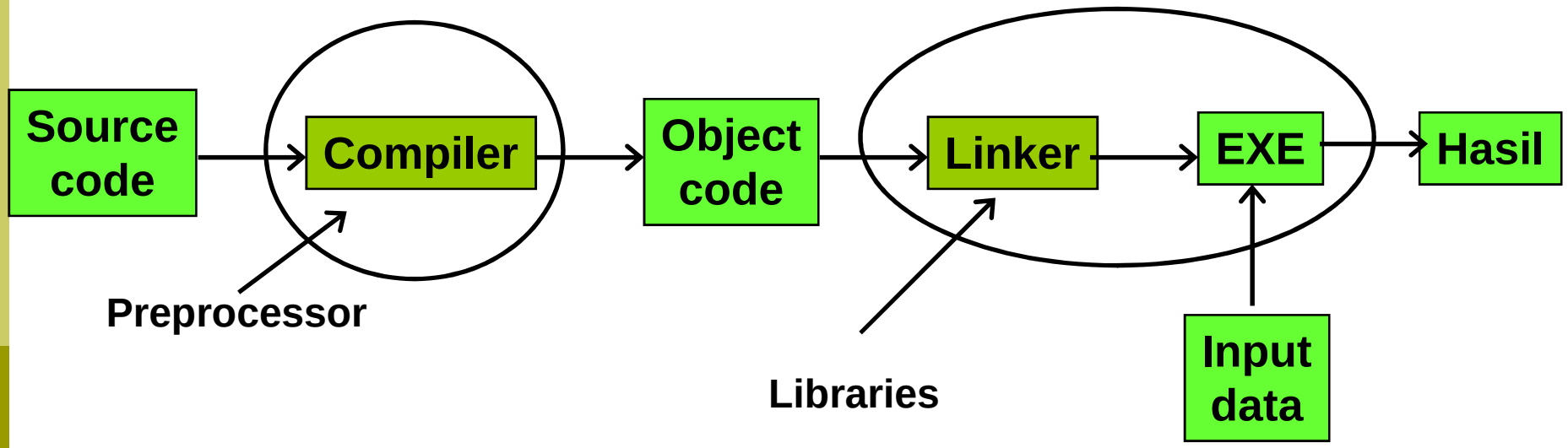
- *Compile time*
  - saat pengubahan *source code* menjadi *object code*
- *Runtime*
  - saat eksekusi *object code*, (dan menerima *input* dari *user*)

Steps in coding, compiling, and executing program



Translator

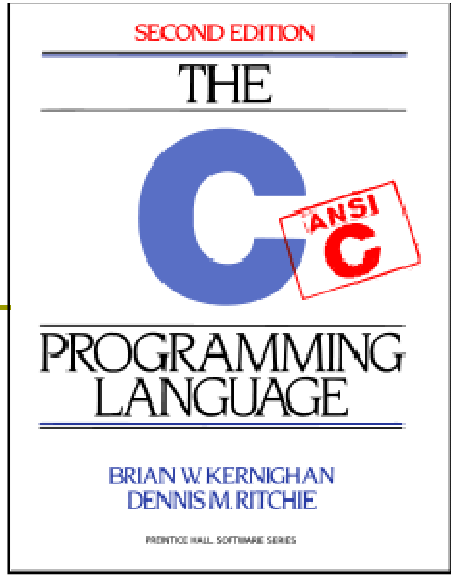
Kompiler (4)



Compile time

Runtime

Bahasa C



- Bahasa pemrograman tingkat tinggi
- 1972:
  - Dirancang oleh **Dennis M Ritchie** di **Bell Laboratories**
- 1978:
  - Dennis dan Brian W. Kernighan mempublikasikan bahasa C melalui “The C Programming Language”
- 1989:
  - Bahasa C distandarisasi **ANSI** (The American National Standard Institute)
  - Standar ISO/IEC 9899:1990 (ANSI C99)

Contoh Program

```
#include <stdio.h>

void main()
{
   printf(“Halo! Selamat Belajar C”);
}
```

```
#include <stdio.h>

int main()
{
   printf(“Halo! Selamat Belajar C”);
   return 0;
}
```

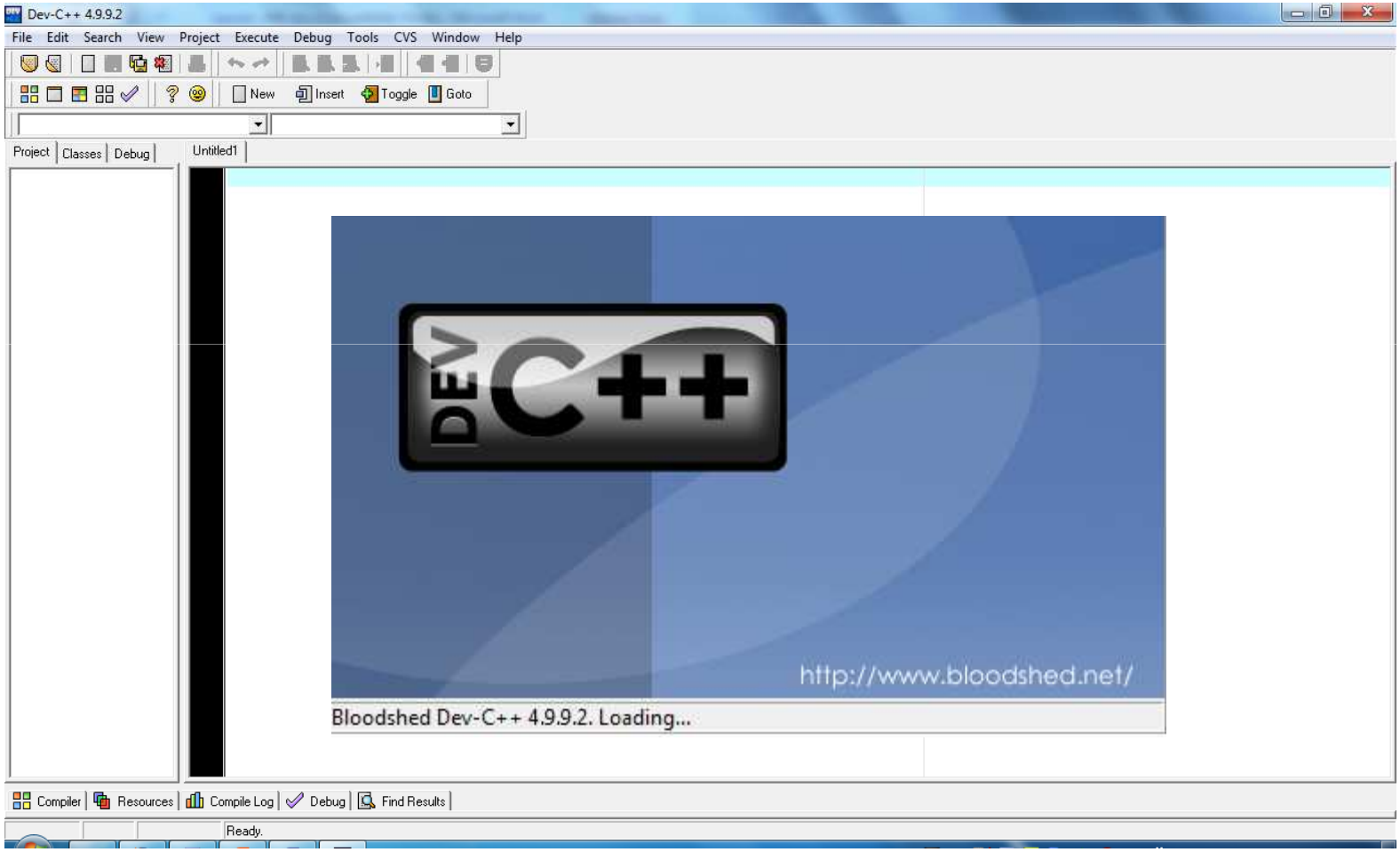
Bahasa C

- Bahasa C dikatakan sebagai bahasa pemrograman **terstruktur**, **prosedural** karena strukturnya menggunakan fungsi-fungsi sebagai bagian program-program (*subroutine / module*).
- Fungsi-fungsi selain **fungsi utama** disebut *subroutine/ module* dan ditulis setelah fungsi utama (**main**) atau diletakkan pada file pustaka (***library***).
- Berekstensi .c
- Dikompilasi menjadi .exe (Windows)

C Compilers

- Dapat dilihat di:
  - http://en.wikipedia.org/wiki/List_of_compilers#C_compilers
- Dev-C++ menggunakan compiler MinGW
  - http://www.mingw.org/
  - Gcc (dan GNU) yang diporting ke Windows
  - Unix-Like

DevC++


Dev-C++ 4.9.9.2
File Edit Search View Project Execute Debug Tools CVS Window Help
New Insert Toggle Goto
Project Classes Debug
Untitled1
DEV C++
http://www.bloodshed.net/
Bloodshed Dev-C++ 4.9.9.2. Loading...
Compiler Resources Compile Log Debug Find Results
Ready.

Bahasa C

- Struktur Program C adalah:
  - Suatu program C minimal harus memiliki function **main()**, tanpa function itu maka program C **tidak dapat dieksekusi** tapi **bisa dikompilasi**.

```
<preprosessor directive>
void main(){
        <statement>;
        <statement>;
        <statement>;
}
```

```
<preprosessor directive>
int main(){
        <statement>;
        <statement>;
        <statement>;
        return 0;
}
```

Statement & Preprosesor Directive

- **Statement** adalah suatu baris instruksi/perintah tertentu.
  - Statement menyebabkan suatu tindakan akan dilakukan oleh komputer.
  - Diakhiri dengan titik koma (;).
- **Preprosessor Directive** adalah bagian yang berisi pengikutsertaan file atau berkas-berkas fungsi, pendefinisian konstanta, atau fungsi makro tertentu.

Contoh suatu program C (2)

```
#include <stdio.h>
int main()
{
    int a,b,c;
    printf(“Isi bilangan pertama:”);
    scanf(“%d”,&a);
    printf(“Isi bilangan kedua:”);
    scanf(“%d”,&b);
    c = a + b;
    printf(“Hasil %d + %d = %d\n",a,b,c);
    return 0
}
```

Statement

Instruksi/ Statement	Tindakan
a = b * c ;	Menghitung
printf(“Antonius Rachmat C”);	Menampilkan literal string
scanf(“%f”,&celcius);	Menerima input data
if(n<0) printf(“negatif”);	Mengendalikan proses

Struktur Program C

- Bagian header berisi: library, tipe data khusus, konstanta, makro
- Selain function ***main()*** dapat ditambahkan function lain

- function sebaiknya ditulis terlebih dahulu sebelum function ***main()***
- Jika tidak harus ditulis judul fungsinya terlebih dahulu diatas fungsi main

```
#include <stdio.h>
int jumlahkan(int a, int b);

int main()
{  printf("Hasil 5 + 3 adalah %d", jumlahkan(5,3));
}

int jumlahkan(int a, int b)
{  return a+b;
}
```

```
#include <stdio.h>

int jumlahkan(int a, int b)
{   return a+b;
}

int main()
{  printf("Hasil 5 + 3 adalah %d", jumlahkan(5,3));
}
```

Identifier

- suatu tempat untuk menyimpan nilai
- Diberi nama unik dan bisa memiliki tipe data
- Dibagi menjadi 2:
  1. Konstanta
  2. Variabel
- Dapat juga merupakan nama suatu elemen dalam program, mis.
  - Nama function
  - Nama prosedur
  - Nama tipe data, dll

Jenis Identifier

1. **Konstanta**
   - Identifier yang nilainya tetap selama program berjalan (dieksekusi)
   - Cara untuk mengubahnya hanya melalui *source code* saja

2. **Variabel**
   - Identifier yang nilainya dapat berubah atau diubah selama program berjalan (dieksekusi)
   - Pengubah: *user* atau proses

Standard Identifier

- **Standard Identifier** adalah identifier-identifier yang biasanya berupa fungsi-fungsi tertentu yang telah diberi makna tertentu oleh compiler bahasa C, tetapi tidak bersifat **reserved** sehingga masih bisa dipakai kembali oleh pemrogram.

```
#include <stdio.h>
#include <conio.h>
int main(){
    printf(“hallo bahasa C”);
    getch();
}
```

ATURAN PENULISAN IDENTIFIER

- Tidak boleh sama dengan nama keyword reserved, function, dan harus unik.
- Maksimum 32 karakter. Bila lebih, maka karakter selebihnya tidak akan diperhatikan oleh komputer.
- Case sensitive : membedakan huruf besar dan kecil
- Karakter pertama harus huruf atau underscore (_), selebihnya boleh angka.
- Tidak boleh mengandung spasi / blank

Keywords

- Adalah identifier yang telah didefinisikan oleh bahasa C secara default
- Sifat:
  - Memiliki arti dan pemakaian tertentu
  - Reserved
  - Ditulis dalam huruf kecil
- Menurut standar ANSI: 32 keywords

Keywords (2)

auto	double	int	switch
break	else	long	typedef
case	enum	register	union
char	extern	return	unsigned
const	float	short	void
continue	for	signed	volatile
default	goto	sizeof	while
do	if	static	struct

Tentang variabel

Penamaan yang salah

`sum$value`	$ is not a valid character.
`piece flag`	Embedded spaces are not permitted.
`3Spencer`	Variable names cannot start with a number.
`int`	`int` is a reserved word.

Camel Case Example:

moneyMadeThisYear = moneyAtEnd – moneyAtStart;

Tipe Data (Basic Types)

Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-128 to 127
int	2 or 4 bytes	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
unsigned int	2 or 4 bytes	0 to 65,535 or 0 to 4,294,967,295
short	2 bytes	-32,768 to 32,767
unsigned short	2 bytes	0 to 65,535
long	4 bytes	-2,147,483,648 to 2,147,483,647
unsigned long	4 bytes	0 to 4,294,967,295

Tipe Data (Basic Types)

Type	Storage size	Value range	Precision
float	4 byte	1.2E-38 to 3.4E+38	6 decimal places
double	8 byte	2.3E-308 to 1.7E+308	15 decimal places
long double	10 byte	3.4E-4932 to 1.1E+4932	19 decimal places

Untuk dapat mengetahui ukuran tipe data dapat digunakan perintah **sizeof(<tipedata>)**

NEXT

- Identifier, Header, Escape Character
- Preprosesor Directive
- Operator
- Komentar
- Input - Ouput