MODUL PRAKTIKUM – Fungsi

A. Tujuan

Setelah mempelajari materi kegiatan pembelajaran ini mahasiswa akan dapat :

- 1) Memahami konsep fungsi dalam pemrograman C++ secara benar.
- 2) Mengenal bentuk fungsi dalam pemrograman C++ secara benar.
- 3) Dapat membuat fungsi sendiri dalam aplikasinya pada pembuatan program secara tepat.
- 4) Dapat mengembangkan bentuk-bentuk fungsi dalam pemrograman secara benar.

B. Materi

Latar Belakang

- Ketika program kita sudah sangat besar dan kompleks maka akan semakin sulit membacanya?
- Oleh karena itu dibutuhkan fungsi yang digunakan untuk membungkus program menjadi bagian-bagian kecil.
- o Tujuannya agar program tidak menumpuk pada fungsi main() saja.

> Fungsi

- Fungsi adalah suatu teknik pemrograman di mana program yang biasanya cukup besar dibagi-bagi menjadi beberapa bagian program yang lebih kecil sehingga akan mudah dipahami dan dapat digunakan kembali, baik untuk program itu sendiri maupun program lain yang memiliki proses yang sama.
- Fungsi yang selalu ada pada program C++ adalah fungsi main() karena fungsi tersebutlah yang pertama kali akan dieksekusi/dijalankan.

Kelebihan Fungsi

- Program lebih pendek
- Mudah dibaca dan dimengerti
- Mudah didokumentasi
- Mengurangi kesalahan dan mudah mencari kesalahan
- Kesalahan yang terjadi bersifat "local"

Tim Dosen | Algoritma & Pemrograman | Universitas Amikom Yogyakarta

Standard Library Function

- Yaitu fungsi-fungsi yang telah disediakan oleh C++ dalam file-file header atau librarynya.
- Misalnya: clrscr()
- Untuk function ini kita harus mendeklarasikan terlebih dahulu library yang akan digunakan, yaitu dengan menggunakan preprosesor.
- o Misalnya: #include <conio.h>

Programmer-Defined Function

- Adalah function yang dibuat oleh programmer sendiri.
- Function ini memiliki nama tertentu yang unik dalam program, letaknya terpisah dari program utama (fungsi main)
- Terdiri dari fungsi void dan non void

Fungsi yang tidak mengembalikan nilai (void)

- Disebut void karena fungsi tersebut tidak mengembalikan suatu nilai keluaran yang didapat dari hasil proses fungsi tersebut.
- Tidak memiliki nilai kembalian fungsi
- o Ciri: tidak adanya keyword return.
 - tidak adanya tipe data di dalam deklarasi fungsi.
 - menggunakan keyword void.

Fungsi yang mengembalikan nilai (nonvoid)

- Disebut non-void karena mengembalikan nilai kembalian yang berasal dari keluaran hasil proses function tersebut.
- Memiliki nilai kembalian
- o Ciri: ada keyword return
 - ada tipe data yang mengawali deklarasi fungsi
 - tidak ada keyword void

Contoh Void (1)

```
#include <iostream>
 2
   using namespace std;
 3
 4
   //membuat fungsi hello()
 5 pvoid hello() {
 6
        cout<<"Hello Selamat Datang!\n";
 7
   1
 8
 9 pint main(){
        //memanggil fungsi hello()
10
11
        hello();
12 \{
```

Contoh Void (2)

```
#include <iostream>
 2
   using namespace std;
 3
   //membuat fungsi hello()
 4
 5 pvoid hello(){
        cout << "Hello Selamat Datang!\n";
 6
 7
   1
 8
 9 pint main(){
10
        //memanggil fungsi hello()
11
        hello();
12
        hello();
13
        hello();
14
        hello();
15
        hello();
16 L}
```

Contoh Void (3)

```
1 #include <iostream>
   using namespace std;
3
4 //membuat fungsi hello() yg memiliki parameter nilai
5 pvoid hello(string nama) {
       cout<<"Hello Selamat Datang "<<nama<<"!\n";
7 1
8
9 pint main(){
10
       //memanggil fungsi hello() dgn memberi nilai parameter
11
       hello ("Andi");
       hello("Ani");
12
13
       hello("Ana");
14
       hello("Ali");
15
       hello("Adi");
```

Contoh Void (4)

```
1 #include <iostream>
 2
   using namespace std;
 3
 4 //membuat fungsi yg memiliki parameter nilai
 5 pvoid fungsiluas (int p, int 1) {
 6
        int luas = p*1;
 7
        cout<<"Hasil Luasnya adalah "<<luas;
 8
   L }
 9
10 pint main() {
11
        //memanggil fungsi dgn memberi nilai parameter
12
        fungsiluas (4,5);
13 \{\}
```

Contoh Void (5)

```
1 #include <iostream>
    using namespace std;
3
4 //membuat fungsi yg memiliki parameter nilai
5 ⊟void fungsiluas(int p, int 1){
        int luas = p*1;
7
        cout<<"Hasil Luasnya adalah "<<luas;
8 4
9
11
        int panjang, lebar;
12
        cout<<"Inputkan nilai panjang = ";</pre>
13
        cin>>panjang;
14
        cout<<"Inputkan nilai lebar = ";</pre>
15
        cin>>lebar;
16
17
        //memanggil fungsi dgn memberi nilai parameter
18
        fungsiluas(panjang,lebar);
19 L}
```

Contoh Void (6)

```
#include <iostream>
 2
  using namespace std;
 3
 4
  //deklarasi fungsi
   void fungsiluas(int p, int l);
 7 pint main(){
         int panjang, lebar;
9
        //memanggil fungsi dgn memberi nilai parameter
10
        cout<<"Inputkan nilai panjang = ";</pre>
11
        cin>>panjang;
12
        cout<<"Inputkan nilai lebar = ";</pre>
13
        cin>>lebar;
14
        fungsiluas(panjang,lebar);
15 L}
16
   //definisi fungsi
18 pvoid fungsiluas(int p, int l){
19
        int luas = p*1;
20
        cout<<"Hasil Luasnya adalah "<<luas;</pre>
21 \[ \}
```

Contoh Non Void

```
#include <iostream>
    using namespace std;
 3
 4
    //deklarasi variabel global
 5 int luas;
 6
7
    //membuat fungsi yg memiliki parameter nilai
8 pint fungsiluas (int p, int 1) {
9
        luas = p*1;
10
        return luas;
11
   L }
12
13 pint main(){
14
        //deklarasi variabel lokal
15
        int panjang, lebar;
16
17
        cout<<"Inputkan nilai panjang = ";
18
        cin>>panjang;
19
        cout<<"Inputkan nilai lebar = ";</pre>
20
        cin>>lebar;
2.1
22
        //memanggil fungsi dgn memberi nilai parameter
23
        fungsiluas(panjang,lebar);
24
        cout<<"Hasil Luasnya adalah "<<luas;</pre>
25
   L }
```

Contoh Rekursif

```
1 #include <iostream>
2 using namespace std;
4 pint rekursiffaktorial (int f) {
        if (f==1) {
6
            return 1;
7
        } else {
8
            return f*rekursiffaktorial(f-1);
9
        }
   L }
10
11
12 pint main(){
13
        int faktorial;
14
15
        cout<<"Inputkan nilai faktorial = ";</pre>
16
        cin>>faktorial;
17
        cout<<"Hasil Faktorialnya adalah "<<rekursiffaktorial(faktorial);</pre>
18
19 1
```