HÀM (FUNCTION)

1. KHÁI QUÁT VỀ HÀM

Phần lớn các chương trình máy tính được xây dựng để giải quyết các bài toán lớn trong thực tế.

Cách tốt nhất để xây dựng và bảo trì một chương trình đó là xây dựng nó từ những thành phần nhỏ được xây dựng đơn lẻ.

Chức năng Hàm (Function) được sử dụng để chia nhỏ chương trình thành các thủ tục nhỏ giải quyết từng chức năng nhỏ

1. KHÁI QUÁT VỀ HÀM

Lợi ích khi chia chương trình thành các hàm nhỏ

Code trở nên mạch lạc, dễ đọc

Dễ debug khi gặp lỗi

Dễ bảo trì khi cần thay đổi một chức năng nhỏ

Có khả năng tái sử dụng lại code

2. CÚ PHÁP (SYNTAX)

```
returntype functionName ( parameter1, parameter2,...){
        //Function body
}
```

Trong đó:

returntype: Kiểu trả về của hàm (int, double, void,...)

functionName: Tên hàm

parameter: Tham số của hàm

function body: Các câu lệnh bên trong của hàm

2. CÚ PHÁP (SYNTAX)

Ví dụ:

Kiểu trả về void, tức hàm này không trả về giá trị nào

xinchao: tên hàm

Hàm này không có tham số

```
void xinchao (){
    printf("Hello 28tech!\n");
}
```

3. GỌI HÀM (FUNCTION CALL)

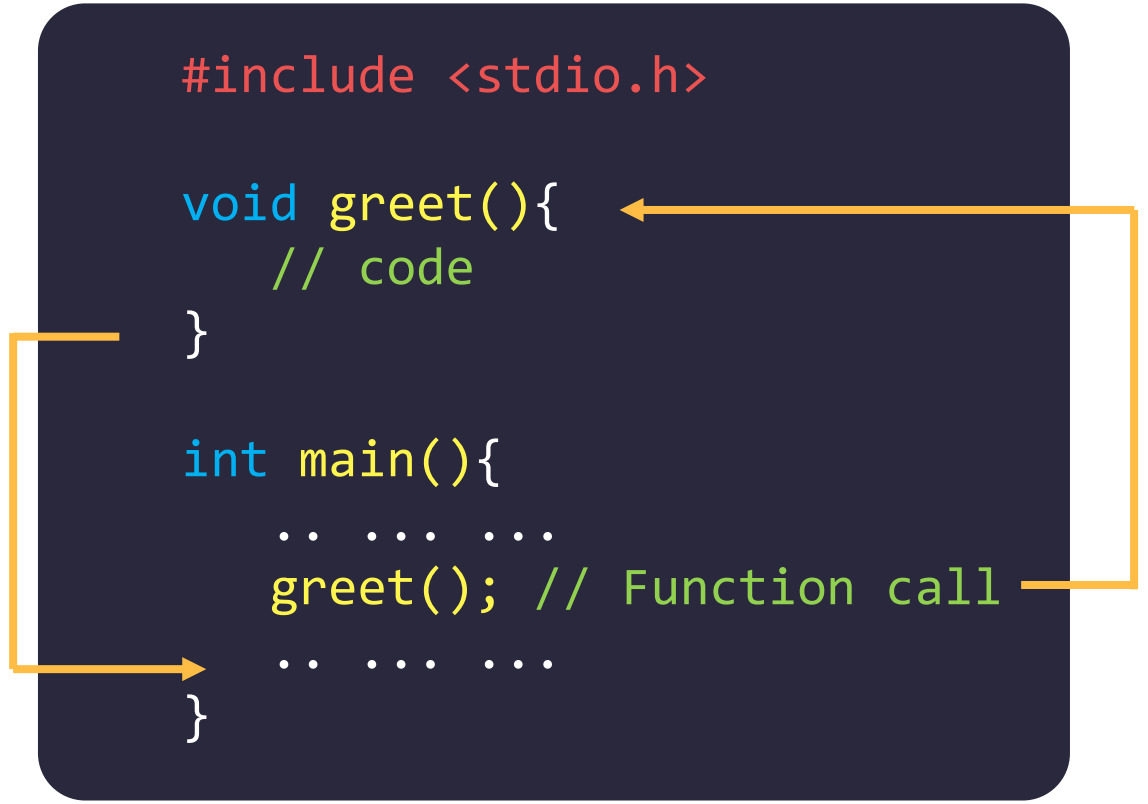
```c
#include <stdio.h>

void greet(){
    printf("Hello 28tech !\n");
}

int main(){
    printf("Truoc khi goi ham \n");
    greet(); // Function call
    printf("Sau khi goi ham");
}
```

OUTPUT :

Truoc khi goi ham
Hello 28tech !
Sau khi goi ham



4. KHAI BÁO NGUYÊN MẪU HÀM (FUNCTION PROTOTYPE)

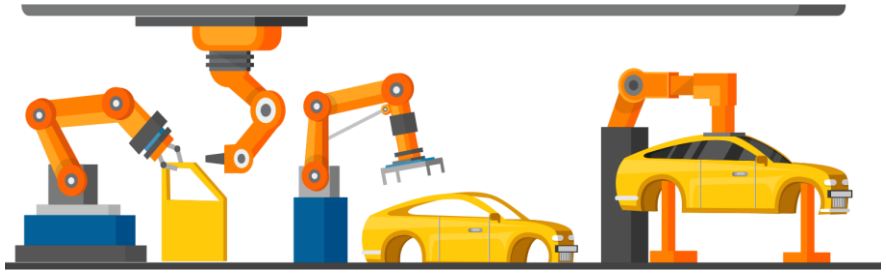
👉 Các bạn có thể khai báo nguyên mẫu hàm ở đầu chương trình để làm cho chương trình rõ ràng hơn cũng như thông báo những hàm.

```
//Function prototype
int max_val(int, int);
//Function declaration
int max_val(int a, int b){
      if (a > b)
         return a;
      else
         return b;
}
```

5. THAM SỐ VÀ GIÁ TRỊ TRẢ VỀ (PARAMETER & RETURN TYPE)

Tham số

Hàm



Giá trị trả về

Tham số là những giá trị được truyền cho hàm khi hàm được gọi.

5. THAM SỐ VÀ GIÁ TRỊ TRẢ VỀ (PARAMETER & RETURN TYPE)

VÍ DỤ

```
void printNumber(int a){
    printf ("%d",a);
}
int main(){
    int n = 10;
    printNumber(n);
}
```

OUTPUT: 10

Giải thích cách hoạt động: Khi bạn gọi hàm printNumber(n) thì n được gọi là Đối số hay tham số thực sự, còn a trong tham số hình thức. Giá trị của n sẽ được gán cho a, vì thế a cũng sẽ có giá trị là 10, và khi câu lệnh bên trong hàm printNumber thực hiện màn hình sẽ hiển thị giá trị 10.

Dự đoán Output của một số chương trình sau:

Program 1

```c
#include <stdio.h>

int tong(int a, int b, int c){
    return a + b + c;
}

int main(){
    int x = 100, y = 200, z = 300;
    printf("%d\n", tong(x, y, z));
    tong(x, y, z);
}
```

Program 2

```c
#include <stdio.h>

void printNum(int a, int b, int c){
    printf("%d %d %d", b, c, a);
}

int main(){
    int x = 100, y = 200, z = 300;
    printNum(x, y, z);
}
```

Dự đoán Output của một số chương trình sau:

Program 3

```c
#include <stdio.h>

int tong(int x, int y, int z){
    return 100;
    int tong = x + y + z;
    return tong;
}

int main(){
    int x = 100, y = 200, z = 300;
    printf("%d", tong(x, y, z));
}
```

Program 4

```c
#include <stdio.h>

int tong(int x, int y, int z){
    return 100;
    int tong = x + y + z;
    return tong;
}

int main(){
    int x = 100, y = 200, z = 300;
    printf("%d", tong(x));
}
```

Dự đoán Output của một số chương trình sau:

Program 5

```c
#include <stdio.h>

int tong(int x, int y, int z){
    return 100;
    int tong = x + y + z;
    return tong;
}

int main(){
    int x = 100, y = 200, z = 300;
    tong(x, y, z);
}
```

Program 6

```c
#include <stdio.h>

void greet(){
   printf("Hello 28tech !\n");
   return;
   printf("C programming !\n");
}

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

6. TRUYỀN THAM TRỊ

XÉT VÍ DỤ

```
#include <stdio.h>

void change(int n){
    n += 100;
    printf("%d\n", n);
}

int main(){
    int a = 200;
    change(a);
    printf("%d", a);
}
```

OUTPUT: 300
200

Khi gọi hàm change(a) thì tham số thực sự của hàm change là a = 200, khi đó giá trị này được gán cho biến n là một biến trong hàm change, 2 biến này hoàn toàn khác nhau và việc bạn cho n += 100 sẽ tăng n lên làm 300. Câu lệnh in ra n trong hàm change sẽ có kết quả là 300, nhưng sau khi hàm change thực thi xong, bạn in ra giá trị của a thì a vẫn là 200. Đây gọi là truyền tham trị

Nếu bạn truyền tham trị thì giá trị của tham số thực sự sẽ không bị thay đổi sau khi hàm kết thúc.

6. TRUYỀN THAM TRỊ

👉 Nếu bạn muốn thay đổi giá trị của n sau khi kết thúc hàm, có 2 cách giải quyết:

1.Sử dụng con trỏ(sẽ học sau)

2.Viết hàm change trả về n + 100 và gán lại cho n như cách bên

```c
#include <stdio.h>

int change(int n){
    n += 100;
    printf("%d\n", n);
    return n;
}

int main(){
    int a = 200;
    a = change(a);
    printf("%d", a);
}
```

OUTPUT: 300
300

7. NHỮNG CHÚ Ý KHI XÂY DỰNG HÀM

1. Hàm này có cần trả về giá trị không, nếu có thì trả về kiểu dữ liệu là gì?

2. Hàm này có bao nhiêu tham số, các tham số có kiểu dữ liệu là gì?

3. Hàm của bạn xây dựng đã đủ tổng quát chưa hay quá quá chi tiết và chỉ phù hợp cho 1 bài toán cụ thể

4. Bạn gọi hàm có đúng thứ tự tham số mà mình mong muốn hay không, kiểu dữ liệu của tham số hình thức và tham số chính thức có hợp lí hay không?

5. Nếu hàm của khác void, thì hàm sẽ kết thúc ngay lập tức khi gặp câu lệnh return một giá trị nào đó, còn nếu hàm void mà các bạn muốn kết thúc tại thời điểm nào đó các bạn có thể sử dụng câu lệnh return; là đủ.