



Asia Cyber  
University

# Algoritma & Pemrograman 1

Function

# Tujuan Pembelajaran

- Mampu menggunakan *Function* dalam program

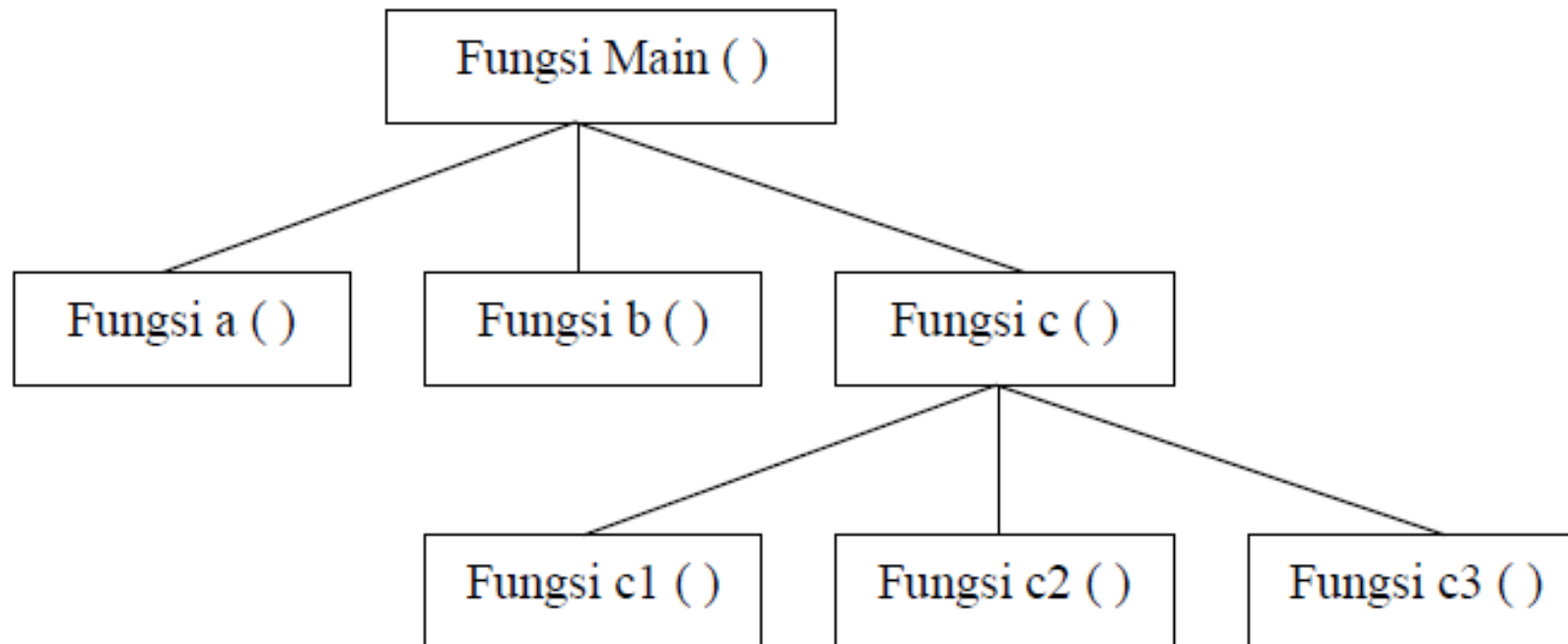
# Function

- Fungsi adalah sekumpulan perintah operasi program yang dapat menerima argumen input dan dapat memberikan hasil output yang dapat berupa nilai ataupun sebuah hasil operasi.
- Hasil akhir fungsi akan berupa sebuah nilai balik (return)
- Nama fungsi yang didefinisikan sendiri oleh pemrogram tidak boleh sama dengan nama build-in function pada compiler C++.
- Fungsi digunakan agar pemrogram dapat menghindari penulisan program berulang-ulang, agar terlihat rapi dan kemudahan dalam debugging program.
- Parameter adalah nama-nama peubah yang dideklarsikan pada bagian header fungsi.
- Pemrogram dapat membuat fungsi yang didefinisikan sendiri olehnya.

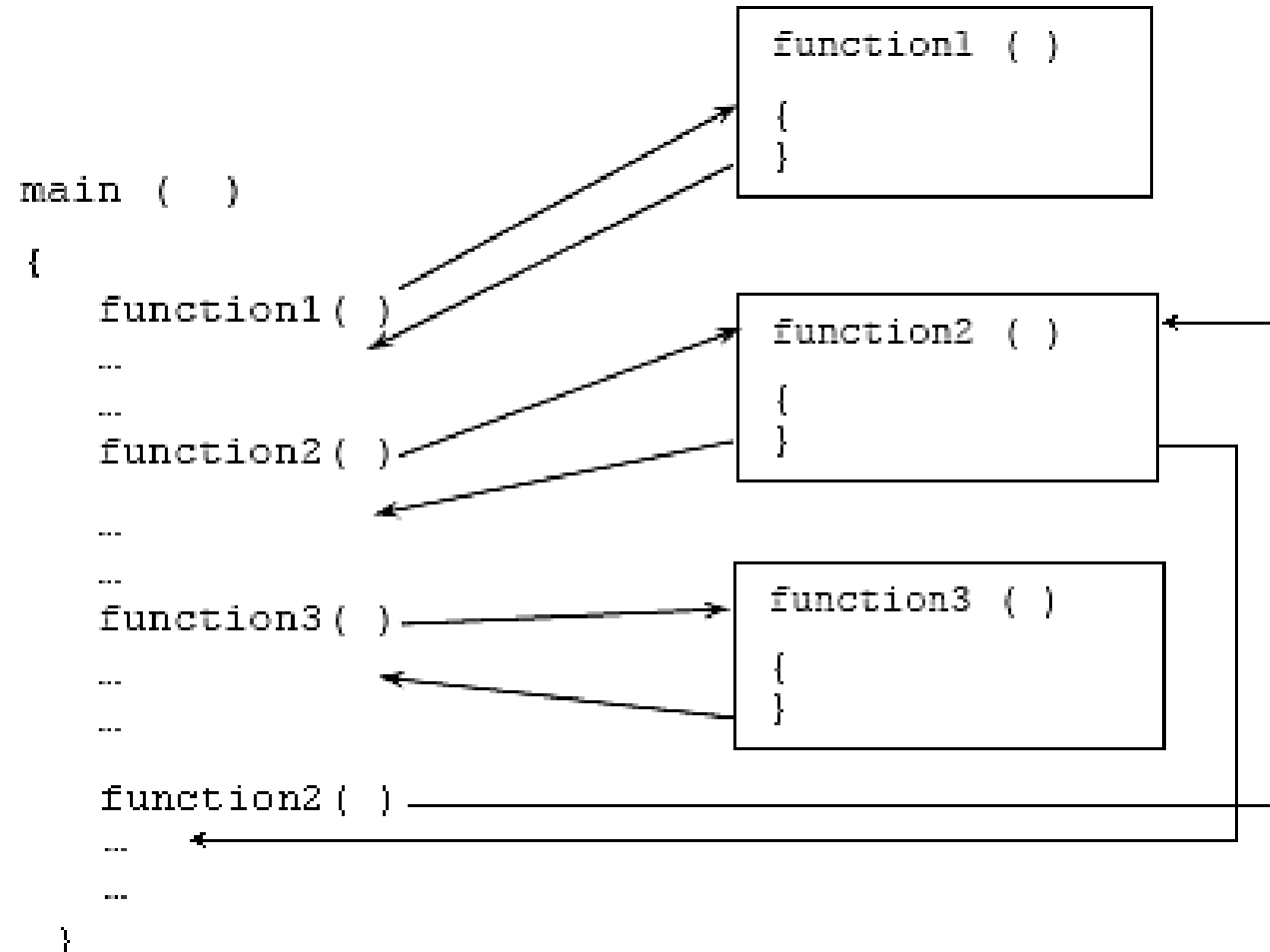
# Function

Bentuk Umum :

```
Deklarasi parameter  
{  
    Isi fungsi  
}
```



# Struktur Function

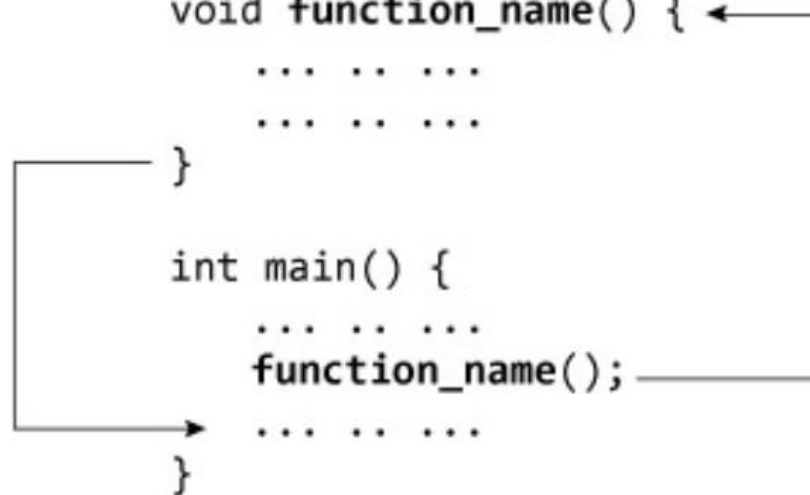


# Struktur Fungsi

```
#include <iostream>

void function_name() {
    ... ..
}

int main() {
    ... ..
    function_name();
}
```

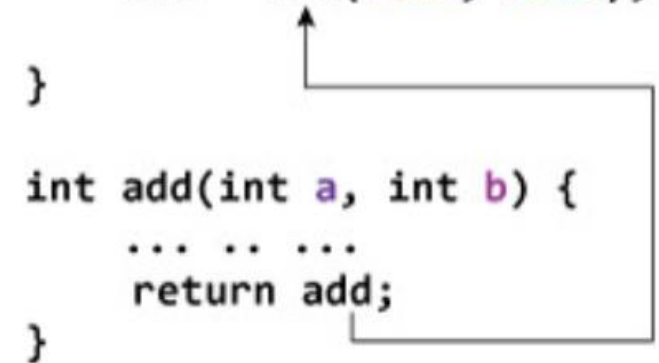
A diagram with two arrows. One arrow starts from the closing brace of the `main` function and points to the opening brace of the `function_name` function. The other arrow starts from the `function_name()` call inside the `main` function and points to the opening brace of the `function_name` function.

```
# include <iostream>
using namespace std;

int add(int, int);

int main() {
    ... ..
    sum = add(num1, num2);
}

int add(int a, int b) {
    ... ..
    return add;
}
```

A diagram with an arrow starting from the `return add;` line inside the `add` function and pointing to the opening brace of the `add` function, indicating a recursive call.

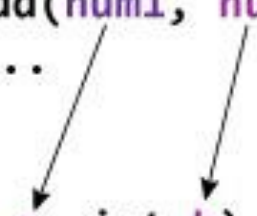
# Struktur Function

```
# include <iostream>
using namespace std;

int add(int, int);

int main() {
    ... ..
    sum = add(num1, num2); // Actual parameters: num1 and num2
    ... ..
}

int add(int a, int b) { // Formal parameters: a and b
    ... ..
    add = a+b;
    ... ..
}
```



The diagram illustrates the relationship between actual and formal parameters. Two arrows originate from the actual parameters 'num1' and 'num2' in the function call within the `main` function. These arrows point to the formal parameters 'a' and 'b' respectively in the function definition of `add`.

# Prototipe Fungsi

- Sebuah fungsi tidak dapat dipanggil kecuali sudah dideklarasikan, deklarasi fungsi dikenal dengan sebutan prototipe fungsi. Prototipe fungsi berupa :
  1. Nama Fungsi
  2. Tipe nilai fungsi
  3. Jumlah dan tipe argumen
- Dan diakhiri dengan titik koma, sebagaimana pada pendeklarasian variabel.



# Prototipe Fungsi

Contoh:

1. **long** kuadrat (**long l**) ;

Pada contoh pertama, fungsi kuadrat ( ) mempunyai argumen bertipe long dan nilai balik bertipe **long**.

2. **void** garis ( ) ;

Pada contoh kedua, fungsi garis ( ) tidak memiliki argumen dan nilai baliknya tidak ada (**void**).

3. **double** maks (**double x, double y**)

Pada contoh ketiga, fungsi maks( ) mempunyai dua buah argumen, dengan masing masing argumen bertipe double.

# Contoh Fungsi

```
# include <iostream.h>

double hasil (int A, int B);

void main()
{
    int x,y;
    double z;

    cout << "Masukkan Nilai x : ";
    cin >> x;

    cout << "Masukkan Nilai y : ";
    cin >> y;

    z = hasil (x,y);

    cout << "Hasil perkaliannya = ";
    cout << x << " x " << y << " = " << z;
}

double hasil (int A, int B)
{
    return (A * B);
} // Statement Mengembalikan Nilai
```

Penjelasan :

Fungsi terdiri atas dua bagian, yaitu judul (*header*) dan isi (*body*). Judul dari sebuah fungsi terdiri dari tipe return (**double**), nama fungsi (hasil) dan list parameter ( **int A, int B**).

Jadi, judul untuk fungsi hasil adalah **double** hasil (**int A, int B**)

Isi dari sebuah fungsi adalah blok kode yang mengikuti judulnya.

Berisi kode yang menjalankan aksi dari fungsi, termasuk pernyataan *return* yang memuat nilai fungsi yang akan dikembalikan ke yang memanggilnya, Isi dari fungsi hasil ( ) adalah

# Contoh Fungsi



Asia Cyber  
University

```
#include <iostream>
using namespace std;

// Function prototype (declaration)
int add(int, int);

int main()
{
    int num1, num2, sum;
    cout<<"Enters two numbers to add: ";
    cin >> num1 >> num2;

    // Function call
    sum = add(num1, num2);
    cout << "Sum = " << sum;
    return 0;
}

// Function definition
int add(int a, int b)
{
    int add;
    add = a + b;

    // Return statement
    return add;
}
```

# Pernyataan Return

- **Bentuk umum** pernyataan return :  
`return ekspresi;`
  - Dengan *ekspresi* adalah sebuah ekspresi yang nilainya dinyatakan untuk sebuah variable yang tipenya sama seperti tipe **return**.
  - Terdapat juga fungsi yang tidak memberikan nilai balik atau tipe **returnnya** void.
  - Pernyataan *return* dari sebuah fungsi mempunyai dua manfaat, yaitu akan mengakhiri fungsi dan mengembalikan nilainya ke program pemanggil.

# Contoh Tanpa Return

Fungsi di bagian atas

```
1  #include <iostream>
2  using namespace std;
3
4  void tampilkan_judul()
5  {
6      cout << "Universitas Siber Asia" << endl;
7      cout << "Fakultas Teknologi Komunikasi dan Infomasi" << endl;
8      cout << "Jl. Sawo Manila Pejaten" << endl;
9  }
10
11 int main()
12 {
13     tampilkan_judul();
14 }
```



# Contoh Tanpa Return

Fungsi di bagian bawah

```
1  #include <iostream>
2  using namespace std;
3
4  void tampilkan_judul();
5
6  int main()
7  {
8      tampilkan_judul();
9  }
10
11 void tampilkan_judul()
12 {
13     cout << "Universitas Siber Asia" << endl;
14     cout << "Fakultas Teknologi Komunikasi dan Infomasi" << endl;
15     cout << "Jl. Sawo Manila Pejaten" << endl;
16 }
```

***Terimakasih***