

написать рекурсивное вычисление чисел фибоначчи

1 1 2 3 5 8 13 21
1 2 3 4 5 6 7

fib(1)=1
fib(2)=1
fib(3)=2

....
fib(n)=fib(n-1)+fib(n-2)

inf fib(int n)//7

1. Какие значения мы вычисляем
2. Какое рекурсивное соотношение
3. Какие начальные значения
4. В каком порядке вычисляются значения
5. Где искать ответ

**БЫСТРОДЕЙСТВИЕ
O(n)**

```
fibarr[1000];
fibarr[1]=1;
fibarr[2]=1;
for(i=3;i<n;i++)
{
    fibarr[i]=fibarr[i-1]+fibarr[i-2];
}

cout<<fibarr[50]<<endl;
```

```
int main()
{
    clock_t start,finish;

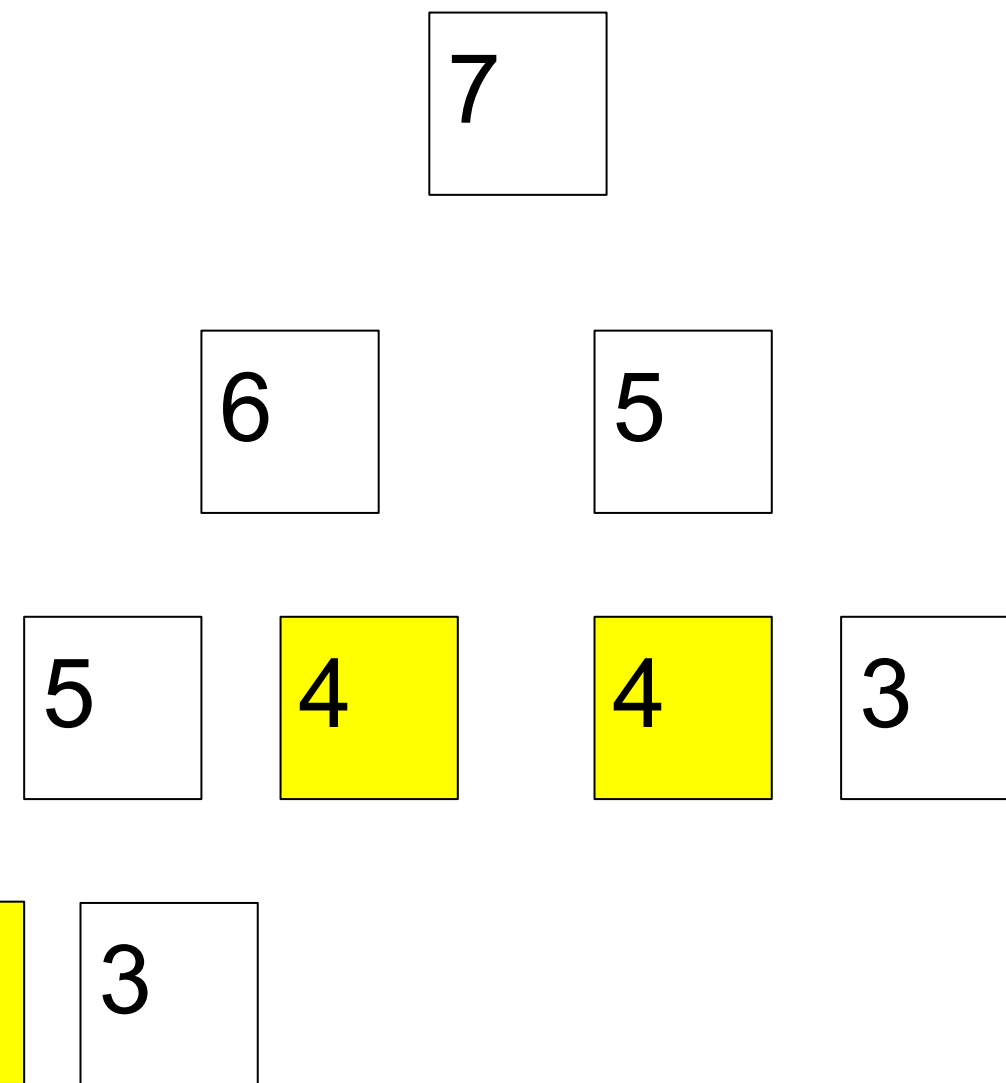
    int Num;
    cin>>Num;
    start=clock();
    long long int fibarr[1000];
    fibarr[1]=1;
    fibarr[2]=1;
    for(int i=3;i<=Num;i++)
    {
        fibarr[i]=fibarr[i-1]+fibarr[i-2];
    }

    cout<<fibarr[Num]<<endl;
    finish=clock();
    int result;
    //result=fib2(Num);

    cout<<(finish-start)<<endl;
}
```

1134903170

global_data=[];



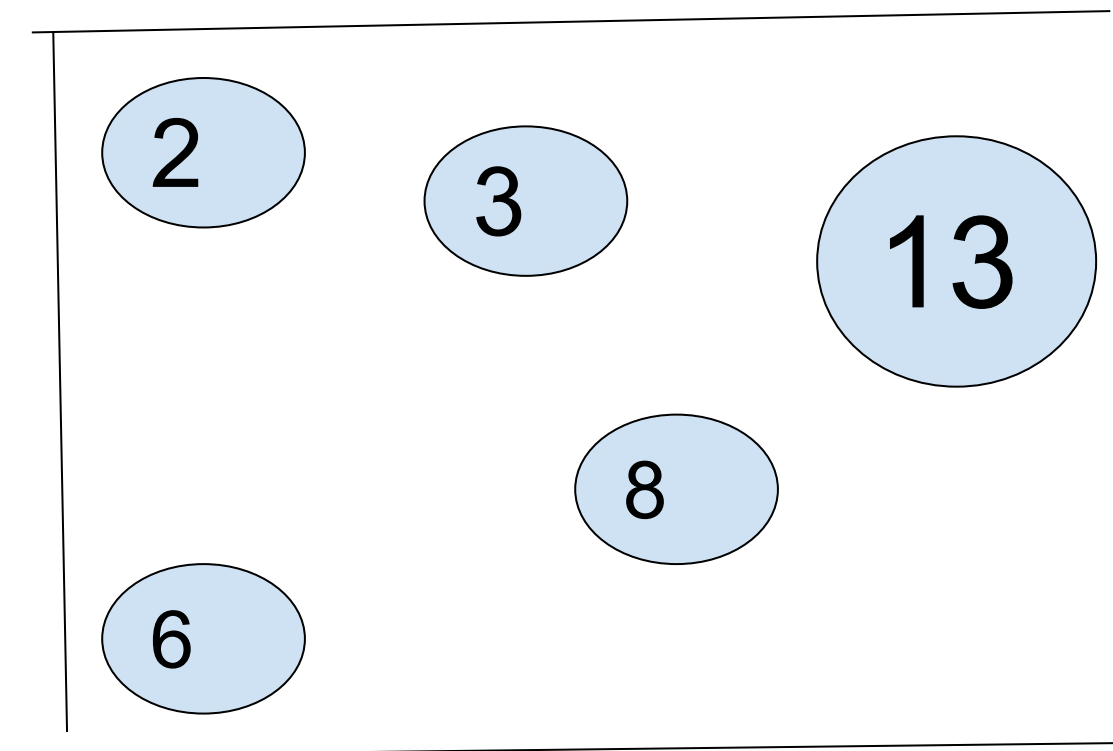
```
#include <iostream>
#include <ctime>
using namespace std;

int fib(int n)
{
    if(n>0)
    {
        if(n==1 || n==2)
        {
            return 1;
        }
        else
        {
            return (fib(n-1)+fib(n-2));
        }
    }
    else
    {
        return 0;
    }
}
```

```
int main()
{
    int Num;
    cin>>Num;
    int result;
    clock_t start,finish;
    start=clock();
    result=fib(Num);
    finish=clock();
    cout<<result<<endl<<(double)(finish-start)/CLOCKS_PER_SEC<<endl;
}
```

//gcc -lstdc++ din1.cpp -o din1.out
//./din1.out

**БЫСТРОДЕЙСТВИЕ
O(1.5^n)**



динамическое
программирование -
МОЖНО ВЫЧИСЛЯТЬ 1
раз

рекуррентная
формула

```
#include <iostream>
#include <ctime>
#include <vector>
#include <map>
using namespace std;
int contains(int value, vector<int> vec)
{
    for(int i = 0; i < vec.size(); i++)
    {
        if(vec[i] == value)
        {
            return i;
        }
    }

    return -1;
}

int contains(int position, map<int,int> mp)
{
    map<int, int> :: iterator it = mp.begin();
    for(;it != mp.end(); it++)
    {
        if(it->first == position)
        {
            return it->first;
        }
    }

    return -1;
}

void print_fiba(map<int,int> &mp)
{
    map<int, int> :: iterator it = mp.begin();
    for(;it != mp.end(); it++)
    {
        cout<<it->first<<" "<<it->second<<" "<<endl;
    }
    cout<<endl;
}

void ps(int n)
{
    for(int i=0;i<n;i++)
    {
        cout<<" ";
    }
}

//int fib(int n,vector<int> &fiba)
int fib(int n, map<int,int> &fiba, int parent)
{
    print_fiba(fiba);
    int position=contains(n,fiba);
    ps(4-parent+1);
    cout<<"n="<<n<<" "<<"position="<<position<<endl;
    if(position>=0)
    {
        return fiba[position];
    }
    else
    {
        if(n>0)
        {
            if(n==1 || n==2)
            {
                fiba[n]=1;
                return 1;
            }
            else
            {
                fiba[n]=(fib(n-1,fiba,n)+fib(n-2,fiba,n));
                return fiba[n];
            }
        }
        else
        {
            fiba[n]=0;
            return 0;
        }
    }
}

int main()
{
    //vector<int> fiba;
    map<int,int> fiba;
    int Num;
    cin>>Num;
    int result;
    clock_t start,finish;
    start=clock();
    result=fib(Num,fiba,Num+1);
    print_fiba(fiba);
    finish=clock();
    cout<<"result="<<result<<endl<<(double)(finish-start)/CLOCKS_PER_SEC<<endl;
}
```