

Modul Praktikum  
Algoritma dan Pemrograman I

# “ Fungsi ”



TIM ASISTEN PEMROGRAMAN  
ANGKATAN 12

Departemen Pendidikan Ilmu Komputer  
Fakultas Pendidikan Matematika dan Ilmu Pengetahuan Alam  
Universitas Pendidikan Indonesia  
2021

## Pendahuluan

Hampir sama seperti prosedur, fungsi adalah sebuah blok/bagian dari kode yang terpisah dari blok/bagian utamanya (biasanya **main**). Akan tetapi, prosedur hanya menjalankan tugas tertentu tanpa menghasilkan nilai apapun. Maka dari itu, **fungsi** adalah **prosedur yang bisa mengembalikan nilai (return) dalam bentuk nilai (value)**.

Suatu blok kode dikatakan sebagai fungsi jika mengembalikan nilai untuk kemudian digunakan. Meskipun biasanya fungsi digunakan untuk mengembalikan nilai ke blok kode utama (main), fungsi juga bisa digunakan untuk mengembalikan nilai yang dibutuhkan oleh prosedur atau fungsi lain.

## Sintaks

Fungsi dideklarasikan seperti prosedur biasa, tetapi **diawali dengan tipe data nilai yang akan dikembalikannya**.

```
returnType functionName(dataType arguments1, dataType arguments2, ...);
```

Contoh:


```
1  #include "stdio.h"
2
3  int foo(int a, int b)
4  {
5      // Code.
6      ...
7
8      return var;
9  }
```

Dengan “**int**” di awal sebagai tipe nilai yang akan dikembalikan, “**foo**” sebagai nama fungsi, dan “(**int a, int b**)” sebagai parameter dari fungsi tersebut.

Sementara itu, **baris kode “return var;”** digunakan untuk mengembalikan nilai dari **fungsi**. Jika tipe nilai yang akan dikembalikan berupa integer, maka var ini harus berupa integer juga.

## Arguments dan Cara Kerja Fungsi

**Arguments** adalah variabel yang akan dijadikan data referensi pada suatu parameter di dalam fungsi. Biasanya, fungsi akan mengembalikan nilai hasil olahan argument di dalam blok kodenya.



Baik arguments maupun tipe data fungsi tidak harus berbentuk integer. Bisa juga tipe data lainnya seperti char, float, bahkan struct dan pointer.

Contoh:

```
1  #include "stdio.h"
2
3  int foo(int a, int b);
4
5  int main()
6  {
7      int x, y;
8
9      // Code.
10     ...
11
12     foo(x, y);
13
14     // Code.
15     ...
16
17     return 0;
18 }
19
20 int foo(int a, int b)
21 {
22     // Code.
23     ...
24
25     return var;
26 }
```

Keterangan:

- `(int a, int b)` merupakan parameter, yang menandakan fungsi memerlukan dua variabel integer sebagai data referensi fungsinya.
- `return var;` menandakan fungsi akan mengembalikan variabel bernama `var`. Meskipun tidak ditulis secara eksplisit, dapat disimpulkan `var` tersebut berupa integer.
- `foo(x, y);` merupakan baris kode di mana fungsi dipanggil. `x` dan `y` merupakan variabel bertipe integer, yang mana digunakan sebagai data referensi dari parameter `a` dan `b`.

## Contoh Kode Program Penggunaan Fungsi dan Beberapa Perbandingannya

Contoh program ini digunakan untuk menghitung pangkat dari suatu nilai.

Kode dasar:

```
1 // Deklarasi pustaka.
2 #include <stdio.h>
3
4 // Fungsi pemangkatan dasar.
5 int powerof(int base, int power)
6 {
7     // Deklarasi variabel kontrol dan nilai hasil.
8     int i,
9         result = 1;
10
11     // Mengalikan bilangan "base" sebanyak "power" kali.
12     for(i = 0; i < power; i++)
13     {
14         result *= base;
15     }
16
17     // Mengembalikan nilai hasil pemangkatan.
18     return result;
19 }
20
21 int main()
22 {
23     // Deklarasi variabel masukan dan nilai hasil.
24     int n, x,
25         result;
26
27     printf("Masukkan angka : ");
28     scanf("%d", &n);
29
30     printf("Masukkan pangkat : ");
31     scanf("%d", &x);
32
33     // Mendapatkan hasil n dipangkatkan oleh x.
34     result = powerof(n, x);
35
36     printf("Hasil : %d\n", result);
37 }
```

Kode tanpa tampungan hasil:

```
21  int main()
22  {
23      // Deklarasi variabel masukan dan nilai hasil.
24      int n, x,
25          result;
26
27      printf("Masukkan angka : ");
28      scanf("%d", &n);
29
30      printf("Masukkan pangkat : ");
31      scanf("%d", &x);
32
33      // Keluaran hasil n dipangkatkan oleh x.
34      printf("Hasil : %d\n", powerof(n, x));
35  }
```

Kode dengan pointer:

```
1  // Deklarasi pustaka.
2  #include <stdio.h>
3
4  // Fungsi pemangkatan dasar.
5  int powerof(int *base, int power)
6  {
7      // Deklarasi variabel kontrol dan nilai sementara.
8      int i, temp;
9
10     // Jika pangkat adalah 0, nilai hasil selalu 1.
11     if(power == 0)
12     {
13         return 1;
14     }
15     else
16     {
17         // Menyimpan nilai basis statis.
18         temp = *base;
19
20         // Mengalikan bilangan "base" sebanyak "power" kali.
21         for(i = 1; i < power; i++)
22         {
23             *base *= temp;
24         }
25
26         // Mengembalikan nilai basis yang sudah dipangkatkan.
27         return *base;
28     }
29 }
```

```

31  ✓ int main()
32  {
33      // Deklarasi variabel masukan dan nilai hasil.
34      int n, x,
35          result;
36
37      printf("Masukkan angka : ");
38      scanf("%d", &n);
39
40      printf("Masukkan pangkat : ");
41      scanf("%d", &x);
42
43      // Keluaran hasil n dipangkatkan oleh x.
44      printf("Hasil : %d\n", powerof(&n, x));
45  }

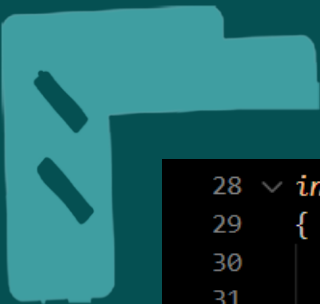
```

Kode menggunakan pass by reference:

```

1  // Deklarasi pustaka.
2  #include <stdio.h>
3
4  // Prosedur pemangkatan dasar.
5  void powerof(int *base, int power)
6  {
7      // Deklarasi variabel kontrol dan nilai sementara.
8      int i, temp;
9
10     // Jika pangkat adalah 0, nilai hasil selalu 1.
11     if(power == 0)
12     {
13         *base = 1;
14     }
15     else
16     {
17         // Menyimpan nilai basis statis.
18         temp = *base;
19
20         // Mengalikan bilangan "base" sebanyak "power" kali.
21         for(i = 1; i < power; i++)
22         {
23             *base *= temp;
24         }
25     }
26 }

```



```
28  ✓ int main()
29  {
30      // Deklarasi variabel masukan dan nilai hasil.
31      int n, x,
32          result;
33
34      printf("Masukkan angka : ");
35      scanf("%d", &n);
36
37      printf("Masukkan pangkat : ");
38      scanf("%d", &x);
39
40      powerof(&n, x);
41
42      // Keluaran hasil n dipangkatkan oleh x.
43      printf("Hasil : %d\n", n);
44  }
```

## Latihan

Buatlah satu fungsi yang menerima satu string, lalu **periksa** apakah **jumlah nilai ASCII** dari string tersebut **ganjil** atau **genap**!

### Contoh Masukan

Halo

### Contoh Keluaran

Nilai ASCII : genap.

### Contoh Masukan

Techi

### Contoh Keluaran

Nilai ASCII : ganjil.

---

Buatlah fungsi yang menerima struct nama dan nilai, lalu kelompokkan nilai ke dalam huruf (A = 100-86, B = 85-71, ...). Selain itu, jumlahkan juga nilai ASCII dari nama. **Dengan masukan array of struct, tampilkan nama, kelompok, dan nilai yang sudah dijumlahkan dengan total nilai ASCII nama tersebut!**

### Contoh Masukan

1

Aldian 90

### Contoh Keluaran

Aldian A 675

### Contoh Masukan

3

Carla 72

Ellona 60

Techi 99

### Contoh Keluaran

Carla B 555

Ellona C 663

Techi A 592





## Penutup

Terima kasih atas kerja sama seluruh pihak yang membantu dalam penyusunan modul ini, semoga apa yang telah didapatkan bisa bermanfaat di masa yang akan datang.

## Daftar Pustaka

Tim Asisten Praktikum Algoritma dan Pemrograman I Angkatan 11. (2020). Fungsi. Modul Praktikum Algoritma dan Pemrograman I. Bandung: Universitas Pendidikan Indonesia.