

**LAPORAN PRAKTIKUM**  
**PERTEMUAN 2**  
**PENGENALAN C++ BAGIAN KEDUA**



**Nama :**

Ilham Lii Assidaq (2311104068)

**Dosen :**

Wahyu Andi Saputra

**PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2024**

## **I. TUJUAN**

- a. Memahami penggunaan pointer dan alamat memori
- b. Mengimplementasikan fungsi dan prosedur dalam program

## **II. TOOL**

Dev C++

## **III. DASAR TEORI**

Pada Praktikum kali ini, mempelajari banyak fitur pada C++ yang berguna untuk mengolah data dan fungsi programming lainnya, yaitu

### **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 terdiri dari satu baris data saja. Cara deklarasi array satu dimensi adalah dengan cara

```
tipe_data nama_var[ukuran]  
int nilai [9];
```

Tipe data menyatakan jenis elemen array (int, string, float, bool, dll)

Ukuran menyatakan jumlah maksimum array. Dalam C++ data array disimpan dalam memori pada lokasi yang berurutan. Elemen pertama memiliki indeks 0 dan 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**

Array dua dimensi mirip dengan tabel dan dapat digunakan untuk menyimpan data berbentuk tabel. Terdiri dari dua bagian, dimensi pertama dan kedua. Akses, deklarasi, inisialisasi, dan penampilan datanya serupa dengan array satu dimensi, namun menggunakan dua indeks.

```
int data_nilai[4][3];  
nilai[2][0] = 10;
```

	0	1	2
0			
1			
2	10		
3			

### 3. Array Berdimensi Banyak

Array berdimensi banyak memiliki lebih dari dua indeks yang menunjukkan dimensinya. Semakin banyak dimensinya, semakin sulit dibayangkan. Cara deklarasi

```
tipe_data nama_var[ukuran1][ukuran2]...[ukuran-N];
```

Contoh:

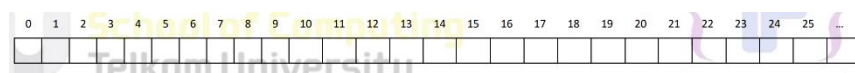
```
int data_rumit[4][6][6];
```

## b. Pointer

### 1. Data dan Memori

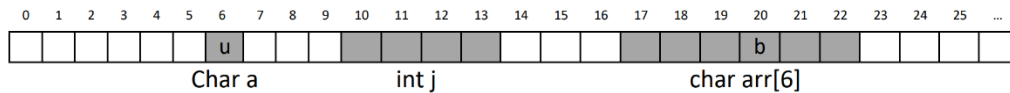
Semua data yang ada digunakan oleh program komputer disimpan di dalam memori (RAM) komputer. Memori dapat digambarkan sebagai sebuah array 1 dimensi yang berukuran sangat besar. Seperti layaknya array, setiap cell memory memiliki “indeks” atau “alamat” unik yang berguna untuk identitas yang biasa kita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

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.



Memori diasumsikan menyimpan 1 byte per cell. Saat komputer pertama kali berjalan, memori kosong. Ketika variabel dideklarasikan, OS mencari cell kosong untuk dialokasikan sebagai memori variabel tersebut

```
char a;
int j;
char arr[6];
arr[3] = 'b'
a = 'u'
```



Variabel `a` dialokasikan di memori pada alamat `x6`, variabel `j` pada alamat `x10-13`, dan variabel `arr` pada alamat `x17-22`. Nilai dari variabel-variabel yang disimpan dalam memori dapat diakses melalui alamat cell yang menyimpannya. Untuk mengetahui alamat memori suatu variabel, kita dapat menggunakan keyword “&” di depan nama variabel yang ingin dicari alamatnya.

C++	Output	Keterangan
<code>Cout &lt;&lt; a &lt;&lt; endl;</code>	'u'	Nilai variabel <code>a</code>
<code>Cout &lt;&lt; &amp;a &lt;&lt; endl;</code>	<code>x6</code>	Alamat variabel <code>a</code>
<code>Cout &lt;&lt; j &lt;&lt; endl;</code>	<code>0</code>	Nilai variabel <code>j</code>
<code>Cout &lt;&lt; &amp;j &lt;&lt; endl;</code>	<code>x10</code>	Alamat variabel <code>j</code>
<code>Cout &lt;&lt; &amp;(arr[4]) &lt;&lt; endl;</code>	<code>x21</code>	Alamat variabel <code>arr[4]</code>

## 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 mendklarasikan pointer sebagai berikut:

```
type *nama_variabel;
```

Contoh:

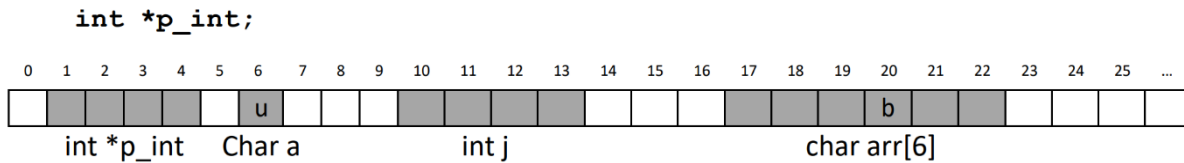
```
int *p_int;
```

```
/* 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 menunjukkan bahwa `p_int` diberi nilai berupa alamat variabel `j`, sehingga `p_int` menunjuk ke `j`. Variabel yang ditunjuk oleh pointer dapat diakses baik melalui variabel itu sendiri maupun melalui pointer. Untuk mendapatkan nilai dari variabel yang ditunjuk pointer, gunakan tanda “\*” di depan nama pointer. Karena pointer adalah variabel, ia juga memerlukan ruang memori dan memiliki alamat sendiri.



C++	Output	Keterangan
<pre>int j,k; j =10; int *p_int; p_int = &amp;j; cout&lt;&lt; j &lt;&lt; endl; cout&lt;&lt; &amp;j &lt;&lt; endl; cout&lt;&lt; p_int &lt;&lt; endl; cout&lt;&lt; &amp;p_int &lt;&lt; endl; cout&lt;&lt; *p_int &lt;&lt; endl; k = *p_int; cout &lt;&lt; k &lt;&lt; endl;</pre>	<pre>10 X6 X6 X1 10 10</pre>	<pre> Nilai variabel j Alamat variabel j Nilai variabel p_int Alamat variabel p_int Nilai variabel yang ditunjuk p_int  Nilai variabel k </pre>

```

D:\ITT SM 3\02_ARRAY\perce x + v
Alamat x = 0x6ffe00
Isi px = 0x6ffe00
Isi X = 87
Nilai yang ditunjuk px = 87
Nilai y = 87

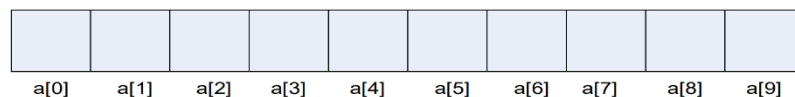
```

### 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 :

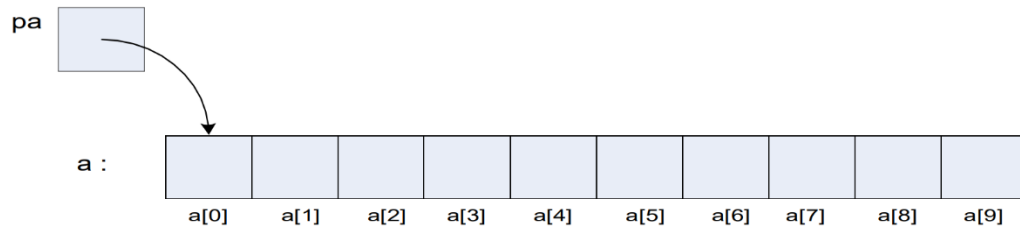
```
int *pa;
```

Maka pertanyaan

```
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` menunjuk ke elemen tertentu dari array, maka `pa + 1` akan menunjuk ke elemen berikutnya, `pa + i` akan menunjuk ke elemen ke-`i` setelah `pa`, dan `pa - i` akan menunjuk ke elemen ke-`i` sebelum `pa`. Jika `pa` menunjuk ke `a[0]`, maka `*(pa + 1)` akan mengandung nilai dari `a[1]`. Dengan demikian, `pa + i` adalah alamat dari `a[i]`, dan `*(pa + i)` berisi nilai dari elemen `a[i]`.

```
1 #include <iostream>
2 #include <conio.h>
3 #define MAX 5
4
5 using namespace std;
6
7 int main() {
8     int i, j;
9     float nilai total, rata_rata;
10    float nilai[MAX];
11
12    static int nilai_tahun[MAX][MAX] = {
13        {0, 2, 2, 0, 0},
14        {0, 1, 1, 1, 0},
15        {0, 3, 3, 3, 0},
16        {4, 4, 0, 0, 4},
17        {5, 0, 0, 0, 5}
18    };
19
20    for (i = 0; i < MAX; i++) {
21        cout << "Masukkan nilai ke-" << i + 1 << ": ";
22        cin >> nilai[i];
23    }
24
25    cout << "\nData nilai siswa:\n";
26
27    for (i = 0; i < MAX; i++) {
28        cout << "Nilai ke-" << i + 1 << " = " << nilai[i] << endl;
29    }
30
31    cout << "\nNilai tahunan:\n";
32
33    for (i = 0; i < MAX; i++) {
34        for (j = 0; j < MAX; j++) {
35            cout << nilai_tahun[i][j] << " ";
36        }
37        cout << "\n";
38    }
39
40    getch();
41    return 0;
42 }
```

## 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 menggunakan `puts()` atau `cout()` :

```
puts(nama);
```

```
cout << nama;
```

Untuk mengakses data string sepertihalnya mengakses data pada array, pengaksesan dilakukan per karakter sesuai dengan indeks setiap karakter dalam string.

Contoh :

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

## **B. Pointer dan string**

Sesuai dengan penjelasan di atas , misalkan ada string :

“Telkom University”

Array karakter diakhiri dengan karakter ‘\0’ untuk menandai akhir string.

Panjang penyimpanan adalah panjang karakter dalam tanda petik dua ditambah satu. Untuk mengakses string dalam program, digunakan pointer karakter. Standar input/output menerima pointer dari awal array, sehingga string dapat diakses 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 adalah array yang cukup besar untuk menampung karakter sequence dan karakter null ‘\0’ untuk inisialisasi. Setiap karakter dalam array dapat berubah, tetapi amessage akan selalu menunjuk ke storage yang sama. Sebaliknya, pmessage adalah pointer yang diinisialisasikan untuk menunjuk konstanta string dan dapat dimodifikasi untuk menunjuk

kemanapun. Namun, hasilnya tidak terdefinisi jika mencoba mengubah isi string yang ditunjuk.

### 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)

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

```
1  #include <iostream>  
2  #include <stdlib.h>  
3  #include <conio.h>  
4  
5  using namespace std;  
6  
7  // Mendeklarasikan prototype fungsi  
8  int maks3(int a, int b, int c);  
9  
10 int main() {  
11     system("cls");  
12     int x, y, z;  
13  
14     // Input nilai bilangan  
15     cout << "Masukkan nilai bilangan ke-1: ";  
16     cin >> x;  
17     cout << "Masukkan nilai bilangan ke-2: ";  
18     cin >> y;  
19     cout << "Masukkan nilai bilangan ke-3: ";  
20     cin >> z;  
21  
22     // Output nilai maksimum  
23     cout << "Nilai maksimumnya adalah: " << maks3(x, y, z) << endl;  
24  
25     getch();  
26     return 0;  
27 }  
28  
29 /* Badan fungsi */  
30 int maks3(int a, int b, int c) {  
31     // Deklarasi variabel lokal dalam fungsi  
32     int temp_max = a;  
33  
34     if (b > temp_max)  
35         temp_max = b;  
36  
37     if (c > temp_max)  
38         temp_max = c;  
39  
40     return temp_max;  
41 }
```



#### d. Prosedur

Dalam C, semua adalah fungsi, termasuk main(). Prosedur di C adalah fungsi yang tidak mengembalikan nilai, biasanya diawali dengan kata kunci void. Berikut adalah bentuk umum sebuah prosedur:

```
1  #include <iostream>
2  #include <conio.h>
3  #include <stdlib.h>
4
5  using namespace std;
6
7  void tulis(int x);
8
9  int main() {
10     system("cls");
11     int jum;
12
13     cout << "Jumlah baris kata = ";
14     cin >> jum;
15
16     tulis(jum);
17
18     getch();
19     return 0;
20 }
21
22 void tulis(int x) {
23     for (int i = 0; i < x; i++) {
24         cout << "Baris ke-" << i + 1 << endl;
25     }
26 }
```

#### e. Parameter Fungsi

##### 1. Parameter Formal dan Parameter Aktual

Parameter formal adalah variabel yang ada pada daftar parameter 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 disalin ke parameter formal, sehingga parameter aktual tidak berubah meskipun parameter formal mengalami perubahan. Berikut adalah contohnya:

```
1  #include <iostream>
2  #include <conio.h>
3  #include <stdlib.h>
4
5  using namespace std;
6
7  void tukar(int x, int y);
8
9  int main() {
10     int a, b;
11     system("cls");
12
13     a = 4;
14     b = 6;
15
16     cout << "Kondisi sebelum ditukar:\n";
17     cout << "a = " << a << " b = " << b << endl;
18
19     tukar(a, b);
20
21     cout << "Kondisi setelah ditukar:\n";
22     cout << "a = " << a << " b = " << b << endl;
23
24     getch();
25     return 0;
26 }
27
28 void tukar(int x, int y) {
29     int temp;
30     temp = x;
31     x = y;
32     y = temp;
33
34     cout << "Nilai akhir pada fungsi tukar:\n";
35     cout << "x = " << x << " y = " << y << endl;
36 }
```

Hasil eksekusi :

Kondisi sebelum tukar a = 4, b = 6

Nilai akhir pada fungsi tukar x = 6, y = 4

Kondisi setelah tukar a = 4, b = 6

Pada pemanggilan fungsi tukar, nilai variabel a dan b tidak berubah. Ini karena saat pemanggilan fungsi, nilai a dan b disalin ke variabel formal x dan y.

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

Pemanggilan dengan pointer merupakan cara untuk melewatkan 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

Cara penulisan :

```
tukar(int *px, int *py) {
    int temp;
    temp = *px;
```

```

        *px = *py;
        *py = temp; ... ...
    }

```

Cara pemanggilan:

```
tukar(&a, &b);
```

Pada ilustrasi tersebut, px adalah variabel pointer yang menunjuk ke variabel integer. Pada pemanggilan fungsi tukar(), &a dan &b menunjukkan alamat dari a dan b. Dengan cara ini, variabel yang diubah dalam fungsi tukar() adalah variabel yang dilewatkan, karena yang dilewatkan adalah alamat, bukan sekadar salinan nilai.

### C. Pemanggilan dengan Referensi (Call by Pointer)

Pemanggilan dengan referensi adalah metode untuk melewatkan alamat suatu variabel ke dalam fungsi, sehingga dapat mengubah nilai dari variabel aktual yang dilewatkan. Dengan cara ini, variabel di luar fungsi dapat diubah. Berikut adalah cara penulisannya:

```

tukar(int &px, int &py) {
    int temp;
    temp = px;
    px = py;
    py = temp;
    ... ...
}

```

Cara pemanggilan:

```
tukar(a, b);
```

untuk melewatkan nilai dengan referensi, argumen disampaikan ke fungsi seperti nilai lainnya. Di parameter, harus dideklarasikan di awal, dan pemanggilan tidak memerlukan parameter tambahan seperti pada call by pointer.

```

1 #include <iostream>
2 #include <conio.h>
3 #include <stdlib.h>
4
5 using namespace std;
6
7 void tukar(int &x, int &y);
8
9 int main() {
10     int a, b;
11     system("cls");
12
13     a = 4;
14     b = 6;
15
16     cout << "Kondisi sebelum ditukar:\n";
17     cout << "a = " << a << " b = " << b << endl;
18
19     tukar(a, b);
20
21     cout << "Kondisi setelah ditukar:\n";
22     cout << "a = " << a << " b = " << b << endl;
23
24     getch();
25     return 0;
26 }
27
28 void tukar(int &x, int &y) {
29     int temp;
30     temp = x;
31     x = y;
32     y = temp;
33
34     cout << "Nilai akhir pada fungsi tukar:\n";
35     cout << "x = " << x << " y = " << y << endl;
36 }

```

## IV. GUIDE

### a. Array 1 Dimensi

```

1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int angka[5] = {1, 2, 3, 4, 5};
7     for (int i = 0; i < 5; i++) {
8         cout << angka[i] << endl;
9     }
10 }

```

D:\VIT SM 3\02\_ARRAY\perc7. x + v

```

1
2
3
4
5

```

### b. Array 2 dimensi

```

1 #include <iostream>
2
3 using namespace std;
4
5 int main() {
6     int matriks[3][3] = {
7         {1, 2, 3},
8         {4, 5, 6},
9         {7, 8, 9}
10    };
11
12    cout << "3X3:" << endl;
13    for (int i = 0; i < 3; i++) {
14        for (int j = 0; j < 3; j++) {
15            cout << matriks[i][j] << " ";
16        }
17        cout << endl;
18    }
19 }

```

D:\VIT SM 3\02\_ARRAY\p9.exe x + v

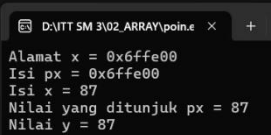
```

21 3X3:
22 1 2 3
23 4 5 6
24 7 8 9
25

```

### c. Pointer

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int x, y;
6     int *px;
7
8     x = 87;
9     px = &x;
10    y = *px;
11
12    cout << "Alamat x = " << &x << endl;
13    cout << "Isi px = " << px << endl;
14    cout << "Isi x = " << x << endl;
15    cout << "Nilai yang ditunjuk px = " << *px << endl;
16    cout << "Nilai y = " << y << endl;
17
18    return 0;
19 }
20
```



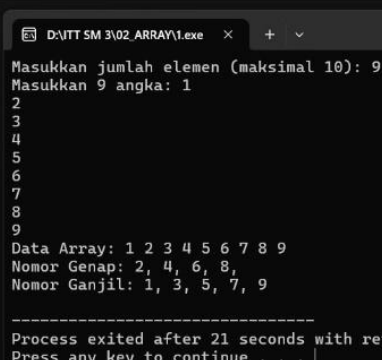
Alamat x = 0x6ffe00  
Isi px = 0x6ffe00  
Isi x = 87  
Nilai yang ditunjuk px = 87  
Nilai y = 87

## V. UNGUIDED

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

```
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,
```

```
1 #include <iostream>
2 #include <vector>
3
4 using namespace std;
5
6 int main() {
7     const int maksimal = 10;
8     int arr[maksimal], n;
9
10    cout << "Masukkan jumlah elemen (maksimal " << maksimal << "): ";
11    cin >> n;
12
13    if (n <= 0 || n > maksimal) {
14        cout << "Jumlah elemen tidak valid." << endl;
15        return 1;
16    }
17
18    cout << "Masukkan " << n << " angka: ";
19    for (int i = 0; i < n; i++) {
20        cin >> arr[i];
21    }
22
23    cout << "Data Array: ";
24    for (int i = 0; i < n; i++) {
25        cout << arr[i] << " ";
26    }
27    cout << endl;
28
29    cout << "Nomor Genap: ";
30    for (int i = 0; i < n; i++) {
31        if (arr[i] % 2 == 0) cout << arr[i] << (i < n-1 ? ", " : "");
32    }
33    cout << endl;
34
35    cout << "Nomor Ganjil: ";
36    for (int i = 0; i < n; i++) {
37        if (arr[i] % 2 != 0) cout << arr[i] << (i < n-1 ? ", " : "");
38    }
39    cout << endl;
40
41    return 0;
42 }
```



Masukkan jumlah elemen (maksimal 10): 9  
Masukkan 9 angka: 1  
2  
3  
4  
5  
6  
7  
8  
9  
Data Array: 1 2 3 4 5 6 7 8 9  
Nomor Genap: 2, 4, 6, 8,  
Nomor Ganjil: 1, 3, 5, 7, 9  
-----  
Process exited after 21 seconds with return code 0  
Press any key to continue . . .

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

```

1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int x, y, z;
6
7     // Input ukuran array tiga dimensi
8     cout << "Masukkan nilai (x, y, z): ";
9     cin >> x >> y >> z;
10
11     // Deklarasi dan input elemen array tiga dimensi
12     int arr[x][y][z];
13     cout << "Masukkan array:" << endl;
14     for (int i = 0; i < x; i++)
15         for (int j = 0; j < y; j++)
16             for (int k = 0; k < z; k++) {
17                 cout << "arr[" << i << "][" << j << "][" << k << "] = ";
18                 cin >> arr[i][j][k];
19             }
20
21     // Menampilkan elemen array
22     cout << "array nya adalah:" << endl;
23     for (int i = 0; i < x; i++)
24         for (int j = 0; j < y; j++)
25             for (int k = 0; k < z; k++)
26                 cout << "arr[" << i << "][" << j << "][" << k << "] = " << arr[i][j][k] << endl;
27
28     return 0;
29 }

```

D:\ITT SM 3\02\_ARRAY\2.exe x +

```

Masukkan nilai (x, y, z): 1
2
3
Masukkan 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] = 4
array nya adalah:
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] = 4
-----
Process exited after 10.23 sec
Press any key to continue . .

```

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

```

1 #include <iostream>
2 using namespace std;
3
4 void hitungNilai(int arr[], int n) {
5     int max = arr[0], min = arr[0];
6     float sum = 0;
7
8     for (int i = 0; i < n; i++) {
9         if (arr[i] > max) max = arr[i];
10        if (arr[i] < min) min = arr[i];
11        sum += arr[i];
12    }
13
14    float avg = sum / n;
15    cout << "Max: " << max << endl;
16    cout << "Min: " << min << endl;
17    cout << "Average: " << avg << endl;
18 }
19
20 int main() {
21     int n;
22
23     cout << "Masukkan jumlah array: ";
24     cin >> n;
25
26     int arr[n];
27     cout << "Masukkan elemen array:" << endl;
28     for (int i = 0; i < n; i++) {
29         cout << "Array ke - " << i + 1 << ": ";
30         cin >> arr[i];
31     }
32
33     hitungNilai(arr, n);
34
35     return 0;

```

D:\ITT SM 3\02\_ARRAY\33.exe x + v

```

Masukkan jumlah array: 9
Masukkan elemen array:
Array ke - 1: 1
Array ke - 2: 2
Array ke - 3: 3
Array ke - 4: 4
Array ke - 5: 5
Array ke - 6: 6
Array ke - 7: 7
Array ke - 8: 8
Array ke - 9: 9
Max: 9
Min: 1
Average: 5
-----
Process exited after 16.78 seconds with return value 0
Press any key to continue . . .

```

## **VI. KESIMPULAN**

Pada praktikum kali ini, modul 2 mempelajari Array 1 dan 2 dimensi, pointer, fungsi dan kombinasi dll.

Praktikum ini saya memahami bahwa array adalah kumpulan elemen dengan tipe data yang sama, yang bisa di akses menggunakan indeks, baik dalam satu dimensi maupun multidimensi. Sementara itu, pointer menjadi salah satu alat yang sangat berguna, karena pointer adalah variabel yang menyimpan alamat memori dari variabel lain. Ini memungkinkan untuk mengakses dan memanipulasi data dengan cara yang lebih efisien.

Kesimpulan akhirnya adalah praktikum kali ini memberikan wawasan tentang fitur C++ terkhususnya Array dan Pointer sebagai tools untuk menciptakan perangkat lunak atau program yang akan saya kerjakan dan semoga bermanfaat di masa depan.