

### Aturan Praktikum Struktur Data

1. **Akun GitHub:** Setiap praktikan wajib memiliki akun GitHub yang aktif dan digunakan selama praktikum berlangsung.
2. **Invite Collaborator:** Setiap praktikan diwajibkan untuk menambahkan collaborator di setiap repository
  - a. Asisten Praktikum: AndiniNH
  - b. Asisten Praktikum: 4ldiputra
3. **Repository Praktikum:** Setiap praktikan diwajibkan untuk membuat satu repository di GitHub yang akan digunakan untuk seluruh tugas dan laporan praktikum. Repository ini harus diatur dengan rapi dan sesuai dengan instruksi yang akan diberikan di lampiran.
4. **Penamaan Folder:** Penamaan folder dalam repository akan dibahas secara rinci di lampiran. Praktikan wajib mengikuti aturan penamaan yang telah ditentukan.

Nomor	Pertemuan	Penamaan
1	Pengalanan Bahasa C++ Bagian Pertama	01_Pengenalan_CPP_Bagian_1
2	Pengenalan Bahasa C++ Bagian Kedua	02_Pengenalan_CPP_Bagian_2
3	Abstract Data Type	03_Abstract_Data_Type
4	Single Linked List Bagian Pertama	04_Single_Linked_List_Bagian_1
5	Single Linked List Bagian Kedua	05_Single_Linked_List_Bagian_2
6	Double Linked List Bagian Pertama	06_Double_Linked_List_Bagian_1
7	Stack	07_Stack
8	Queue	08_Queue
9	Assessment Bagian Pertama	09_Assessment_Bagian_1
10	Tree Bagian Pertama	10_Tree_Bagian_1
11	Tree Bagian Kedua	11_Tree_Bagian_2
12	Asistensi Tugas Besar	12_Asistensi_Tugas_Besar
13	Multi Linked List	13_Multi_Linked_List
14	Graph	14_Graph
15	Assessment Bagian Kedua	15_Assessment_Bagian_2
16	Tugas Besar	16_Tugas_Besar

#### 5. Jam Praktikum:

- Jam masuk praktikum adalah **1 jam lebih lambat** dari jadwal yang tercantum. Sebagai contoh, jika jadwal praktikum adalah pukul 06.30 - 09.30, maka aturan praktikum akan diatur sebagai berikut:
  - **06.30 - 07.30:** Waktu ini digunakan untuk **Tugas Praktikum dan Laporan Praktikum** yang dilakukan di luar laboratorium.
  - **07.30 - 08.30:** Sesi ini mencakup **tutorial, diskusi, dan kasus problem-solving**. Kegiatan ini berlangsung di dalam laboratorium dengan alokasi waktu sebagai berikut:
    - **60 menit pertama:** Tugas terbimbing.
    - **60 menit kedua:** Tugas mandiri.

6. **Pengumpulan Tugasn Pendahuluan:** Tugas Pendahuluan (TP) wajib dikumpulkan melalui GitHub sesuai dengan format berikut:

**nama\_repo/nama\_pertemuan/TP\_Pertemuan\_Ke.md**

Sebagai contoh:

**STD\_Yudha\_Islalmi\_Sulistya\_XXXXXXXX/01\_Running\_Modul/TP\_01.md**

7. **Pengecekan Tugas Pendahuluan:** Pengumpulan laporan praktikum akan diperiksa **1 hari sebelum praktikum selanjutnya** dimulai. Pastikan tugas telah diunggah tepat waktu untuk menghindari sanksi.

8. **Struktur Laporan Praktikum**

1. **Cover :**

**LAPORAN PRAKTIKUM**  
**Modul 2**  
**PENGENALAN BAHASA C++ (BAGIAN KEDUA)**



**Disusun Oleh:**  
**KAFKA PUTRA RIYADI -2311104041**  
**Kelas**  
**SE-07-02**

**Dosen :**  
**Wahyu Andi Saputra,S.pd,M.Eng**

**PROGRAM STUDI S1 SOFTWARE ENGINEERING**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY**  
**PURWOKERTO**  
**2024**

2. **Tujuan**

1. Memahami penggunaan *pointer* dan alamat memori
2. Mengimplementasikan fungsi dan prosedur dalam program

### 3. Landasan Teori

#### Array

Array adalah kumpulan-kumpulan variabel yang menyimpan data dengan tipe yang sama atau data-data yang tersusun secara linear dimana di dalamnya terdapat elemen dengan tipe yang sama. Indeks dalam array menyatakan elemen yang disimpan dan panjang atau length menyatakan total elemen yang tersimpan. Dalam array, untuk membedakan satu variabel dengan variabel lain berdasarkan subscript, bilangan dalam kurung siku [...] disebut subscript, dengan subscript masing-masing elemen dapat diakses. Untuk mengakses array di dalam bahasa C++ (dan bahasa pemrograman lain secara umum), Anda menggunakan indeks yang menunjukkan posisi elemen di dalam array tersebut. Indeks array dimulai dari **0**, artinya elemen pertama berada pada indeks **0**, elemen kedua pada indeks **1**, dan seterusnya.

#### Pointer

dalam bahasa pemrograman (termasuk C++) adalah sebuah variabel yang menyimpan alamat memori dari variabel lain. Dengan pointer, kita bisa mengakses dan memanipulasi data yang disimpan di alamat memori tersebut.

### 4. Guided

#### 2.1.1 Array Satu Dimensi

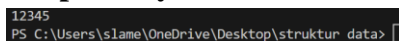
Array satu dimensi adalah kumpulan elemen data yang tersimpan dalam satu baris atau kolom, dan semua elemen di dalamnya memiliki tipe data yang sama. Setiap elemen dalam array diakses melalui indeks yang merepresentasikan posisi elemen tersebut. Indeks dimulai dari 0 dan bertambah satu untuk setiap elemen berikutnya.

#### CONTOH:



```
1 int main(){
2     int nilai[5]={1,2,3,4,5};
3     cout << nilai[0];
4     cout << nilai[1];
5     cout << nilai[2];
6     cout << nilai[3];
7     cout << nilai[4];
8
9 }
```

#### Outputannya:



```
12345
PS C:\Users\slame\OneDrive\Desktop\struktur_data>
```

#### 2.1.2 Array Dua Dimensi

Array 2 dimensi adalah struktur data yang terdiri dari baris dan kolom, di mana setiap elemen dapat diakses menggunakan dua indeks: satu untuk baris dan satu untuk kolom. Secara konsep, array 2 dimensi bisa dianggap sebagai tabel dengan nilai-nilai yang diatur dalam bentuk grid.

**CONTOH:**

```
1 int main(){
2     // Array 2 dimensi
3     int nilai[3][4]={
4         {1,2,3,4},
5         {5,6,7,8},
6         {9,10,11,12}
7     };
8
9     for(int i=0; i<3; i++){
10        for (int j=0; j<4; j++){
11            cout << nilai[i][j] << " ";
12        }
13        cout << endl;
14    }
15 }
16
```

**Outputannya:**

```
1 2 3 4
5 6 7 8
9 10 11 12
PS C:\Users\sleme\OneDrive\Desktop\struktur data>
```

**2.2 Pointer****2.2.1 Data dan Memori**

Semua data yang ada digunakan oleh program komputer disimpan di dalam memori (RAM) komputer. Memori dapat digambarkan sebagai sebuah *array* 1 dimensi yang berukuran sangat besar. Seperti layaknya *array*, setiap *cell memory* memiliki “indeks” atau “alamat” unik yang berguna untuk identitas yang biasa kita sebut sebagai “*address*”

**2.2.2 Pointer dan Alamat**

Variabel *pointer* merupakan dasar tipe variabel yang berisi *integer* dalam format heksadesimal. *Pointer* digunakan untuk menyimpan alamat memori variabel lain sehingga *pointer* dapat mengakses nilai dari variabel yang alamatnya ditunjuk

**CONTOH:**

```
1 int main(){
2     // #POINTER
3
4     int x,y;
5     int *px;
6     x = 87;
7     px = &x;
8     y= *px;
9
10    cout << "Alamat x = " << &x << endl;
11    cout << "isi px = " << px << endl;
12    cout << "isi x = " << x << endl;
13    cout << "Nilai y= " << y << endl;
14    getch();
15    return 0;
16 }
```

### Outputannya:

```
Alamat x = 0x5ffe8c
isi px = 0x5ffe8c
isi x = 87
Nilai y= 87
PS C:\Users\slame\OneDrive\Desktop\struktur_data>
```

## 2.4 Prosedur

Dalam C sebenarnya tidak ada prosedur, semua berupa fungsi, termasuk main() pun adalah sebuah fungsi. Jadi prosedur dalam C merupakan fungsi yang tidak mengembalikan nilai, biasa diawali dengan kata kunci void di depan nama prosedur.

### 2.5.2 Cara melewati Parameter

#### A. Pemanggilan dengan Nilai (*call by value*)

Pada pemanggilan dengan nilai, nilai dari parameter aktual akan disalin kedalam parameter formal, jadi parameter aktual tidak akan berubah meskipun parameter formalnya berubah.

#### B. Pemanggilan dengan Pointer (*call by pointer*)

Pemanggilan dengan *pointer* merupakan cara untuk melewati alamat suatu variabel ke dalam suatu fungsi. Dengan cara ini dapat merubah nilai dari variabel aktual yang dilewatkan ke dalam fungsi. Jadi cara ini dapat merubah variabel yang ada diluar fungsi.

### CONTOH:

```
1  int penjumlahan(int a, int b){
2      return a + b;
3  }
4  void greet(string name){
5      cout<< "Hello, " << name << "!" << endl;
6  }
7
8
9  int main(){
10
11     int hasil = penjumlahan(5, 3);
12     cout << "hasilnya adalah" << hasil << endl;
13
14     greet("alice");
15
16 }
```

### Outputannya:

```
hasilnya adalah8
Hello, alice!
PS C:\Users\slame\OneDrive\Desktop\struktur_data>
```

#### C. Pemanggilan dengan Referensi (*Call by Reference*)

Pemanggilan dengan referensi merupakan cara untuk melewati alamat suatu variabel kedalam suatu fungsi. Dengan cara ini dapat merubah nilai dari variabel aktual yang

dilewatkan ke dalam fungsi.

Cara pemanggilannya:

- Gunakan tanda & dalam deklarasi parameter fungsi.
- Saat memanggil fungsi, Anda melewati variabel seperti biasa (tanpa tanda & di sisi pemanggilan fungsi), tetapi variabel itu akan dilewatkan sebagai referensi.

## 5. Unguided

1. Buatlah program untuk menampilkan Output seperti berikut dengan data yang diinputkan

**JAWAB:**

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int n;
6
7      // #Meminta jumlah elemen yang akan diinput
8      cout << "Masukkan jumlah elemen: ";
9      cin >> n;
10
11     int array[n];
12
13     // Memasukkan elemen array dari user
14     cout << "Masukkan " << n << " elemen:" << endl;
15     for(int i = 0; i < n; i++) {
16         cin >> array[i];
17     }
18
19     // #Menampilkan data array
20     cout << "Data Array: ";
21     for(int i = 0; i < n; i++) {
22         cout << array[i] << " ";
23     }
24     cout << endl;
25
26     // #Menampilkan angka genap
27     cout << "Nomor Genap: ";
28     for(int i = 0; i < n; i++) {
29         if(array[i] % 2 == 0) {
30             cout << array[i] << ", ";
31         }
32     }
33     cout << endl;
34
35     // #Menampilkan angka ganjil
36     cout << "Nomor Ganjil: ";
37     for(int i = 0; i < n; i++) {
38         if(array[i] % 2 != 0) {
39             cout << array[i] << ", ";
40         }
41     }
42     cout << endl;
43
44     return 0;
45 }
46
```

### Outputannya:

```
Masukkan jumlah elemen: 10
Masukkan 10 elemen:
1
2
3
4
5
6
7
8
9
10
Data Array: 1 2 3 4 5 6 7 8 9 10
Nomor Genap: 2, 4, 6, 8, 10,
Nomor Ganjil: 1, 3, 5, 7, 9,
PS C:\Users\sIame\OneDrive\Desktop\struktur_data> |
```

- Buatlah program Input array tiga dimensi tetapi jumlah atau ukuran elemennya diinputkan oleh user!

### JAWAB:

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int x, y, z;
6
7     // Minta pengguna memasukkan ukuran dimensi
8     cout << "Masukkan ukuran dimensi pertama (x): ";
9     cin >> x;
10    cout << "Masukkan ukuran dimensi kedua (y): ";
11    cin >> y;
12    cout << "Masukkan ukuran dimensi ketiga (z): ";
13    cin >> z;
14
15    // Deklarasi array 3D dengan ukuran dinamis
16    int array[x][y][z];
17
18    // Minta input elemen untuk array 3D
19    cout << "Masukkan elemen array 3D:" << endl;
20    for (int i = 0; i < x; i++) {
21        for (int j = 0; j < y; j++) {
22            for (int k = 0; k < z; k++) {
23                cout << "Elemen [" << i << "][" << j << "][" << k << "]: ";
24                cin >> array[i][j][k];
25            }
26        }
27    }
28
29    // Menampilkan elemen array 3D
30    cout << "Inflamen Array 3D adalah:" << endl;
31    for (int i = 0; i < x; i++) {
32        for (int j = 0; j < y; j++) {
33            for (int k = 0; k < z; k++) {
34                cout << "Elemen [" << i << "][" << j << "][" << k << "] = " << array[i][j][k] << endl;
35            }
36        }
37    }
38
39    return 0;
40 }
41
```

### Outputannya:

```
Masukkan elemen array 3D:
Elemen [0][0][0]: 1 2 3
Elemen [0][0][1]: Elemen [0][0][2]: Elemen [0][1][0]: 4 5 6
Elemen [0][1][1]: Elemen [0][1][2]:
Elemen Array 3D adalah:
Elemen [0][0][0] = 1
Elemen [0][0][1] = 2
Elemen [0][0][2] = 3
Elemen [0][1][0] = 4
Elemen [0][1][1] = 5
Elemen [0][1][2] = 6
PS C:\Users\sIame\OneDrive\Desktop\struktur_data> |
```



3. Buatlah program menu untuk mencari nilai Maksimum, Minimum dan Nilai rata – rata dari suatu array dengan input yang dimasukan oleh user!

Jawab:

```
1 #include <iostream>
2 #include <vector>
3 #include <limits>
4
5 using namespace std;
6
7 vector<int> getArray() {
8     int n, input;
9     vector<int> arr;
10
11     cout << "Berapa banyak elemen dalam array? ";
12     cin >> n;
13
14     cout << "Masukkan " << n << " angka: ";
15     for (int i = 0; i < n; ++i) {
16         cin >> input;
17         arr.push_back(input);
18     }
19     return arr;
20 }
21
22 int findMax(const vector<int>& arr) {
23     int max_val = numeric_limits<int>::min();
24     for (int num : arr) {
25         if (num > max_val) {
26             max_val = num;
27         }
28     }
29     return max_val;
30 }
31
32 int findMin(const vector<int>& arr) {
33     int min_val = numeric_limits<int>::max();
34     for (int num : arr) {
35         if (num < min_val) {
36             min_val = num;
37         }
38     }
39     return min_val;
40 }
41
42 double findAvg(const vector<int>& arr) {
43     double sum = 0;
44     for (int num : arr) {
45         sum += num;
46     }
47     return sum / arr.size();
48 }
49
50 int main() {
51     vector<int> array = getArray();
52
53     cout << "Nilai maksimum: " << findMax(array) << endl;
54     cout << "Nilai minimum: " << findMin(array) << endl;
55     cout << "Nilai rata-rata: " << findAvg(array) << endl;
56
57     return 0;
58 }
```

Outputannya:

```
Berapa banyak elemen dalam array? 5
Masukkan 5 angka: 2 6 7 8 9
Nilai maksimum: 9
Nilai minimum: 2
Nilai rata-rata: 6.4
PS C:\Users\slame> |
```

## 6. Kesimpulan

Setelah saya mempelajari dan mengerjakan laporan praktikum ini saya mendapatkan sedikit ilmu tentang jenis jenis array mulai dari 1 dimensi, 2 dimensi dan array berdimensi banyak serta cara melewati parameter ada 3 cara yaitu:

1. **Pemanggilan dengan Nilai (*call by value*)**
2. **Pemanggilan dengan Pointer (*call by pointer*)**
3. **Pemanggilan dengan Referensi (*Call by Reference*)**