天津大学

数据结构上机实验报告

题目:递归和非递归实现 **Fibanacci** 数列

学生姓名 王雨朦

学生学号 **2016229082**

学院名称 国际工程师学院

专业 计算机

时间 **2016/10/10**

目　　录

第一章　需求分析

1.1 原题表述

fibonacci 数列指的是这样一个数列：0,1,1,2,3,5,8,13,21,34,55,89,144……,特别指出：第 0 项是 0，第 1 项是第一个 1，这个数列从第 2 项开始，每一项都等于前两项之和。分别利用递归的方法和非递归的方法求 fibonacci 数列，并比较递归算法和非递归算法的不同。

1.2 解决方案

1） 用递归算法实现 fibonacci 数列；
2） 用非递归算法实现 fibonacci 数列；
3） 比较递归和非递归的优缺点。

第二章　概要设计

2.1 抽象数据类型

Fibonacci 数列递归和非递归算法:
unsigned long fib(int n)//递归算法
unsigned long fib_b(int n)//非递归

2.2 算法

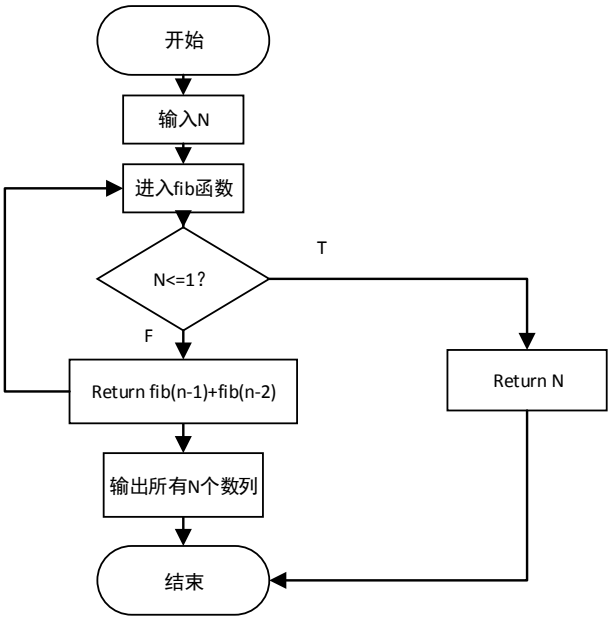

Figure 2-1 递归算法

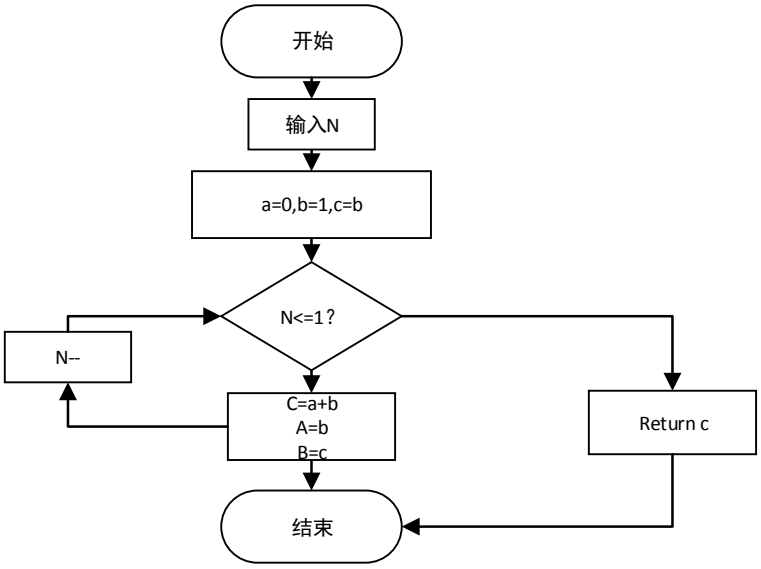

Figure 2-2 非递归算法

第三章　详细设计

3.1 程序代码

```
#include <iostream>
#include <cstdio>
#include <cstdlib>
using namespace std;
//递归
unsigned long fib(int n){
    if(n<=1) return n;
    else
        return fib(n-1)+fib(n-2);
}
unsigned long fib_b(int n){
    unsigned long a=0;
    unsigned long b=1,c;
    if(n<=1) return n;
    else{
        for(int i=2;i<=n;i++){
            c=a+b;
            a=b;
            b=c;
        }
        return c;
    }
}
int main(){
    int n;
    printf("您想输出斐波那契数列的前几个数？\n");
    scanf("%d",&n);
    printf("递归输出前%d 个数为：\n",n);
    for(int i =0; i<n; i++){
        cout<<fib(i)<<" ";
    }
    cout<<endl;

    printf("非递归输出前%d 个数为：\n",n);
    for(int i =0; i<n; i++){
        cout<<fib_b(i)<<" ";
    }
    cout<<endl;}
```

第四章　调试分析

4.1 调试过程

Bug 名称	循环中上界应取闭区间
Bug 描述	非递归算法中，计算 c=a+b 时应计算到第 N 次才是第 n 个 fibonacci 数列中的数
Bug 原因	思绪没有理清楚
Bug 解决方案	将 i<N 改为 i<=n
Bug 总结	上下边界值，思考不清楚

Bug 名称	无法输出数列中的数
Bug 描述	递归算法中，想在计算第 n 个 fib 值的时候把它前面的递归计算到的值都顺便输出
Bug 原因	没有一个确定的条件能判断每次递归的时候已经计算出来值了并且将其输出。
Bug 解决方案	计算 n 次 fib 的值，一一输出
Bug 总结	或许可以实现开销更小的输出值得算法

第五章　测试结果

5.1 测试过程

测试编号	1
测试对象	递归计算
测试输入参数	N=20
测试步骤	查看递归算法结果
测试预期结果	0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181
测试输出结果	0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181
测试分析	程序正确

```
您想输出斐波那契数列的前几个数?
20
递归输出前20个数为:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181
```

测试编号	2
测试对象	非递归计算
测试输入参数	N=22
测试步骤	输出递归和非递归计算出来的值看是不是一样
测试预期结果	0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946
测试输出结果	0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946
测试分析	程序正确

```
您想输出斐波那契数列的前几个数?
22
递归输出前22个数为:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946
非递归输出前22个数为:
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181 6765 10946

Process returned 0 (0x0)   execution time : 1.340 s
Press any key to continue.
```

第六章　未来展望与思考

6.1 比较递归和非递归算法的性能

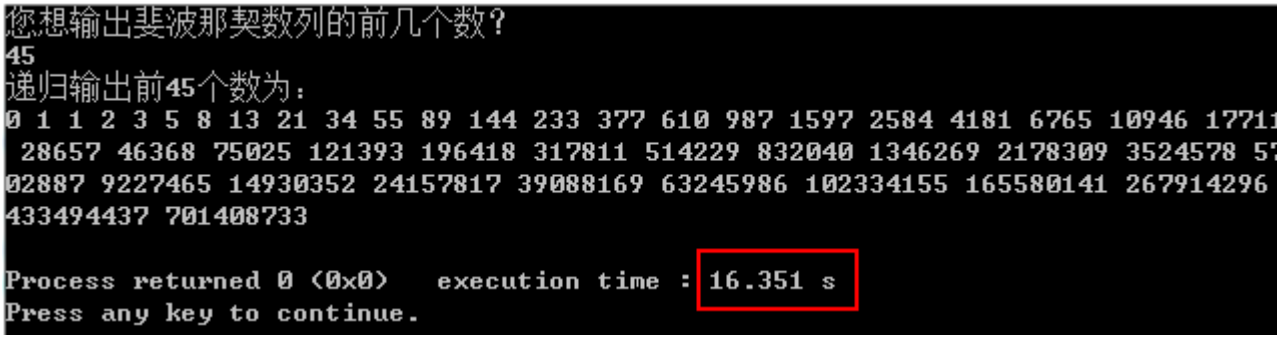

Figure6- 1 递归算法的计算时间

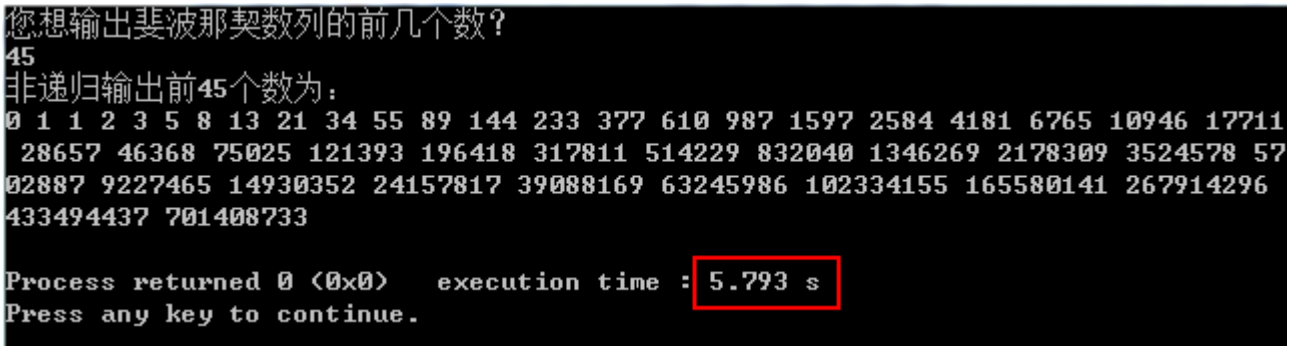

Figure6- 2 非递归算法的计算时间

对比以上两图，可知：当计算量小时，两种算法的性能差别不大。
但是稍微增加一点计算量的时候，递归算法的空间消耗就能够很明显的提现出来。

6.2 思考递归的优劣性

递归好处：代码更简洁清晰，可读性更好。

递归坏处：由于递归需要系统堆栈，所以空间消耗要比非递归代码要大很多。而且，如果递归深度太大，可能系统撑不住。

总之，调用函数要消耗资源，所以递归空间和时间消耗都大，在计算能力和空间有限的背景下，尽量不要使用递归算法，或多将递归算法改写为非递归算法。