Sum of AP (n) = 1+2+…+n

遞歸
Recursion

```
int sumofAP (int n){
        int k, sum=0;


        return sum;
}
```

```
int sumofAP (int n){
    if (n<=0)
         return
    else
         return
}
```

```
n = sumofAP(4);
printf ("sum = %i\n", n);
```

```
sumofAP(4) = 4+3+2+1
sumofAP(3) =   3+2+1
sumofAP(2) =     2+1
sumofAP(1) =       1
sumofAP(0) = 0
```

```
=   +sumofAP(  )
=   +sumofAP(  )
=   +sumofAP(  )
=   +sumofAP(  )
=0
```

```
int fibonacci (int n){
        int i, a=1,b=1,c;
        if (n<=2) return (1);





        return (c);
}
```

1,2,3,4,5, 6,7

1,1,2,3,5, 8,13,...

a,b,c

a,b,c

每個數c
都是前兩數之和(a+b)

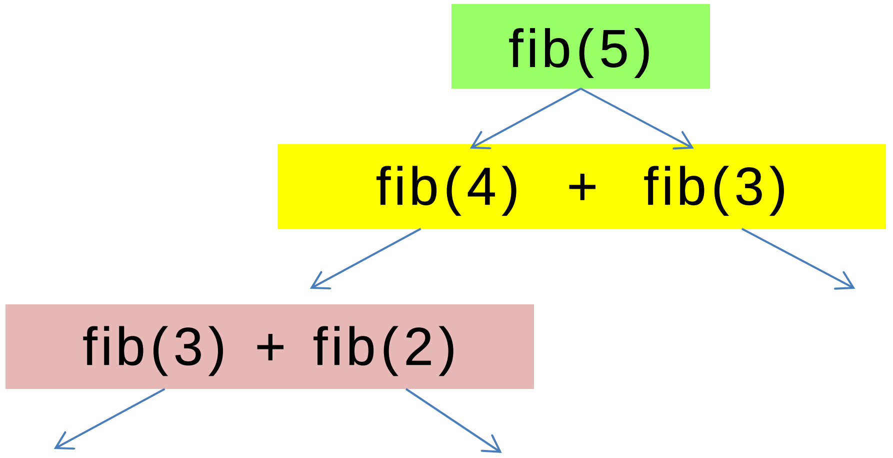
fibonacci(n)
= fibonacci(   ) + fibonacci(   )

Recursion

1,2,3,4,5, 6,7

```
int fibonacci (int n){
        if (                )
                return (        );
        else
                return (                                        );
}
```

1,1,2,3,5, 8,13,...



```
a          b          c
234   %   144   =   90
          144   %   90   =   54
                    90   %   54   =   36
                             54   %   36   =   18
                                      36   %   18   =   0
                                      a          b          c
```

HCF = Highest Common Factor
GCD = Greatest Common Divisor

```
             234   144
int gcd (int a, int b){
        int c=9;
        while (c>0){



        }
        return (b);
}
```

Recursion

```
int gcd (int a, int b){
        if (               )
                return (b);
        else
                return (                    );
}
```

(n!) Factorial   5! = 5×4×3×2×1

5

```
long factorial (int n){
        int i;
        long result=1;



        return (result);
}
```

```
   factorial(5)
= 5 * factorial(4)
= 5 * 4 * factorial(3)
= 5 * 4 * 3 * factorial(2)
= 5 * 4 * 3 * 2 * factorial(1)
= 5 * 4 * 3 * 2 * 1
```

Recursion

```
long factorial (int n){
        if (              )
                return 1;
        else
                return (                              );
}
```

2 10

```
double power (double x, int n){
          int i, result=1;


          return (result);
}
```

t = power(2,10);

$= 2\times2\times2\times2\times2\times2\times2\times2\times2\times2$

$2^{10} = 1024$

$2^{-5} = \frac{1}{2^5}$

Recursion

```
double power (double x, int n){
          if (n==0)
                    return
          else if(n>0)
                    return
          else
                    return
}
```

```
void toBin (int n){






}
```

toBin(12)
→ 1100
toBin(65)
→ 100 0001

toHex(58)
→ 4A

```
void toHex (int n){
        char hexchar[] = "0123456789ABCDEF";







}
```

```
void reverseChar(){
        char ch;








}
```

輸入
reverse
→ esrever

toWord(123)
→ One Two Three

```
void toWord (int n){
        char *wd[ ] = {
                "Zero", "One", "Two", "Three", "Four",
                "Five", "Six", "Seven", "Eight", "Nine" };







}
```