

**LAPORAN PRAKTIKUM**  
**Struktur Data dan Algoritma**

**MODUL III**  
**POINTER**



**Disusun oleh:**  
**Bintang Rizqi Pasha**  
**21102056**  
**S1 IF-09-B**

**PROGRAM STUDI S1**  
**INFORMATIKA FAKULTAS**  
**INFORMATIKA**  
**INSTITUT TEKNOLOGI TELKOM**  
**PURWOKERTO PURWOKERTO**

**2021**

## DASAR TEORI

Pointer adalah sebuah variabel atau object yang menunjuk ke variabel atau objek lainnya. Sebelumnya pernah dijelaskan mengenai variabel, menyatakan bahwa. “variabel merupakan sebuah representasi dari alamat memori pada komputer”. Dan pointer hanyalah variabel yang menyimpan alamat memori, memori tersebut dapat berasal dari variabel, objek dan lain-lain. dengan pointer kita dimungkinkan untuk menunjuk suatu memori, mendapatkan isi dari memori dan mengubah isi dari memori yang ditunjuk. ada dua hal yang perlu anda ketahui. dalam pointer terdapat dua macam operator yang akan kita gunakan, yaitu address-of (&) dan dereference operator (\*).

### Pointer Sebagai Alamat Dari Variabel

Misalkan kamu memiliki variabel x dan terletak di memori 0x000001. Jika kamu ingin memasukkan nilai 100 ke dalam variabel x, maka processor harus membawa nilai 100 tersebut kedalam variabel x yang terletak di alamat memori 0x000001. Hal yang perlu kamu ketahui adalah, setiap variabel ternyata memiliki ukuran byte yang berbeda-beda dalam memori. Sebagai contoh suatu variabel bertipe int memiliki ukuran 4 byte dalam memori. Maka variabel tersebut akan menempati 4 kapling lokasi dalam memori, misalkan 0x000001, 0x000002, 0x000003, dan 0x000004. Jika terdapat dua buah variabel bertipe int yang bersebelahan, maka alamat variabel pertama terletak di 0x000001 dan variabel kedua terletak di alamat 0x000005.

### Macam-macam Operator pada Pointer

Address-of Operator (&), adalah operator yang memungkinkan kita untuk mendapatkan/melihat alamat memori yang dimiliki oleh variabel tersebut. Cara menggunakannya adalah dengan meletakkan tanda & di depan identitas saat pemanggilan variabel. Hal itu akan membuat compiler memberikan alamat memori bukan isi/nilai dari memori tersebut.

Dereference Operator (\*), adalah operator yang memungkinkan mendapatkan isi/nilai dari sebuah memori berdasarkan alamat memori.

### Bentuk penulisan pointer

```
tipeData *identitas;
```

```
//atau  
tipeData *identitas = &var;
```

### **Pointer Sebagai Parameter Suatu Fungsi**

Seperti halnya dengan array, pointer dapat digunakan sebagai parameter suatu fungsi. Karena sifat pointer yang hanya sebagai penunjuk, maka setiap perubahan yang terjadi pada parameter, sebenarnya terjadi pada variabel yang ditunjuk bukan pada variabel pointer.

### **Cara Mengakses Pointer**

Variabel pointer adalah variabel yang memiliki alamat memori sebagai nilai dari variabel pointer tersebut. Dan pada pointer kita dimungkinkan untuk mengakses nilai dari pointer itu sendiri dan mengakses nilai dari alamat memori yang dimiliki(ditunjuk) oleh pointer.

Pointer merupakan variabel, untuk mengakses pointer tidak jauh beda dengan cara mengakses variabel. Untuk mengakses nilai dari pointer kita hanya cukup memanggil identitas dari pointer tersebut.

```
pInt
```

pemanggilan itu akan menghasilkan nilai dari pointer yang berupa alamat memori dari variabel yang ditunjuk oleh pointer tersebut.

Karena pointer hanya dapat memiliki nilai berupa alamat memori, untuk mengubah nilai dari pointer atau mengubah tujuan dari pointer kita membutuhkan operator address-of (&) pada operand sumber.

```
pInt = &myVar
```

operand sumber akan menghasilkan alamat memori dari myVar, dan hal itu merupakan nilai yang dibutuhkan oleh variabel pointer.

### **Ukuran pointer**

Setiap kita mendirikan pointer, pointer itu akan membutuhkan memori. Dan besar memori itu sama pada setiap tipe data yang digunakan. Besar memori dari pointer tergantung pada mesin kompiler. jika compiler merupakan 32bit maka pointer akan memakan memori sebanyak 4 bytes, Jika menggunakan 64bit maka pointer memakan memori sebanyak 8 bytes.

Sumber :

“Pengertian Pointer.” *Belajar C++*, <https://www.belajarcpp.com/tutorial/cpp/pointer/>. Diakses pada 16 April 2022.

“Pointer Pada Pemrograman C++ – SinauArduino.” *SinauArduino*, 27 April 2016, <https://www.sinauarduino.com/artikel/pointer-pada-pemrograman-cpp/>. Diakses pada 16 April 2022.

## LATIHAN KELAS - GUIDED

### 1. Guided 1 Source code

```
#include <iostream>

//Nama : Bintang Rizqi Pasha
//Kelas : IF 09 B
//NIM : 21102056

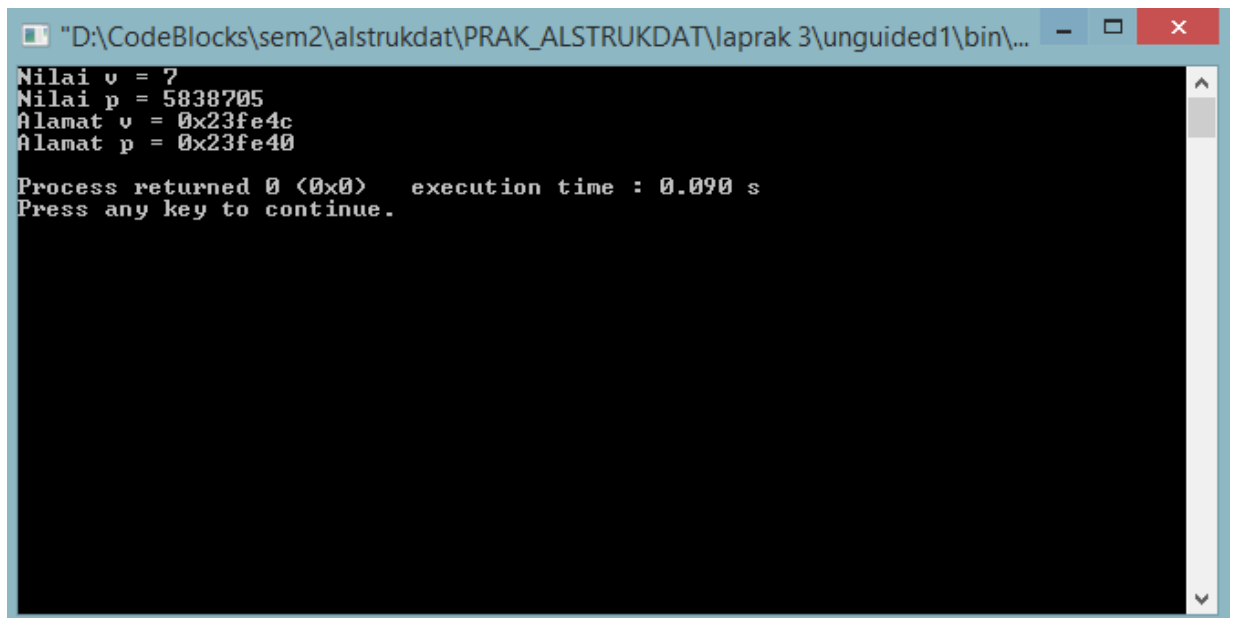
using namespace std;

int main()
{
    int v=7;
    int *p;

    cout << "Nilai v = " << v << endl;
    cout << "Nilai p = " << *p << endl; //nilai akan random
    cout << "Alamat v = " << &v << endl;
    cout << "Alamat p = " << &p << endl;

    return 0;
}
```

### Screenshot program



The screenshot shows a Windows command prompt window titled "D:\CodeBlocks\sem2\alstrukdat\PRAK\_ALSTRUKDAT\laprak 3\unguided1\bin\...". The output of the program is displayed as follows:

```
Nilai v = 7
Nilai p = 5838705
Alamat v = 0x23fe4c
Alamat p = 0x23fe40

Process returned 0 (0x0)   execution time : 0.090 s
Press any key to continue.
```

### Deskripsi program

Program dibuat untuk mengetahui nilai alamat dari suatu variabel dan variabel yang sudah berpointer, untuk variabel memakai int v=7; dan untuk variabel berpointer int \*p; dan nilai \*p tidak di inisialisasi. Dan dicetak Nilai v dan p tetapi bedanya untuk v (variabel biasa) hanya dipanggil "v" tetapi untuk nilai p dipanggil \*p (dipanggil dengan pointer asterik )

dan untuk memanggil alamat dari variabel menggunakan pointer "&" yakni ampersand. Kemudian ketika di run akan tampil nilai v dan p, serta alamat dari v dan p.

## 2. Guided 2

### Source code

```
#include <iostream>

//Nama : Bintang Rizqi Pasha
//Kelas : IF 09 B
//NIM : 21102056

using namespace std;

int main()
{
    int value1 =5, value2 =15;
    int *mypointer;

    cout << "Nilai value 1 = " << value1 <<endl; //cetak nilai awal 5
    cout << "Nilai value 2 = " << value2 <<endl; //cetak nilai awal 15
    cout << endl;

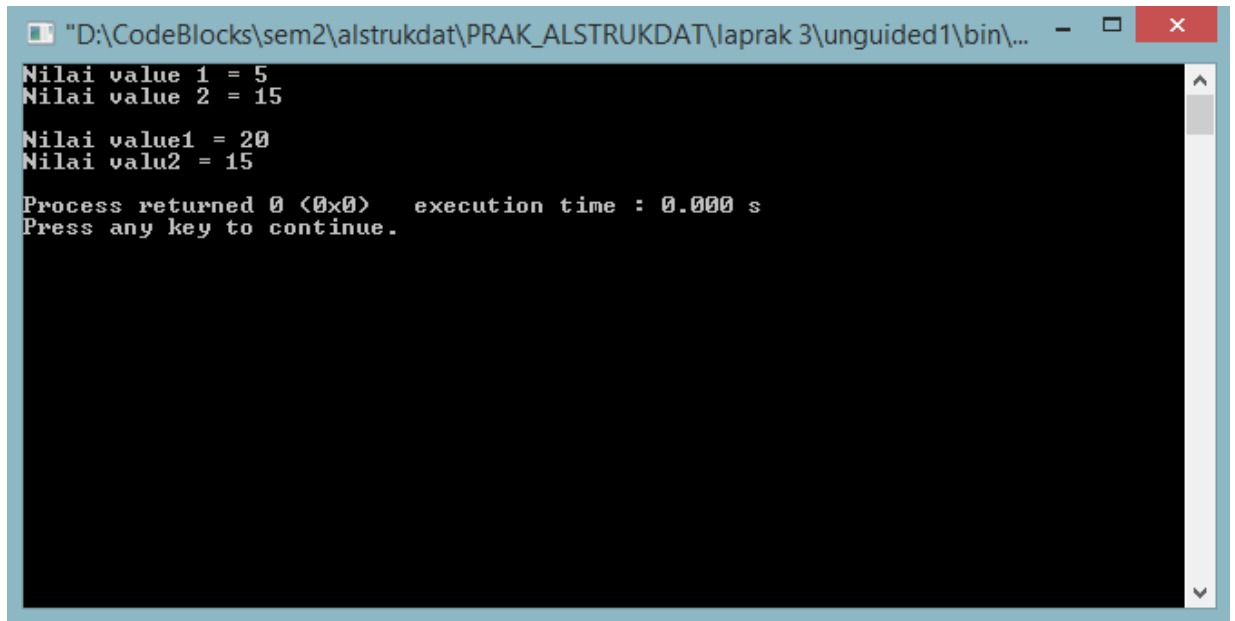
    //inisialisasi ulangan dengan pointer
    mypointer = &value1; //alamat pointer di value1
    *mypointer = 10; // ganti nilai value1 jadi 10 karena alamat mypointer
sama

    mypointer == &value2; //alamat pointer di value2
    *mypointer = 20; //ganti nilai value2 jadi 20 karena alamat mypointer
sama

    cout << "Nilai value1 = "<< value1<<endl; //print value1 nilainya 10
    cout << "Nilai value2 = "<< value2<<endl; //print value2 nilainya 20

    return 0;
}
```

### Screenshot program

A screenshot of a Windows command prompt window. The title bar shows the file path "D:\CodeBlocks\sem2\alstrukdat\PRAK\_ALSTRUKDAT\laprak 3\unguided1\bin\...". The command prompt has a black background with white text. The output of the program is as follows:

```
Nilai value 1 = 5
Nilai value 2 = 15

Nilai value1 = 20
Nilai valu2 = 15

Process returned 0 (0x0)   execution time : 0.000 s
Press any key to continue.
```

### Deskripsi program

Program mendeklarasikan value1 yang nilainya 15 dan value2 yang nilainya 15 dengan tipe data integer, kemudian mendeklarasikan variabel int \*mypointer; gunanya untuk menginisialisasi ulang dengan pointer, cara kerjanya dengan mengakses alamat dari value1 dan menginisiasikannya \*mypointer menjadi bilangan apa aja yang diinginkan maka ketika value1 ditampilkan ulang maka nilainya dengan otomatis akan terganti dengan apa yang sudah diinisialisasi oleh \*mypointer. Begitu juga dengan value2, alamatnya akan diakses oleh \*mypointer menggunakan ampersand seperti &value2 (sama halnya dengan &value1 yang sebelumnya) maka ketika di inisialisasi \*mypointer menjadi suatu bilangan yang nilainya 20, sehingga nilai dari value2 akan berubah, yang tadinya value2 bernilai 15 berubah menjadi 20.

### 3. Guided 3 Source code

```
#include <iostream>

//Nama : Bintang Rizqi Pasha
//Kelas : IF 09 B
//NIM : 21102056

using namespace std;

int main()
{
    int data[] = {1,2,3,4,5};
    int *pData = data; //pData jadi array karena pointer trus nilai data di
    inisialisasi ke *pData

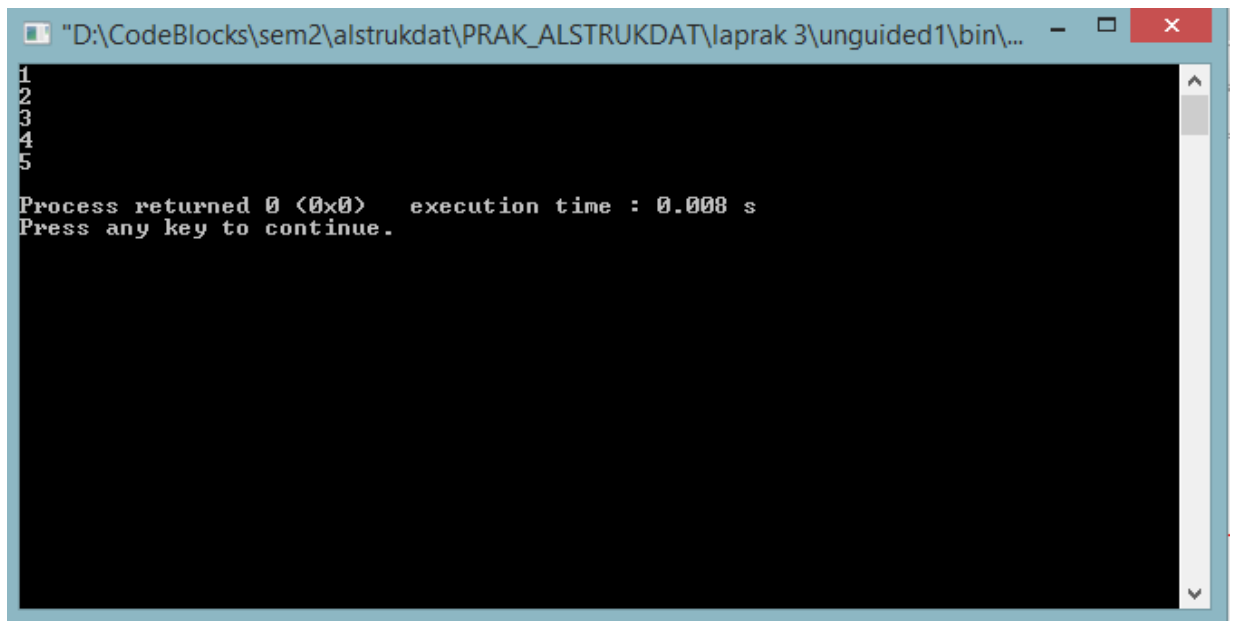
    //jadi pData bakal bisa punya array yg sama kek data
    cout << pData[0] <<endl; //nyoba akses array 0
    cout << pData[1] <<endl;
    cout << pData[2] <<endl;
```



```
cout << pData[3] <<endl;
cout << pData[4] <<endl;

return 0;
}
```

### Screenshot program



```
1
2
3
4
5
Process returned 0 (0x0) execution time : 0.008 s
Press any key to continue.
```

### Deskripsi program

Program yakni mendeskripsikan cara kerja dari array dan diakses dengan memakai pointer dari variabel yang lain. Maka cara kerjanya yakni mempunyai sebuah array misal `data[] = {1,2,3,4,5}`; kemudian dideklarasikan ulang dengan `int *pData` sebagai sebuah pointer untuk mengakses `data[]` maka diperlukan sintaks sebagai berikut.

```
int *pData = data;
```

asterik dari `pData` menunjuk pada nilai `pData` itu dan mengambilnya dari variabel `data` yang sudah memiliki nilai berupa array 1 dimensi, sehingga nilai dari `pData` akan mengikuti menjadi array sedemikian sehingga `pData` dapat ditampilkan layaknya sebuah array biasa seperti.

```
cout << pData[0] <<endl;
```

nilai dari variabel `pData` dapat dipanggil layaknya sebuah array biasa.

## LATIHAN KELAS - UNGUIDED

### 1. Unguided 1 Source code

```
#include <iostream>

//Nama : Bintang Rizqi Pasha
//Kelas : IF 09 B
//NIM : 21102056

using namespace std;

int main()
{
    //tampilkan semua nilai data/elemen dari array melalui pointer yang menunjuknya
    //untuk unguided 1
    int data[] = {1,2,3,4,5,6,7,8,9,10}; //data
    //inisialisasi nilai array data ke pData
    int *pData = data;

    //print nilai pData
    cout << endl << "Unguided 1" << endl << endl;
    for (int i=0; i<10; i++){
        cout << "Nilai data elemen ke-" << i << " : ";
        cout << pData[i] << endl;
    }
    cout << endl;

    //selesai
    return 0;
}
```

### Screenshot program

## Deskripsi program

Program akan menampilkan semua nilai data/elemen dari array melalui pointer yang menunjuknya dengan cara menginisialisasi variabel pointer seperti `int *pData = nilai variabel yang sudah dibuat sebelumnya (data);`

Maka nilai data/elemen dari array akan diproses melalui pointer dan menjadi nilai dari suatu pointer tersebut yang kemudian dapat di ditampilkan dengan hanya mengakses elemen pointernya saja. Dikarenakan nilai `*pData` berupa array maka dapat ditampilkan dengan melakukan perulangan for loop biasa dengan nilai awal nol dan nilai batas seperti panjang array dan increment dan hanya dengan memanggil `pData[i]` maka semua nilai dalam `pData` akan terpanggil sesuai urutan dalam perulangan for loop.

## 2. Unguided 2 Source code

```
#include <iostream>

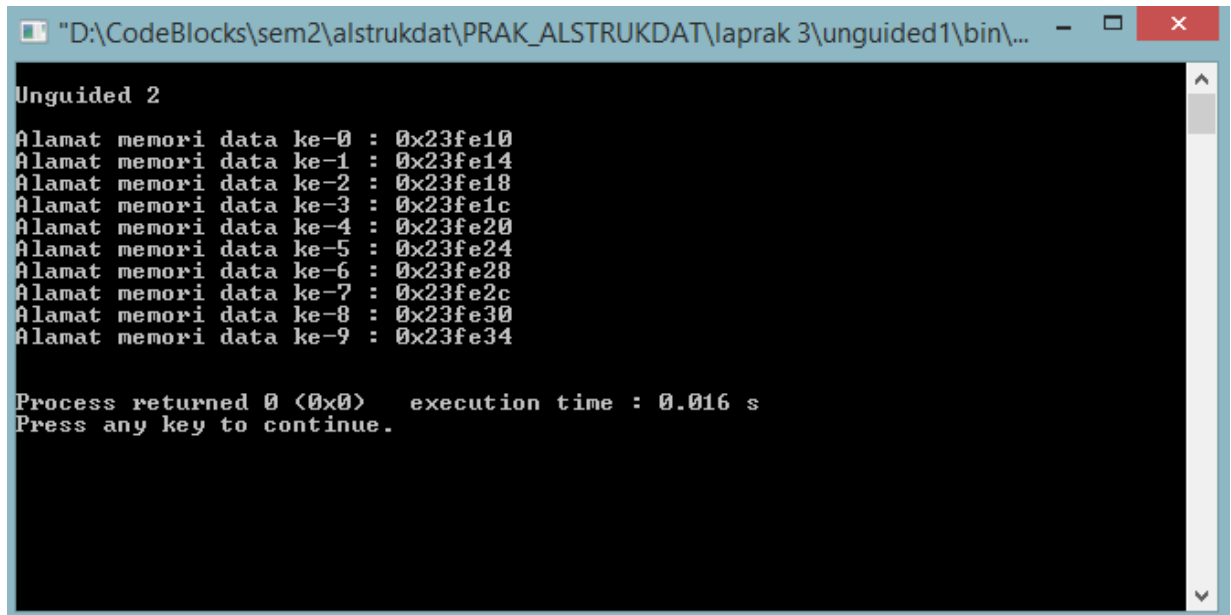
//Nama : Bintang Rizqi Pasha
//Kelas : IF 09 B
//NIM : 21102056

using namespace std;

int main()
{
    //tampilkan semua alamat data/elemen dari array melalui pointer yang menunjuknya
    //untuk unguided 1
    int data[] = {1,2,3,4,5,6,7,8,9,10}; //data
    //inisialisasi nilai array data ke pData
    int *pData = data;

    //print alamat memori data dalam pData
    cout << endl << "Unguided 2" << endl << endl;
    for (int i=0; i<10; i++){
        cout << "Alamat memori data ke-" << i << " : " ;
        cout << &pData[i] << endl;
    }
    cout << endl;
    //selesai
    return 0;
}
```

## Screenshot program



```
"D:\CodeBlocks\sem2\alstrukdat\PRAK_ALSTRUKDAT\laprak 3\unguided1\bin\... - □ ×

Unguided 2

Alamat memori data ke-0 : 0x23fe10
Alamat memori data ke-1 : 0x23fe14
Alamat memori data ke-2 : 0x23fe18
Alamat memori data ke-3 : 0x23fe1c
Alamat memori data ke-4 : 0x23fe20
Alamat memori data ke-5 : 0x23fe24
Alamat memori data ke-6 : 0x23fe28
Alamat memori data ke-7 : 0x23fe2c
Alamat memori data ke-8 : 0x23fe30
Alamat memori data ke-9 : 0x23fe34

Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

### Deskripsi program

Program akan menampilkan nilai semua alamat data/element dari array melalui pointer yang menunjuk suatu variabel dengan cara menginisialisasi variabel pointer seperti `int *pData = nilai variabel yang sudah dibuat sebelumnya (data)`; Setelah itu melakukan percetakan alamat dari `pData` menggunakan pointer ampersand yang disandingin dengan variabel `pData[i]` maka alamat dari elemen `pData` akan ditampilkan. Untuk menampilkan alamat dari `pData` menggunakan perulangan `for` loop dengan nilai awal 0 dan batas akhir yakni panjang array dan memakai increment dengan didalamnya memakai fungsi `cout` dan `endl` seperti.

```
cout << "Alamat memori data ke-"<< i<< " : " ;
cout << &pData[i]<< endl;
```

Maka alamat akan ditampilkan secara berurutan dan keseluruhan.

### 3. Unguided 3 Source code

```
#include <iostream>

//Nama : Bintang Rizqi Pasha
//Kelas : IF 09 B
//NIM : 21102056

using namespace std;

int main()
{
    //tampilkan semua nilai data/element dari array melalui pointer yang
    menunjuknya
    //untuk unguided 1
    int data[] = {1,2,3,4,5,6,7,8,9,10}; //data
    //inisialisasi nilai array data ke pData
    int *pData = data;
    int *mypointer; //untuk re-inisialisasi nilai array

    //untuk re-inisialisasi nilai pData pakai *mypointer
    cout << endl<< "Unguided 3"<< endl<< endl;
    for(int i=0; i<10; i++){
```

```

//menetapkan alamat re-inisialisasi
mypointer = &pData[i];
if (i==0){
    *mypointer = 29;
}if (i==2){
    *mypointer = 17;
}else if (i==3){
    *mypointer = 12;
}else if (i==6){
    *mypointer = 13;
}else if (i==8){
    *mypointer = 67;
}

//print nilai pData
cout << "Nilai data elemen ke-"<< i<< " : ";
cout << pData[i]<<endl;
}
//selesai
return 0;
}

```

### Screenshot program

```

D:\CodeBlocks\sem2\alstrukdat\PRAK_ALSTRUKDAT\laprak 3\unguided1\bin\...
Unguided 3
Nilai data elemen ke-0 : 29
Nilai data elemen ke-1 : 2
Nilai data elemen ke-2 : 17
Nilai data elemen ke-3 : 12
Nilai data elemen ke-4 : 5
Nilai data elemen ke-5 : 6
Nilai data elemen ke-6 : 13
Nilai data elemen ke-7 : 8
Nilai data elemen ke-8 : 67
Nilai data elemen ke-9 : 10
Process returned 0 (0x0) execution time : 0.008 s
Press any key to continue.

```

### Deskripsi program

Program untuk menginisialisasi ulang dari variabel yang ada contohnya variabel array data[] dengan nilai {1,2,3,4,5,6,7,8,9,10}; dan diakses dengan variabel pointer \*pData dengan diinisialisasi data; maka \*pData akan mempunyai nilai yang sama dan alamat yang sama juga dengan array data[] dan di deklarasikan juga \*mypointer dengan tipe data integer untuk re-inisialisasi nilai array. Untuk re-inisialisasi nilai pData pakai \*mypointer menggunakan perulangan for loop dengan mengakses alamat pData nya terlebih dahulu mypointer = &pData[i]; maka nilai dari mypointer yang nantinya dapat diganti berdasarkan urutan dari perulangan tersebut. Dipakai percabangan if, else if untuk me re-inisialisasi nilai pData dengan pointer

mypointer dengan

`*mypointer = 29;`

contohnya seperti itu, tetapi dipakaikan percabangan indeks mana yang ingin dire-inisialisasi dengan nilai 29 maka dari itu digunakanlah percabangan ini seperti.

```
if (i==0){
    *mypointer = 29;
}if (i==2){
    *mypointer = 17;
}else if (i==3){
    *mypointer = 12;
}else if (i==6){
    *mypointer = 13;
}else if (i==8){
    *mypointer = 67;
}
```

Yang artinya nanti untuk nilai  $i = 0$  maka nilai `pData` akan menjadi 29, untuk nilai  $i=2$  maka nilai `pData` akan menjadi 17, untuk nilai  $i =3$  maka nilai `pData` akan menjadi 12, untuk nilai  $i=6$  maka nilai `pData` akan menjadi 13, untuk nilai  $i=8$  maka nilai `pData` akan menjadi 67. Dengan mencetak nilai `pData` menggunakan perulangan `for loop` yang tadinya 1,2,3,4,5,6,7,8,9,10. Akan berubah setelah dilakukan program seperti yang tadi mengikuti nilai yang sudah di re-inisialisasi ulang sebelumnya. Maka didapatkan nilai `pData` = 29,2,17,12,5,6,13,8,67,10.