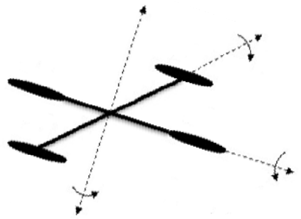
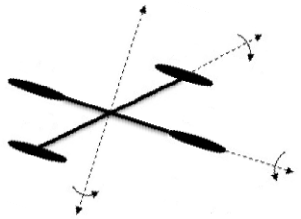
재귀 호출



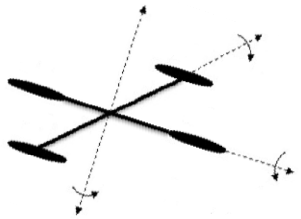
- Factorial 계산

```
factorial(0) = 1;
factorial(n) = n*factorial(n-1);
```

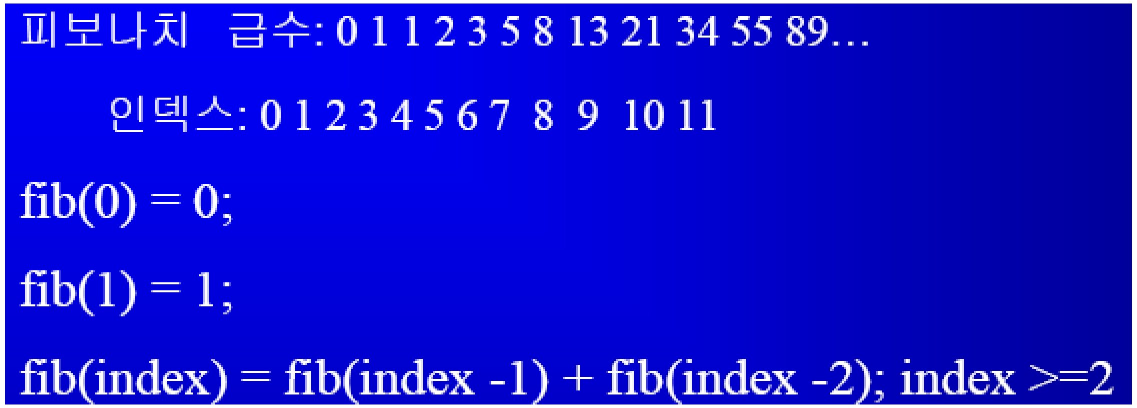
```
factorial(3) = 3 * factorial(2)
             = 3 * (2 * factorial(1))
             = 3 * ( 2 * (1 * factorial(0)))
             = 3 * ( 2 * ( 1 * 1)))
             = 3 * ( 2 * 1)
             = 3 * 2
             = 6
```



```
#include <iostream>
using namespace std;

// Return the factorial for a specified index
int factorial(int);

int main()
{
  // Prompt the user to enter an integer
  cout << "Please enter a non-negative integer: ";
  int n;
  cin >> n;

  // Display factorial
  cout << "Factorial of " << n << " is " << factorial(n);

  system("pause");
  return EXIT_SUCCESS;
}

// Return the factorial for a specified index
int factorial(int n)
{
  if (n == 0) // Base case
    return 1;
  else
    return n * factorial(n - 1); // Recursive call
}
```



- 피보나치 수열



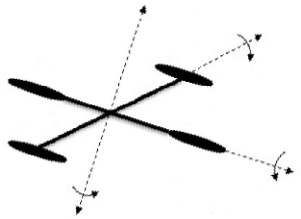
피보나치 급수: 0 1 1 2 3 5 8 13 21 34 55 89…

인덱스: 0 1 2 3 4 5 6 7 8 9 10 11

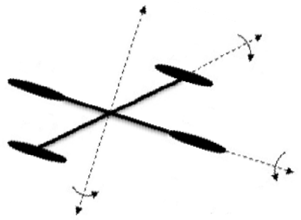
fib(0) = 0;

fib(1) = 1;

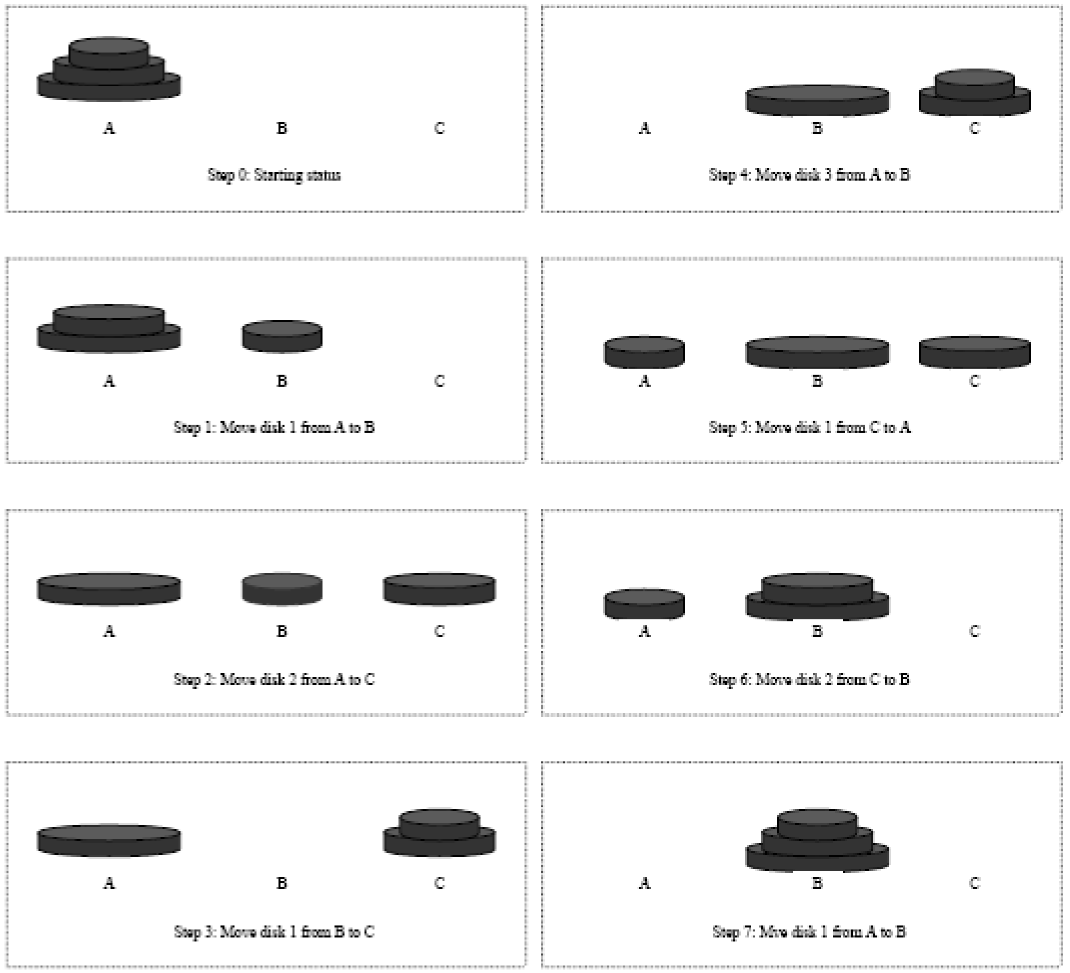
fib(index) = fib(index -1) + fib(index -2); index >=2



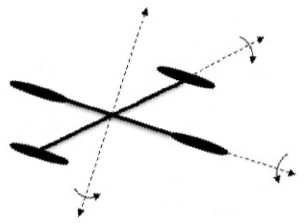
```
#include <iostream>
using namespace std;

// The function for finding the Fibonacci number
int fib(int);

int main()
{
  // Prompt the user to enter an integer
  cout <<  "Enter an index for the Fibonacci number: ";
  int index;
  cin >> index;

  // Display factorial
  cout << "Fibonacci number at index " << index << " is “ << fib(index) << endl;

  system("pause");
  return EXIT_SUCCESS;
}

// The function for finding the Fibonacci number
int fib(int index)
{
  if (index == 0) // Base case
    return 0;
  else if (index == 1) // Base case
    return 1;
  else // Reduction and recursive calls
    return fib(index - 1) + fib(index - 2);
}
```



- 하노이 탑



A
B
C
Step 0: Starting status
Step 1: Move disk 1 from A to B
Step 2: Move disk 2 from A to C
Step 3: Move disk 1 from B to C
Step 4: Move disk 3 from A to B
Step 5: Move disk 1 from C to A
Step 6: Move disk 2 from C to B
Step 7: Mve disk 1 from A to B



```
#include <iostream>
using namespace std;

/* The function for finding the solution to move n disks
    from fromTower to toTower with auxTower */
void moveDisks(int n, char fromTower,
    char toTower, char auxTower)
{
  if (n == 1) // Stopping condition
    cout << "Move disk " << n << " from " << fromTower << " to " << toTower << endl;
  else
  {
    moveDisks(n - 1, fromTower, auxTower, toTower);
    cout << "Move disk " << n << " from " << fromTower << " to " << toTower << endl;
    moveDisks(n - 1, auxTower, toTower, fromTower);
  }
}

int main()
{
  // Read number of disks, n
  cout << "Enter number of disks: ";
  int n;
  cin >> n;

  // Find the solution recursively
  cout << "The moves are: " << endl;
  moveDisks(n, 'A', 'B', 'C');

  system("pause");
  return EXIT_SUCCESS;
}
```