题目 斐波那契数列

考点 递归和循环 热点指数 99449 通过率 29.53%

具体题目

大家都知道斐波那契数列，现在要求输入一个整数n，请你输出斐波那契数列的第n项（从0开始，第0项为0）。

```
/*
整体思路：考虑负数，大数，算法的复杂度，空间的浪费
*/
public class Solution {
    public int Fibonacci(int n) {
        //方法1：用递归，系统会让一个超大的n来让Stack Overflow，所以
        //递归就不考虑了

        //使用迭代法，用fn1和fn2保存计算过程中的结果，并复用起来
        int fn1 = 1;
        int fn2 = 1;

        //考虑出错情况
        if (n <= 0) {
            return 0;
        }
        //第一和第二个数直接返回
        if (n == 1 || n == 2) {
            return 1;
        }
        //当n>=3时，走这里，用迭代法算出结果
        //这里也说明了，要用三个数操作的情况，其实也可以简化为两
        //个数，从而节省内存空间
        while (n-- > 2) {
            fn1 += fn2;
            fn2 = fn1 - fn2;
        }
        return fn1;
    }
}


//就记录前面计算的n-1和n-2的值嘛
public class Solution {
   public static int Fibonacci(int n) {
         if (n <= 1)
             return n;
         int res = 0;
         int n1 = 0;
         int n2 = 1;
         for (int i=2; i<=n; i++){
             res = (n1 + n2);
             n1 = n2;
             n2 = res;
         }
        return res;
     }
}


  /*
```

```
  *方法一：递归，不考虑，有大量的重复计算，会导致内存溢出
  /*
  public class Solution {
      public int Fibonacci(int n) {
          if(n<=0) {
              return 0;
          }
if(n==1) {
              return 1;
          }
          return Fibonacci(n-2)+Fibonacci(n-1);
      }
  }
  */
  /*
  *方法二：使用迭代法，用fn1和fn2保存计算过程中的结果，并复用起来
  */
  public class Solution {
      public int Fibonacci(int n) {
          int fn1 = 1;
          int fn2 = 1;
          if(n <= 0 ) {
              return 0;
          }
          if(n==1 || n==2) {
              return 1;
          }
          while(n>2) {
              fn1 += fn2;
              fn2 = fn1-fn2;
              n--;
          }
          return fn1;
      }
  }
```