LAPORAN PRAKTIKUM MODUL 2

Penganalan Bahasa C++ Bagian Kedua

DISUSUN OLEH :

ZIVANA AFRA YULIANTO 2211104039

S1SE-06-02

DOSEN :

WAHYU ANDI SAPUTRA

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO 2024

Tujuan

1. **Memahami Konsep Fungsi dan Prosedur**

   Tujuan utama adalah agar peserta mampu memahami dan mengimplementasikan fungsi serta prosedur dalam C++. Peserta akan belajar bagaimana mendefinisikan, memanggil, dan memanfaatkan fungsi/prosedur untuk modularisasi program, sehingga kode menjadi lebih terstruktur dan mudah dipahami.

2. **Menguasai Penggunaan Array**

   Peserta diharapkan mampu memahami konsep array sebagai struktur data yang menyimpan banyak nilai dengan tipe data yang sama. Modul ini bertujuan agar peserta dapat mendeklarasikan, menginisialisasi, dan memanipulasi array dalam program untuk menyelesaikan masalah yang melibatkan koleksi data.

3. **Memahami Konsep Pointer dan Implementasinya**

   Modul ini bertujuan agar peserta memahami konsep pointer, termasuk cara mendeklarasikan pointer, mengambil alamat memori suatu variabel, dan melakukan manipulasi data melalui pointer. Peserta akan belajar bagaimana pointer digunakan untuk mengakses dan mengubah nilai variabel serta elemen array secara efisien.

4. **Mengaplikasikan Pointer dengan Fungsi dan Array**

   Setelah memahami dasar-dasar pointer, peserta akan diajarkan bagaimana pointer dapat diintegrasikan dengan fungsi dan array untuk membuat program yang lebih dinamis dan fleksibel, seperti penggunaan pointer sebagai parameter fungsi atau pengelolaan array melalui pointer.

Landasan Teori

- **Fungsi dan Prosedur** Fungsi dalam C++ adalah blok kode yang dirancang untuk menjalankan tugas tertentu. Fungsi bertujuan untuk modularisasi program dengan membagi tugas besar menjadi sub-tugas yang lebih kecil, sehingga kode lebih mudah dipelihara dan dibaca. Fungsi memiliki parameter yang bisa menerima nilai dari pemanggil dan mengembalikan nilai setelah proses selesai. Prosedur, di sisi lain, adalah fungsi yang tidak mengembalikan nilai (void function). Dengan fungsi dan prosedur, kita dapat mengurangi pengulangan kode dan meningkatkan efisiensi pengembangan.

- **Array** Array adalah struktur data yang memungkinkan penyimpanan beberapa elemen bertipe sama dalam satu variabel. Indeks array dimulai dari nol, yang memungkinkan akses acak ke setiap elemen. Array sangat efisien untuk pengelolaan data yang berbentuk koleksi, seperti daftar angka, string, atau objek. Dalam implementasinya, array berguna untuk menyimpan dan mengelola banyak data dengan cara yang mudah dan terstruktur.

- **Pointer** Pointer adalah variabel yang menyimpan alamat memori dari variabel lain. Dengan menggunakan pointer, kita bisa mengakses dan memanipulasi data secara langsung melalui alamat memori tersebut, sehingga memberikan kontrol yang lebih dalam terhadap manajemen memori. Pointer juga memungkinkan fungsi untuk mengubah nilai variabel yang ada di luar cakupan (scope) fungsi tersebut, karena mengoperasikan variabel secara "by reference" bukan "by value."

- **Pointer dan Array** Pointer dan array memiliki hubungan yang sangat erat dalam C++. Sebuah array adalah sekumpulan data yang disimpan secara berurutan di memori, dan nama array sebenarnya adalah pointer ke elemen pertama dari array tersebut. Dengan pointer, kita dapat menavigasi dan mengakses elemen array dengan cara yang efisien. Pointer juga dapat digunakan untuk operasi dinamis pada array, seperti pengalokasian memori secara dinamis untuk array dengan ukuran yang tidak diketahui pada saat kompilasi.

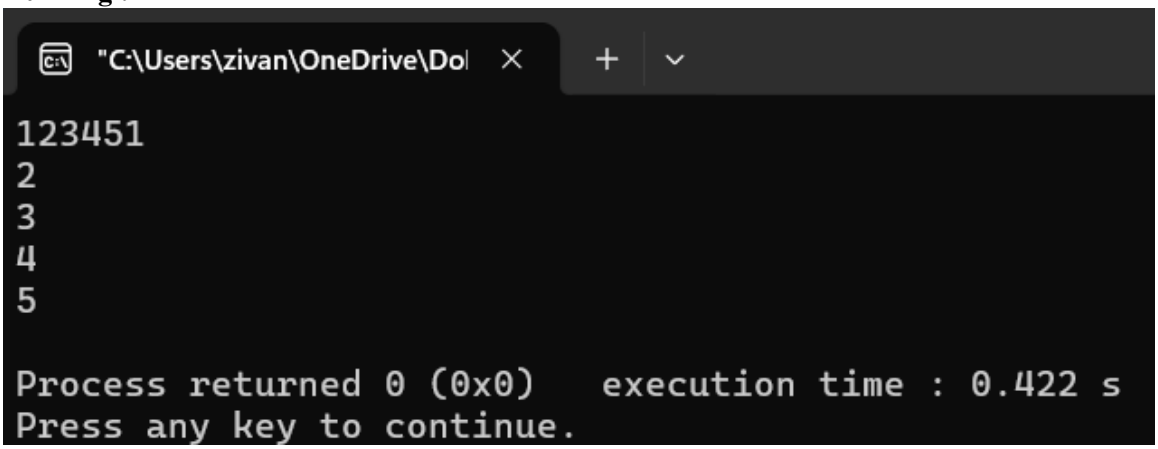
Guided

1. Array

Code :

```
int main(){
    int nilai[5] = {1, 2, 3, 4, 5};
    cout << nilai[0];
    cout << nilai[1];
    cout << nilai[2];
    cout << nilai[3];
    cout << nilai[4];
    int nilai[5] = {1, 2, 3, 4, 5};
    for (int i = 0; i < 5; i++){
        cout << nilai[i] << endl;
    }
}
```

Running :



Deskripsi :

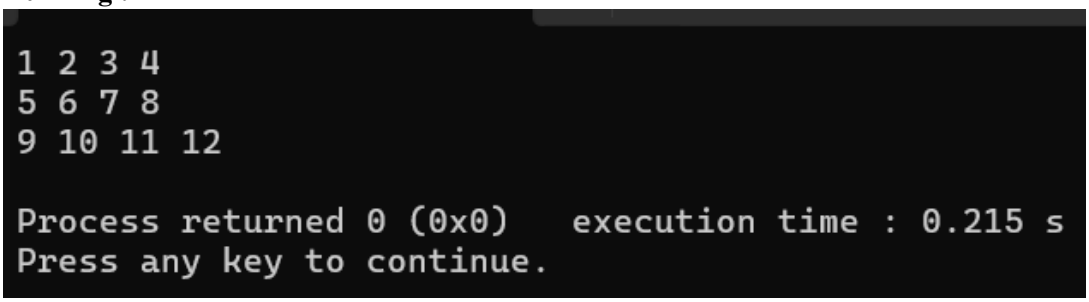
- Deklarasi dan Inisialisasi Array:
    - Kode mendeklarasikan array bernama nilai dengan lima elemen bertipe integer, yaitu {1, 2, 3, 4, 5}.
- Menampilkan Elemen Array Secara Manual:
    - Baris cout << nilai[0]; hingga cout << nilai[4]; menampilkan masing-masing elemen array secara manual satu per satu.
- Menampilkan Elemen Array Menggunakan Loop:
    - Kode mendeklarasikan array nilai lagi (harus dihapus karena pengulangan deklarasi array tidak valid).
    - Setelah itu, for loop digunakan untuk menampilkan semua elemen array dengan cara iteratif, dari indeks 0 hingga 4.
- Kesalahan Deklarasi Ganda:
    - Ada kesalahan dalam kode di mana array nilai dideklarasikan dua kali. Baris kedua deklarasi (int nilai[5] = {1, 2, 3, 4, 5};) harus dihapus untuk membuat kode berjalan dengan benar.

2. Array 2 Dimensi

Code :

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{    int nilai[3][4] = {
        {1, 2, 3, 4},
        {5, 6, 7, 8},
        {9, 10, 11, 12},
    };
    for (int i = 0; i < 3; i++)
    {
        for (int j = 0; j < 4; j++)
        {
            cout << nilai[i][j] << " ";
        }
        cout << endl;
    }
}
```

Running :



Deskripsi :

1. Deklarasi dan Inisialisasi Array 2 Dimensi: Kode ini mendeklarasikan sebuah array 2 dimensi bernama nilai dengan ukuran 3 baris dan 4 kolom. Array ini diinisialisasi dengan nilai-nilai berikut:
   `Baris 1: {1, 2, 3, 4}`
   `Baris 2: {5, 6, 7, 8}`
   `Baris 3: {9, 10, 11, 12}`

2. Loop untuk Menampilkan Elemen Array: Kode menggunakan dua loop bersarang (for loop) untuk mengiterasi melalui setiap elemen dalam array. Loop `luar (for (int i = 0; i < 3; i++))` mengiterasi melalui setiap baris, sedangkan loop dalam `(for (int j = 0; j < 4; j++))` mengiterasi melalui setiap kolom dalam baris tersebut.
3. Menampilkan Elemen Array: Di dalam loop dalam, `cout << nilai[i][j] << " ";` digunakan untuk mencetak setiap elemen dari array diikuti oleh spasi. Setelah semua elemen dalam satu baris ditampilkan, `cout << endl;` digunakan untuk memindahkan output ke baris baru, sehingga setiap baris dari array ditampilkan pada baris terpisah di konsol.
4. Output Akhir: Hasil akhir dari eksekusi program ini adalah menampilkan nilai-nilai dalam array dalam format tabel, sebagai berikut:
   `1 2 3 4`
   `5 6 7 8`
   `9 10 11 12`

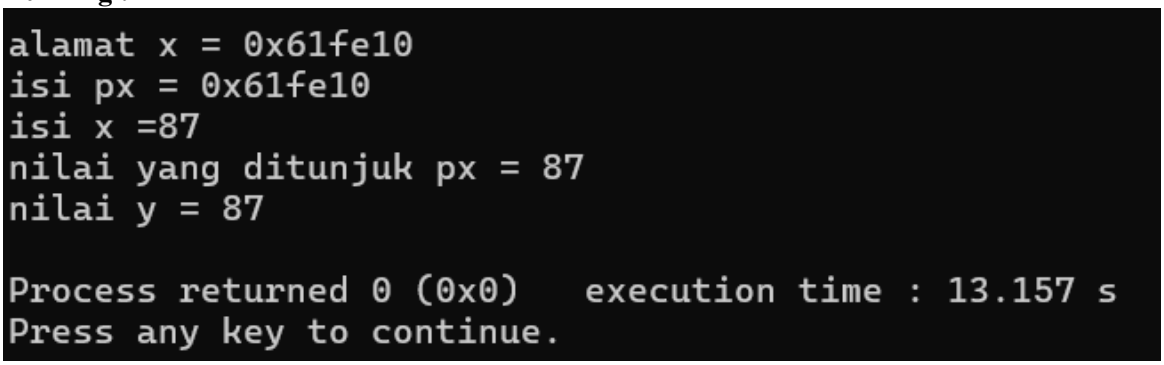
3. Pointer

Code :

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
    int x, y;
    int *px;
    x = 87;
    px = &x;
    y = *px;

    cout << "alamat x = " << &x << endl;
    cout << "isi px = " << px << endl;
    cout << "isi x =" << x << endl;
    cout << "nilai yang ditunjuk px = " << *px << endl;
    cout << "nilai y = " << y << endl;
    getch();
}
```

Running :



Deskripsi :

- Deklarasi Variabel dan Pointer:
  - Program ini mendeklarasikan dua variabel bertipe integer, yaitu `x` dan `y`, serta sebuah pointer bertipe integer `px` (`int *px`).
- Inisialisasi Variabel dan Pointer:
  - Nilai variabel `x` diatur menjadi `87` (`x = 87`).
  - Pointer `px` diinisialisasi untuk menyimpan alamat dari variabel `x` (`px = &x`), sehingga `px` menunjuk ke lokasi memori di mana `x` disimpan.
- Penugasan Nilai melalui Pointer:
  - Nilai yang ditunjuk oleh pointer `px` (yaitu nilai `x`) ditugaskan ke variabel `y` (`y = *px;`). Dengan ini, `y` juga akan memiliki nilai `87`.
- Output Program:
  - `cout << "alamat x = " << &x << endl;` menampilkan alamat memori dari variabel `x`.
  - `cout << "isi px = " << px << endl;` menampilkan alamat yang disimpan dalam pointer `px`, yang sama dengan alamat `x`.
  - `cout << "isi x = " << x << endl;` menampilkan nilai dari `x`, yaitu `87`.
  - `cout << "nilai yang ditunjuk px = " << *px << endl;` menampilkan nilai yang ditunjuk oleh pointer `px`, yaitu `87` (nilai `x`).

- `cout << "nilai y = " << y << endl;` menampilkan nilai dari `y`, yang juga `87` (hasil dari penugasan sebelumnya).

- Fungsi `getch()`:
  - `getch();` digunakan untuk menunggu input dari pengguna sebelum menutup konsol, sehingga pengguna dapat melihat output sebelum jendela program ditutup.

4. Fungsi

Code :

```
#include <iostream>
#include <conio.h>

using namespace std;

// fungi
int penjumlahan(int a, int b)
{
    return a + b;
}

int main()
{
    // pemanggilan fungsi
    int hasil = penjumlahan(5, 3);
    cout << hasil << endl;
}
```

Running :

```
8

Process returned 0 (0x0)   execution time : 0.187 s
Press any key to continue.
```

Deskripsi :

- Deklarasi dan Definisi Fungsi:
  - Kode ini mendeklarasikan sebuah fungsi bernama `penjumlahan` yang menerima dua parameter bertipe integer (`int a` dan `int b`). Fungsi ini mengembalikan hasil penjumlahan dari kedua parameter tersebut (`return a + b;`).
- Fungsi **`main`**:
  - Di dalam fungsi `main`, pemanggilan fungsi `penjumlahan` dilakukan dengan memberikan argumen `5` dan `3`. Hasil dari pemanggilan fungsi ini disimpan dalam variabel `hasil`.
- Menampilkan Hasil:
  - `cout << hasil << endl;` digunakan untuk menampilkan nilai yang disimpan dalam `hasil`, yang merupakan hasil penjumlahan dari `5` dan `3`. Jadi, output program ini akan menampilkan `8`.
- Fungsi **`getch()`**:

- Meskipun `getch();` dideklarasikan dengan `#include <conio.h>`, fungsi ini tidak digunakan dalam kode ini. Jika ingin menambahkan penundaan sebelum program berakhir, `getch()` dapat digunakan di akhir fungsi `main`.

5. Procedur

Code :

```
#include <iostream>
#include <conio.h>

using namespace std;

// prosedure
void greet(string name)
{
    cout << "hallo" << name << "!" << endl;
}

int main()
{
    // pemanggilan prosedure
    greet("alice");
}
```

Running :

```
hallo alice!

Process returned 0 (0x0)   execution time : 0.151 s
Press any key to continue.
```

Deskripsi :

- Deklarasi dan Definisi Prosedur:
  - Kode ini mendeklarasikan sebuah prosedur bernama `greet` yang menerima satu parameter bertipe string (`string name`). Prosedur ini mencetak pesan sapaan yang menyapa nama yang diberikan dengan format: "Hallo [nama]!" menggunakan `cout`.
- Fungsi **`main`**:
  - Di dalam fungsi `main`, prosedur `greet` dipanggil dengan argumen `"alice"`, sehingga prosedur akan mengeksekusi dan menampilkan pesan sapaan untuk nama tersebut.
- Output Program:
  - Ketika prosedur `greet` dipanggil, output yang dihasilkan adalah:
    `hallo alice!`
- Fungsi **`getch()`**:
  - Meskipun `getch();` dideklarasikan dengan `#include <conio.h>`, fungsi ini tidak digunakan dalam kode ini. Jika ingin menambahkan penundaan sebelum program berakhir, `getch()` dapat digunakan di akhir fungsi `main`.

Unguided

1. **Nomor 1**

Code :

```
#include <iostream>
#include <vector>
using namespace std;

int main()
{
    int n;
    cout << "Masukkan jumlah elemen array: ";
    cin >> n;

    vector<int> arr(n); // Gunakan vector untuk array dinamis
    cout << "Masukkan elemen array:\n";
    for (int i = 0; i < n; i++)
        cin >> arr[i];

    cout << "\nData Array : ";
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";

    cout << "\nNomor Genap  : ";
    for (int i = 0; i < n; i++)
        if (arr[i] % 2 == 0)
            cout << arr[i] << ", ";

    cout << "\nNomor Ganjil : ";
    for (int i = 0; i < n; i++)
        if (arr[i] % 2 != 0)
            cout << arr[i] << ", ";

    cout << endl;
    return 0;
}
```

Running :

```
Masukkan jumlah elemen array: 6
Masukkan elemen array:
1 2 3 4 5 6

Data Array : 1 2 3 4 5 6
Nomor Genap  : 2, 4, 6,
Nomor Ganjil : 1, 3, 5,

Process returned 0 (0x0)   execution time : 12.100 s
Press any key to continue.
```

Deskripsi :

Deklarasi Variabel dan Input Pengguna:

- Program dimulai dengan mendeklarasikan variabel n, yang akan menyimpan jumlah elemen dari array. Pengguna diminta untuk memasukkan nilai n melalui konsol.

Inisialisasi Vector:

- Sebuah vector<int> arr(n); diciptakan untuk menyimpan elemen array secara dinamis berdasarkan jumlah yang dimasukkan oleh pengguna. Ini menggantikan penggunaan array statis.

Input Elemen Array:

- Program meminta pengguna untuk memasukkan elemen-elemen array satu per satu melalui loop for, yang kemudian disimpan dalam arr.

Menampilkan Data Array:

- Setelah semua elemen dimasukkan, program mencetak semua data dalam array menggunakan loop for.

Pemisahan Angka Genap dan Ganjil:

- Program menggunakan dua loop for terpisah untuk mencetak angka genap dan angka ganjil dari array. Angka genap ditampilkan jika arr[i] % 2 == 0, dan angka ganjil jika arr[i] % 2 != 0.

Output Program:

- Program menampilkan tiga bagian output:
  - Data array yang dimasukkan oleh pengguna.
  - Daftar nomor genap yang terdapat dalam array.
  - Daftar nomor ganjil yang terdapat dalam array.

Akhir Program:

- Program diakhiri dengan return 0;, menandakan bahwa eksekusi program selesai tanpa kesalahan.

2. **Nomor 2**
   Code :

```
#include <iostream>
using namespace std;

int main()
{
    int x, y, z;

    // Input ukuran array
    cout << "Masukkan ukuran array 3D (x y z): ";
    cin >> x >> y >> z;

    // Alokasi memori untuk array 3D
    int ***arr = new int **[x];
    for (int i = 0; i < x; i++)
    {
        arr[i] = new int *[y];
        for (int j = 0; j < y; j++)
        {
            arr[i][j] = new int[z];
        }
    }

    // Input elemen array
    cout << "Masukkan elemen array:" << endl;
    for (int i = 0; i < x; i++)
    {
        for (int j = 0; j < y; j++)
        {
            for (int k = 0; k < z; k++)
            {
                cout << "arr[" << i << "][" << j << "][" << k << "] = ";
                cin >> arr[i][j][k];
            }
        }
    }

    // Menampilkan array
    cout << "\nArray yang diinput:" << endl;
    for (int i = 0; i < x; i++)
    {
        for (int j = 0; j < y; j++)
        {
            for (int k = 0; k < z; k++)
            {
                cout << arr[i][j][k] << " ";
            }
            cout << endl;
        }
        cout << endl;
    }

    // Dealokasi memori
    for (int i = 0; i < x; i++)
    {
        for (int j = 0; j < y; j++)
        {
            delete[] arr[i][j];
        }
        delete[] arr[i];
    }
    delete[] arr;

    return 0;
}
```

Running :

```
Masukkan ukuran array 3D (x y z): 1 2 3
Masukkan elemen array:
arr[0][0][0] = 1
arr[0][0][1] = 2
arr[0][0][2] = 3
arr[0][1][0] = 4
arr[0][1][1] = 5
arr[0][1][2] = 6

Array yang diinput:
1 2 3
4 5 6


Process returned 0 (0x0)   execution time : 12.850 s
Press any key to continue.
```

Deskripsi :

Input Ukuran Array:

- Program meminta pengguna untuk memasukkan ukuran dari array 3 dimensi (x, y, z).

Alokasi Memori:

- Menggunakan alokasi dinamis (new), program membuat array 3D berdasarkan ukuran yang diberikan.

Input Elemen Array:

- Pengguna diminta untuk memasukkan elemen-elemen untuk setiap posisi dalam array melalui tiga loop bersarang.

Menampilkan Array:

- Program mencetak isi array 3D dengan format terstruktur.

Dealokasi Memori:

- Memori yang dialokasikan dibebaskan menggunakan delete[] untuk mencegah kebocoran memori.

Akhir Program:

- Program diakhiri dengan return 0;, menandakan eksekusi selesai tanpa kesalahan.

3. Nomor 3

Code :

```
#include <iostream>
#include <vector>
#include <limits> // Untuk std::numeric_limits
using namespace std;

void findMax(const vector<int> &arr)
{
    int maxVal = numeric_limits<int>::min();
    for (int val : arr)
    {
        if (val > maxVal)
        {
            maxVal = val;
        }
    }
    cout << "Nilai Maksimum: " << maxVal << endl;
}

void findMin(const vector<int> &arr)
{
    int minVal = numeric_limits<int>::max();
    for (int val : arr)
    {
        if (val < minVal)
        {
            minVal = val;
        }
    }
    cout << "Nilai Minimum: " << minVal << endl;
}

void findAverage(const vector<int> &arr)
{
    double total = 0;
    for (int val : arr)
    {
        total += val;
    }
    double average = total / arr.size();
    cout << "Nilai Rata-rata: " << average << endl;
}

int main()
{
    int n;

    // Meminta pengguna memasukkan jumlah elemen
    cout << "Masukkan jumlah elemen array: ";
    cin >> n;

    vector<int> arr(n); // Membuat vector dengan ukuran n

    // Meminta pengguna untuk memasukkan elemen array
    cout << "Masukkan elemen array:\n";
    for (int i = 0; i < n; i++)
    {
        cout << "Elemen [" << i << "]: ";
        cin >> arr[i];
    }
```

```
int choice;
    do
    {
        // Menu pilihan
        cout << "\nMenu:\n";
        cout << "1. Cari Nilai Maksimum\n";
        cout << "2. Cari Nilai Minimum\n";
        cout << "3. Cari Nilai Rata-rata\n";
        cout << "4. Keluar\n";
        cout << "Pilih opsi (1-4): ";
        cin >> choice;

        switch (choice)
        {
        case 1:
            findMax(arr);
            break;
        case 2:
            findMin(arr);
            break;
        case 3:
            findAverage(arr);
            break;
        case 4:
            cout << "Keluar dari program." << endl;
            break;
        default:
            cout << "Pilihan tidak valid! Silakan pilih lagi." << endl;
        }
    } while (choice != 4);

    return 0;
}
```

Running :

```
Masukkan jumlah elemen array: 3
Masukkan elemen array:
Elemen [0]: 1
Elemen [1]: 2
Elemen [2]: 3

Menu:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-rata
4. Keluar
Pilih opsi (1-4): 1
Nilai Maksimum: 3

Menu:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-rata
4. Keluar
Pilih opsi (1-4): 2
Nilai Minimum: 1

Menu:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-rata
4. Keluar
Pilih opsi (1-4): 3
Nilai Rata-rata: 2

Menu:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-rata
4. Keluar
Pilih opsi (1-4): 4
Keluar dari program.

Process returned 0 (0x0)   execution time : 8.275 s
Press any key to continue.
```

Deskripsi :

Input Jumlah Elemen:

- Program meminta pengguna untuk memasukkan jumlah elemen yang ingin dimasukkan ke dalam array.

Inisialisasi Vector:

- Sebuah vector arr dibuat dengan ukuran yang ditentukan oleh pengguna untuk menyimpan elemen integer.

Input Elemen Array:

- Pengguna diminta untuk memasukkan elemen-elemen ke dalam array melalui loop.

Menu Pilihan:

- Program menampilkan menu yang memungkinkan pengguna memilih opsi untuk mencari nilai maksimum, minimum, atau rata-rata dari elemen array, atau keluar dari program.

Fungsi Pencarian:

- Tiga fungsi (findMax, findMin, findAverage) diimplementasikan untuk:
  - Mencari dan menampilkan nilai maksimum dari array.
  - Mencari dan menampilkan nilai minimum dari array.
  - Menghitung dan menampilkan rata-rata nilai dari elemen array.

Pengulangan Menu:

- Menu diulang menggunakan loop do-while hingga pengguna memilih untuk keluar dengan memasukkan opsi 4.

Kesimpulan

Melalui modul ini, peserta diharapkan memperoleh pemahaman yang mendalam tentang empat konsep penting dalam pemrograman C++: **fungsi**, **prosedur**, **array**, dan **pointer**. Fungsi dan prosedur membantu peserta memecah tugas besar menjadi bagian-bagian yang lebih kecil, memungkinkan pengembangan kode yang lebih modular, terstruktur, dan efisien. Array memudahkan pengelolaan koleksi data yang berjenis sama, sementara pointer memberikan fleksibilitas dalam mengelola memori dan memungkinkan manipulasi data secara langsung.

Integrasi antara pointer dan array memberikan kekuatan tambahan untuk mengakses dan memanipulasi data dengan lebih efisien. Selain itu, pemahaman tentang pointer sangat penting untuk bekerja dengan struktur data dinamis, pengoptimalan memori, dan pemrograman tingkat lanjut di C++. Kesimpulannya, pemahaman akan konsep-konsep dasar ini membekali peserta dengan keterampilan yang esensial dalam pengembangan program yang lebih kompleks dan efisien di masa depan.