



ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

# Function and structured programming

Department of Information System  
SoICT, HUST

# Function

- Is a block of declarations and statements which is assigned a name
- A function is a sub-program
- A program is a function with the name **main** and can **call** to sub-programs
- These sub-programs can use other functions

# Example

*Function definition*

```
#include <stdio.h>
```

```
// Function prints a greeting
```

```
void sayHello ( void )  
{  
    printf("Hello World!\n");  
}
```

```
// Calling the greeting function
```

```
int main(void)  
{  
    sayHello();  
    return 0;  
}
```

*Function call*

# Why using functions?

- Functions allow divide a problem into smaller problems
  - Allow solving the difficult problem easier
- A program is clearer when using functions
  - We only need to know what a function does without caring how it is done
- They allow generalize some groups of statements that repeat many times
  - Prevent repeatedly writing a group of statements many time

# Building function

- Writing a function needs to specify:
  - the name of the function
  - its parameters
  - what it returns
  - block of statements to be carried out when the function is called
- The block of statements is called the “function body”

# Factorial function

*Function's name*

*Function's body*

```
#include <stdio.h>
```

```
int factorial (int a)  
{
```

```
    int i, fac=1;  
    for(i=1; i<=a; i++)  
        fact = fac * i;  
    return fac;  
}
```

```
int main( void ) {  
    int num;
```

```
    printf("Input an integer:");  
    scanf ("%d", &num);
```

```
    printf("%d!=%d\n",  
           num, factorial(num));
```

```
}
```

# Function parameters

- Parameters are information passed to a function
- “Formal” parameters are local variables declared inside the function declaration.
- “Actual” parameters are values passed to the function when it is called
- Parameters are local variables of the function. Their values are defined each time the function is called.
  - Parameters have different values at each time the function is called
  - Parameters can only be accessed inside the function
  - When calling the function, values for all parameters must be defined
- Note:
  - Parameters are passed by copying the value of the actual parameters to the formal parameters.
  - Changes to formal parameters do not affect the value of the actual parameters.

# Example of parameter

```
#include <stdio.h>

int addOne ( int i )
{
    i = i + 1;
    return i;
}

int main(void)
{
    int i = 3;

    printf("%d\n", addOne(i) );
    printf("%d\n", i);

    return 0;
}
```

Declare a parameter  
as a local variable

Change the value  
of the local variable

Passing the value of **i**  
in function **main** for  
the function

Output:

4

3



# Example

```
void badSwap ( int a, int b )
{ int temp;
  temp = a;
  a = b;
  b = temp;
  printf("Called environment: %d %d\n",a,b);
}

int main(void)
{ int a = 3, b = 5;
  printf("Calling environment: %d %d\n",a,b);
  badSwap ( a, b );
  printf("Calling environment: %d %d\n",a,b);
  return 0;
}
```

# Return value

- **return** statement is used to return a value for a function
- A function can have several return statements. The first return that the program meets will terminate the function.
- A function that returns nothing must be declared with the return type **void**
  - In this case, no **return** is needed

# Declare and define a function

- A definition of the function that describes all members of the function including main body of the function
- A function declaration only has to declare:
  - Function's name
  - Argument's type
  - Return type
- Create a function declaration by using prototype.

Example:

```
int addOne (int) ;  
void sayHello(void) ;
```

# Role of prototype

- A function can be defined after being used, however it has to declare before being used.
- It allows to call a function without knowing its definition.
  - Example, the prototype of the function `printf()` is declared in the file `stdio.h`

# Factorial function

*Prototype*

```
#include <stdio.h>
```

```
int factorial (int);
```

```
int main( void ) {
```

```
    int num;
```

```
    printf("Enter an integer number:");
```

```
    scanf("%d",&num);
```

```
    printf("%d!=%d\n",  
          num, factorial(num));
```

```
}
```

*Definition*

```
int factorial (int a) {
```

```
    int i, gt=1;
```

```
    for(i=1; i<=a; i++)
```

```
        gt = gt * i;
```

```
    return gt;
```

# Global variable

- Variables declared in a function body (local variables) are only accessible while the function is executing.
- Global variables are variables declared outside the functions. They accessible in any function **after** their declaration to the end of that source file.
- Example:  
    int global;  
    void f(void) { global = 0; }  
    void f(void) { global = 1; }

# Variables with the same name

- When the global variable and the local variable has the same name, the local variable has a higher priority than the global one.
- Example

```
int i; //global variable
void f() {
    int i; //local variable
    i++; // only change value of the local variable i
}
void g() {
    i++; // change value of the global variable i
}
```

# Function library

- C provides some functions such as input, output, mathematic, memory management, string processing, etc.
- To use these functions, their prototypes are needed to be declared in the program.
- Such prototypes are written in header files (.h). We only need to #include them in the program



# math.h

- Include a set of mathematic functions with the prototypes:

```
double sin(double x);  
double cos(double x);  
double tan(double x);  
...  
double log(double x);  
double sqrt(double x);  
double pow(double x, double y);  
int ceil(double x);  
int floor(double x);  
...
```

# Exercise

- Given two function prototypes:  
`int nhapso() ;`  
`int max(int a, int b) ;`
- Write function definitions and the main program using the above functions for finding the maximum values for 3 numbers entered from keyboard.



25 YEARS ANNIVERSARY  
**SOICT**

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG  
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

**Thank you  
for your  
attentions!**



[soict.hust.edu.vn/](http://soict.hust.edu.vn/)



[fb.com/groups/soict](https://fb.com/groups/soict)

