

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



**Disusun Oleh:**

**Nama: RifqiMRamdani**

**NIM: 2311104044**

**SE07-02**

**Dosen :**

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

**PROGRAM STUDI S1 SOFTWARE ENGINEERING**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY**

**PURWOKERTO**

**2024**

## 1. Tujuan

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

## 2. Landasan Teori

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.

## 3. Guided

### Array Satu Dimensi

Adalah array yang hanya terdiri dari satu larik data saja. Cara pendeklarasian array satu dimensi:

```
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.

nilai[5] → elemen ke-5 dari *array* nilai. Contoh memasukkan data ke dalam *array*:

```
nilai[4] = 90;           /*memasukkan 90 ke dalam array nilai indeks ke-4*/  
cin << nilai[4]         /*membaca input-an dari keyboard*/
```

### Array Dua Dimensi

Bentuk array dua dimensi ini mirip seperti tabel. Jadi array dua dimensi bisa digunakan untuk menyimpan data dalam bentuk tabel. Terbagi menjadi dua bagian, dimensi pertama dan dimensi kedua. Cara akses, deklarasi, inisialisasi, dan menampilkan data 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			

Gambar 2-1 Ilustrasi Array Dua Dimensi

Array Berdimensi Banyak Merupakan array yang mempunyai indeks banyak, lebih dari dua. Indeks inilah yang menyatakan dimensi array. Array berdimensi banyak lebih susah dibayangkan, sejalan dengan jumlah dimensi dalam array. Cara deklarasi

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

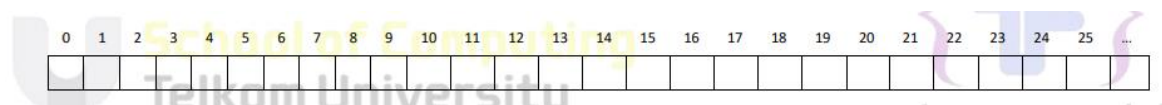
Contoh:

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

### Data dan Memori

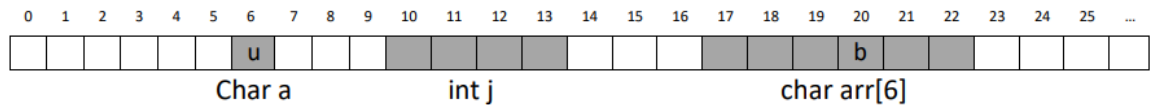
Semua data yang ada digunakan oleh program komputer disimpan di dalam memori (RAM) komputer. Memori dapat digambarkan sebagai sebuah array 1 dimensi yang berukuran sangat besar. Seperti layaknya array, setiap cell memory memiliki “indeks” atau “alamat” unik yang berguna untuk identitas yang biasa kita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



Digambarkan sebuah memory diasumsikan setiap cell menyimpan 1 byte data. Pada saat komputer pertama kali berjalan keadaan memori adalah kosong. Saat variabel dideklarasikan, OS akan mencari cell kosong untuk dialokasikan sebagai memori variabel tersebut.

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



Pada contoh di atas variabel `a` dialokasikan di memori alamat `x6`, variabel `j` dialokasikan di alamat `x10-13`, dan variabel `arr` dialokasikan di alamat `x17-22`. Nilai variabel yang ada di dalam memori dapat dipanggil menggunakan alamat dari cell yang menyimpannya. Untuk mengetahui alamat memori tempat di mana suatu variabel dialokasikan, kita bisa menggunakan keyword “&” yang ditempatkan di depan nama variabel yang ingin kita cari 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>	0	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>

## 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 */
```

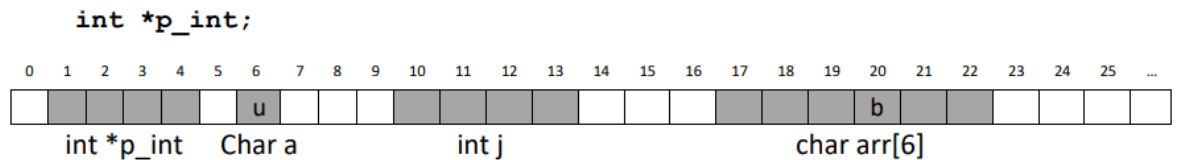
Agar suatu *pointer* menunjuk ke variabel lain, mula-mula *pointer* harus diisi dengan alamat memori yang ditunjuk.

```
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



Gambar 2-4 Ilustrasi Alokasi *Pointer*

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>

Berikut ini contoh program sederhana menggunakan *pointer*:

```

1  #include <iostream>
2  #include <conio.h>
3  using namespace std;
4
5  int main(){
6      int x,y; //x dan y bertipe int
7      int *px; //px merupakan variabel pointer menunjuk ke variabel int
8      x =87;
9      px=&x;
10     y=*px;
11     cout<<"Alamat x= "<<&x<<endl;
12     cout<<"Isi px= "<<px<<endl;
13     cout<<"Isi X= "<<x<<endl;
14     cout<<"Nilai yang ditunjuk px= "<<*px<<endl;
15     cout<<"Nilai y= "<<y<<endl;
16     getch();
17     return 0;
18 }
```

```

Alamat x= 0022FF14
Isi px= 0022FF14
Isi X= 87
Nilai yang ditunjuk px= 87
Nilai y= 87

```

## 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]`.



Gambar 2-6 Array

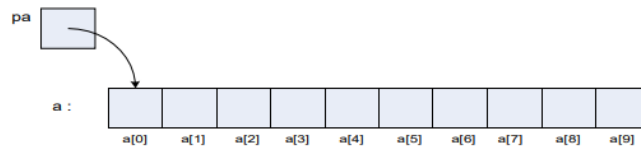
Notasi  $a[i]$  akan merujuk elemen ke- $i$  dari *array*. Jika  $pa$  merupakan *pointer* yang menunjuk variabel bertipe *integer*, yang di deklarasikan sebagai berikut :

```
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  $a[i]$ .

```

1  #include <iostream>
2  #include <conio.h>
3  #define MAX 5
4  using namespace std;
5
6  int main(){
7      int i,j;
8      float nilai_total, rata_rata;
9      float nilai[MAX];
10     static int nilai_tahun[MAX][MAX]=
11     {
12         {0,2,2,0,0},
13         {0,1,1,1,0},
14         {0,3,3,3,0},
15         {4,4,0,0,4},
16         {5,0,0,0,5}
17     };
18     /*inisialisasi array dua dimensi */
19     for (i=0; i<MAX; i++){
20         cout<<"masukkan nilai ke-"<<i+1<<endl;
21         cin>>nilai[i];
22     }
23     cout<<"\ndata nilai siswa :\n";
24     /*menampilkan array satu dimensi */
25     for (i=0; i<MAX; i++)
26         cout<<"nilai k-"<<i+1<<"=" <<nilai[i]<<endl;
27     cout<<"\n nilai tahunan : \n";
28
29     /* menampilkan array dua dimensi */
30     for(i=0; i<MAX; i++){
31         for(j=0; j<MAX; j++)
32             cout<<nilai_tahun[i][j];
33         cout<<"\n";
34     }
35     getch();
36     return 0;
37 }

```

## 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*/
```

## 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.

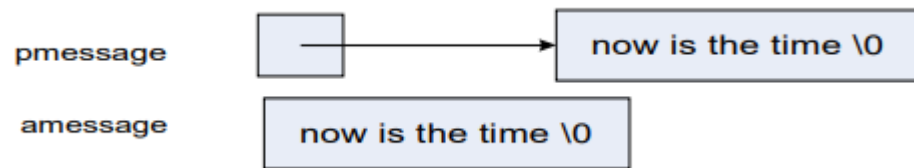
Ada perbedaan yang sangat penting diantara pernyataan berikut :

```
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 menginisiasinya. Tiap-tiap karakter dalam array bisa saja berubah tapi variabel amessage akan selalu menunjuk apada storage yang sama. Di sisi lain, pmessage merupakan pointer, diinisiasikan menunjuk konstanta string,



pointer bisa di modifikasi untuk menunjuk kemanapun, tapi hasilnya tidak akan terdefinisi jika kamu mencoba untuk mengubah isi string.



## 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:

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

Algoritma	C++
Program coba_fungsi Kamus x,y,z : integer function maks3(input: a,b,c : integer) : integer Algoritma input(x,y,z) output( maks3(x,y,z) ) function maks3(input:a,b,c : integer) : integer kamus temp_max : integer algoritma temp_max ← a if (b>temp_max) then temp_max ← b if (c>temp_max) then temp_max ← c → temp_max	<pre> #include &lt;conio.h&gt; #include &lt;iostream&gt; #include &lt;stdlib.h&gt; using namespace std; int maks3(int a, int b, int c); /*mendeklarasikan prototype fungsi */ int main(){     system("cls");     int x,y,z;     cout&lt;&lt;"masukkan nilai bilangan ke-1     =";     cin&gt;&gt;x;     cout&lt;&lt;"masukkan nilai bilangan ke-2     =";     cin&gt;&gt;y;     cout&lt;&lt;"masukkan nilai bilangan ke-3     =";     cin&gt;&gt;z;     cout&lt;&lt;"nilai maksimumnya adalah ="     &lt;&lt;maks3(x,y,z) ;     getch();     return 0; } /*badan fungsi */ int maks3(int a, int b, int c){     /* deklarasi variabel lokal dalam     fungsi */     Int temp_max =a;     if(b&gt;temp_max)         temp_max=b;     if(c&gt;temp_max)         temp_max=c;     return (temp_max); } </pre>

## 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. Bentuk umum sebuah prosedur:

```
void nama_prosedur (daftar_parameter) {  
    blok_pernyataan_prosedur ;  
}
```

Algoritma	C++
<pre>Program coba_prosedur Kamus     jum : integer     procedure tulis(input:         x: integer) Algoritma     input(jum)     tulis(jum)  procedure tulis(input: x: integer) kamus     i : integer algoritma     i traversal [1..x]     output("baris ke-", i+1)</pre>	<pre>#include &lt;iostream&gt; #include &lt;conio.h&gt; #include &lt;stdlib.h&gt;  using namespace std; /*prototype fungsi */ void tulis(int x); int main() {     System("cls");     int jum;     cout &lt;&lt; " jumlah baris kata=";     cin &gt;&gt; jum;     tulis(jum);     getch();     return 0; } /*badan prosedur*/ void tulis(int x){     for (int i=0; i&lt;x; i++)         cout&lt;&lt;"baris ke-"&lt;&lt;i+1&lt;&lt;endl; }</pre>

## Parameter Fungsi

### Paramater 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

## 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. Untuk lebih jelasnya perhatikan contoh berikut:

Algoritma	C++
<pre>Program coba_parameter_by_value Kamus   a,b : integer    procedure tukar(input:     x,y : integer) Algoritma   a ← 4   b ← 6   output(a,b)   tukar(a,b)   output(a,b)  procedure tukar(input:x,y : integer) kamus   temp : integer algoritma   temp ← x   x ← y   y ← temp   output(x,y)</pre>	<pre>#include &lt;iostream&gt; #include &lt;conio.h&gt; #include &lt;stdlib.h&gt;  using namespace std; /*prototype fungsi */ void tukar(int x, int y);  int main () {   int a, b;   system("cls");   a=4; b=6;   cout &lt;&lt; "kondisi sebelum ditukar \n";   cout &lt;&lt; " a = "&lt;&lt;a&lt;&lt;" b = "&lt;&lt;b&lt;&lt;endl;   tukar(a,b);   printf("kondisi setelah ditukar \n");   cout &lt;&lt; " a = "&lt;&lt;a&lt;&lt;" b = "&lt;&lt;b&lt;&lt;endl;   getch();   return 0; }  void tukar (int x, int y) {   int temp;   temp = x;   x = y;   y = temp;   cout &lt;&lt; "nilai akhir pada fungsi tukar \n";   cout &lt;&lt; " x = "&lt;&lt;x&lt;&lt;" y = "&lt;&lt;y&lt;&lt;endl; }</pre>

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
```

Jelas bahwa pada pemanggilan fungsi tukar, yang melewati variabel a dan b tidak merubah nilai dari variabel tersebut. Hal ini dikarenakan ketika pemanggilan fungsi tersebut nilai dari a dan b disalin ke variabel formal yaitu x dan y.

### 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.  
Cara penulisan :

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

Cara pemanggilan: tukar  
(&a, &b);

Pada ilustrasi tersebut, \*px merupakan suatu variabel pointer yang menunjuk ke suatu variabel interger. Pada pemanggilan fungsi tukar(), &a dan &b menyatakan “alamat a” dan “alamat b”. dengan cara diatas maka variabel yang diubah dalam fungsi tukar() adalah variabel yang dilewatkan dalam fungsi itu juga, karena yang dilewatkan dalam fungsi adalah alamat dari variabel tersebut, jadi bukan sekedar disalin.

### **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. 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);
```

untuk melewati nilai dengan referensi, argumen dilalui ke fungsi seperti nilai lain. Jadi pada akhirnya, harus mendeklarasikan di parameter awal, serta untuk pemanggilan tidak perlu menggunakan paramter tambahan seperti pada *call by pointer*.

Algoritma	C++
Program coba_parameter_by_reference  Kamus a,b : integer  procedure tukar(input/output: x,y : integer)  Algoritma a ← 4 b ← 6 output(a,b)  tukar(a,b)  output(a,b)  procedure tukar(input/output :x,y : integer) kamus temp : integer  algoritma temp ← x x ← y y ← temp  output(x,y)	<pre> #include &lt;iostream&gt; #include &lt;conio.h&gt; #include &lt;stdlib.h&gt;  using namespace std; /*prototype fungsi */ void tukar(int &amp;x, int &amp;y); int main () {     int a, b;     system("cls");     a=4;  b=6;     cout &lt;&lt; "kondisi sebelum ditukar \n";     cout &lt;&lt; " a = "&lt;&lt;a&lt;&lt;" b = "&lt;&lt;b&lt;&lt;endl;     tukar(a,b);     printf("kondisi setelah ditukar \n");     cout &lt;&lt; " a = "&lt;&lt;a&lt;&lt;" b = "&lt;&lt;b&lt;&lt;endl;     getch();     return 0; }  void tukar (int &amp;x, int &amp;y) {     int temp;     temp = x;     x = y;     y = temp;     cout&lt;&lt; "nilai akhir pada fungsi tukar \n";     cout &lt;&lt; " x = "&lt;&lt;x&lt;&lt;" y = "&lt;&lt;y&lt;&lt;endl; </pre>

Call By  
Reference

	}
--	---

Algoritma	C++
Program coba_parameter_by_pointer Kamus a,b : integer procedure tukar(input/output: x,y : integer) Algoritma a ← 4 b ← 6 output(a,b) tukar(a,b) output(a,b)  procedure tukar(input/output :x,y : integer) kamus temp : integer algoritma temp ← x x ← y y ← temp output(x,y)	<pre> #include &lt;iostream&gt; #include &lt;conio.h&gt; #include &lt;stdlib.h&gt;  using namespace std; /*prototype fungsi */ void tukar(int *x, int *y); int main () {     int a, b;     system("cls");     a=4; b=6;     cout &lt;&lt; "kondisi sebelum ditukar \n";     cout &lt;&lt; " a = "&lt;&lt;a&lt;&lt;" b = "&lt;&lt;b&lt;&lt;endl;     tukar(&amp;a,&amp;b);     printf("kondisi setelah ditukar \n");     cout &lt;&lt; " a = "&lt;&lt;a&lt;&lt;" b = "&lt;&lt;b&lt;&lt;endl;     getch();     return 0; }  void tukar (int *x, int *y) {     int temp;     temp = *x;     *x = *y;     *y = temp;     cout &lt;&lt; "nilai akhir pada fungsi tukar \n";     cout &lt;&lt; " x = "&lt;&lt;x&lt;&lt;" y = "&lt;&lt;y&lt;&lt;endl; } </pre> <div data-bbox="1300 544 1466 719" style="border: 1px solid blue; padding: 5px; display: inline-block;"> <i>Call by Pointer</i> </div>

## LATIHAN PRAKTIKUM DIKELAS

```
main.cpp x
1  #include <iostream>
2
3  using namespace std;
4
5  int main() {
6
7      int nilai[5]={1,2,3,4,5};
8      cout << nilai[0];
9      cout << nilai[1];
10     cout << nilai[2];
11     cout << nilai[3];
12     cout << nilai[4];
13
14
15 }
16
17
```

Maka akan menghasilkan output

```
"D:\TUGAS SEMESTER 3\guide x + v
12345
Process returned 0 (0x0)   execution time : 0.273 s
Press any key to continue.
|
```

2.

```

#include <iostream>
using namespace std;

int main() {

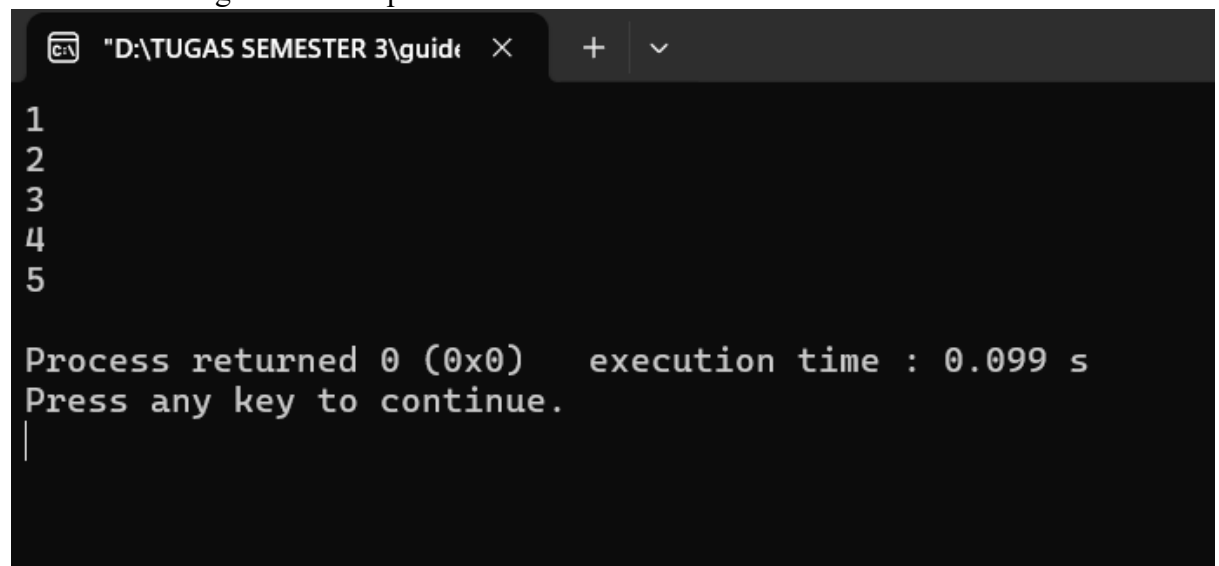
    int nilai[5] = {1, 2, 3, 4, 5};

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

    return 0;
}

```

Maka akan menghasilkan output



```

1
2
3
4
5

Process returned 0 (0x0)   execution time : 0.099 s
Press any key to continue.
|

```

3.

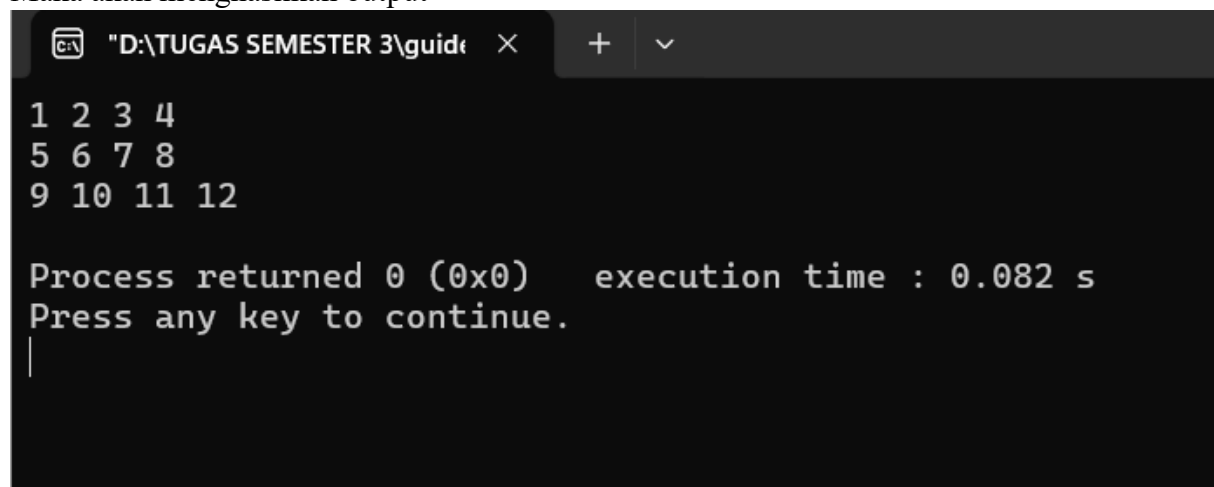


```

45     #include <iostream>
46     using namespace std;
47
48     int main() {
49         int nilai[3][4] = {
50             {1, 2, 3, 4},
51             {5, 6, 7, 8},
52             {9, 10, 11, 12}
53         };
54
55         for (int i = 0; i < 3; i++) {
56             for (int j = 0; j < 4; j++) {
57                 cout << nilai[i][j] << " ";
58             }
59             cout << endl;
60         }
61
62         return 0;
63     }
64
65

```

Maka akan menghasilkan output



The screenshot shows a terminal window with the following output:

```

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

Process returned 0 (0x0)   execution time : 0.082 s
Press any key to continue.
|

```

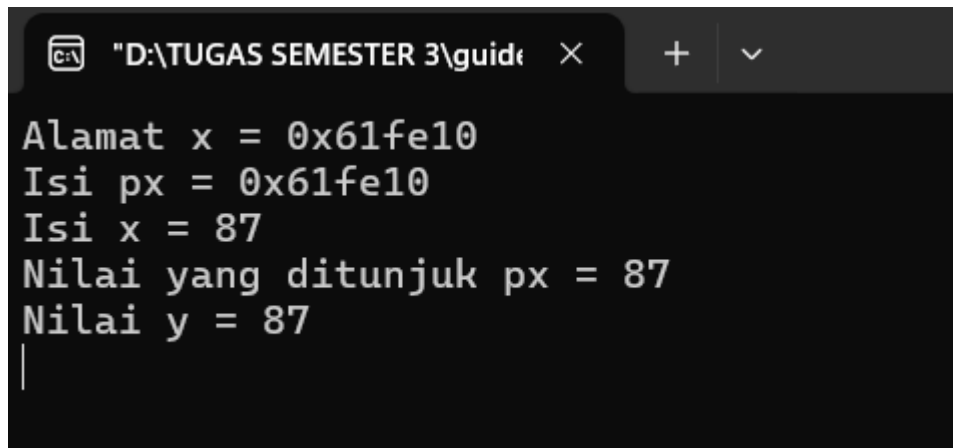
4.

```

75
76 #include <iostream> // Include iostream for cout
77 using namespace std;
78
79 int main() {
80     int x, y;
81     int *px;
82
83     x = 87;
84     px = &x;
85
86     y = *px;
87     // Print the results
88     cout << "Alamat x = " << &x << endl;
89     cout << "Isi px = " << px << endl;
90     cout << "Isi x = " << x << endl;
91     cout << "Nilai yang ditunjuk px = " << *px << endl;
92     cout << "Nilai y = " << y << endl;
93
94     cin.get();
95     return 0;
96 }
97
aa

```

Maka akan menghasilkan output



```

D:\TUGAS SEMESTER 3\guida
Alamat x = 0x61fe10
Isi px = 0x61fe10
Isi x = 87
Nilai yang ditunjuk px = 87
Nilai y = 87
|

```

5.

```

#include <iostream>
using namespace std;

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

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

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

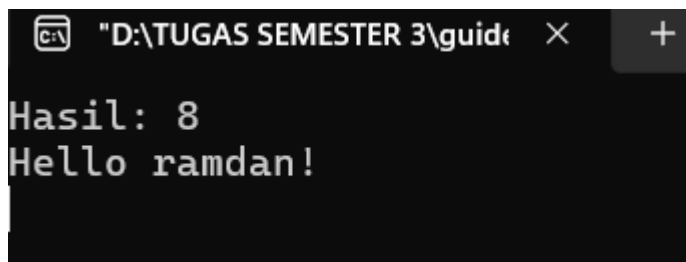
    greet("ramdan");

    cin.get();

    return 0;
}

```

Maka akan menghasilkan output



The screenshot shows a Windows command prompt window with the title bar "D:\TUGAS SEMESTER 3\guide". The output displayed in the window is:

```

Hasil: 8
Hello ramdan!

```

#### 4. Unguided

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

```
main.cpp X main.cpp X *main.cpp X
1  #include <iostream>
2  #include <vector>
3
4  using namespace std;
5
6  int main() {
7      int n;
8
9      cout << "Masukkan jumlah elemen array: ";
10     cin >> n;
11
12     vector<int> dataArray(n);
13
14     cout << "Masukkan " << n << " elemen array: " << endl;
15     for (int i = 0; i < n; i++) {
16         cout << "Elemen " << (i + 1) << ": ";
17         cin >> dataArray[i];
18     }
19
20     cout << "Data Array: ";
21     for (int i = 0; i < n; i++) {
22         cout << dataArray[i] << " ";
23     }
24     cout << endl;
25
26     cout << "Nomor Genap: ";
27     for (int i = 0; i < n; i++) {
28         if (dataArray[i] % 2 == 0) {
29             cout << dataArray[i] << ", ";
30         }
31     }
32     cout << endl;
33
34     cout << "Nomor Ganjil: ";
35     for (int i = 0; i < n; i++) {
36         if (dataArray[i] % 2 != 0) {
37             cout << dataArray[i] << ", ";
38         }
39     }
40     cout << endl;
41
42     return 0;
43 }
```

Maka akan menghasilkan outputnya

```
"D:\TUGAS SEMESTER 3\UNGI" × + v
Masukkan jumlah elemen array: 10
Masukkan 10 elemen array:
Elemen 1: 1
Elemen 2: 2
Elemen 3: 3
Elemen 4: 4
Elemen 5: 5
Elemen 6: 6
Elemen 7: 7
Elemen 8: 8
Elemen 9: 9
Elemen 10: 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,

Process returned 0 (0x0)   execution time : 27.449 s
Press any key to continue.
|
```

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

```
*main.cpp X
1  #include <iostream>
2
3  using namespace std;
4
5  int main() {
6      int x, y, z;
7
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     int array3D[x][y][z];
16
17     cout << "Masukkan elemen-elemen array: " << endl;
18     for (int i = 0; i < x; i++) {
19         for (int j = 0; j < y; j++) {
20             for (int k = 0; k < z; k++) {
21                 cout << "Array[" << i << "][" << j << "][" << k << "]: ";
22                 cin >> array3D[i][j][k];
23             }
24         }
25     }
26
27     cout << "\nArray 3D yang dimasukkan adalah: " << endl;
28     for (int i = 0; i < x; i++) {
29         for (int j = 0; j < y; j++) {
30             for (int k = 0; k < z; k++) {
31                 cout << "Array[" << i << "][" << j << "][" << k << "] = " << array3D[i][j][k] << endl;
32             }
33         }
34     }
35
36     return 0;
37 }
38
```

Maka akan menghasilkan outputnya

Masukkan ukuran dimensi pertama (x): 2

Masukkan ukuran dimensi kedua (y): 2

Masukkan ukuran dimensi ketiga (z): 2

Masukkan elemen-elemen array:

Array[0][0][0]: 1

Array[0][0][1]: 2

Array[0][1][0]: 3

Array[0][1][1]: 4

Array[1][0][0]: 5

Array[1][0][1]: 6

Array[1][1][0]: 7

Array[1][1][1]: 8

Array 3D yang dimasukkan adalah:

Array[0][0][0] = 1

Array[0][0][1] = 2

Array[0][1][0] = 3

Array[0][1][1] = 4

Array[1][0][0] = 5

Array[1][0][1] = 6

Array[1][1][0] = 7

Array[1][1][1] = 8

Process returned 0 (0x0) execution time : 6.425 s

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

```
main.cpp X *main.cpp X
1  #include <iostream>
2  using namespace std;
3
4  int cariMaksimum(int arr[], int n) {
5      int maksimum = arr[0];
6      for (int i = 1; i < n; i++) {
7          if (arr[i] > maksimum) {
8              maksimum = arr[i];
9          }
10     }
11     return maksimum;
12 }
13 int cariMinimum(int arr[], int n) {
14     int minimum = arr[0];
15     for (int i = 1; i < n; i++) {
16         if (arr[i] < minimum) {
17             minimum = arr[i];
18         }
19     }
20     return minimum;
21 }
22 double cariRataRata(int arr[], int n) {
23     int total = 0;
24     for (int i = 0; i < n; i++) {
25         total += arr[i];
26     }
27     return static_cast<double>(total) / n;
28 }
29
30 int main() {
31     int n, pilihan;
32
33     cout << "Masukkan jumlah elemen array: ";
34     cin >> n;
35
36     int arr[n];
37
38     cout << "Masukkan " << n << " elemen array: " << endl;
```



```

39     cout << "Masukkan " << n << " elemen array: " << endl;
40     for (int i = 0; i < n; i++) {
41         cout << "Elemen " << i + 1 << ": ";
42         cin >> arr[i];
43     }
44
45     do {
46         cout << "\nMenu Pilihan:" << endl;
47         cout << "1. Cari Nilai Maksimum" << endl;
48         cout << "2. Cari Nilai Minimum" << endl;
49         cout << "3. Cari Nilai Rata-Rata" << endl;
50         cout << "4. Keluar" << endl;
51         cout << "Masukkan pilihan (1-4): ";
52         cin >> pilihan;
53
54         switch (pilihan) {
55             case 1:
56                 cout << "Nilai Maksimum: " << cariMaksimum(arr, n) << endl;
57                 break;
58             case 2:
59                 cout << "Nilai Minimum: " << cariMinimum(arr, n) << endl;
60                 break;
61             case 3:
62                 cout << "Nilai Rata-Rata: " << cariRataRata(arr, n) << endl;
63                 break;
64             case 4:
65                 cout << "Keluar dari program." << endl;
66                 break;
67             default:
68                 cout << "Pilihan tidak valid, silakan coba lagi." << endl;
69         }
70
71     } while (pilihan != 4);
72     return 0;
73 }
74

```

Maka akan menghasilkan outputnya

```
"D:\TUGAS SEMESTER 3\UNGI" X + v
Elemen 4: 15
Elemen 5: 3

Menu Pilihan:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Masukkan pilihan (1-4): 1
Nilai Maksimum: 15

Menu Pilihan:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Masukkan pilihan (1-4): 2
Nilai Minimum: 3

Menu Pilihan:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Masukkan pilihan (1-4): 3
Nilai Rata-Rata: 8

Menu Pilihan:
1. Cari Nilai Maksimum
2. Cari Nilai Minimum
3. Cari Nilai Rata-Rata
4. Keluar
Masukkan pilihan (1-4): 4
Keluar dari program.

Process returned 0 (0x0)   execution time : 76.714 s
Press any key to continue.
|
```

## 5. Kesimpulan

Penggunaan pointer dan alamat memori dalam bahasa pemrograman C++ sangat penting untuk memahami cara kerja data di dalam memori. Pointer memungkinkan programmer untuk mengakses dan memanipulasi data secara langsung, memberikan fleksibilitas dalam pengelolaan memori. Selain itu, implementasi fungsi dan prosedur dalam program membantu dalam modularisasi kode, membuatnya lebih mudah dibaca, dipelihara, dan digunakan kembali. Dengan menguasai konsep ini, programmer dapat membuat program yang efisien dan terstruktur dengan baik.