FUNCTION AND PROCEDURES

TODAY

- FUNCTION
- PROCEDURES

PROCEDURES

- Procedures merupakan blok kode program yang melakukan tugas tertentu.
- Procedures tidak memiliki nilai pengembalian

PROCEDURES

- Procedures merupakan blok kode program yang melakukan tugas tertentu.
- Procedures tidak memiliki nilai pengembalian

PROCEDURES

- Berikut ini merupakan contoh prosedur

```c
void mhs(){
    printf("Hay saya merupakan prosedur\n");
    printf("Prosedur ini dapat anda panggil tanpa menggunakan argumen");
}

int main(){
    mhs();
}
```

PROCEDURES

- Berikut ini merupakan contoh prosedur
- Terdapat 2 argument yang digunakan pada procedure ini

```c
void mhs(int a, int b){
    int hasil = a * b;
    printf("Hasil adalah %d", hasil);
}

int main(){
    int p, l;
    printf("Masukkan nilai p dan l : ");
    scanf("%d %d", &p, &l);
    mhs(p, l);
}
```

PROCEDURES

- Contoh lain dari procedures

```
void mhs(){
    char nama[20];
    printf("Masukkan nama anda : ");
    scanf("%s", nama);
    printf("Nama Mahasiswa : %s", nama);
}

int main(){
    mhs();
}
```

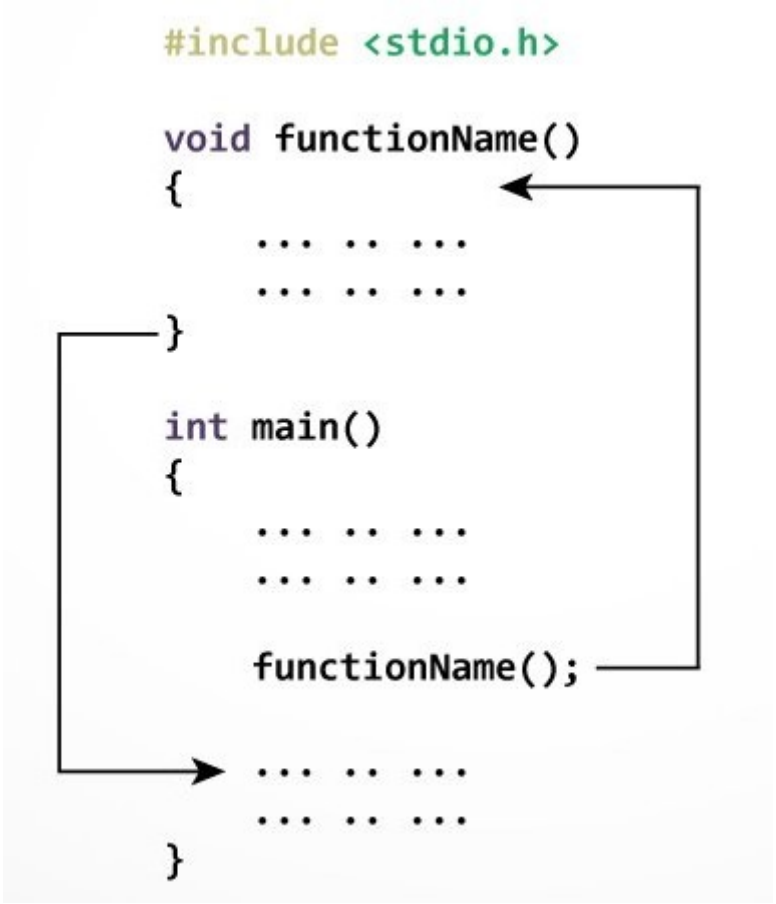
FUNCTION

- Function merupakan blok kode program yang melakukan tugas tertentu.
- Function memiliki nilai pengembalian
- Standard library function
- User-defined function

ADVANTAGEG

- Program akan lebih mudah dipahami, dipelihara, dan di-debug.
- Kode yang dapat digunakan kembali yang dapat digunakan di program lain
- Sebuah program besar dapat dibagi menjadi modul-modul yang lebih kecil. Oleh karena itu, sebuah proyek besar dapat dibagi di antara banyak programmer.

FUNCTION



FUNCTION

- Function terdiri dari nama function dan argument
- Berikut ini merupakan contoh fungsi yang dibangun tanpa argument.

```c
#include <stdio.h>

// function fedinition
int penjumlahan(){

    int result = 10 + 10;

    return result;
}

int main(){

    //call function
    printf("%d", penjumlahan());

    return 0;
}
```

FUNCTION

- Membuat function dengan return string dapat menggunakan contoh berikut.
- Khusus return bertipe string, Dalam c kita tidak dapat melakukan return terhadap nama variable string tersebut

```
int mhs(){
    return "martin";
}
int main(){
    printf("%s", mhs());
    return 0;
}
```

FUNCTION

- Function terdiri dari nama function dan argument
- Berikut ini merupakan contoh function yang menggunakan 2 argumen

```c
#include <stdio.h>

// function fedinition
int penjumlahan(int a, int b){

    int result = a + b;

    return result;
}

int main(){

    int p, l, total;
    printf("Masukkan nilai p : ");
    scanf("%d", &p);
    printf("Masukkan nilai l : ");
    scanf("%d", &l);

    //call function
    total = penjumlahan(p, l);
    printf("Hasil = %d", total);

    return 0;
}
```

FUNCTION

- Contoh lain penggunaan function

```c
int aritmatika(){
    int p, l;
    printf("Masukkan nilai p dan l : ");
    scanf("%d %d", &p, &l);

    int result = p * l;

    return result;
}

int main(){


    printf("%d", aritmatika());

    return 0;
}
```

FUNCTION

- Fungsi dapat melakukan pengembalian nilai lebih dari 1.
- Terdapat beberapa cara untuk melakukannya.
- Berikut ini merupakan teknik dengan menggunakan array.

```c
#include <stdio.h>

// function fedinition
int aritmatika(int a, int b, int result[2]){

    result[0] = a + b;
    result[1] = a - b;

    return result;
}

int main(){

    int p, l, total[2];
    printf("Masukkan nilai p : ");
    scanf("%d", &p);
    printf("Masukkan nilai l : ");
    scanf("%d", &l);

    //call function
    aritmatika(p, l, total);
    printf("penjumlahan = %d \npengurangan = %d", total[0], total[1]);

    return 0;
}
```

FUNCTION

- Fungsi dapat melakukan pengembalian nilai lebih dari 1.
- Terdapat beberapa cara untuk melakukannya.
- Berikut ini merupakan teknik dengan menggunakan Structures.

```c
#include <stdio.h>

// function fedinition
struct addReduc{
    int add, reduc;
};

typedef struct addReduc Struct;

Struct aritmatika(int a, int b){
    Struct x;
    x.add = a + b;
    x.reduc = a - b;

    return x;

}

int main(){

    Struct result;
    int p, l;
    printf("Masukkan nilai p : ");
    scanf("%d", &p);
    printf("Masukkan nilai l : ");
    scanf("%d", &l);

    //call function
    result = aritmatika(p, l);
    printf("penjumlahan = %d \npengurangan = %d", result.add, result.reduc);

    return 0;
}
```