

HÀM (Function)

NỘI DUNG BUỔI HỌC

ĐỊNH NGHĨA

WHAT?

CÔNG DỤNG

WHY?

CÁCH SỬ DỤNG

HOW?

BÀI TẬP

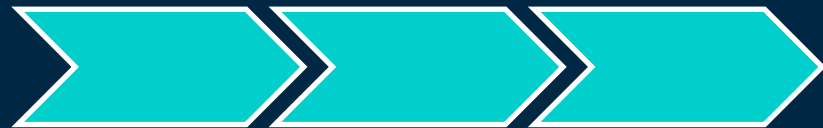
!?

ĐỊNH NGHĨA

WHAT?

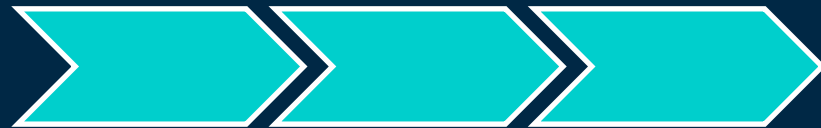
Hàm Toán

1. Nhận vào một số Tham số
2. Trả về kết quả



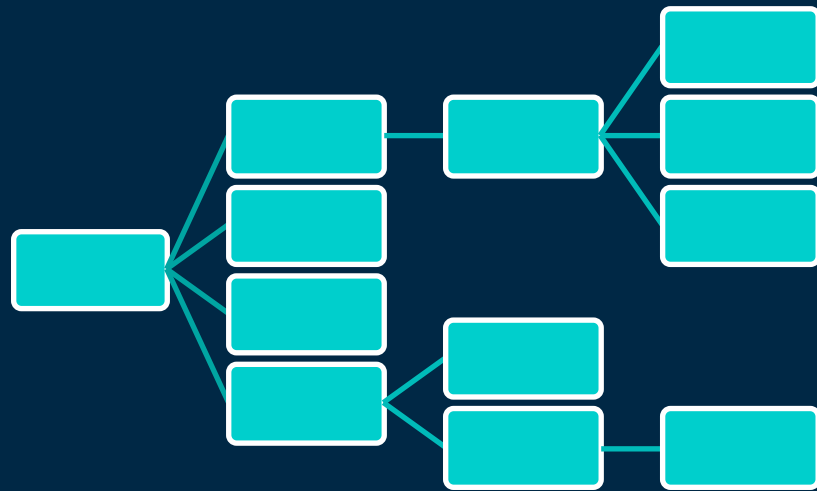
Hàm Toán

1. Nhận vào một số Tham số
2. Trả về kết quả



Hàm Tin

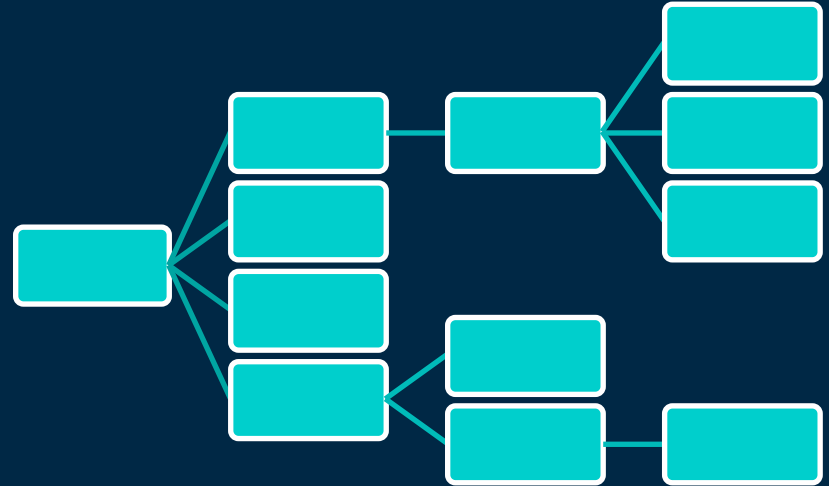
1. Nhận vào tham số (có thể không có)
2. Thực hiện hành động/phép tính
3. Trả về kết quả (số, mảng) (có thể không có)



Hàm Tin

ĐỊNH NGHĨA CHÍNH XÁC:

Nhóm các câu lệnh được thực hiện khi hàm được gọi



The background is a dark blue field with several thin white vertical lines. Small squares in teal, pink, and orange are scattered throughout. A large pink square on the right contains the text 'WHY?'. A horizontal bar at the bottom is composed of a pink segment on the left and an orange segment on the right.

CÔNG DỤNG

WHY?

Tái sử dụng

Khi cần thực hiện chuỗi lệnh nào đó nhiều lần, ở nhiều phần khác nhau trong chương trình

⇒ Giúp tránh lặp lại code

```
1  #include <stdio.h>
2
3  // Kiểm tra xem số a có nguyên tố không
4  bool is_prime(int a){
5      for (int i = 2; i*i <= a; i++){
6          if (a % i == 0) return false;
7      }
8      return true;
9  }
10
11 // In ra tính chẵn/lẻ của a
12 void print_even(int a){
13     if (a % 2 == 0) printf("Even");
14     else printf("Odd");
15 }
16
17 // Tính số Fibonacci thứ i
18 int fibonacci(int n){
19     if (n == 0) return 0;
20     if (n == 1) return 1;
21     return fibonacci(n-1) + fibonacci(n-2);
22 }
```


Tái sử dụng

Khi cần thực hiện chuỗi lệnh nào đó nhiều lần, ở nhiều phần khác nhau trong chương trình

⇒ Giúp tránh lặp lại code

Dễ quản lý

Chia chương trình thành nhiều phần nhỏ ⇒ Dễ kiểm soát, dễ đọc

```
1  #include <stdio.h>
2
3  // Kiểm tra xem số a có nguyên tố không
4  bool is_prime(int a){
5      for (int i = 2; i*i <= a; i++){
6          if (a % i == 0) return false;
7      }
8      return true;
9  }
10
11 // In ra tính chẵn/lẻ của a
12 void print_even(int a){
13     if (a % 2 == 0) printf("Even");
14     else printf("Odd");
15 }
16
17 // Tính số Fibonacci thứ i
18 int fibonacci(int n){
19     if (n == 0) return 0;
20     if (n == 1) return 1;
21     return fibonacci(n-1) + fibonacci(n-2);
22 }
```

CÁCH SỬ DỤNG

HOW?

CẤU TRÚC

The background is a dark blue gradient. It features an abstract pattern of small squares in white, light blue, and light orange, some of which are solid and others are outlines. Thin white vertical lines of varying lengths are scattered across the background, creating a modern, minimalist aesthetic.

CẤU TRÚC

Kiểu dữ liệu trả về

Tên hàm

Các tham số (đầu vào)

Khối lệnh cần thực hiện

```
1  #include <stdio.h>
2
3  // Kiểm tra xem số a có nguyên tố không
4  bool is_prime(int a){
5      for (int i = 2; i*i <= a; i++){
6          if (a % i == 0) return false;
7      }
8      return true;
9  }
10
11 // In ra tính chẵn/lẻ của a
12 void print_even(int a){
13     if (a % 2 == 0) printf("Even");
14     else printf("Odd");
15 }
16
17 // Tính số Fibonacci thứ i
18 int fibonacci(int n){
19     if (n == 0) return 0;
20     if (n == 1) return 1;
21     return fibonacci(n-1) + fibonacci(n-2);
22 }
```

Kiểu dữ liệu trả về

Tên hàm


Các tham số (đầu vào)

Khối
lệnh
cần
thực
hiện

```
3 // kiểm tra xem số a có nguyên tố không
4 bool is_prime(int a){
5     for (int i = 2; i*i <= a; i++){
6         if (a % i == 0) return false;
7     }
8     return true;
9 }
```

Kiểu dữ liệu trả về

Kiểu dữ liệu nhận lại sau khi gọi hàm
(Là **void** nếu hàm không trả về giá trị)



```
3 // kiểm tra xem số a có nguyên tố không
4 bool is_prime(int a){
5     for (int i = 2; i*i <= a; i++){
6         if (a % i == 0) return false;
7     }
8     return true;
9 }
```

Kiểu dữ liệu trả về

Tên hàm


Các tham số (đầu vào)

Khối
lệnh
cần
thực
hiện

```
3 // kiểm tra xem số a có nguyên tố không
4 bool is_prime(int a){
5     for (int i = 2; i*i <= a; i++){
6         if (a % i == 0) return false;
7     }
8     return true;
9 }
```

Tên hàm

- Một tên ngắn gọn, thể hiện mục đích
- Có một số điều kiện đặc biệt như tên biến



```
3 // Kiểm tra xem số a có nguyên tố không
4 bool is_prime(int a){
5     for (int i = 2; i*i <= a; i++){
6         if (a % i == 0) return false;
7     }
8     return true;
9 }
```


Kiểu dữ liệu trả về

Tên hàm

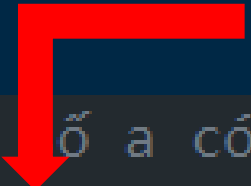
Các tham số (đầu vào)

Khối
lệnh
cần
thực
hiện

```
3 // kiểm tra xem số a có nguyên tố không
4 bool is_prime(int a){
5     for (int i = 2; i*i <= a; i++){
6         if (a % i == 0) return false;
7     }
8     return true;
9 }
```

- Đóng vai trò là các biến đầu vào (input) của hàm.
- Cần kiểu dữ liệu trước tên (như biến).
- Số lượng tùy ý.

Các tham số
(đầu vào)



```
3 // Kiểm tra xem số a có nguyên tố không
4 bool is_prime(int a){
5     for (int i = 2; i*i <= a; i++){
6         if (a % i == 0) return false;
7     }
8     return true;
9 }
```

Kiểu dữ liệu trả về

Tên hàm

Các tham số (đầu vào)

Khối
lệnh
cần
thực
hiện

```
3 // kiểm tra xem số a có nguyên tố không
4 bool is_prime(int a){
5     for (int i = 2; i*i <= a; i++){
6         if (a % i == 0) return false;
7     }
8     return true;
9 }
```

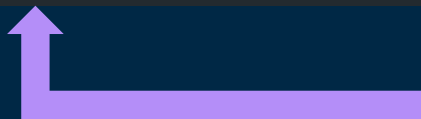
Khối lệnh gồm các câu
lệnh cần thực hiện

Thân hàm

Khối lệnh cần thực hiện



```
5   for (int i = 2; i*i <= a; i++){  
6       |   if (a % i == 0) return false;  
7       |   }  
8       return true;
```



Trả về kết quả (dừng
hàm)

Kiểu dữ liệu trả về

Tên hàm

Các tham số (đầu vào)

Khối
lệnh
cần
thực
hiện

```
3 // kiểm tra xem số a có nguyên tố không
4 bool is_prime(int a){
5     for (int i = 2; i*i <= a; i++){
6         if (a % i == 0) return false;
7     }
8     return true;
9 }
```

CÁCH SỬ DỤNG

CÁCH SỬ DỤNG

```
1  #include <iostream>
2  using namespace std;
3
4  // Hàm có thể lượng tham số tùy ý
5  int max_of_4(int a1, int a2, int a3, int a4){
6      // Ta sử dụng (gọi) hàm max là hàm có sẵn của C/C++
7      int max_a1_a2 = max(a1, a2);
8      int max_a3_a4 = max(a3, a4);
9      // Ta trả về giá trị là max của 2 biến vừa khai báo
10     return max(max_a1_a2, max_a3_a4);
11 }
12
13 int main(){
14     int a = 2, b = 3, c = 1, d = -1;
15     int m = max_of_4(a, b, c, d);
16     printf("%i", m);
17 }
```

CÁCH SỬ DỤNG: KHAI BÁO

Kiểu dữ liệu trả về

Tên hàm

Các tham số (đầu vào)

Thân
hàm

```
4 // Hàm có thể lượng tham số tùy ý
5 int max_of_4(int a1, int a2, int a3, int a4){
6     // Ta sử dụng (gọi) hàm max là hàm có sẵn của C/C++
7     int max_a1_a2 = max(a1, a2);
8     int max_a3_a4 = max(a3, a4);
9     // Ta trả về giá trị là max của 2 biến vừa khai báo
10    return max(max_a1_a2, max_a3_a4);
11 }
```


CÁCH SỬ DỤNG: GỌI HÀM

```
13  int main(){
14      int a = 2, b = 3, c = 1, d = -1;
15      int m = max_of_4(a, b, c, d);
16      printf("%i", m);
17  }
```

1. Tên hàm (max_of_4)
2. Danh sách các tham số, phân tách bằng dấu (,)
3. Gán giá trị trả về của hàm vào biến nào đó (m) (nếu có)

ĐỆ QUY

The background is a dark blue gradient. It features several thin, vertical white lines of varying lengths scattered across the frame. Interspersed among these lines are small squares in three colors: light blue, light orange, and light pink. Some of these squares are solid, while others are outlined. The overall aesthetic is minimalist and modern.

ĐỊNH NGHĨA

Là một hàm tự gọi chính nó

và tự gọi chính nó

và tự gọi chính nó

và tự gọi chính nó

```
4  int giai_thua(int n){
5      if (n==1) return 1;
6      return n * giai_thua(n-1);
7  }
8
9  int fibonacci(int n){
10     if (n == 1 || n == 2) return 1;
11     return fibonacci(n-1) + fibonacci(n-2);
12 }
13
14 int UCLN(int a, int b){
15     if (a == 0) return b;
16     if (b == 0) return a;
17     if (a > b){
18         return UCLN(a-b, b);
19     }
20     else{
21         return UCLN(a, b-a);
22     }
23 }
```

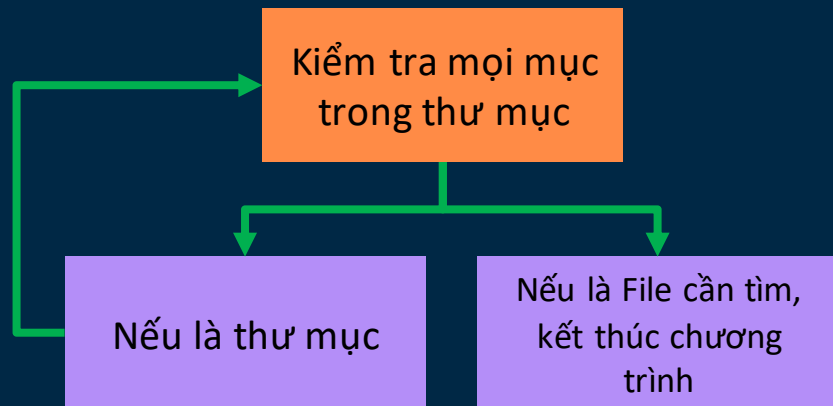
BÀI TOÁN VÍ DỤ

BÀI TOÁN VÍ DỤ

Cho một thư mục (Folder máy tính)

Mỗi thư mục có thể bao gồm các thư mục con, và/hoặc các File

Tìm File (P001.cpp)

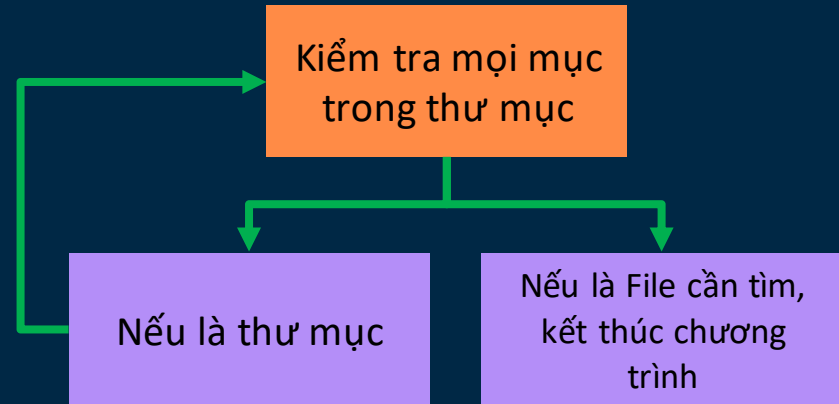


BÀI TOÁN VÍ DỤ

Cho một thư mục (Folder máy tính)

Mỗi thư mục có thể bao gồm các thư mục con, và/hoặc các File

Tìm File (P001.cpp)



```
SEARCH_FILE(current_folder, search_key)
    for item in current_folder:
        if item is file:
            if item.name == search_key:
                exit(); // Thoát chương trình
            else if item is folder: // item là một folder con
                SEARCH_FILE(item, search_key)
```


BÀI TOÁN VÍ DỤ

Cho một thư mục (Folder máy tính)

Mỗi thư mục có thể bao gồm các thư mục con, và/hoặc các File

Tìm File (P001.cpp)





Tài liệu



Sinh



Lý



Codefun



Hoá



Toán





Sinh



Quang
hợp

Slide.pptx

Báo cáo.docx



Hồ hấp

Phối.mp4



Quang
hợp

Slide.pptx

Báo cáo.docx



Hồ hấp



Phởi.mp4



Lý





Codefun

Bài WA

CHD1A.cpp

CHD2A.cpp

CHD3A.cpp

Bài AC

P001.cpp

P002.cpp

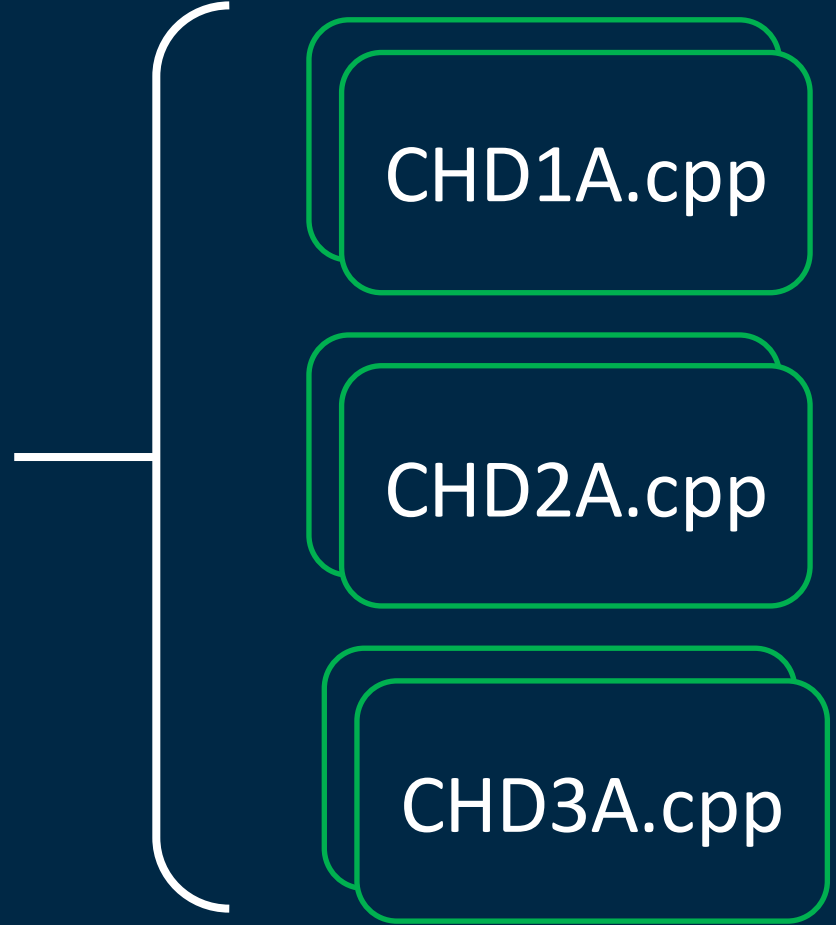
P003.cpp

Bài TLE

P332.cpp

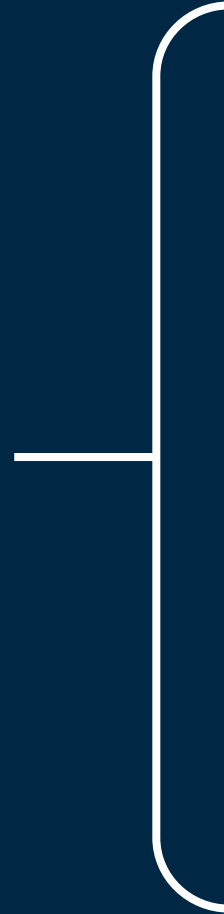


Bài WA





Bài AC



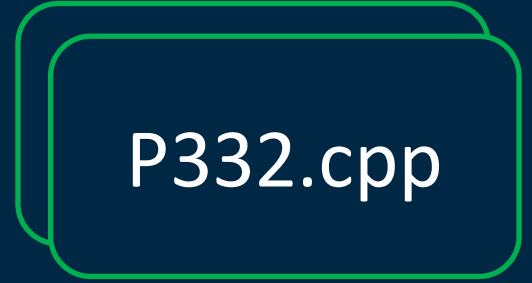
P001.cpp

P002.cpp

P003.cpp



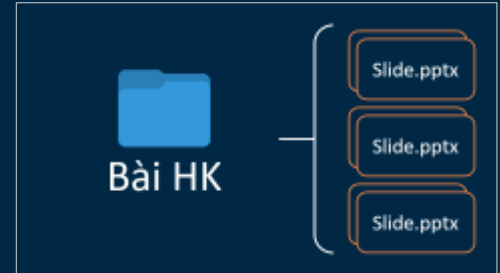
Bài TLE



P332.cpp



Hoá



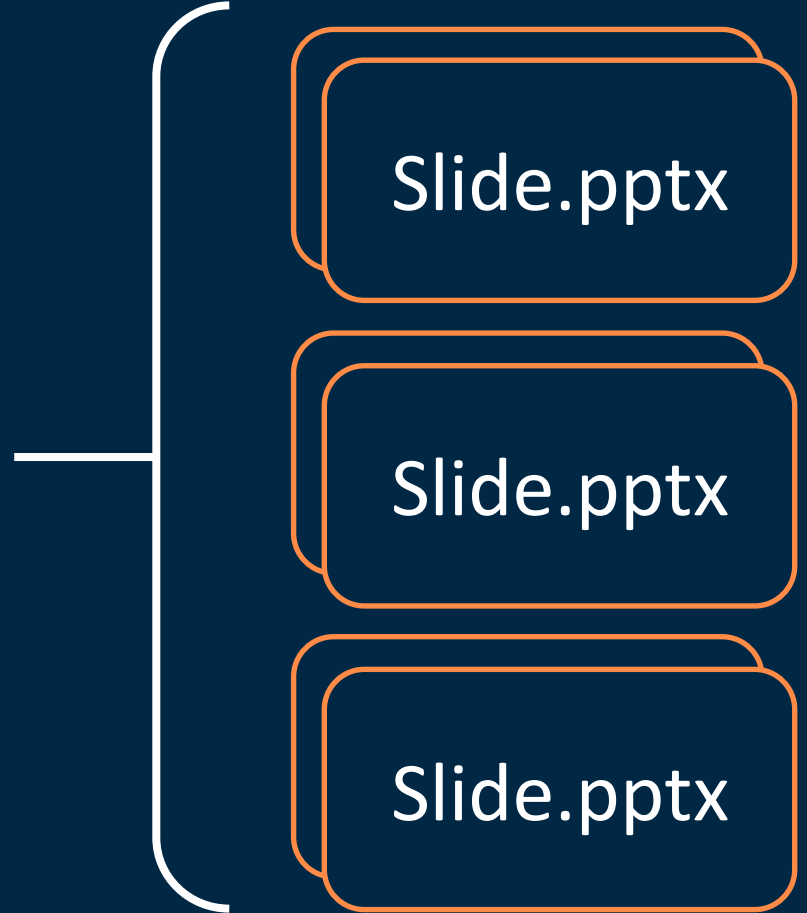


Toán



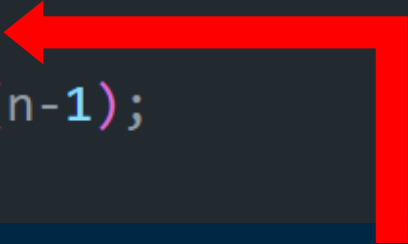


Bài HK



TÍNH DỪNG

```
4  int giai_thua(int n){  
5      if (n==1) return 1;  
6      return n * giai_thua(n-1);  
7  }
```



Mỗi hàm đệ quy đều cần có điều kiện dừng

Với hàm trên, điều kiện dừng là: $n=1$

Bỏ dòng này, hàm sẽ không bao giờ dừng

MỘT SỐ VÍ DỤ

```
25  int sum(int a[], int len){
26      if (len == 0){
27          return 0;
28      }
29      return a[len-1] + sum(a, len-1);
30  }
```

```
4   int giai_thua(int n){
5       if (n==1) return 1;
6       return n * giai_thua(n-1);
7   }
8
9   int fibonacci(int n){
10      if (n == 1 || n == 2) return 1;
11      return fibonacci(n-1) + fibonacci(n-2);
12  }
13
14  int UCLN(int a, int b){
15      if (a == 0) return b;
16      if (b == 0) return a;
17      if (a > b){
18          return UCLN(a-b, b);
19      }
20      else{
21          return UCLN(a, b-a);
22      }
23  }
```

BÀI TẬP

!?