

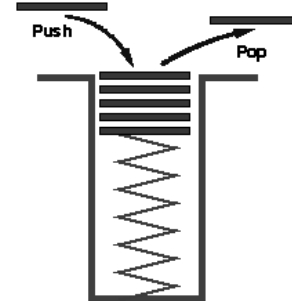
C u trúc d li u và thu t toán

Stacks

Stacks

- Stacks là một dạng danh sách (mảng) có bất biến về hướng như LIFO
- Hai phép toán
 - `int push(Stack s, void *item);`
 - Thêm một phần tử vào đỉnh của stack
 - `void *pop(Stack s);`
 - Lấy một phần tử từ đỉnh của stack
- Thường thì một máy xấp xỉ
- Các phép toán khác

```
int IsEmpty( Stack s );  
/* Return TRUE if empty */  
void *Top( Stack s );  
/* Return the item at the top,  
   without deleting it */
```



Stacks – Thuật toán

- **Mảng (Arrays)**
 - Cung cấp một stack với cách ghi nhớ
 - Hỗ trợ việc mở rộng *nhúng* đáp ứng các nhu cầu động thời gian
 - Ghi nhớ với cách bố trí và ràng buộc
 - Bộ nhớ trong máy tính cá nhân
 - Kích thước máy x p a, etc
- **Phép toán push, pop**
 - Các biến của AddToC..., DeleteFromC...
- **Danh sách liên kết có hướng**
- **Stack:**
 - *Vấn đề là một danh sách liên kết với hướng ngược lại!*

Stacks – m t s v n liên quan

- **Stacks trong ch ng trình tin h c**
 - Là khóa call / return trong functions & procedures
 - Khuôn d ng Stack cho phép các l i g i qui
 - Call: push stack frame
 - Return: pop stack frame
- **Khuôn d ng Stack**
 - Các tham s c a Function
 - Return a ch
 - Các bi n c c b (Local variables)

Ví d : Ph ng th c trao i d li u

- C dùng 1 stack l u tr các bi n c c b và các chuy n các tham s cho hàm v i m i l n g i hàm th c hi n
- ❶ Hàm g i (O) c t các tham s vào stack.
 - ❷ G i th c hi n hàm c g i (F).
 - ❸ F nh n l y các tham s t stack
 - ❹ F t o các bi n c c b ng v i các tham s trên stack
 - ❺ Khi k t thúc, F c p nh t giá tr các tham s (ref) và tr i u khi n cho O
 - ❻ O nh n l y các giá tr m i c a tham s c ng nh giá tr tr v

Ph ng th c trao i d li u

```
#include <stdio.h>
double power(int, int);
int main(void)
{
    int    x = 2;
    double d;
    d = power(x, 5);
    printf("%lf\n", d);
    return 0;
}
double power(int n, int p)
{
    double result = n;
    while(--p > 0)
        result *= n;
    return result;
}
```

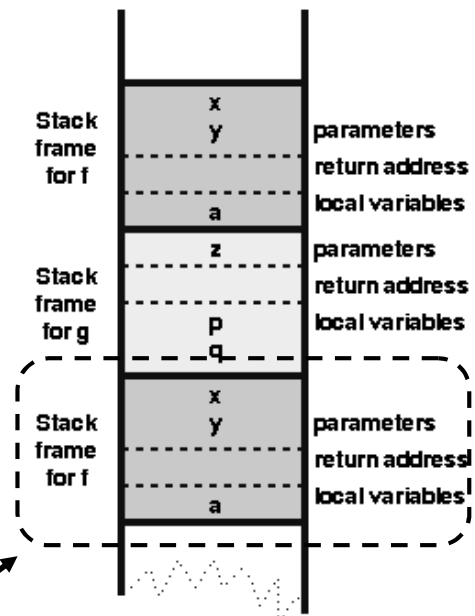
32.0	power: result
2	power: n
5	power: p
32.0	main: d
2	main: x

Stack Frames - Functions in HLL

- Program

```
function f( int x, int y) {  
    int a;  
    if ( term_cond ) return ...;  
    a = ...;  
    return g( a );  
}
```

```
function g( int z ) {  
    int p, q;  
    p = ... ; q = ... ;  
    return f(p,q);  
}
```



Context
for execution of f

qui

- Là công nghệ rất hữu ích
 - nhúng các hàm toán học
 - nhúng cấu trúc dữ liệu
 - Các cấu trúc qui tắc tự nhiên bởi các hàm qui!

qui

- Là công nghệ rất hữu ích
 - nghiên cứu các hàm toán học
 - nghiên cứu cấu trúc dữ liệu
 - Các cấu trúc qui tắc tự nhiên bởi các hàm qui!
- Các hàm nghiên cứu qui
 - factorial
 - Fibonacci
 - GCD bài thuật toán Euclid
 - Bài toán Fourier
 - Trò chơi
 - Tháp Hanoi (Towers of Hanoi)
 - C (Chess)

qui – Ví d

- Dãy s (Fibonacci)

Mã gi
(Pseudo-code)

```
fib( n ) = if ( n = 0 ) then 1  
           else if ( n = 1 ) then 1  
           else fib(n-1) + fib(n-2)
```

C

```
int fib( n ) {  
    if ( n < 2 ) return 1;  
    else return fib(n-1) + fib(n-2);  
}
```

Gi i pháp n gi n, d hi u
(Simple, elegant solution!)

qui – Ví d

- Dãy s Fibonacci

```
C    int fib( n ) {  
        if ( n <= 1 ) return 1;  
        return fib(n-1) + fib(n-2);  
    }
```

Nh ng, trong Fibonacci, th i gian ch y t i!!!!

Tuy nhiên, nhi u hàm qui khác,
VD tìm ki m nh phân, r t n gi n, rõ ràng và *hi u qu* !