BAB II  
LANDASAN TEORI

**1.1. Struktur Dasar Bahasa C**

Setiap kali ingin membuat program C, kita harus menggunakan line berikut ini:

#include <stdio.h>

Int main(){

Return 0

}

Kalau kita perhatikan, struktur program tersebut dibagi menjadi dua, yaitu bagian Include, dan fungsi main.Fungsi main() adalah fungsi utama dalam program. Fungsi ini akan dieksekusi pertama kali saat program dijalankan. Karena itu, kita harus menuliskan logika program di dalam fungsi ini. Pada contoh di atas kita membuat fungsi main dengan deklarasi seperti ini.Pada program C, #include berfungsi untuk mengimpor fungsi-fungsi yang sudah didefinisikan pada header file. Header file adalah file yang berisi definisi fungsi yang sudah dibuat. Tujuannya agar bisa digunakan pada program C yang lainnya. File ini berekstensi .h.

**1.2. Deklarasi di Bahasa C**

Variabel, merupakan suatu tempat di memori untuk menyimpan data sementara. Data yang ada dalam variabel dapat diubah sesuai kemauan programmer. Deklarasi pada Bahasa C dilakukan dengan mengetik , “tipe data <spasi> nama variabel” contoh, “int a” untuk mendeklarasikan variabel dengan tipe data integer atau bilangan bulat. Constant, hampir sama dengan variable constant merupakan suatu tempat di dalam memori yang dibuat untuk menyimpan data. Namun bedanya dalam constant datanya tidak dapat diubah. Deklarasi pada Bahasa C dilakukan dengan mengetik “const <spasi> nama constant <spasi> nilai konstanta”, contoh “const pp 1,5”. Function, merupakan kelompok statement yang bersama-sama menyelesaikan suatu masalah atau tugas. Dalam c kita memiliki satu function default, yaitu main(), yang selalu akan dijalankan pertama saat program di-compile. Untuk mendeklarasikan function kita menggunakan “return\_type <spasi> function\_name(parameter list);”. Command ini akan memberi tahu compiler tentang nama fungsi dan cara untuk memanggil fungsi tersebut. Body dari function akan dijelaskan secara terpisah.

Macro, merupakan cara untuk mendefinisikan nilai yang konstan tanpa bisa diubah dalam kode. Hampir sama dengan Constant, namun Macro tidak memerlukan tempat di memori, serta tidak memerlukan inisialisasi di setiap fungsi yang ada. Yang menyebabkan macro lebih hemat memori dibanding Constant. Cara untuk mendefinisikan constant adalah dengan

“#define <spasi> nama <spasi> nilai” contoh,

“#define dimapp 3,5”

**1.3. Operator Dalam Bahasa C**

Arithmetic Operator, merupakan operator dasar yang kita kenal sehari-hari, berhubungan dengan operasi aritmatika seperti penjumlahan, pengurangan, perkalian, dan pembagian. Contoh dari operatornya adalah, +, -, \*, dll Relational Operator, merupakan operator yang membandingkan relasi antara dua operand. Misalnya, mengecek apakah dua operand nilainya sama, berbeda, lebih besar dari, kurang dari, dan lainnya. Contoh dari operatornya adalah, ==, !=, >, <

Logical Operator, merupakan operator yang digunakan untuk menghubungkan dua atau lebih ekspresi yang memiliki nilai kebenaran, kemudian menghasilkan kesimpulan dari kumpulan ekspresi tersebut. Misalnya, Logical AND, OR, dan NOT, contoh operatornya, &&, ||, dan !

Bitwise Operator, merupakan operator yang melakukan manipulasi data pada tingkat bit. Merupakan operator yang bisa juga digunakan untuk shifting bits. Bitwise operator tidak bisa diaplikasikan pada floar atau double. Contohnya Bitwise AND(&) , Bitwise OR(|), dll Assignment Operator, merupakan operato yang memanipulasi operand sekaligus meng-assign hasil dari manipulasi ke operand tersebut. Hampir mirip dengan Arithmetic Operator, mungkin bisa dibilang merupakan pemanjangannya. Contoh dari operator ini adalah \*=(Multiply left operant assign to left operand), =, assign values from right side operands to left side operand. Conditional Operator, merupakan operator yang mengecek apakah kondisi ekspresi terpenuhi, kemudian mengeksekusi ekspresi lain dengan syarat apakah ekspresi True atau False. Contoh penulisan, ekspresi 1? Ekspresi 2: ekspresi 3. Saat ekspresi satu terpenuhi maka ekspresi 2 akan dieksekusi, saat tidak terpenuhi maka ekspresi 3 akan dieksekusi.

**1.4. Input Output dalam Bahasa C**

Dalam Bahasa C, kita bisa menggunakan printf(), untuk meng-outputkan nilai ke console. Data di-ouput dengan menggunakan juga beberapa format specifier. Seperti %f, %d, %c, dll.

Untuk menginputkan data kita bisa menggunakan fungsi scanf(), dengan tambahan beberapa type specifier. Namun sebelumnya kita harus sudah mendeklarasikan variable yang ingin kita gunakan sebagai tempat dan menggunakan type specifier yang cocok dengan variabel tersebut.

**1.5 Perulangan While dan For loop**

**1.5.1.** **Perulangan While dan Do-while:**

Perulangan while secara intuitif, akan bekerja dengan cara mengulang *statement* yang ada di dalam bracket while. Hingga kondisi while tersebut tidak terpenuhi. Jika secara intuitif saya terjemahkan, (semasih *statement*), maka lakukan {*statement*}. Saat statement sudah tidak terpenuhi lagi maka looping akan berhenti. While biasa digunakan saat kita tidak tahu berapa kali looping akan terjadi. Contoh penggunaan while adalah,

1. Pada pembuatan menu,

While dapat digunakan untuk looping sebuah menu, hingga user keluar dari program. While biasa digunakan karena kita tidak tahu berapa kali user akan mengulang menu yang dia inginkan. Contoh implementasi,

….

While(i != 1){

Scanf(“%d”, menu);

….

If(i == 3){

i == 1;

}

….

}

Program ini akan mengulang terus menu hingga user menginput 3, untuk keluar dari program tersebut.

1. Perulangan yang banyaknya tidak diketahui,

Fungsi utama while adalah sebagai cara untuk melakukan perulangan saat kita tidak tau berapa kali harus terjadi perulangan tersebut, Contoh implementasinya,

…

while(n > 0) {

System.out.println(n + "...");

waitASecond ();

n -= 1;

}

Perulangan while memiliki alternatif lain yaitu do-while. Jika while akan mengecek condition terlebih dahulu sebelum melaksanakan statement. Do while akan melaksanakan statement terlebih dahulu setelah itu baru mengecek.

**1.5.2. Perulangan For:**

Perulangan For digunakan saat kita sudah tahu pasti berapa kali perulangan tersebut akan terjadi. For loop juga digunakan saat kita memerlukan berapa kali atau informasi tentang ini loop ke-berapa, baik untuk di print atau untuk operasi. Perulangan For bisa di set dengan tiga statement, statement yang menyatakan variabel yang akan di increment, lalu Batasan increment tersebut, dan jumlah increment setiap loop. Berikut contohnya dalam Bahasa c.

…..

For(i=0;i<5;i++){

……

}

Statement for loop diatas akan melakukan looping sebanyak 4 kali, dimulai dari I = 0, hingga I = 4. For loop biasa digunakan untuk looping dengan jumlah yang sudah kita ketahui, atau, jumlah looping tersebut bisa dicari dengan apa-apa saja yang sudah kita tahu sekarang. Contohnya,

1. Looping Benda di dalam Array

Kita bisa mencari jumlah benda yang ada di dalam array dengan menggunakan fungsi len(), karena kita sudah tahu berapa kali kita harus melakukan looping, kita bisa melakukan operasi pada array.

…..

Char Array[15]

For(i=0; i<=len(array); i++){

print(“%s”, array);

}

…..

1. Looping Saat kita memerlukan Informasi loop

Kita bisa menggunakan for loop untuk mendapatkan informasi tentang ini looping ke berapa, dan ingin mencetak atau melakukan operasi sejumlah tertentu. Contoh,

…..

For(i=0; i<= 5; i++){

Print(“%d”, i)

}

**1.6. Percabangan If dan Switch branching**

**1.6.1 If Branching**

If branching digunakan saat kita ingin melakukan percabangan yang tidak terlalu banyak dan kita ingin hanya mengeksekusi satu case saja. Atau saat kita memerlukan percabangan yang mengecek banyak kondisi dari lebih dari satu variabel. Contoh,

….

If(n == a && c == b){

Printf(“Keqing Wangy”)

}

Else if(n == c && c == a){

Printf(“Ganyu Wangy”)

}

Else{

Printf(“Klee Wangy Wangy Wangy!”)

}

…..

**1.6.2. Switch Branching**

Percabangan switch digunakan saat kita memerlukan banyak percabangan yang tidak menggunakan terlalu banyak kondisi, switch case biasanya hanya support percabangan satu variabel. Switch merupakan sintaks untuk menentukan variabel apa yang akan di cek, dan case merupakan condition yang akan dicocokkan. Sementara default merupakan line yang akan dijalankan jika tidak ada case yang terpenuhi. Selanjutnya di beberapa Bahasa pemrograman lain seperti python switch dihilangkan dan diganti dengan tipe data dictionary. Berikut contohnya pada Bahasa c,

…

Switch(n){

Case “keqing”:

Printf(“Keqing wangy”);

Break;

Case “ganyu”:

Prinf(“Ganyu wangy”);

Break;

Case “klee”:

Printf(“Moshi-moshi keisatsu desu?”);

Break;

Default:

Printf(“Rip Gacha, try again next time);

Break;

…..

**1.7. Array**

**1.7.1. Definisi Array**

Array adalah tempat untuk menyimpan kumpulan data. Dalam pengertian saya, array bisa diibaratkan sebagai lemari yang berisi banyak lubang. Namun setiap lemari tersebut hanya bisa menyimpan satu jenis barang saja. Char, int, atau float saja tidak bisa digabungkan dengan tipe data yang lain dalam satu lemari atau satu array.

**1.7.2. Deklarasi Array**

Array dapat dideklarasikan dengan mengetik, tipe\_data nama\_array[ukuran],tipe data meliputi, char, int, float, Boolean, dan double. Deklarasi array ini merupakan contoh untuk deklarasi array dalam 1 dimensi. Berikut merupakan cara deklarasi array dan contohnya jika ingin membuat array lebih dari satu dimensi.

Deklarasi Array 2 Dimensi:

tipe\_data nama\_array[ukuran\_x][ukuran\_y]

Seperti jumlah dimensinya array membutuhkan dua ukuran yaitu ukuran y dan ukuran x. Dan akan menghasilkan array 2 dimensi seperti di bawah.

[0, 0, 0, 0]

[0, 0, 0, 0]

**1.7.3. Deklarasi Array of Char dalam C**

#include <stdio.h>

#Declare Global Variable AOC

Char string[10] = “Dima”;

Void main(){

Printf(“%s”, string);

}

Kode diatas akan mendeklarasikan array of char atau string, dan memberi nilai dima pada array tersebut, kemudian mencetaknya. Bisa kita lihat penggunaan format.

tipe\_data nama\_array[ukuran\_x]

Kita menggunakan char sebagai tipe data karena char atau huruf merupakan penyusun string atau kata. Selanjutnya nama array saya gunakan string karena itu merupakan nama paling intuitif yang bisa kita pakai. Selanjutnya ukuran array saya gunakan 10 karena cukup untuk menampung data yang saya inginkan.

**1.8. File Handling**

File handling dalam Bahasa c dibentuk oleh kesatuan dari beberapa fungsi file handling yang saling bekerja sama. Dalam program ini File Handling merupakan hal yang utama, karena merupakan inti dari yang program ini ingin capai. Fungsi ini memiliki fungsi masing-masing. Beberapa diantaranya:

Membuka dan Menutup file (fopen dan fclose):

Dapat digunakan dengan cara:

Nama\_pointer = fopen(“nama\_file”, “r”);

Ada beberapa mode dalam fopen yang bisa digunakan meliputi:

|  |  |
| --- | --- |
| r | opens a text file in read mode |
| w | opens a text file in write mode |
| a | opens a text file in append mode |
| r+ | opens a text file in read and write mode |
| w+ | opens a text file in read and write mode |
| a+ | opens a text file in read and write mode |
| rb | opens a binary file in read mode |
| wb | opens a binary file in write mode |
| ab | opens a binary file in append mode |
| rb+ | opens a binary file in read and write mode |
| wb+ | opens a binary file in read and write mode |
| ab+ | opens a binary file in read and write mode |

Dan fclose digunakan untuk menutup file dengan cara:

Fclose(pointer\_name);

Menulis data ke file (fprintf):

Data dapat ditulis ke file dengan pertama dibuka dengan fopen lalu selanjutnya melakukan operasi yang kita inginkan, salah satunya fprintf.

Fungsi ini dapat digunakan dengan cara,

Fprintf(fp, “Hello\n”);

Data yang dituliskan dengan fprintf akan diformat dengan formatting dan tidak akan memberikan newline secara otomatis. Alternatifnya ada fputs() dan fputc(), perbedaannya fputs akan menghasilkan newline secara otomatis dan fputc akan menuliskan single character.

Membaca data di file(fscanf):

Fungsi ini dapat digunakan dengan cara,

fscanf(fp, "%s", buff)

Biasanya digunakan dalam loop contoh,

while(fscanf(fp, "%s", buff)!=EOF)

Data yang dibaca dalam fscanf dapat disimpan dalam array of char yang disini saya namai buff. Data ini akan disimpan termasuk dalam new linenya. Kita bisa menambah syarat pada looping, contoh, jika kita tidak ingin newline kita bisa menambahkan != “\n”. Alternatifnya ada fgets() dan fgetc(), perbedaannya fputs akan menghasilkan newline secara otomatis dan fputc akan menuliskan single character. Perbedaannya fungsi fgets akan membaca hingga newline.