**数据结构实验报告**

姓名：郭宇瑶 学号：U201717031 班级：软工1704班

**实验一 求整数和、切pizza和Hanoi塔等问题的求解**

一、实验描述

用C语言编程实现求整数和，切pizza以及Hanoi塔等问题的求解，在程序中加入clock ()来计算求解时间，使用不同的输入值得到对应的时间值。分析算法的时间复杂度并与测量结果进行比较，如果存在差异，解释原因。

问题1——求整数和：输入一个正整数n，求1~n的所有正整数之和，即1+2+···+n。

问题2——切披萨：一块披萨（无厚度）切n刀，最多可以切成几块？输入正整数n，返回最多切成的块数。

问题3——Hanoi塔：把圆盘从下面开始按大小顺序重新摆放在另一根柱子上。移动的规则如下：1. 每次只能从一个柱子的最上面移动一个碟子到另外一个柱子上。2. 不能将大碟子放到小碟子的上面。

二、实验设计

1.求整数和问题：利用递归来计算n项和的值，输入10组大小分别为100,200,300,400,500,600，700,800，900，1000的n；

2.切pizza问题：利用递归的方式来计算切n刀后的pizza块数，输入10组大小分别为100,200,300,400,500,600，700,800，900，1000的n；

3.Hanoi塔问题：利用递归来求解Hanoi问题，输入大小为10至20的n；

4，定义两个时间类型变量start和end，分别用于记录一个函数执行前后的时间；

5，对于求整数和与切pizza问题，记录程序执行10000次所需的时间；对于Hanoi塔问题，记录程序执行100次所需的时间；分别用合适的Excel图表对它们进行描述。

三、实验实现过程

1.整数求和问题：

（1）定义int型变量n，把n作为求和函数的参数；

（2）定义time\_t类型的变量start并调用clock（）函数来记录函数开始执行时间；

（3）定义方法int sum(int n) //利用递归方法求解；

（4）定义time\_t类型的变量end并调用clock（）函数来记录函数结束时间。

（5）将求值函数体循环10000次，在函数执行前，加上代码： start=clock(); 函数执行后，加上代码：end=clock()；

（6）打印（int）(end-start)；

（7）重复5 次实验，得到平均数据；

（8）记录相应10组实验结果并绘制相应Excel图表。

2.切pizza问题：

（1）定义int型变量n，把n作为计算块数函数的参数；

（2）定义time\_t类型的变量start并调用clock（）函数来记录函数开始执行时间；

（3）定义方法sum(int n) //利用循环来求和；

（4）定义time\_t类型的变量end并调用clock（）函数来记录函数结束执行时间；

（5）将求值函数体循环10000次，在函数执行前，加上代码： start=clock(); 函数执行后，加上代码：end=clock()；

（6）打印（int）(start-end)；

（7）重复5 次实验，得到平均数据；

（8）记录相应10组实验结果并绘制相应Excel图表。

3.Hanoi塔问题：

（1）定义int型变量n作为Hanoi塔的层数，n取10-20；

（2）定义time\_t类型的变量start并调用clock（）函数来记录开始时间；

（3）定义方法void hanoiTower (int n, char a, charb, char c)，利用递归来求解问题，并将方法循环执行100次；

（4）定义clock\_t类型的变量end并调用clock（）函数来记录函数结束执行时间；

（5）打印（int）(end-start)；

（6）重复5 次实验，得到平均数据；

（7）记录相应10组数据并求2的对数，绘制出相应Excel图表。

四、实验数据

1. 求整数和问题图表

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 整数值 | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | 1000 |
| 时间平均值（clock） | 23 | 50.2 | 92 | 127 | 150.6 | 188 | 208.2 | 227.2 | 258.6 | 272.6 |

1. 切pizza问题图表

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 切的次数 | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | 1000 |
| 时间平均值(clock) | 30.2 | 55.2 | 81.4 | 103.8 | 120.6 | 146.2 | 170 | 192.2 | 208.6 | 227 |

1. Hanoi塔问题图表

|  |  |  |
| --- | --- | --- |
| n | 平均时间(clock) | 以2为底时间的对数 |
| 10 | 1.8 | 0.847996907 |
| 11 | 4.2 | 2.070389328 |
| 12 | 8.8 | 3.137503524 |
| 13 | 18 | 4.169925001 |
| 14 | 34.4 | 5.10433666 |
| 15 | 71.8 | 6.165911939 |
| 16 | 127.6 | 6.995484519 |
| 17 | 247.8 | 7.953032377 |
| 18 | 548.2 | 9.098558518 |
| 20 | 2027.2 | 10.98527271 |

五、实验结论

1，算法时间复杂度分析

对于用递归方法求n项整数和与用迭代方法求切pizza所得的块数，它们的时间复杂度T(N)都等于O(N)；Hanoi塔问题使用递归方法，有T(N)=2T(N-1)+1， T(1)=1消去系数和常量得T(N)=O(2^N)；

2，与测量结果进行比对

求整数和问题与切pizza问题最后得到的结果曲线均为一条直线，n与时间t成线性关系，与算法时间复杂度分析所得的T(N)=O(N)吻合；Hanoi塔问题最后得到的结果曲线大致呈现出线性增长，与前面的分析吻合，曲线存在波动可能与CPU的内存分配与运行速度有关（n增大，2^n迅速增大，递归占用内存也迅速增加）。

六、代码

1.求整数和

/\*\*

\* 求整数和问题，递归算法

\* @param n 所给整数上限

\* @return 和

\*/

int sum(int n)

{

if(0==n)

return 0;

return sum(n-1)+n;

}

1. 切披萨问题

/\*\*

\* 切披萨问题，递归算法

\* @param n 刀数

\* @return 块数

\*/

int numPcs(int n){

if(n ==1)

return 2;

return numPcs(n-1)+n;

}

1. Hanio塔问题

void hanoiTower (int n, char a, char b, char c)

{

if (1 == n)

{

printf ("%c -> %c\n", a, c);

return;

}

hanoiTower (n-1, a, c, b);

hanoiTower (n-1, b, a, c);

return;

}