Intermediate C Programming

Lesson 9

Recursion

Today’s outline

- Recursion
- Exercise

Recursion

- What is recursion?

- Recursive function: A function that calls itself
- Recursion : The process of calling function itself

```
int quicksort( ... ){
    ....
    quicksort( );
    quicksort( );
}
```

Recursion

- The factorial of n

$$\begin{aligned}
1! &= 1 \\
2! &= 1! \times 2 \\
3! &= 2! \times 3 = (1! \times 2) \times 3 \\
n! &= (n-1)! \times n = (n-2)! \times (n-1) \times n = \cdots
\end{aligned}$$

- Fibonacci

$$\begin{aligned}
f(0) &= 0 \\
f(1) &= 1 \\
f(2) &= f(1) + f(0) = 0 + 1 = 1 \\
f(3) &= f(2) + f(1) = (f(1) + f(0)) + f(1) = 1 + 1 = 2 \\
f(n) &= f(n-1) + f(n-2) = \cdots
\end{aligned}$$

Recursion

- The factorial of n

```c
int factorial( int x ){
    int xx;
    if( x == 0 || x == 1 ){
        printf( "1" );
        return 1;
    } else {
        printf( "%d * (", x );
        xx = x * factorial( x-1 );
        printf( ")" );
        return xx;
    }
}
```

Recursion

- factorial.c

```
#include <stdio.h>
int factorial( int x ){
    int xx;
    if( x == 0 || x == 1 ){
        printf( "1" );
        return 1;
    } else {
        printf( "%d * (", x );
        xx = x * factorial( x-1 );
        printf( ")" );
        return xx;
    }
}
int main(){



         return 0;
}
```

Recursion

- Fibonacci

```
int fibonacci( int x ){
    if( x == 0 ){
        return 0;
    } else if( x == 1 ){
        return 1;
    } else {
        return fibonacci( x-1 )+fibonacci( x-2 );
    }
}
```

Recursion

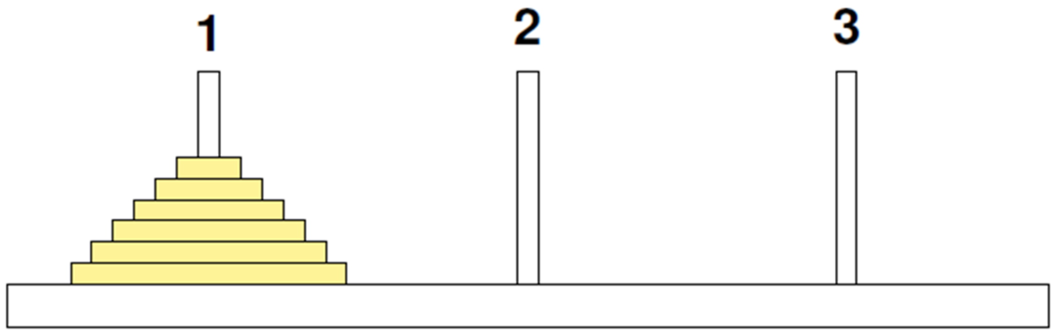
- Fibonacci.c

```
#include <stdio.h>
int fibonacci( int x ){
    if( x == 0 ){
        return 0;
    } else if( x == 1 ){
        return 1;
    } else {
        return fibonacci( x-1 )+fibonacci( x-2 );
    }
}
int main(){
    int i, data;
    printf( ">> " );
    scanf( "%d", &data );



    return 0;
}
```

Recursion

Tower of Hanoi

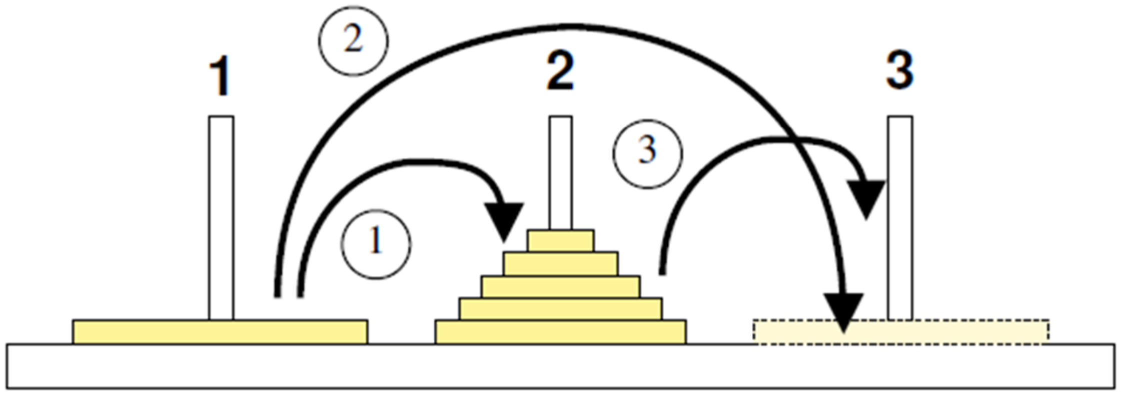
- Only one disk can be moved at a time.
- Each time the moving consists of 1) taking the upper disk from one of the stacks and 2) placing it on top of another stack.
- None of the disks can be placed on top of a smaller disk.



Recursion

Tower of Hanoi (for example:64 disks in total)

```
hanoi( 64, 1, 3, 2 );
```



- Move 63 disks from stack 1 to stack 2 `hanoi( 63, 1, 2, 3 )`
- Move 64'th disk from stack 1 to stack 3
- Move 63 disks from stack 2 to stack 3 `hanoi( 63, 2, 3, 1 )`

Recursion

Tower of Hanoi

```
int steps=0;
void hanoi( int nth, int from, int to, int tmp ){
    if( nth > 0 ){
        hanoi( nth-1, from, tmp, to );
        printf( "disk(%d): %d -> %d\n", nth, from, to );
        steps++;
        hanoi( nth-1, tmp, to, from );
    }
}
```

Exercise

Finish the program of the Tower of Hanoi