

I. COVER

LAPORAN PRAKTIKUM
Modul 2
Pemograman lanjut C++ dan Pemahaman Array, Pointer Fungsi, Prosedur



Disusun Oleh:
Marvel Sanjaya Setiawan
(2311104053)

Kelas:
SISE 07 02

Dosen :
Wahyu Andi Saputra
PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
2024

II. TUJUAN

Melatih kemampuan dalam memprogram C++ dengan menekankan penggunaan pointer, pengelolaan memori, serta pembuatan fungsi dan prosedur.

III. Landasan Teori

Array : Menyimpan kumpulan data sejenis dalam satu variabel.

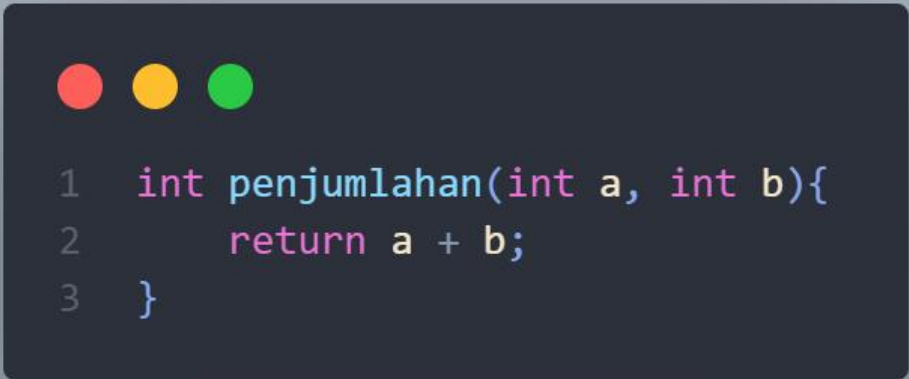
Pointer : Variabel yang menunjuk ke alamat memori suatu data.

Fungsi : Blok kode yang dapat dipanggil berulang kali.

Prosedur : Digunakan untuk membagi program menjadi bagian-bagian yang lebih kecil.

IV. GUIDED

1. Fungsi untuk penjumlahan dua bilangan :



```
1  int penjumlahan(int a, int b){  
2      return a + b;  
3  }
```

Fungsi penjumlahan(int a, int b):

- int: Fungsi ini akan menghasilkan angka bulat.
- penjumlahan: Nama fungsi.
- (int a, int b): Menerima dua angka bulat sebagai input (disebut a dan b).
- return a + b;: Menjumlahkan a dan b, lalu mengembalikan hasilnya.

Fungsi ini dirancang khusus untuk menambahkan dua bilangan bulat dan mengembalikan hasil penjumlahannya.

2. Fungsi untuk memanggil sapaan :

```
1 void greet(string name){  
2     cout << "Hello, " << name << "!" << endl;  
3 }
```

Fungsi greet ini berfungsi untuk menampilkan pesan sapaan yang dipersonalisasi. Kamu bisa memasukkan nama seseorang sebagai input, dan fungsi ini akan mencetak kalimat "Hello, [nama]!".

3 .Menggunakan fungsi penjumlahan :

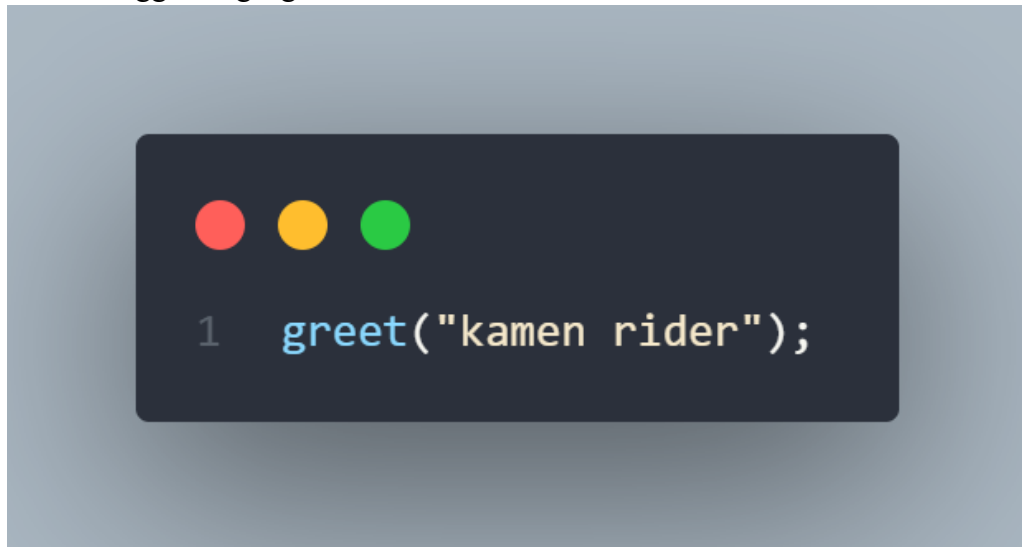
```
1 int main(){  
2     int hasil = penjumlahan(5, 3);  
3  
4     cout << "Hasil: " << hasil << endl;  
5 }
```

Panggil Fungsi: `penjumlahan(5, 3)` memanggil fungsi penjumlahan dengan memberikan nilai 5 dan 3 sebagai input.

Hitung dan Simpan: Fungsi penjumlahan akan menghitung jumlah dari 5 dan 3 (yaitu 8), lalu hasil penjumlahan ini disimpan ke dalam variabel `hasil`.

Tampilkan Hasil: `cout << "Hasil: " << hasil << endl;` mencetak kalimat "Hasil: 8" ke layar, di mana angka 8 adalah nilai yang tersimpan di variabel `hasil`.

4. Memanggil fungsi greet :



greet: Ini adalah perintah untuk menjalankan sebuah fungsi yang sudah dibuat sebelumnya. Fungsi ini khusus untuk memberikan salam.

"kamen rider": Ini adalah informasi tambahan yang diberikan kepada fungsi greet. Informasi ini akan digunakan oleh greet untuk menentukan siapa yang akan disapa.

5. Menampilkan elemen Array 1 dimensi :




Membuat Daftar Angka: Membuat sebuah daftar yang bisa menampung 5 angka bulat, dan diberi nama daftar berupa "nilai". Lalu, mengisi daftar itu dengan angka 1, 2, 3, 4, dan 5.

Mengulang: Membuat sebuah perulangan yang akan berjalan sebanyak 5 kali.

Menampilkan Angka: Pada setiap perulangan, mengambil satu angka dari daftar "nilai" dan menampilkannya di layar bersama dengan nomor urutnya.

6. Array 2 dimensi :




```
1  int main(){
2      int nilai[3][4]={
3          {1,2,3,4},
4          {5,6,7,8},
5          {9,10,11,12}
6  };
7      return 0;
8  };
```

Kode ini membuat sebuah array dua dimensi yang berisi 12 angka yang disusun dalam bentuk tabel.

`int nilai[3][4]`: Ini seperti membuat sebuah kotak besar yang dibagi menjadi 3 baris dan 4 kolom. Setiap kotak kecil di dalam kotak besar ini akan diisi dengan angka bulat.

`{{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}}`: Ini adalah angka-angka yang akan mengisi kotak-kotak kecil tadi. Angka-angka ini disusun dalam bentuk baris dan kolom.

7. Loop untuk menampilkan array 2 dimensi :




```
1  int main(){
2      for(int i=0; i<3; i++){
3          for(int j=0; j<4; j++){
4              cout<<nilai[i][j]<<" "<<endl;
5          }
6      }
7      cout<<endl;
8
9      return 0;
10 }
```

Kode ini digunakan untuk menampilkan semua elemen dalam sebuah array dua dimensi secara sistematis, baris per baris dan kolom per kolom.

Cara Kerja:

1. Perulangan Baris:
 - Kode akan mengulang sebanyak 3 kali untuk setiap baris pada tabel.
 - Pada setiap perulangan baris, nilai *i* akan bertambah 1 untuk berpindah ke baris berikutnya.
2. Perulangan Kolom:
 - Di dalam perulangan baris, terdapat perulangan kolom yang akan mengulang sebanyak 4 kali untuk setiap kolom pada baris tersebut.
 - Pada setiap perulangan kolom, nilai *j* akan bertambah 1 untuk berpindah ke kolom berikutnya.
3. Menampilkan Angka:
 - Pada setiap kombinasi *i* dan *j*, nilai pada posisi `nilai2d[i][j]` (yaitu angka pada baris ke-*i* dan kolom ke-*j*) akan ditampilkan di layar, diikuti oleh spasi.
 - Setelah semua kolom pada satu baris selesai ditampilkan, kursor akan pindah ke baris baru.

8. Pointer :



```
1  int main(){
2      int x,y;
3      int *px;
4      x=87;
5      px=&x;
6      y=*px;
7
8      return 0;
9  }
```

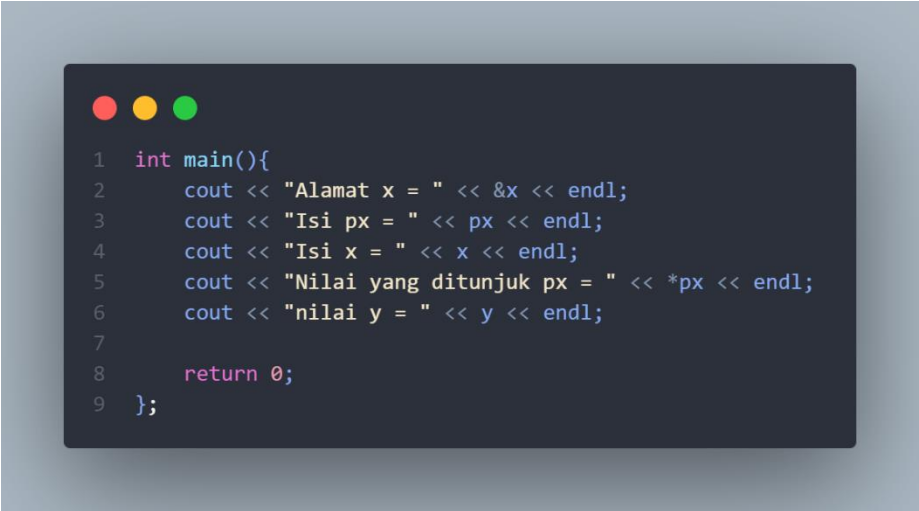
Kode ini menunjukkan cara kita bisa menggunakan alamat untuk mengakses dan menyalin nilai dari satu variabel ke variabel lainnya.

Penjelasan Lebih Detail:

Pointer: Bayangkan pointer sebagai sebuah kertas catatan yang berisi alamat sebuah rumah.

Dereferensi: Mengambil nilai yang ada di alamat yang ditunjuk oleh pointer itu seperti kita pergi ke rumah yang alamatnya tercatat di kertas dan melihat apa yang ada di dalam rumah tersebut.

9. Menampilkan nilai pointer dan variabel :




```
1  int main(){
2      cout << "Alamat x = " << &x << endl;
3      cout << "Isi px = " << px << endl;
4      cout << "Isi x = " << x << endl;
5      cout << "Nilai yang ditunjuk px = " << *px << endl;
6      cout << "nilai y = " << y << endl;
7
8      return 0;
9  };
```

Kode ini seperti kita sedang memeriksa identitas dan isi dari tiga kotak (variabel). Kita melihat di mana kotak itu berada (alamat), apa yang tertulis di label kotak (nilai pointer), dan apa yang ada di dalam kotak (nilai variabel).

Penjelasan Lebih Detail:

- **&x:** Ini seperti kita menanyakan alamat rumah dari seseorang bernama x.
- **px:** Ini seperti sebuah catatan yang berisi alamat rumah x.
- ***px:** Ini seperti kita pergi ke rumah yang alamatnya tercatat di px dan melihat apa yang ada di dalamnya.

V. UNGUIDED



```
1  int main() {
2      int arr[10];
3
4      cout << "Masukkan 10 angka: ";
5      for (int i = 0; i < 10; i++) {
6          cin >> arr[i];
7      }
8
9      cout << "Data Array: ";
10     for (int i = 0; i < 10; i++) {
11         cout << arr[i] << " ";
12     }
13     cout << endl;
14
15     cout << "Nomor Genap: ";
16     for (int i = 0; i < 10; i++) {
17         if (arr[i] % 2 == 0) {
18             cout << arr[i] << ", ";
19         }
20     }
21     cout << endl;
22
23     cout << "Nomor Ganjil: ";
24     for (int i = 0; i < 10; i++) {
25         if (arr[i] % 2 != 0) {
26             cout << arr[i] << ", ";
27         }
28     }
29     cout << endl;
30
31     return 0;
32 }
```

```
Masukkan 10 angka: 1 2 3 4 5 6 7 8 9 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 : 11.571 s
Press any key to continue.
```

Kode ini memberikan contoh sederhana tentang bagaimana cara menggunakan komputer untuk mengolah data (dalam hal ini, angka-angka) dan menampilkan hasil yang diinginkan.

Penjelasan Lebih Detail:

- Baris 1-3: Mendeklarasikan variabel arr sebagai array yang dapat menampung 10 bilangan bulat.
- Baris 4-8: Meminta pengguna untuk memasukkan 10 angka dan menyimpannya ke dalam array arr.
- Baris 9-13: Menampilkan semua angka yang telah disimpan dalam array arr.
- Baris 14-19: Menampilkan semua angka genap dalam array arr.
- Baris 21-26: Menampilkan semua angka ganjil dalam array arr.

```

1  int main() {
2      int x, y, z;
3      cout << "Masukkan ukuran dimensi pertama: ";
4      cin >> x;
5      cout << "Masukkan ukuran dimensi kedua: ";
6      cin >> y;
7      cout << "Masukkan ukuran dimensi ketiga: ";
8      cin >> z;
9
10     int ***array = new int**[x];
11     for (int i = 0; i < x; i++) {
12         array[i] = new int*[y];
13         for (int j = 0; j < y; j++) {
14             array[i][j] = new int[z];
15         }
16     }
17
18     cout << "\nMasukkan elemen array:\n";
19     for (int i = 0; i < x; i++) {
20         for (int j = 0; j < y; j++) {
21             for (int k = 0; k < z; k++) {
22                 cout << "array[" << i << "]" << j << "]" << k << "]: ";
23                 cin >> array[i][j][k];
24             }
25         }
26     }
27
28     cout << "\nIsi array:\n";
29     for (int i = 0; i < x; i++) {
30         for (int j = 0; j < y; j++) {
31             for (int k = 0; k < z; k++) {
32                 cout << "array[" << i << "]" << j << "]" << k << "]: " << array[i][j][k] << endl;
33             }
34         }
35     }
36
37     for (int i = 0; i < x; i++) {
38         for (int j = 0; j < y; j++) {
39             delete[] array[i][j];
40         }
41         delete[] array[i];
42     }
43     delete[] array;
44
45     return 0;
46 }

```

```

Masukkan ukuran dimensi pertama: 2
Masukkan ukuran dimensi kedua: 3
Masukkan ukuran dimensi ketiga: 2

Masukkan elemen array:
array[0][0][0]: 1
array[0][0][1]: 2
array[0][1][0]: 3
array[0][1][1]: 4
array[0][2][0]: 5
array[0][2][1]: 6
array[1][0][0]: 7
array[1][0][1]: 8
array[1][1][0]: 9
array[1][1][1]: 10
array[1][2][0]: 11
array[1][2][1]: 12

Isi array:
array[0][0][0]: 1
array[0][0][1]: 2
array[0][1][0]: 3
array[0][1][1]: 4
array[0][2][0]: 5
array[0][2][1]: 6
array[1][0][0]: 7
array[1][0][1]: 8
array[1][1][0]: 9
array[1][1][1]: 10
array[1][2][0]: 11
array[1][2][1]: 12

Process returned 0 (0x0)   execution time : 33.435 s
Press any key to continue.

```

Kode ini memungkinkan kita membuat sebuah struktur data (array tiga dimensi) yang dapat menyimpan data dalam bentuk tiga dimensi. Kita bisa mengatur ukuran kotak ini sesuai dengan kebutuhan kita, dan kemudian mengisi serta melihat isi kotak tersebut.

Array tiga dimensi: Bayangkan sebuah rubik. Setiap kotak kecil pada rubik adalah sebuah elemen dalam array tiga dimensi.

Nested loop: Perulangan bersarang digunakan untuk mengakses setiap elemen dalam array tiga dimensi, sama seperti kita memeriksa setiap kotak kecil pada rubik satu per satu.

Input dan output: Program berinteraksi dengan pengguna untuk mendapatkan input (ukuran kotak dan nilai untuk setiap kotak) dan menampilkan output (isi kotak).



```
1  int main() {
2      int n, max, min, sum = 0;
3      float avg;
4
5      cout << "Masukkan jumlah elemen array: ";
6      cin >> n;
7
8      int arr[n];
9
10     cout << "Masukkan elemen array:\n";
11     for (int i = 0; i < n; i++) {
12         cin >> arr[i];
13     }
14
15     max = arr[0];
16     min = arr[0];
17     for (int i = 1; i < n; i++) {
18         if (arr[i] > max) {
19             max = arr[i];
20         }
21         if (arr[i] < min) {
22             min = arr[i];
23         }
24         sum += arr[i];
25     }
26
27     avg = (float)sum / n;
28
29     cout << "\nNilai maksimum: " << max << endl;
30     cout << "Nilai minimum: " << min << endl;
31     cout << "Nilai rata-rata: " << avg << endl;
32
33     return 0;
34 }
```

```
Masukkan jumlah elemen array: 5
Masukkan elemen array:
10
20
30
40
50

Nilai maksimum: 50
Nilai minimum: 10
Nilai rata-rata: 28

Process returned 0 (0x0)   execution time : 36.716 s
Press any key to continue.
```

Program ini seperti seorang kalkulator pintar yang dapat menghitung berbagai hal terkait dengan sekumpulan angka. Kita hanya perlu memberikan angka-angka tersebut, dan program akan langsung memberikan hasil perhitungan yang kita butuhkan.

Penjelasan Lebih Detail:

- **Array:** Bayangkan array sebagai sebuah kotak yang berisi beberapa laci. Setiap laci berisi satu angka.
- **Nilai maksimum:** Angka terbesar di dalam kotak.
- **Nilai minimum:** Angka terkecil di dalam kotak.
- **Rata-rata:** Hasil dari menjumlahkan semua angka dalam kotak, kemudian membaginya dengan jumlah laci.

VI. KESIMPULAN

Array, pointer, fungsi, dan prosedur adalah konsep kunci dalam C++. **Array** untuk menyimpan data, **pointer** untuk mengakses data langsung, sementara **fungsi** dan **prosedur** untuk membuat kode lebih terstruktur.