

作业 8

必做题

第一题

主测试程序：

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#define REPETITION 600000
int ditui(void);
char digui(void);
int main() {
    int iRoundMain; /*主函数重复运行程序循环变量*/
    int iTimeDitui=0; /*递推用时*/
    int iTimeDigui=0; /*递归用时*/
    int iTimeNull=clock(); /*基准值*/
    for (iRoundMain=0; iRoundMain<REPETITION; iRoundMain++) ditui(); /*递推程序执行*/
    iTimeDitui=(clock()-iTimeNull);
    printf("递推得到的结果: %d, %d次计算用时%d\n", ditui(), REPETITION, iTimeDitui); /*递推结果输出*/
    iTimeNull=clock(); /*时间归零*/
    for (iRoundMain=0; iRoundMain<REPETITION; iRoundMain++) digui(); /*递归程序执行*/
    iTimeDigui=(clock()-iTimeNull);
    printf("递归得到的结果: %d, %d次计算用时%d\n", digui(), REPETITION, iTimeDigui); /*递归结果输出*/
    system("pause");
    return '\0';
}
```

递推程序：

```
int ditui() { /*递推主函数*/
    int iRound=0; /*数组位置循环变量*/
    char cOrder=1; /*报数结果*/
    char cAlive[40]; /*每个人的生存状态, 1活, 0死*/
    short sAliveNumber=40; /*活着的人数*/
    for (iRound=0; iRound<=39; iRound++) { /*初值为所有人都活着*/
        cAlive[iRound]=1;
    }
    while (sAliveNumber>1) { /*循环至只剩一人*/
        for (iRound=0; iRound<=39; iRound++) { /*遍历各个人, 不论死活*/
            if (cAlive[iRound]==1) {
                if (cOrder==3) { /*报数为3杀死*/
                    cAlive[iRound]=0;
                    cOrder=1;
                }
            }
        }
        sAliveNumber=0;
        for (iRound=0; iRound<=39; iRound++)
            if (cAlive[iRound]) sAliveNumber++;
        cOrder++;
    }
}
```

```

        sAliveNumber--;
    }
    else cOrder++;/*报数为1、2继续存活*/
}
}
}
for (iRound=0;cAlive[iRound]==0;iRound++){continue;}/*找到活着的序号*/
return iRound+1;
}
递归程序：
char digui() /*递归主函数*/
{
    char cNumber[500];
    int fReset(char cN[]);
    char fKill(char cN[], short, short);
    fReset(cNumber);
    return fKill(cNumber, 0, 40);
}
int fReset(char cN[]) /*这个函数用来把1~40填充入数组
char cRoundReset=0;
for (;cRoundReset<=39;cRoundReset++){
    cN[cRoundReset]=cRoundReset+1;
}
return 0;
}
char fKill(char cN[], short sCheckPosition, short sAddPosition) /*这个函数表示每
一次报数1~3
//CheckPosition:本次该报数的人的序号所在数组位置
//AddPosition:本次报数该添加的数组位置
if (sAddPosition-sCheckPosition>=2) {
    cN[sAddPosition]=cN[sCheckPosition];/*报数为1下一轮还活着*/
    cN[sAddPosition+1]=cN[sCheckPosition+1];/*报数为2下一轮还活着*/
    return fKill(cN, sCheckPosition+3, sAddPosition+2);/*报数为3下一轮被跳过
*/
}
else return cN[sCheckPosition];/*只剩一个人*/
}

```

运行结果：（用时单位毫秒）

```

递推得到的结果：28, 600000次计算用时455
递归得到的结果：28, 600000次计算用时439
请按任意键继续. . . |

```

```

递推得到的结果：28, 600000次计算用时471
递归得到的结果：28, 600000次计算用时455
请按任意键继续. . . |

```

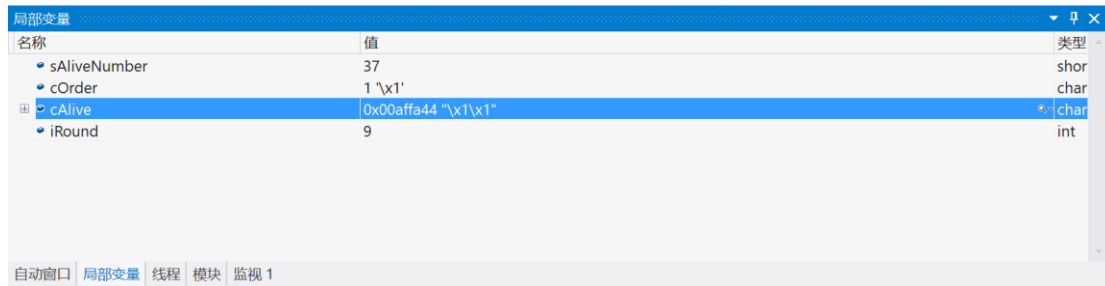
```
递推得到的结果：28，600000次计算用时487
递归得到的结果：28，600000次计算用时440
请按任意键继续. . . |
```

可见递推法用时长于递归法。

分析原因：

递推法每一次都需要遍历 40 个人（无论死活），而递归法每次只需要考虑活人，因此需要判断的次数减小，程序运行时间缩短。

“调用窗口”截图：



名称	值	类型
• sAliveNumber	37	short
• cOrder	1 '\x1'	char
• cAlive	0x00affa44 '\x1\x1'	char
• iRound	9	int

自动窗口 | 局部变量 | 线程 | 模块 | 监视 1

第二题

源代码：

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#define AVERAGE_REPETITION 3/*为了重复取平均进行的次数*/
int GoDown(int);
int iI0(char);
int iTimeOnce(char, int);
int iTimeAverage(char, int);
int iI0(char cRoundMax) /*输入n，输出结果的函数*/
{
    int iN; /*代表n的变量iN定义*/
    char cRound; /*代表执行的循环变量*/
    for (cRound=0; cRound<cRoundMax; cRound++) {
        printf("请输入n: ");
        scanf("%d", &iN);
        printf("n的方案有%d种。 \n", GoDown(iN));
    }
}
int GoDown(int n) /*计算方法数的函数*/
{
    switch(n) {
        /*初始情况*/
        case 1: return 1; break;
        case 2: return GoDown(1)+1; break;
        case 3: return GoDown(1)+GoDown(2)+1; break;
        /*高次情况*/
        default: return GoDown(n-3)+GoDown(n-2)+GoDown(n-1); break;
    }
}
```

```

    }
}
long long int llTimeOnce(char cN,int iRepetition){/*测一次时间*/
    int iTimeRef=clock();/*时间基准*/
    int iRoundTime=0;/*循环变量*/
    int llReturn=0;/*输出内容变量*/
    for (;iRoundTime<iRepetition;iRoundTime++) GoDown(cN);/*执行程序*/
    llReturn=clock()-iTimeRef;/*输出变量计算*/
    return llReturn;
}
double rTimeAverage(char cN,int iRepetition){/*计算一个n的用时平均值*/
    int iRoundAverage=0;/*循环变量*/
    double rTimeSum=0;/*累加器，最后求平均*/
    for (;iRoundAverage<AVERAGE_REPETITION;iRoundAverage++)
rTimeSum+=(double)llTimeOnce(cN,iRepetition);
    return (rTimeSum/(double)AVERAGE_REPETITION)/(double)iRepetition;
}
int main(){/*主函数*/
    iIO(16);/*依次测试5~20*/
    printf("\nn=15的计算用时: %.10lf",rTimeAverage(15,700));
    printf("\nn=25的计算用时: %.10lf",rTimeAverage(25,50));
    printf("\nn=35的计算用时: %.10lf\n",rTimeAverage(35,3));
    system("pause");
    return 1;
}

```

测试结果:

包含了题目的两个部分，其中 15、25、35 的测试时间均已取过平均，用时单位毫秒

```
请输入n: 5
n的方案有13种。
请输入n: 6
n的方案有24种。
请输入n: 7
n的方案有44种。
请输入n: 8
n的方案有81种。
请输入n: 9
n的方案有149种。
请输入n: 10
n的方案有274种。
请输入n: 11
n的方案有504种。
请输入n: 12
n的方案有927种。
请输入n: 13
n的方案有1705种。
请输入n: 14
n的方案有3136种。
请输入n: 15
n的方案有5768种。
请输入n: 16
n的方案有10609种。
请输入n: 17
n的方案有19513种。
请输入n: 18
n的方案有35890种。
请输入n: 19
n的方案有66012种。
请输入n: 20
n的方案有121415种。

n=15的计算用时: 0.0747619048
n=25的计算用时: 32.2266666667
n=35的计算用时: 16014.0000000000
请按任意键继续. . . |
```

选做题

第一题

源代码:

```
#include <stdio.h>
#include <stdlib.h>
#define MIN 100/2
#define MAX 1000/4
short sNum[6];/*存储两种脚数的六位的数组*/
```

```

short sCalculate(short);
short sCheck(short);
short sResult();
int main() {
    int iRound=MIN; /*循环遍历所有只数*/
    for (;iRound<MAX;iRound++) {
        sCalculate(iRound); /*更新脚六位数组*/
        if (sResult()) printf("只数为%d, 鸡脚数为%d, 兔脚数为%d.\n", iRound, iRound*2, iRound*4);
    }
    system("pause");
    return '/0';
}

short sCalculate(short sAmount) { /*对每一个可能的只数存储脚的六位*/
    short sChickenFeet=sAmount*2; /*鸡脚数*/
    short sRabbitFeet=sAmount*4; /*兔脚数*/
    short sRound; /*循环变量*/
    for (sRound=0;sRound<3;sRound++) {
        sNum[sRound]=sChickenFeet%10;
        sChickenFeet/=10;
    }
    for (sRound=0;sRound<3;sRound++) {
        sNum[sRound+3]=sRabbitFeet%10;
        sRabbitFeet/=10;
    }
    return 0;
}

short sCheck(short sNumber) { /*检查脚六位是否存在该数*/
    short sRoundCheck=0; /*循环变量*/
    short sFlag=0; /*指示是否存在, 0不存在, 1存在*/
    for (;sRoundCheck<6;sRoundCheck++) {
        if (sNum[sRoundCheck]==sNumber) {
            sFlag=1;
            return 1;
        }
    }
    return 0;
}

short sResult() { /*检查脚六位是否符合要求*/
    short sRoundResult=0; /*循环变量*/
    short sSumFlag=0; /*累加sCheck输出结果*/
    for (;sRoundResult<6;++sRoundResult) sSumFlag+=sCheck(sRoundResult);
    return sSumFlag/6;
}

```

运行结果:

```
只数为76, 鸡脚数为152, 兔脚数为304。
请按任意键继续. . . |
```

第二题

源代码:

```
#include<stdio.h>
#include<stdlib.h>
#define n 50
long long fllS(void);
int main() {
    double rS=(double)fllS();
    double rNS=n*(n+1)*(n+2)/120.0*(8*n*n+11*n+1);
    printf("S=%.01f\tNS=%.01f\n", rS, rNS);
    if (rS>rNS) printf("S>NS, output=1\n");
    else if (rS==rNS) printf("S=NS, output=0\n");
    else printf("S<NS, output=-1\n");
    system("pause");
    return 0;
}
long long fllS() { /*计算S*/
    long long llSumS=0; /*累加器*/
    short sRoundS1=1, sRoundS2; /*循环变量*/
    for (;sRoundS1<=n;sRoundS1++) {
        for (sRoundS2=1;sRoundS2<=sRoundS1;sRoundS2++)
            llSumS+=sRoundS1*sRoundS2*sRoundS2;
    }
    return llSumS;
}
```

运行结果:

```
S=22708855      NS=22708855
S=NS, output=0
请按任意键继续. . . |
```