LAPORAN PRAKTIKUM

Modul 02

“PENGENALAN BAHASA C++ (BAGIAN KEDUA)”

Disusun Oleh:

Dimastian Aji Wibowo (2311104058)

SE-07-02

Dosen :

Wahyu Andi Saputra, S.Pd., M.Eng.

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

PURWOKERTO

2024

1. **Tujuan**
    - Memahami penggunaan *pointer* dan alamat memori.
    - Mengimplementasikan fungsi dan prosedur dalam program.
2. **Landasan Teori**

A. Array

Array merupakan kumpulan data dengan nama yang sama dan setiap elemen bertipe data sama. Untuk mengakses setiap komponen / elemen array berdasarkan indeks dari setiap elemen.

1. Array Satu Dimensi

Array satu dimensi adalah *array* yang hanya terdiri dari satu larik dengan cara pendeklarasiannya seperti berikut:

```
tipe_data nama_var[ukuran]
```

Keterangan:

Tipe_data → menyatakan jenis elemen array (int, char, float, dll).

Ukuran → menyatakan jumlah maksimum array.

Contoh:

```
Int nilai[10];
```

Menyatakan bahwa array nilai mengandung 10 elemen dan bertipe integer. Dalam C++ data array disimpan dalam memori pada lokasi yang berurutan. Elemen pertama memiliki indeks 0 dan elemen selanjutnya memiliki indeks 1 dan seterusnya. Jadi jika terdapat array dengan 5 elemen maka elemen pertama memiliki indeks 0 dan elemen terakhir memiliki indeks 4.

2. Array Dua Dimensi

Bentuk *array* dua dimensi ini mirip seperti tabel yang menjadikan *array* dua dimensi bisa digunakan untuk menyimpan data dalam bentuk tabel dengan terbagi menjadi dua bagian, dimensi pertama dan dimensi kedua. Cara akses, deklarasi, inisialisasi, dan menampilkan data masih sama dengan *array* satu dimensi, hanya saja indeks yang digunakan ada dua.

Contoh:

int data_nilai[4][3];

nilai[2][0] = 10;

	0	1	2
0			
1			
2	10		
3			

3. Array Berdimensi Banyak

Array berdimensi banyak merupakan array yang mempunyai indeks banyak lebih dari dua. Indeks inilah yang menyatakan dimensi dari *array*. *Array* berdimensi banyak lebih susah dibayangkan, sejalan dengan jumlah dimensi dalam *array*.

Contoh:

int data_rumit[4][6][6]...[N];

Array sebenarnya masih banyak pengembangannya untuk menyimpan berbagai bentuk data, pengembangan *array* misalnya *array* tak berukuran.

B. Pointer

1. Data dan Memori

Semua data a 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”.

Saat program berjalan, Sistem Operasi (OS) akan mengalokasikan space memory untuk setiap variabel, objek, atau array yang kita buat. Lokasi pengalokasian memori bisa sangat teracak sesuai proses yang ada di dalam OS masing-masing.

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.

Cara pendeklarasian variabel pointer adalah sebagai berikut:

type *nama_variabel;

Contoh:

```
int *p_int;
/* p_int merupakan variabel pointer yang menunjuk ke data bertipe int */
```

Agar suatu pointer menunjuk ke variabel lain, mula-mula pointer harus diisi dengan alamat memori yang ditunjuk.

```
p_int = &j;
```

Pernyataan di atas berarti bahwa p_int diberi nilai berupa alamat dari variabel j. Setelah pernyataan tersebut di eksekusi maka dapat dikatakan bahwa p_int menunjuk ke variabel j. Jika suatu variabel sudah ditunjuk oleh pointer. Maka, variabel yang ditunjuk oleh pointer dapat diakses melalui variabel itu sendiri ataupun melalui pointer.

Untuk mendapatkan nilai dari variabel yang ditunjuk pointer, gunakan tanda * di depan nama variabel pointer. Pointer juga merupakan variabel, karena itu pointer juga akan menggunakan space memory dan memiliki alamat sendiri.

3. Pointer dan Array

Ada keterhubungan yang kuat antara array dan pointer. Banyak operasi yang bisa dilakukan dengan array juga bisa dilakukan dengan pointer. Pendeklarasian array: int a[10];

Mendefinisikan array sebesar 10, kemudian blok dari objek array tersebut diberi nama a[0],a[1],a[2],…….....a[9]. Notasi a[i] akan merujuk elemen ke-i dari array. Jika pa merupakan pointer yang menunjuk variabel bertipe integer , yang di deklarasikan sebagai berikut :

```
int *pa;
```

maka pernyataan :

pa=&a[0];

akan membuat pa menunjuk ke alamat dari elemen ke-0 dari variabel a. Sehingga, pa akan mengandung alamat dari a[0].

Sekarang, pernyataan :

x=*pa;

Akan menyalinkan isi dari $a[0]$ ke variabel x

Jika pa akan menunjuk ke elemen tertentu dari array, maka pendefinisian $pa + 1$ akan menunjuk elemen berikutnya, $pa + i$ akan menunjuk elemen ke-i setelah pa, sedangkan $pa - i$ akan menunjuk elemen ke-i sebelum pa sehingga jika pa menunjuk ke $a[0]$ maka $*(pa + 1)$ akan mengandung isi elemen ke $a[1]$. $pa + i$ merupakan alamat dari $a[i]$, dan $*(pa + i)$ akan mengandung isi dari elemen $a[i]$.

4. Pointer dan String

a. String

String merupakan bentuk data yang sering digunakan dalam bahasa pemrograman untuk mengolah data teks atau kalimat. Dalam bahasa C pada dasarnya string merupakan kumpulan dari karakter atau array dari karakter.

Deklarasi variabel string:

```
char nama[50];
```

50 → menyatakan jumlah maksimal karakter dalam string.

Memasukkan data string dari keyboard:

```
gets(nama_array);
```

contoh: gets(nama);

jika menggunakan cin():

contoh:

```
cin>>nama;
```

Inisialisasi string:

```
char nama[]= {‘s’,’t’,’r’,’u’,’k’,’d’,’a’,’t’,’\0’};
```

Merupakan variabel nama dengan isi data string “strukdat”.

Bentuk inisialisasi yang lebih singkat:

```
char nama[]=”strukdat”;
```

Menampilkan string bisa nggunakan puts() atau cout() :

```
puts(nama);
cout << nama;
```

Untuk mengakses data string sepertihalnya mengakses data pada array, pengaksesan dilakukan perkarakter sesuai dengan indeks setiap karakter dalam string dengan contoh :

```
Cout<<nama[3]; /*menampilkan karakter ke-3 dari string*/
```

b. Pointer dan String

Sesuai dengan penjelasan di atas , misalkan ada string :

“I am string"

Merupakan array dari karakter. Dalam representasi internal, array diakhiri dengan karakter ‘\0’ sehingga program dapat menemukan akhir dari program. Panjang dari storage merupakan panjang dari karakter yang ada dalam tanda petik dua ditambah satu. Ketika karakter string tampil dalam sebuah program maka untuk mengaksesnya digunakan pointer karakter. Standar input/output akan menerima pointer dari awal karakter array sehingga konstanta string akan diakses oleh pointer mulai dari elemen pertama.

Jika pmessage di deklarasikan :

```
char *pmessage ;
```

Maka pernyataan berikut :

```
pmessage = "now is the time";
```

Akan membuat pmessage sebagai pointer pada karakter array. Ini bukan copy string, hanya pointer yang terlibat. C tidak menyediakan operator untuk memproses karakter string sebagai sebuah unit.

Ada perbedaan yang sangat penting diantara pernyataan berikut :

```
char amessage[]="now is the time"; //merupakan array
char *pmessage= "now is the time"; //merupakan pointer
```

Variabel amessage merupakan sebuah array, hanya cukup besar untuk menampung karakter-karakter sequence tersebut dan karakter null ‘\0’ yang menginisialisasinya. Tiap-tiap karakter dalam array bisa saja berubah tapi variabel amessage akan selalu menunjuk apada storage yang sama. Di sisi lain, pmessage merupakan pointer, diinisialisasikan menunjuk konstanta string, pointer bisa di modifikasi untuk menunjuk kemanapun, tapi hasilnya tidak akan terdefinisi jika kamu mencoba untuk mengubah isi string

C. Fungsi

Fungsi merupakan blok dari kode yang dirancang untuk melaksanakan tugas khusus dengan tujuan:

1. Program menjadi terstruktur, sehingga mudah dipahami dan mudah dikembangkan. Program dibagi menjadi beberapa modul yang kecil
2. Dapat mengurangi pengulangan kode (duplikasi kode)sehingga menghemat

ukuran program.

Pada umumnya fungsi memerlukan masukan yang dinamakan sebagai parameter. Masukan ini selanjutnya diolah oleh fungsi. Hasil akhir fungsi berupa sebuah nilai (nilai balik fungsi). Berikut merupakan bentuk umum sebuah fungsi:

```
tipe_keluaran nama_fungsi(daftar_parameter) {
        blok pernyataan fungsi ;
}
```

Jika penentu_tipe fungsi merupakan tipe dari nilai balik fungsi, bila tidak disebutkan maka akan dianggap (default) sebagai int.

D. 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. Berikut merupakan bentuk umum sebuah prosedur:

```
void nama_prosedur (daftar_parameter) {
        blok pernyataan prosedur ;
}
```

E. Parameter Fungsi

1. Paramater Formal dan Parameter Aktual

Parameter formal adalah variabel yang ada pada daftar paramerter ketika mendefinisikan fungsi. Pada fungsi maks3() contoh diatas, a, b dan merupakan parameter formal.

```
float perkalian (float x, float y) {
        return (x*y);
}
```

Pada contoh di atas x dan y adalah parameter formal. Adapun parameter aktual adalah parameter (tidak selamanya menyatakan variabel) yang dipakai untuk memanggil fungsi.

```
X = perkalian(a, b);
Y = perkalian(10,30);
```

Dari pernyataan diatas a dan b merupakan parameter aktual, begitu pula 10 dan 30. parameter aktual tidak harus berupa variabel, melainkan bisa berupa konstanta atau

ungkapan.

2. Cara melewatkan 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 melewatkan alamatsuatu 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.

c. Pemanggilan dengan Referensi (Call by Reference)

Pemanggilan dengan referensi merupakan cara untuk melewatkan alamat suatu variabel kedalam 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.

3. Guided

A. Array Satu Dimensi

1. Menuliskan tipe data dari array diikuti mendeklarasikan jumlah array dan nilai dari array per indeksnya.
2. Menampilkan nilai array dengan memanggil nama array beserta indeksnya.
3. Dapat menggunakan perulangan untuk menampilkan nilai array dengan sekaligus.

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

    for(int i=0; i<5;i++){
        cout<<nilai[i]<<endl;
    }
}
```

4. Berikut merupakan output dari program tersebut.

B. Array Dua Dimensi

1. Sama seperti array satu dimensi, yang membedakan hanya hanya jumlah pendeklarasian array yang sebelumnya hanya satu sekarang menjadi dua.
2. Diikuti menuliskan nilai array sesuai dengan banyak arraynya.
3. Menampilkan nilai array dengan menggunakan perulangan agar lebih efisien.

```
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]<<""<<endl;
        }
    }
}
```

4. Berikut merupakan output dari program tersebut.

```
1
2
3
4
5
6
7
8
9
10
11
12
```

C. Pointer

1. Mendeklarasikan dua variabel yang bertipe integer.
2. Menginisialisasi variabel x dengan suatu nilai sekaligus mengisi pointer px dengan alamat memori x menggunakan &.
3. Mengisi variabel y dengan nilai yang ditunjuk oleh pointer px dan arena px menunjuk ke x, maka y akan mendapatkan nilai yang sama dengan x, maka operator * digunakan untuk mengakses nilai yang ditunjuk oleh pointer.
4. Menampilkan hasil dari operasi tersebut.
5. Menggunakan getch() untuk menunggu input keyboard sebelum program ditutup.
6. Menuliskan return 0 untuk menunjukkan akhir dari fungsi main().

```
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();
    return 0;
}
```

7. Berikut merupakan output dari program tersebut.

```
Alamat x = 0x61fe10
Isi px = 0x61fe10
Isi x = 87
Nilai yang ditunjuk px = 87
nilai y = 87
```

D. Fungsi

1. Membuat fungsi yang akan digunakan, sebagai contoh disini membuat fungsi penjumlahan dua bilangan integer dan fungsi untuk menampilkan pesan.
2. Setelah fungsi selesai dibuat, maka membuat fungsi main() dan memanggil hasil dari setiap fungsi yang telah dibuat diluar dari fungsi main().

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

void greet(string name){
    cout<<"Halo "<<name<<"!"<<endl;
}

int main(){

    int hasil = penjumlahan(5, 3);
    cout<<"Hasil "<<hasil<<endl;

    greet("World!");
}
```

3. Berikut merupakan output dari program tersebut.

```
Hasil 8
Halo World!!
```

4. Unguided

1. Program membuat array dengan input user dan memisahkan nilai array dengan bilangan genap dan bilangan ganjil

1. Menuliskan fungsi main().
2. Mendeklarasikan variabel diikuti membuat fungsi untuk mengambil input dari user yang berupa integer untuk jumlah elemen array dan menginisialisasi setiap indeksnya juga

```cpp
int main() {
    vector<int> dataArray;
    int jumlahData, input;

    cout << "Masukkan jumlah elemen data array: ";
    cin >> jumlahData;

    cout << "Masukkan elemen data array: " << endl;
    for (int i = 0; i < jumlahData; i++) {
        cin >> input;
        dataArray.push_back(input);
    }

    cout << "Data array: ";
    for (int i = 0; i < jumlahData; i++) {
        cout << dataArray[i];
        if (i != jumlahData - 1) {
            cout << ", ";
        }
    }
    cout << endl;
```

3. Menampilkan data array, menampilkan nomor genap dengan menggunakan if else, dan menampilkan nomor ganjil menggunakan if else.
4. Menuliskan return 0;

```
    cout << "Nomor genap: ";
    for (int i = 0; i < jumlahData; i++) {
        if (dataArray[i] % 2 == 0) {
            cout << dataArray[i];
            if (i != jumlahData - 2) {
                cout << ", ";
            }
        }
    }
    cout << endl;

    cout << "Nomor ganjil: ";
    for (int i = 0; i < jumlahData; i++) {
        if (dataArray[i] % 2 != 0) {
            cout << dataArray[i];
            if (i != jumlahData - 2) {
                cout << ", ";
            }
        }
    }
    cout << endl;
    return 0;
}
```

5. Berikut merupakan output dari program tersebut.

```
Masukkan jumlah elemen data array: 10
Masukkan elemen data array:
11
12
13
14
15
16
17
18
19
20
Data array: 11, 12, 13, 14, 15, 16, 17, 18, 19, 20
Nomor genap: 12, 14, 16, 18, 20,
Nomor ganjil: 11, 13, 15, 17, 19
```

2. Program pembuat array tiga dimensi dengan ukuran dari input user

1. Mendeklarasikan variabel x, y, z untuk mewakili array tiga dimensi dengan integer.
2. Membuat program untuk mengambil input dari user untuk ukuran dari array tiga dimensi tersebut dilanjutkan mendeklarasikan array tiga dimensi yang sudah diinputkan oleh user.
3. Meminta user untuk menginput nilai dari array yang sudah dibuat satu per satu dengan perulangan menyesuaikan dari ukuran array yang telah diinputkan oleh user.
4. Menampilkan array yang telah diinputkan oleh user dengan menggunakan perulangan agar lebih efisien dan cepat.
5. Menuliskan return 0 untuk mengakhiri fungsi main().

```
int main() {
    int x, y, z;
    cout << "Masukkan ukuran dimensi pertama (x): ";
    cin >> x;
    cout << "Masukkan ukuran dimensi kedua (y): ";
    cin >> y;
    cout << "Masukkan ukuran dimensi ketiga (z): ";
    cin >> z;

    int array[x][y][z];
    cout << "Masukkan elemen-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 << "Elemen [" << i << "][" << j << "][" << k << "]: ";
                cin >> array[i][j][k];
            }
        }
    }
    cout << "\nArray tiga dimensi yang diinputkan:" << endl;
    for (int i = 0; i < x; i++) {
        for (int j = 0; j < y; j++) {
            for (int k = 0; k < z; k++) {
                cout << "Elemen [" << i << "][" << j << "][" << k << "] = " << array[i][j][k] << endl;
            }
        }
    }
    return 0;
}
```

6. Berikut merupakan output dari program tersebut.

```
Masukkan ukuran dimensi pertama (x): 2
Masukkan ukuran dimensi kedua (y): 2
Masukkan ukuran dimensi ketiga (z): 2
Masukkan elemen-elemen array:
Elemen [0][0][0]: 1
Elemen [0][0][1]: 2
Elemen [0][1][0]: 3
Elemen [0][1][1]: 4
Elemen [1][0][0]: 5
Elemen [1][0][1]: 6
Elemen [1][1][0]: 7
Elemen [1][1][1]: 8

Array tiga dimensi yang diinputkan:
Elemen [0][0][0] = 1
Elemen [0][0][1] = 2
Elemen [0][1][0] = 3
Elemen [0][1][1] = 4
Elemen [1][0][0] = 5
Elemen [1][0][1] = 6
Elemen [1][1][0] = 7
Elemen [1][1][1] = 8
```

3. **Program menu untuk mencari nilai Maksimum, Minimum dan Nilai rata – rata dari suatu array dengan input yang dimasukan oleh user**
    1. Membuat fungsi untuk mencari nilai maksimum dari array menggunakan perulangan dengan membandingkan setiap nilai dari array per indeksnya mana yang lebih besar dan akan berhenti saat nilai dari suatu array merupakan nilai maksimum.
    2. Membuat fungsi untuk mencari nilai mininum dari array seperti fungsi untuk mencari nilai maksimum, yang membedakan hanya nilai array yang dibandingkan adalah yang lebih kecil sampai menemukan nilai terkecil.
    3. Membuat fungsi untuk menghitung rata – rata menggunakan perulangan untuk menambah semua nilai dari array lalu membaginya dengan seluruh jumlah array.

```
int cariMaksimum(int arr[], int n) {
    int maks = arr[0];
    for (int i = 1; i < n; i++) {
        if (arr[i] > maks) {
            maks = arr[i];
        }
    }
    return maks;
}
int cariMinimum(int arr[], int n) {
    int min = arr[0];
    for (int i = 1; i < n; i++) {
        if (arr[i] < min) {
            min = arr[i];
        }
    }
    return min;
}
float cariRataRata(int arr[], int n) {
    int sum = 0;
    for (int i = 0; i < n; i++) {
        sum += arr[i];
    }
    return static_cast<float>(sum) / n;
}
```

4. Membuat fungsi main() diikuti pendeklarasian variabel untuk menyimpan jumlah elemen array yang menggunakan input user dan membuat fungsi input.
5. Mendeklarasikan array dan membuat input nilai dari array dari user berdasarkan jumlah array yang telah diinputkan oleh user dengan perulangan.
6. Membuat menu untuk memilih fungsi apa yang dibutuhkan untuk user dengan do while dan switch, dengan setiap case mewakili setiap fungsi yang telah dibuat.

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

    int arr[n];
    cout << "Masukkan elemen-elemen array: " << endl;
    for (int i = 0; i < n; i++) {
        cout << "Elemen ke-" << i + 1 << ": ";
        cin >> arr[i];
    }
    do {
        cout << "\nMenu Operasi Array: " << endl;
        cout << "1. Cari Nilai Maksimum" << endl;
        cout << "2. Cari Nilai Minimum" << endl;
        cout << "3. Cari Nilai Rata-Rata" << endl;
        cout << "4. Keluar" << endl;
        cout << "Pilih menu (1-4): ";
        cin >> pilihan;
```

7. Membuat case terakhir untuk keluar dari program dan default jika input untuk memilih menu fungsi yang dimasukan tidak valid.
8. Menuliskan return 0 untuk mengakhiri fungsi main().

```
        switch (pilihan) {
            case 1:
                cout << "Nilai Maksimum: " << cariMaksimum(arr, n) << endl;
                break;
            case 2:
                cout << "Nilai Minimum: " << cariMinimum(arr, n) << endl;
                break;
            case 3:
                cout << "Nilai Rata-Rata: " << cariRataRata(arr, n) << endl;
                break;
            case 4:
                cout << "Keluar dari program." << endl;
                break;
            default:
                cout << "Pilihan tidak valid, silakan coba lagi." << endl;
        }
    } while (pilihan != 4);
    return 0;
}
```

9. Berikut merupakan output dari program tersebut.

Telkom University Purwokerto

```
Masukkan jumlah elemen array: 5
Masukkan elemen-elemen array:
Elemen ke-1: 1
Elemen ke-2: 2
Elemen ke-3: 3
Elemen ke-4: 4
Elemen ke-5: 5

Menu Operasi Array:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Pilih menu (1-4): 1
Nilai Maksimum: 5

Menu Operasi Array:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Pilih menu (1-4): 2
Nilai Minimum: 1
```

```
Menu Operasi Array:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Pilih menu (1-4): 1
Nilai Maksimum: 5

Menu Operasi Array:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Pilih menu (1-4): 2
Nilai Minimum: 1

Menu Operasi Array:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Pilih menu (1-4): 3
Nilai Rata-Rata: 3
```

```
Menu Operasi Array:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Pilih menu (1-4): 4
Keluar dari program.
```

5. Kesimpulan

Dalam praktikum ini, mempelajari konsep dasar yang penting seperti array, pointer, fungsi, prosedur, dan penggunaan parameter. Array digunakan untuk menyimpan sejumlah elemen dalam satu variabel, mempermudah akses dan manipulasi data secara terstruktur. Pointer diperkenalkan sebagai variabel unik yang menyimpan alamat data di memori. Ini memungkinkan untuk berinteraksi langsung dengan data di memori. Fungsi dan prosedur dipelajari untuk mengorganisasi kode dengan lebih baik, memisahkan logika program ke dalam blok-blok yang dapat digunakan kembali, dengan parameter sebagai alat untuk mengirimkan data antara fungsi/prosedur dan bagian lain dari program. Praktik ini meningkatkan pemahaman tentang bagaimana komponen berinteraksi satu sama lain, meningkatkan efisiensi dan modularitas dalam penulisan program.