Program Studi Teknik Elektro ITB

Nama Kuliah (Kode) : Praktikum Pemecahan Masalah dengan C (EL2208)
Tahun / Semester : 2023-2024 / Genap
Modul : 1 – Overview of C Language
Nama Asisten / NIM : _____
Nama Praktikan / NIM : Pradigta Hisyam Ramadhan / 18322008

Tugas Pendahuluan

SOAL KONSEP

1. Library-library yang terdapat dalam bahasa C dan contohnya:

stdlib.h : Merupakan header standar untuk engakses library yang digunakan untuk mendefinisikan beberapa tipe data, makro, dan fungsi-fungsi umum seperti konversi angka, alokasi memori, dan lain-lain.
Contoh:
Tipe data: size_t, wchar_t, div_t, ldiv_t
Makro : NULL, RAND_MAX, EXIT_FAILURE, EXIT_SUCCESS, dan MB_CUR_MAX
Fungsi : rand(), srand()

stdio.h : Merupakan header standar yang mendefinisikan beberapa tipe data, makro, dan fungsi-fungsi untuk melakukan input dan output.
Contoh:
Tipe data: size_t, FILE, fpos
Makro : NULL, EOF, stderr, stdin, stdout, *etc*.
Fungsi : printf(), scanf(), *fopen(), getchar(), *etc.*

math.h : Merupakan header yang mendefinisikan berbagai macam fungsi matematis dan satu makro. Semua fungsi yang disediakan pada library ini menggunakan data bertipe *double*, baik untuk argumen fungsi maupun hasil *return* fungsi.
Contoh:
Makro : HUGE_VAL, digunakan ketika hasil angka perhitungan tidak dapat ditampilkan ke terminal.
Fungsi : cos(), acos(), cosh(), exp(), *etc.*

string.h : Merupakan header file yang mendefinisikan satu tipe data, satu makro, dan beberapa fungsi terkait manipulasi array karakter.
Contoh:
Tipe data: size_t
Makro : NULL
Fungsi : *strcat(), *strcpy(), *memchr(), *etc*.

2. Beberapa jenis operator dalam bahasa C adalah
Arithmetic Operators
1. '+' : Melakukan penjumlahan
2. '-' : Melakukan pengurangan
3. '*' : Melakukan perkalian
4. '\' : Melakukan pembagian
5. '%' : Mengambil nilai sisa hasil bagi
6. '++' : Menambah nilai dengan tipe data *int* dengan 1
7. '--' : Mengurangi nilai dengan tipe data *int* dengan 1

Contoh penggunaan operator aritmetika:

Praktikum > C tempCodeRunnerFile.c > main(void)

```c
#include <stdio.h>

int main(void)
{
    int a = 5, b = 4, sum;

    printf("a + b = %d", a+b);

    return 0;
}
```

TERMINAL PORTS

```
PS D:\(A) College Materials - ITB 22'\(A) Biomedical Engineering\Semester 4\Pemecahan Masalah dengan C\Praktikum> cd "d:
College Materials - ITB 22'\(A) Biomedical Engineering\Semester 4\Pemecahan Masalah dengan C\Praktikum\" ; if ($?) { gcc
CodeRunnerFile.c -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
a + b = 9
```

Relational Operators

1. '==' : Membandingkan kesamaan dua nilai operan.
2. '!=' : Membandingkan ketaksamaan dua nilai operan.
3. '>' : Membandingkan dua nilai yang lebih besar antara dua operan.
4. '>=' : Membandingkan dua nilai yang lebih besar atau sama dengan antara dua operan.
5. '<' : Membandingkan dua nilai yang lebih kecil antara dua operan.
6. '<=' : Membandingkan dua nilai yang lebih kecil atau sama dengan antara dua operan.

Contoh penggunaan operator relasional:

Praktikum > C tempCodeRunnerFile.c > main(void)

```c
#include <stdio.h>
#include <stdbool.h>

int main(void)
{
    int a = 5, b = 4;
    bool condition = (a == b);

    printf("5 == 4 ?\n");

    if (condition == 0)
    {
        printf("False");
    }
    else
    {
        printf("True");
    }
    return 0;
}
```

TERMINAL PORTS

```
PS D:\(A) College Materials - ITB 22'\(A) Biomedical Engineering\Semester 4\Pemecahan Masalah dengan C\Praktikum> cd
"d:\(A) College Materials - ITB 22'\(A) Biomedical Engineering\Semester 4\Pemecahan Masalah dengan C\Praktikum\" ; if
 ($?) { gcc tempCodeRunnerFile.c -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
5 == 4 ?
False
PS D:\(A) College Materials - ITB 22'\(A) Biomedical Engineering\Semester 4\Pemecahan Masalah dengan C\Praktikum>
```

Logical Operators

1. '&&' : operator AND, *return* True jika kedua kondisi bernilai True (atau tidak nol).
2. '||' : operator OR, *return* True jika salah satu dari dua kondisi bernilai True (atau tidak nol).
3. '!': operator NOT, melakukan negasi dari suatu logika.

Contoh penggunaan operator logika:

3. Pengulangan/*loops* dalam pemrograman sering digunakan untuk melakukan tugas yang akan akan dilakukan secara berulang, tanpa harus mengetik kode yang sama di setiap eksekusinya. Contohnya ketika kita ingin menghitung nilai suatu fungsi polinomial, alih-alih mengalikan suatu bilangan berpangkat dengan menuliskan hasil perkaliannya secara berulang, kita dapat menggunakan pengulangan/loops untuk menyederhanakan proses tersebut. Terdapat beberapa jenis pengulangan, beberapa diantaranya adalah:

Counting-loop: Merupakan jenis *loop* yang digunakan apabila jumlah repetisi yang dibutuhkan sudah diketahui dengan pasti. Pada bahasa C, *counting loop* dapat digunakan dengan menggunakan struktur *while loop* ataupun *for loop*.

Sentinel-controlled loop: Merupakan jenis *loop* yang menerima input dari *user* dan akan diterminasi apabila sebuah nilai spesial/unik diberikan oleh *user*. Struktur *while loop* dan *for loop* dapat digunakan pada bahasa C untuk mengimplementasikan jenis *loop* ini.

Input validation loop: Merupakan salah satu jenis *loop* yang interaktif karena program akan meminta nilai input secara repetitif ke *user*. Apabila pada suatu saat nilai yang berada pada daerah masukan yang valid diberikan, maka *loop* akan diterminasi.

Dalam bahasa C sendiri, terdapat tiga syntax yang dapat digunakan untuk melakukan *looping*. Yaitu:
1. **For loop,** digunakan apabila jumlah pasti pengulangan diketahui.
2. **While loop,** digunakan apabila pengulangan akan dilakukan secara terus menerus selama kondisi bernilai benar. Pengecekan kondisi akan dilakukan pada awal perulangan dan perulangan akan berhenti apabila kondisi yang diberikan bernilai salah.
3. **Do-While loop,** mirip dengan while loop, akan tetapi pengecekan kondisi terdapat di bagian akhir setiap perulangan.

4. Terdapat tiga tipe data standar *predefined* yang dimiliki oleh bahasa pemrograman C. Untuk jenis-jenis tipe data lainnya, *user* dapat menggunakan library yang sudah disediakan oleh bahasa C. Ketiga tipe data tersebut adalah:
int: Merupakan tipe data yang digunakan untuk merepresentasikan bilangan integer (*ex:* 0, 1, 2, *etc.*).
double: Merupakan tipe data yang digunakan untuk merepresentasikan bilangan real (*ex:* 3.14159, 0.0005, 150.0, *etc.*).

Kedua tipe data *int* dan *double* dapat digunakan untuk melakukan operasi aritmetik seperti penjumlahan, pengurangan, perkalian, dan pembagian. Selain itu, kedua tipe ini dapat digunakan untuk operasi perbandingan.

char: Merupakan tipe data yang merepresentasikan sebuah karakter, baik berupa huruf, angka, ataupun simbol spesial. Jenis data ini diindikasi dengan adanya tanda petik di akhir dan awal karakter. ASCII *code* merupakan kode khusus untuk merepresentasikan setiap nilai *char* sebagai integer (*ex:* 'A' = 65, 'a' = 97, *etc.*).

SOAL PEMROGRAMAN

Kode Soal secara keseluruhan, dengan fungsi bubbleSort menggunakan pengulangan *for-loop*

```c
#include <stdio.h>
#include <stdbool.h>  // Header untuk mengakses tipe data boolean
#define LOWER_RANGE 0
#define UPPER_RANGE 100

// Fungsi swap untuk menukar dua elemen pada array
void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}

// Fungsi untuk melakukan bubble sort
void bubbleSort(int arr[], int n)
{
    for (int i = 0; i < n - 1; i++)
    {
        for (int j = 0; j < n - i -1; j++)
        {
            // Jika elemen saat ini lebih besar dari elemen berikutnya, tukar
            if (arr[j] > arr[j + 1])
            {
                swap(&arr[j], &arr[j+1]);
            }
        }
    }
}

// Fungsi untuk menampilkan elemen-elemen array
void displayArray(int arr[], int size) {
    for (int i = 0; i < size; i++) {
        printf("%d\n", arr[i]);
    }
    printf("\n");
}

int main(void)
{
    // Deklarasi variabel
```

```
    int n;
    bool validInput = true;  // Flag untuk mengetahui validitas input

    // Menerima input bilangan N
    printf("N = ");
    scanf("%d", &n);

    // deklarasi matriks berukuran N untuk menampung nilai yang diberikan
    int arr[n];

    // Menerima N buah bilangan dari rentang yang sudah ditentukan
    for (int i = 0; i < n; i++)
    {
        // Ambil nilai dari user
        int temp = 0;    // Variabel penampung sementara
        printf("Nilai ke-%d:\t", i+1);
        scanf("%d", &temp);
        if (temp >= LOWER_RANGE && temp <= UPPER_RANGE)
        {
            // Nilai temp dimasukkan ke array
            arr[i] = temp;
        }
        else
        {
            printf("Invalid input!");
            validInput = false;
            break;
        }
    }

    // Cek validitas nilai yang dimasukkan pengguna
    if (validInput)
    {
        // Lakukan bubble sort untuk mengurutkan array
        bubbleSort(arr, n);

        // Tampilkan hasilnya ke layar
        printf("Urutan bilangan dari nilai terkecil sampai terbesar\n");

        displayArray(arr, n);
    }

    return 0;
}
```

Kode dari fungsi bubbleSort apabila menggunakan pengulangan while-loop

```
void bubbleSort(int arr[], int n)
    int i = 1;
    while (i < n ){
        int j = 0;
        while (j < n - 1){
```

```
            // Jika elemen saat ini lebih besar dari elemen berikutnya, tukar
posisi
            if (arr[j] > arr[j + 1]){
                swap(&arr[j], &arr[j + 1]);
            }
            j++;
        }
        i++;
    }
```

Hasil Eksekusi Program

Case #1

```
TERMINAL  PORTS                                                    Code

PS D:\(A) College Materials - ITB 22'\(A) Biomedical Engineering\Semester 4\Pemecahan Masalah dengan C\Praktikum> cd "d:\(A) College Materials
 - ITB 22'\(A) Biomedical Engineering\Semester 4\Pemecahan Masalah dengan C\Praktikum\" ; if ($?) { gcc TPmodul1.c -o TPmodul1 } ; if ($?) { .
\TPmodul1 }
N = 5
Nilai ke-1:     35
Nilai ke-2:     20
Nilai ke-3:     55
Nilai ke-4:     83
Nilai ke-5:     67

Urutan bilangan dari nilai terkecil sampai terbesar
20
35
55
67
83
```

Case #2

```
TERMINAL  PORTS                                                    Code

PS D:\(A) College Materials - ITB 22'\(A) Biomedical Engineering\Semester 4\Pemecahan Masalah dengan C\Praktikum> cd "d:\(A) College Materials
 - ITB 22'\(A) Biomedical Engineering\Semester 4\Pemecahan Masalah dengan C\Praktikum\" ; if ($?) { gcc TPmodul1.c -o TPmodul1 } ; if ($?) { .
\TPmodul1 }
N = 10
Nilai ke-1:     100
Nilai ke-2:     90
Nilai ke-3:     80
Nilai ke-4:     70
Nilai ke-5:     60
Nilai ke-6:     50
Nilai ke-7:     40
Nilai ke-8:     30
Nilai ke-9:     20
Nilai ke-10:    10

Urutan bilangan dari nilai terkecil sampai terbesar
10
20
30
40
50
60
70
80
90
100
```