**PENUGASAN PRAKTIKUM STRUKTUR DATA**

**PERTEMUAN 2**

1. **sizeof()** adalah sebuah operator untuk mengetahui jumlah memori (byte) yang diperlukan oleh suatu tipe data pada bahasa C. Gunakan **sizeof()** untuk mengetahui ukuran memori pada berbagai tipe data pada bahasa pemrograman C, seperti **char, int, float, double**. Catat hasil yang Anda dapatkan. Bandingkan dengan hasil yang didapatkan oleh teman-teman Anda. Diskusikan apakah yang menyebabkan hasil yang didapatkan berbeda-beda.

**Jawab:**

*Source Code:*

#include <stdio.h>

int **main**(int *argc*, char const \**argv*[])

{

**printf**("%lu\n", sizeof(char));

**printf**("%lu\n", sizeof(int));

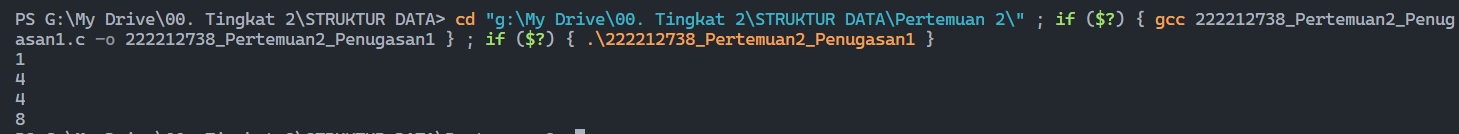
**printf**("%lu\n", sizeof(float));

**printf**("%lu", sizeof(double));

    return 0;

}

*Output:*



Penjelasan:

Di PC yang saya gunakan, sizeof() dari char sebesar 1 byte, integer sebesar 4 byte, float sebesar 4 byte, dan double sebesar 8 byte. Hal ini tentunya bisa saja berbeda antar komputer, bergantung dari sistem yang digunakan (32/64 bit) maupun compiler yang digunakan.

1. Ketik program berikut pada IDE Anda, lalu simpan.

*/\* Aturan Scope pada Bahasa C \*/*

#include <stdio.h>

int **main**(int *argc*, char const \**argv*[])

{

    {

        int x = 10, y = 20; *//(1)*

        {

**printf**("x = %d, y = %d\n", x, y); *//print var x pada (1) dan var y pada (1)*

            {

                int y = 40; *//(2)*

                x++; *//increment var x pada (1)*

                y++; *//increment var y pada (2)*

**printf**("x = %d, y = %d\n", x, y); *//print var x pada (1) dan var y pada (2)*

            }

**printf**("y = %d, y = %d\n", x, y); *//print var x pada (1) dan var y pada (1)*

        }

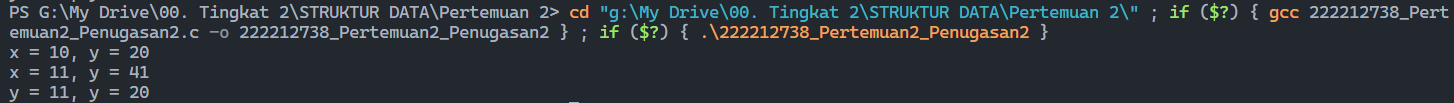
    }

    return 0;

}

**Jawab:**

*Output:*

**

Penjelasan:

Perbedaan tersebut terjadi karena adanya *block scope* yang ditandai dengan { … }. Pada perintah print yang pertama, compiler akan mencetak sesuai dengan value pada variable (1). Namun variable y diinisialisasi lagi pada *block scope* selanjutnya (2), sehingga pada *block scope* tersebut, jika dipanggil variable y maka akan menggunakan variable y pada (2) walaupun sudah pernah diinisiasi sebelumnya. Sehingga, pada *block scope* tersebut meng-*increment* variable x pada (1) dan variable y pada (2). Lalu pada perintah print yang ketiga, maka variable x yang digunakan yaitu pada (1) yang sudah di­-*increment* sebelumnya dan variable y sesuai pada (1).

1. Buatlah program untuk menginput nilai elemen-elemen Matriks A berukuran 3x4 dan mencetak Matriks A tersebut. Contoh ouput sebagai berikut.



**Jawab:**

*Source Code:*

#include <stdio.h>

int **main**(int *argc*, char const \**argv*[])

{

    int matriks[3][4];

    for (int i = 0; i < 3; i++)

    {

        for (int j = 0; j < 4; j++)

        {

**printf**("Masukkan nilai element matriks ke %d, %d: ", i + 1, j + 1);

**scanf**("%d", &matriks[i][j]);

        }

    }

**printf**("\nHasil matriks:\n");

    for (int i = 0; i < 3; i++)

    {

        for (int j = 0; j < 4; j++)

        {

**printf**("%d ", matriks[i][j]);

        }

**printf**("\n");

    }

    return 0;

}

*Output:*

