

Bahasa C dan C++

1. Pendahuluan

1.1. Sejarah C & C++

During the 60s, while computers were still in an early stage of development, many new programming languages appeared. Among them, ALGOL 60, was developed as an alternative to FORTRAN but taking from it some concepts of structured programming which would later inspire most procedural languages, such as CPL and its successors (like C++). ALGOL 68 also influenced directly in the development of data types in C. Nevertheless ALGOL was an unspecific language and its abstraction made it little practical to solve most commercial tasks.

In 1963 the CPL (Combined Programming language) appeared with the idea of being more specific for concrete programming tasks of that time than ALGOL or FORTRAN. Nevertheless this same specificity made it a big language and, therefore, difficult to learn and implement.

In 1967, Martin Richards developed the BCPL (Basic Combined Programming Language), that signified a simplification of CPL but kept the most important features the language offered. Although it continued being an abstract and somewhat large language.

In 1970, Ken Thompson, immersed in the development of UNIX at Bell Labs, created the B language. It was a port of BCPL for a specific machine and system (DEC PDP-7 and UNIX), and was adapted to his particular taste and necessities. The final result was an even greater simplification of CPL, although dependent on the system. It had great limitations like it did not compile to executable code but threaded-code, which generates slower code in execution, and therefore was inadequate for the development of an operating system. Reason why from 1971, Denis Ritchie, from the Bell Labs team, began the development of a B compiler which, among other things, was able to generate executable code directly. This "New B", finally called C, introduced in addition, some other new concepts to the language like data types (char).

In 1973, Denis Ritchie, had developed the bases of C. The inclusion of types, its handling, as well as the improvement of arrays and pointers, along with later demonstrated capacity of portability without becoming a high-level language, contributed to the expansion of the C language. It was established with the book "The C Programming Language" by Brian Kernighan and Denis Ritchie, known as the White Book, and that served as de facto standard until the publication of formal ANSI standard (ANSI X3J11 committee) in 1989.

In 1980, Bjarne Stroustrup, from Bell labs, began the development of the C++ language, that would receive formally this name at the end of 1983, when its first manual was going to be published. In October 1985, the first commercial release of the language appeared as well as the first edition of the book "The C++ Programming Language" by Bjarne Stroustrup.

During the 80s the C++ language was being refined until it became a language with its own personality. All that with very few losses of compatibility with the code with C, and without resigning to its most important characteristics. In fact, the ANSI standard for the C language published in 1989 took good part of the contributions of C++ to structured programming.

From 1990 on, ANSI committee X3J16 began the development of a specific standard for C++. In the period elapsed until the publication of the standard in 1998, C++ lived a great expansion in its use and today is the preferred language to develop professional applications on all platforms.

source: <http://www.cplusplus.com/info/history.html>

2. Pembahasan

Kalian sudah mengenal bahasa pemrograman Pascal kan? Jadi adalah mudah untuk memahami bahasa pemrograman lain. Contohnya adalah bahasa C dan C++ ini. Untuk memulainya, mari kita lihat contoh program (yang sudah me-'legenda') Hello World:

Pascal:
<pre>begin writeln('Hello, world.');</pre>
C:
<pre>#include <stdio.h> int main() { printf("Hello, world.\n"); return 0; }</pre>
C++:
<pre>#include <iostream> int main() { std::cout << "Hello, world.\n"; return 0; }</pre>

2.1. Gambaran Umum Sebuah Program

```
Program Pascal;
{ini bagian 'header' program, berisi library-library yang
diperlukan ataupun komentar-komentar ga jelas...}

uses crt;

{berikut adalah bagian deklarasi kostanta, type bentukan, dan
variabel global yang akan digunakan}
const
  pi = 3.1415926;
  {...;}
type
  angka : integer;
  {...;}
var
  i, j : integer;
  {...;}

{berikut adalah bagian deklarasi fungsi-fungsi dan prosedur yang
akan dipakai}
function kuadrat(x : integer) : longint;
begin
  kuadrat := x*x;
end;

procedure tukar(var a:integer, var b:integer);
var temp : integer;
begin
  temp := a;
  a := b;
  b := temp;
end;

{akhirnya, main program}
begin
  writeln('Selamat datang');
  {...;}
end;
```

Inilah Pascal yang selama ini kalian kenal. Sudah tidak asing kan dengan source code di atas? Coba bandingkan dengan source code C dan C++ di bawah ini.

```
/* C */
/*ini bagian 'header' program, berisi library-library yang
diperlukan ataupun komentar-komentar ga jelas...*/
#include <stdio.h>
#include <string.h>
#include <math.h>
/* ... */

/*berikut adalah bagian deklarasi kostanta, type bentukan, dan
variabel global yang akan digunakan*/
const float pi = 3.1415926;
/* const ...; */
typedef int angka;
/* typedef ...; */
int i, j;
/* ...; */

/*berikut adalah bagian deklarasi fungsi-fungsi dan prosedur
yang akan dipakai*/
long int kuadrat(int x)
{
    return (x*x);
}

void tukar(int *a, int *b);
int temp;
{
    temp := *a;
    *a := *b;
    *b := temp;
}

/*akhirnya, main program*/
int main()
{
    printf("Selamat datang!\n");
    /* ...; */
    return 0;
}
```

Beberapa hal yang perlu diingat mengenai bahasa C:

- Komentar dalam C menggunakan `/* ... */` dan bisa memuat beberapa baris sekaligus.
- Sebuah program dalam C harus memiliki fungsi bernama `main()`, fungsi ini adalah fungsi spesial karena di situlah main program kita.
- Dalam C sebenarnya tidak ada procedure, procedure didefinisikan sebagai function yang mengembalikan nilai void (void = sesuatu yang tidak ada).
- Setiap baris perintah harus diakhiri dengan titik koma `;` kecuali yang diawali dengan kres `#`.
- Blok program diawali kurung-kurawal-buka `{` dan diakhiri kurung-kurawal-tutup `}`.
- Bahasa C adalah bahasa yang case-sensitive, jadi perhatikan dalam penamaan variabel, type, maupun pemanggilan fungsi-fungsi yang sudah ada.
- Perintah terakhir di dalam `main()` yaitu `return 0` digunakan untuk mengindikasikan bahwa program tersebut sukses berjalan. Pada beberapa kontes pemrograman, hal ini merupakan salah satu syarat diterimanya program. (Program yang tidak me-return exit code 0 dianggap mengalami runtime error.)

```

//C++:
//ini bagian 'header' program, berisi library-library yang
//diperlukan
#include <iostream>
#include <fstream>
#include <string>
//#include ...
//...

//berikut adalah bagian deklarasi konstanta, type bentukan,
//dan variabel global yang akan digunakan
const float pi = 3.1415926;
const ...;
typedef int angka;
typedef ...;
int i,j;
//...;

//berikut adalah bagian deklarasi fungsi-fungsi
//dan prosedur yang akan dipakai
long int kuadrat(int x)
{
    return (x*x);
}

void tukar(int *a, int *b);
int temp;
{
    temp := *a;
    *a := *b;
    *b := temp;
}

//akhirnya, main program
int main()
{
    std::cout << "Selamat datang!" << endl;
    //...;
    return 0;
}

```

Beberapa hal yang perlu diingat mengenai bahasa C++:

- Komentar dalam C++ menggunakan /* ... */ untuk memuat beberapa baris sekaligus atau menggunakan // untuk mengabaikan semua karakter di sebelah kanannya sebanyak 1 baris.
- Sebuah program dalam C++ harus memiliki fungsi bernama main(), fungsi ini adalah fungsi spesial karena di situlah main program kita.
- Dalam C++ sebenarnya juga tidak ada procedure, procedure didefinisikan sebagai function yang mengembalikan nilai void..
- Setiap baris perintah harus diakhiri dengan titik koma ';' kecuali yang diawali dengan kres '#'.
- Blok program diawali kurung-kurawal-buka '{' dan diakhiri kurung-kurawal-tutup '}'.
- Bahasa C++ juga bahasa yang case-sensitive, jadi perhatikan dalam penamaan variabel, type, maupun pemanggilan fungsi-fungsi yang sudah ada.
- Perintah terakhir di dalam main() yaitu return 0 digunakan untuk mengindikasikan bahwa program tersebut sukses berjalan.

2.2.Type, Variabel, dan Konstanta

2.2.1. Type

Type data numerik bilangan bulat:

Nilai Min	Nilai Max	Pascal	C	C++
-128	127	shortint	char	char
-32,768	32,767	integer	int	int
-2,147,483,648	2,147,483,647	longint	long	long
0	255	byte	unsigned char	unsigned char
0	65,535	word	unsigned int	unsigned int
0	4,294,967,295	longword	unsigned long	unsigned long

Type data numerik bilangan desimal:

Nilai negatif	Nilai positif	Pascal	C	C++
-3.4×10^{-38} s/d -3.4×10^{-38}	3.4×10^{-38} s/d $3.4 \times 10^{+38}$	real	float	float
-1.7×10^{-308} s/d -1.7×10^{-308}	1.7×10^{-308} s/d $1.7 \times 10^{+308}$	double	double	double

Ingat, seringkali terjadi ketidaktepatan pada representasi bilangan desimal. Untuk mengetes kesamaan nilai 2 buah bilangan, selalu gunakan perbandingan dengan epsilon (sebuah nilai yang sangat kecil). Contoh: untuk mengetes apakah bilangan desimal r1 sama dengan r2, gunakan: $\text{abs}(r2-r1) < \text{epsilon}$, dengan epsilon (misalnya) $1e-04$.

Type data karakter:

- C: char
- C++: char
- Pascal char

Type data boolean:

Pascal	C	C++
type kondisi : boolean;	tidak ada	typedef bool kondisi;

- Pada C++, **bool** dapat berisi **true** ataupun **false**, mirip Pascal.
- Pada C, tidak ada type boolean. Sebagai gantinya type bilangan bulat apapun bisa dipakai sebagai ganti boolean dengan aturan false adalah 0 dan true adalah angka selain 0.

Type data array:

Pascal	C	C++
Type tabel = array[0..10] of integer;	typedef int tabel[11];	typedef int tabel[11];
type matrix = array[0..20,0..30] of real;	typedef float matrix[21][31];	typedef float matrix[21][31];

- Format deklarasinya : **typedef** <type-elemen> <nama-type>[<banyaknya elemen + 1>];
- Pada C dan C++ indeks array selalu dimulai dari 0.
- Jadi jika ditulis **int** tabel[11] artinya ada tabel[0] ... tabel[10].
- Contoh cara pengaksesan : tabel[2], matrix[5][29]

Type data string:

Pascal	C	C++
<code>type name = string[30];</code>	<code>typedef char name[31];</code>	<code>typedef std::string name;</code>

- Untuk C, tidak ada type data khusus string, sehingga string didefinisikan sebagai array of char.
- Pada Pascal, `name:string[30]` artinya karakter diisi dari `name[1]..name[30]` sedangkan dalam C `char name[31]` artinya karakter diisi dari `name[0]..name[30]`.
- Untuk C++, untuk menggunakan type string harus ada `#include <string>` di awal program harus menggunakan `std::string`.

Type data struct (record):

Pascal	C	C++
<pre>type siswa = record nama:string[30]; kelas:integer; end;</pre>	<pre>typedef struct { char nama[31]; int kelas; } siswa;</pre>	<pre>typedef struct { std::string nama; int kelas; } siswa;</pre>

- Misal deklarasi variabel: `siswa anto;`
- Maka akses ke elemennya menggunakan titik '.' contoh: `anto.nama`, `anto.kelas`

Type data enumerasi:

Pascal	C	C++
<pre>type hari = (senin, selasa, rabu, kamis, jumat, sabtu);</pre>	<pre>typedef enum { senin, selasa, rabu, kamis, jumat, sabtu; } hari;</pre>	<pre>typedef enum { senin, selasa, rabu, kamis, jumat, sabtu; } hari;</pre>

- Jadi format: `typedef enum {<elemen-elemen> <nama-type>;`
- Untuk contoh di atas, pada C dan C++ `senin` akan bernilai 0, `selasa=1`, `rabu=2` dst... jika ditulis `typedef enum { senin = 3, selasa = 7, rabu, kamis, jumat, sabtu = 20; } hari;` maka `senin` bernilai 3, `selasa=7`, `rabu=8`, `kamis=9`, `jumat=10`, `sabtu=20`.
- Jadi jika nilai pada elemen tersebut tidak didefinisikan, akan dianggap +1 dari elemen sebelumnya.

Type data pointer:

Pascal	C	C++
<pre>type address : ^integer; type addr : ^siswa; {type siswa ada di atas}</pre>	<pre>typedef int *address; typedef siswa *addr; /*type siswa ada di atas*/</pre>	<pre>typedef int *address; typedef siswa *addr; //type siswa ada di atas</pre>

- Ketika kita menggunakan Pascal, kita hampir tidak perlu berurusan dengan pointer, tapi di C mau tidak mau kita harus berurusan dengan tipe data satu ini.

- Pointer tidak menyimpan nilai, pointer hanya menyimpan alamat memory dari nilai yang disimpan, jadi bukan nilai itu sendiri.
- Misal address P; maka `*P = 10` akan membuat nilai yang ditunjuk oleh P berubah menjadi 10, tapi P itu sendiri tidak berubah, P tetap menunjuk alamat yang sama.

2.2.2. Variabel

Syntax untuk deklarasi dan assignment variabel dalam C & C++ berbeda dengan Pascal.

Pascal	C	C++
<pre>angka : integer; hasil : real; tabel: array[0..10] of char; nama: string[30];</pre>	<pre>int angka; float hasil; char tabel[11]; char nama[31];</pre>	<pre>int angka; float hasil; char tabel[11]; std::string nama;</pre>
<pre>angka := 10; hasil := 23.98; tabel[7] := 'S'; nama := 'alexander';</pre>	<pre>angka = 10; hasil = 23.98; tabel[7] = 'S'; strcpy(nama,"alexander");</pre>	<pre>angka = 10; hasil = 23.98; tabel[7] = 'S'; nama = "alexander";</pre>

- Jika pada Pascal bagian untuk deklarasi variabel harus diawali dengan `var` dan berada sebelum main program, hal ini tidak ada dalam C & C++, kita bisa mendeklarasikan variabel di mana pun.
- Aturan penamaan variabel:
 - o case-sensitive
 - o diawali dengan huruf
 - o boleh terdiri dari huruf, angka, atau underscore '_'
 - o tidak boleh ada variabel ganda, maupun konflik dengan reserved word
- Operator untuk assignment adalah '=' jadi jangan sampai tertukar dengan operator komparasi '=='.
- Pada C & C++ kita bisa menyingkat deklarasi dan assignment sekaligus, contoh: `int angka = 10; float hasil = 23.98;`
- Format deklarasi variabel sangat mirip dengan format deklarasi type. Bedanya hanya tidak adanya kata kunci `typedef`.

2.2.3. Konstanta

Pascal	C	C++
<pre>const pi = 3.1415926;</pre>	<pre>Ada 2 cara: const float pi = 3.1415926; atau: #define pi 3.1415926</pre>	<pre>Ada 2 cara: const float pi = 3.1415926; atau: #define pi 3.1415926</pre>

- Pada C & C++ konstanta biasa harus tetap dideklarasikan type-nya.
- `#define` adalah makro, di mana fungsinya mirip find&replace pada text editor.
- Formatnya: `#define <nama> <nilai>`, cara kerjanya adalah mencari <nama> di seluruh code dan mengubahnya menjadi <nilai>.

2.3. Input dan Output

2.3.1. Output

Pascal	C	C++
<pre>write('Apa kabar?'); writeln('Baik-baik saja');</pre>	<pre>printf("Apa kabar?"); printf("Baik-baik saja\n");</pre>	<pre>std::cout << "Apa kabar?"; std::cout << "Baik-baik saja\n"</pre>
<pre>var i: integer; ...</pre>	<pre>int i; ...</pre>	<pre>int i; ...</pre>

<code>writeln('nilai i adalah ', i);</code>	<code>printf("nilai i adalah %d\n", i);</code>	<code>std::cout << "nilai i adalah " << i << "\n";</code>
<code>var c:char; f: real;</code>	<code>char c; float f;</code>	<code>char c; float f;</code>
<code>... writeln('c = ', c, ' f = ', f);</code>	<code>... printf("c = %c f = %f\n", c, f);</code>	<code>... std::cout << "c = " << c << "f = " << f << "\n";</code>

- Untuk C, perlu **#include** <stdio.h> di awal program.
- Untuk C, output menggunakan fungsi **printf** (print formatted)
Formatnya: **printf**("<string beserta format variabel>", <variabel>, <variabel>...);
Jadi, bentuk dari keluaran dituliskan dulu formatnya sebagai parameter pertama printf dalam bentuk string (diapit oleh petik). Yang membedakan text biasa dengan format variabel adalah tanda %.

Daftar format yang ada:

%i atau %d	int
%li atau %ld	long
%u	unsigned int
%ul	unsigned long
%c	char
%f	float
%lf	long float
%s	string

- Selain format, ada karakter-karakter khusus, contoh: '\n' untuk akhir baris

\a	menimbulkan suara 'beep'
\b	karakter backspace
\n	karakter ENTER (newline)
\r	kembali ke kolom paling awal di baris yang sama
\t	karakter tab
\\	karakter \ (backslash)
\?	karakter ? (question mark)
\'	karakter ' (petik tunggal)
\"	karakter " (petik)
\ooo	karakter ASCII urutan ke ooo (dalam octal)
\xhh	karakter ASCII urutan ke hh (dalam heksa)

- Untuk C++, perlu **#include** <iostream> di awal program.
- Untuk C++, output menggunakan **std::cout**
- Karakter-karakter khusus pada C juga berlaku pada C++.

2.3.2. Input

Pascal	C	C++
<code>var i: integer;</code>	<code>int i;</code>	<code>int i;</code>
<code>f: real;</code>	<code>float f;</code>	<code>float f;</code>
<code>cc: char;</code>	<code>char cc;</code>	<code>char cc;</code>
<code>read(i);</code>	<code>scanf("%d", &i);</code>	<code>std::cin >> i;</code>
<code>read(f);</code>	<code>scanf("%f", &f);</code>	<code>std::cin >> f;</code>
<code>read(cc);</code>	<code>scanf("%c", &cc);</code>	<code>std::cin >> cc;</code>

- Untuk C, perlu **#include** <stdio.h> di awal program.
- Untuk C, input menggunakan **scanf** (scan formatted).
Format yang digunakan hampir sama dengan **printf**, namun setelah string format, yang

diisi bukan nama variabel, melainkan pointer dari array tersebut. Makanya untuk memasukkan nilai desimal (%f) kedalam variabel f digunakan pointer f (&f).

- Untuk C++, perlu **#include** <iostream> di awal program.
- Untuk C++, input menggunakan **std::cin**.

2.4. Operator

2.4.1. Operasi Bilangan Bulat

Operator yang digunakan pada bilangan bulat dan menghasilkan bilangan bulat juga:

Definisi	Pascal	C	C++
Penjumlahan	+	+	+
Pengurangan	-	-	-
Perkalian	*	*	*
Hasil bagi	div	/	/
Sisa bagi	mod	%	%
Increment	inc(x);	x++; atau ++x;	x++; atau ++x;
x < x + a	inc(x, a);	x+=a;	x+=a;
Decrement	dec(x);	x--; atau --x;	x--; atau --x;
x < x - a	dec(x, a);	x-=a;	x-=a;

- Ada perbedaan antara x++ dengan ++x dan antara x-- dengan --x meskipun hasil akhirnya sama.
- Jika dijalankan: `int x = 1; printf("%d", x++);` maka yang keluar adalah angka 1. Nilai x setelah itu adalah 2.
- Tapi jika dijalankan: `int x = 1; printf("%d", ++x);` maka yang keluar adalah 2. Nilai x setelah itu adalah 2.
- Begitu juga dengan x-- dan --x.

2.4.2. Operasi Pecahan

Operator yang digunakan pada bilangan pecahan dan menghasilkan pecahan juga:

Definisi	Pascal	C	C++
Penjumlahan	+	+	+
Pengurangan	-	-	-
Perkalian	*	*	*
Pembagian	/	/	/

- Pada C & C++, operator pembagi '/' menggunakan lambang yang sama. Jika / dioperasikan pada 2 buah bilangan bulat, maka otomatis menjadi operasi div, sedangkan jika salah satu operand-nya pecahan maka otomatis menjadi operasi pembagian biasa.

2.4.3. Operasi Boolean

Operator yang digunakan untuk memanipulasi true dan false:

Definisi	Pascal	C	C++
Negasi	not	!	!
Konjungsi	and	&&	&&
Disjungsi inklusif	or		
Disjungsi eksklusif	xor	^	^

2.4.4. Operasi Perbandingan

Operator yang digunakan untuk membandingkan nilai bilangan dengan bilangan lain:

Definisi	Pascal	C	C++
Sama dengan	=	==	==
Tidak sama dengan	<>	!=	!=
Kurang dari	<	<	<
Kurang dari sama dengan	<=	<=	<=
Lebih dari	>	>	>
Lebih dari sama dengan	>=	>=	>=

- Jangan sampai tertukar antara operator assignment '=' dengan operator pembandingan '=='. Hal ini sering terjadi pada orang yang terbiasa dengan Pascal dan beralih ke C.

2.5. Kondisional

2.5.1. If..then dan if..then..else

Pascal	C	C++
<pre>if (a > b) then begin a := b; end;</pre>	<pre>if (a > b) { a = b; }</pre>	<pre>if (a > b) { a = b; }</pre>
<pre>if ((y mod x) = 0) then begin write('Divisible'); end else begin write('Not divisible'); end;</pre>	<pre>if ((y % x) == 0) { printf("Divisible"); } else { printf("Not divisible"); }</pre>	<pre>if ((y % x) == 0) { std::cout<<"Divisible"; } else { std::cout<<"Not divisible"; }</pre>

- Syntax if..then..else dalam C & C++ sama persis.
- Bedanya dengan Pascal adalah tidak adanya keyword **then** jadi instruksi setelah if() akan dilaksanakan jika true.

2.5.2. Case..of

Pascal	C	C++
<pre>case bulan of 1 : begin ... end; 2 : begin .. end; ... end;</pre>	<pre>switch (bulan) { case 1: ... break; case 2: ... break; ... }</pre>	<pre>switch (bulan) { case 1: ... break; case 2: ... break; ... }</pre>
<pre>case bulan of 1, 3, 5, 7, 8, 10, 12: begin days := 31; end; 4, 6, 9, 11: begin days := 30; end; 2: begin ... end;</pre>	<pre>switch (bulan) { case 1: case 3: ... case 12: days = 31; break; case 4: ... case 11: days = 30; break; }</pre>	<pre>switch (bulan) { case 1: case 3: ... case 12: days = 31; break; case 4: ... case 11: days = 30; break; }</pre>

<pre>end;</pre>	<pre>case 2: ... }</pre>	<pre>case 2: ... }</pre>
-----------------	----------------------------	----------------------------

- Syntax case..of pada C & C++ sama persis.
- Pada C & C++, harus menggunakan keyword switch dan case.
- Pada C & C++ harus digunakan perintah break karena jika tidak ada break, switch akan membaca sisa perintah yang ada di bawahnya.

2.5.3. Operator ? Ternary

Ada cara lain untuk menuliskan if..then..else yaitu dengan operator ternary '?'.
Formatnya: <kondisi> ? <if-true> : <if-false>

Contoh: x = (a < b) ? a : b;

Contoh: x = (a < b) ? a : b;

jika a = 5 dan b = 7 maka baris ini akan menjadi x = a;

tapi jika a = 10 dan b = 5 maka baris ini akan menjadi x = b;

Baris di atas setara dengan **if** (a < b) {x = a;} **else** {x = b;};

Keunggulan dari ternary adalah singkat, tetapi kelemahannya adalah tidak bisa untuk kondisi yang agak kompleks.

2.6. Pengulangan

2.6.1. Loop For

Pascal	C	C++
<pre>var i: integer; ... for i := 1 to 10 do begin ... end;</pre>	<pre>int i; ... for (i = 1; i <= 10; i++) { ... }</pre>	<pre>int i; ... for (i = 1; i <= 10; i++) { ... }</pre>

- Syntax for dalam C & C++ sama persis.
- Format: **for** (<initial value>; <stop condition>; <incremental>) {}

2.6.2. Loop While

Pascal	C	C++
<pre>var found: boolean; ... while (not found) do begin ... end;</pre>	<pre>unsigned char found; ... while (!found) { ... }</pre>	<pre>bool found; ... while (!found) { ... }</pre>

- Syntax while dalam C & C++ sama persis.
- Format: **while** (<kondisi-lanjut>) {}
- Tidak ada keyword **do** pada while, deretan perintah setelah while() itulah yang akan dieksekusi.

2.6.3. Loop Repeat

Pascal	C	C++
<pre>repeat ... until (i > j);</pre>	<pre>do ... }</pre>	<pre>do ... }</pre>

```
while (i <= j); while (i <= j);
```

- Repeat..until dalam C & C++ adalah do..while dan syntaxnya sama untuk C maupun C++.
- Format: `do {} while (<kondisi-lanjut>);`
- Keyword `do..while` tidak bertindak sebagai begin..end sehingga diperlukan { dan } jika ingin mengeksekusi beberapa baris perintah.

2.7.Fungsi dan Prosedur

2.7.1. Fungsi

Pascal	C	C++
<pre>function max(a,b:integer): integer; begin if (a > b) then begin max := a; end else begin max := b; end; end;</pre>	<pre>int max(int a, int b) { if (a > b) return a; else return b; }</pre>	<pre>int max(int a, int b) { if (a > b) return a; else return b; }</pre>

- Format deklarasi suatu fungsi pada C & C++ sama.
- Format: `<type-keluaran> <nama-fungsi>(<type-var> <nama-var> [, <type> <nama>, ...]) {}`
- Perhatikan bahwa di akhir deklarasi nama fungsi tidak ada titik-koma ‘;’.
- Yang membedakan antara fungsi dan variabel adalah fungsi harus ditulis menggunakan ‘()’ meskipun tidak ada parameter.

2.7.2. Prosedur

Pascal	C	C++
<pre>procedure tulis(n : integer); var i : integer; begin for i := 1 to n do writeln('i=', i); end;</pre>	<pre>void tulis(int n) { int i; for (i = 1; i <= 10; i++) printf("i=%d\n", i); }</pre>	<pre>void tulis(int n) { int i; for (i = 1; i <= 10; i++) std::cout<<"i="<<i<< "\n"; }</pre>
<pre>procedure tambah2 (var n : integer); begin n := n + 2; end;</pre>	<pre>void tambah2(int * n) { *n = *n + 2; } /* pemanggilan tambah2(&n); */</pre>	<pre>void tambah2(int & n) { n = n + 2; } /* pemanggilan tambah2(n); */</pre>

- Sebenarnya tidak ada prosedur dalam C maupun C++.
- Prosedur didefinisikan sebagai fungsi yang tidak mengembalikan nilai (void).
- Arti dari type `void` adalah tidak ada nilai yang dikembalikan.
- Di dalam bahasa C, untuk passing variabel ke dalam sebuah fungsi hanya bisa mempassing nilai variabel tersebut (passing by value), tidak bisa mempassing variabel

tersebut.

Oleh karena itu yang di-pass ke dalam fungsi adalah sebuah pointer dari sebuah variabel (`int * n`). Untuk mempass variabel tersebut, kita memanggil fungsi dan memasukkan alamat dari variabel yang ingin diubah nilainya. (`&n`). Di dalam fungsi kita memakai isi dari variabel yang ditunjuk oleh pointer. (`*n`)

- Di dalam C++ sudah ada metode untuk mem-passing variabel yakni dengan tipe referensi.

2.8.Type Casting

Untuk melakukan operasi antara 2 atau lebih variabel yang ber-type berbeda, diperlukan konversi agar semua variabel tersebut memiliki type yang sama, baru setelah itu dapat dioperasikan.

Contoh, potongan source dalam C & C++:

```
int x = 121;
float y;
```

```
y = x / 19;
```

maka y akan berisi nilai 6 karena operator ‘/’ akan menjadi div jika dioperasikan pada 2 bilangan bulat. Pemecahannya, digunakan typecasting, yaitu mengubah suatu variabel/nilai sehingga menjadi type sesuai kemauan kita. Namun perubahan type ini hanya sementara.

Contoh:

```
int x = 121;
float y;
```

```
y = (float)x / (float)19;
```

sekarang y akan berisi 6.368421 karena operator ‘/’ dilakukan pada 2 bilangan desimal. Namun variabel x tetap bertipe integer.

Format typecasting: `(<nama-type>)<nama-variabel-atau-nilai>`

`<nama-type>` dalam hal ini tidak harus type dasar yang sudah ada, bisa juga merupakan type bentukan yang kita buat.

Contoh:

`typedef enum`

```
{
  minggu, senin, selasa, rabu, kamis, jumat, sabtu;
} hari;
hari h = (hari)1;
```

maka h berisi senin karena senin equal dengan nilai 1 dan angka 1 tersebut diubah menjadi type hari ketika di-assign ke h.

Pada kasus casting dari char ke int, akan diberikan nilai ASCII dari karakter tersebut. (misalnya, ‘A’ akan menjadi angka 65 karena A ada di urutan 65 dalam tabel ASCII)

3. Penutup

3.1. Tabel Singkat Perbandingan syntax Pascal, C, dan C++

Deskripsi	Pascal	C	C++
assignment	<code>var x:integer; x := a+b;</code>	<code>int x = a+b;</code>	<code>int x = a+b;</code>
increment	<code>inc(i) atau i := i+1</code>	<code>i++ atau ++i</code>	<code>i++ atau ++i</code>
output	<code>writeln('i=',i);</code>	<code>printf("i=%d\n",i);</code>	<code>std::cout<<"i="<<i<<"\n";</code>
input	<code>read(i);</code>	<code>scanf("%d",&i);</code>	<code>std::cin>>i</code>
operasi numerik	<code>var i:integer; i div 10 i mod 10 i / 10</code>	<code>int i; i % 10 i / 10 (float)i / 10.0</code>	<code>int i; i % 10 i / 10 (float)i / 10.0</code>
kondisional	<code>if (a <> b) then begin sama := false; end else begin sama := true; end;</code>	<code>if (a != b) { sama = 0; } else { sama = 1; }</code>	<code>if (a != b) { sama = false; } else { sama = true; }</code>
pengulangan	<code>for i := 1 to 10 do begin fak := fak*i; end;</code>	<code>for (i = 1;i<=10;i++) { fak*=i; }</code>	<code>for (i = 1;i<=10;i++) { fak*=i; }</code>
fungsi	<code>function plus(x:integer):boolean; begin plus:=(x>=0); end;</code>	<code>int plus(int x) { return (x>=0); }</code>	<code>bool plus(int x) { return (x>=0); }</code>
prosedur	<code>procedure tambah(var x:integer); begin x:=x+1; end;</code>	<code>void tambah(int *x) { *x++; }</code>	<code>void tambah(int &x) { x++; }</code>

3.2. Referensi

- Kernighan, Brian W. and Ritchie, Dennis M. 1988. *C Programming Language 2nd Edition*. Prentice-Hall.
- Stroustrup, Bjarne. 1997. *C++ Programming Language Third Edition*. Addison-Wesley: .
- Schildt, Herbert. 1998. *C++: The Complete Reference*. McGraw-Hill.