高级语言程序设计实验

综合实验报告

课程名称：高级语言程序设计实验

学 期：2022年秋季学期

年级班级：2022电子信息类11班

学生姓名：唐嘉骏

学 号：20221120044

课程教师：尉洪

时 间：2022.12.15

云南大学信息学院

**A题：行列式计算及DNA序列统计特征挖掘**

一、实验目的

1、用C语言程序实现，数学学科中2~10阶行列式的输入和计算输出。

2、用C语言程序实现，发掘A、B两类样本DNA序列片段的统计特征。

二、实验方法

1、利用二维数组存储行列式，考虑使用行列式的代数余子式降阶展开性质完成行列式的计算；考虑设计递归函数，实现行列式的降阶计算；

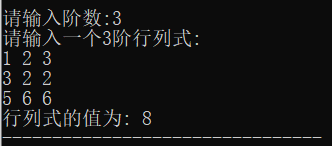
2、给定A、B两类样本DNA序列片段，见附录；考虑编程实现，统计DNA序列字符串中各种子串的出现次数；利用各种子串的出现频度，形成A、B各类DNA序列的统计规律。

三、实验设备软件： 开发环境CodeBlocks 或Visual C++6.0。

四、实验内容要求及实验记录（流程图、程序、结果数据等）

1、用二维数组表示行列式，编写函数实现计算并输出行列式的值。要求程序主函数，输入一个N阶行列式（N<10），调用函数计算行列式的值，至少给出3个不同阶数行列式的计算结果。（提示：行列式值的计算函数包括两个输入参数：数组名，行列式的阶数；函数返回值是行列式的值）（50分）

**[程序输入输出格式示例]**

****

源代码：

#include<stdio.h>

#include<math.h>

#define MAX 10 //定义数组大小常量

double detCalc(double [][10],int );//函数声明

int main()

{

double num[MAX][MAX];

int n;

printf("请输入阶数:");

scanf("%d",&n);//输入阶数

printf("请输入一个%d阶行列式:\n",n);

int i,j;

for(i=0;i<n;i++) //循环输入为数组元素赋值

{

for(j=0;j<n;j++)

{

scanf("%lf",&num[i][j]);

}

}

double res=detCalc(num,n);//定义res变量用于接收函数返回值

printf("行列式的值为:%lg",res);//输出结果

return 0;

}

double detCalc(double a[10][10],int n)//函数定义

{

int i,j,k;

double res=0;

double b[10][10]; //用来存放余子式

if(n==1)

{

res=a[0][0];//若最后为一阶直接返回该元的值

}

else

{

for(j=0;j<n;j++)//按行展开

{

for(i=0;i<n-1;i++)//内部两层循环用于构造余子式

{

for(k=0;k<n-1;k++)

{

if(k<j)

{

b[i][k]=a[i+1][k];

}

if(k>=j)

{

b[i][k]=a[i+1][k+1];

}

}

}

res+=pow(-1,j)\*a[0][j]\*detCalc(b,n-1); /\*用pow函数进行代数余子式正负判断并进行递归调用\*/

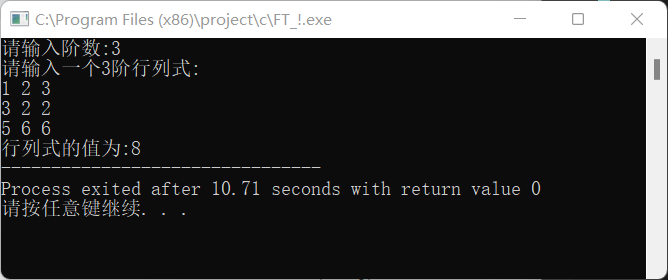
}

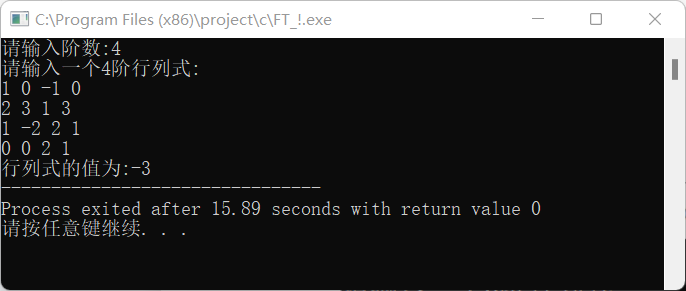
}

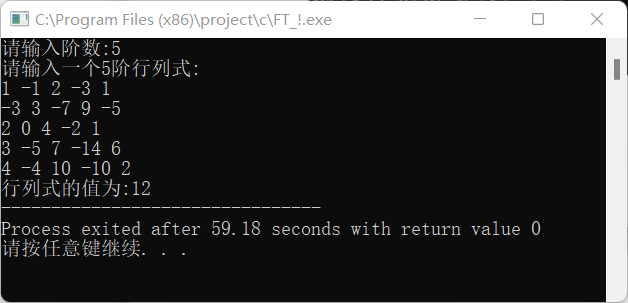
return res;//返回行列式计算结果

}

测试：







2、DNA序列是由4种字符A，T，C，G简化表示的4种碱基按一定的顺序排列而成，充分发现DNA序列的规律性和结构对理解DNA全序列是十分有意义的。请给出AB两类各序列片段中各子串的出现次数，发现AB各类DNA序列片段的分类统计规律。（50分）

**[统计规律参考表格格式]**

表1. 样本DNA序列片段统计特征

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 统计特征 | 样本串1 | 样本串2 | 样本串3 | 样本串4 | 样本串5 | 共同特征 |
| A类  1字串 | a--->33  t--->15  c--->19  g--->44 | a--->30  t--->17  c--->18  g--->46 | a--->30  t--->7  c--->24  g--->50 | a--->47  t--->32  c--->12  g--->20 | a--->26  t--->12  c--->26  g--->47 | a,g碱基较多  t,c碱基较少 |
| B类  1字串 | 1. -->39   t--->55  c--->5  g--->11 | a--->36  t--->55  c--->3  g--->16 | a--->28  t--->57  c--->11  g--->14 | a--->33  t--->55  c--->9  g--->13 | a--->32  t--->71  c--->0  g--->7 | a,t碱基较多  c,g碱基较少 |
| A类  2字串 | gg--->21  ga--->13  aa--->10  ac--->10  ag--->9 | gg--->24  ga--->11  aa--->11  cg--->10  ac--->8 | gg--->26  cg--->16  ga--->15  gc--->8  ag--->6 | aa--->21  at--->13  ta--->12  tt--->12  ga--->7 | gg--->24  cg--->12  ga--->11  gc--->10  ac--->9 | gg子串出现次数最多，ga，aa，cg其次，其余二字串占比较少 |
| B类  2字串 | tt--->33  ta--->18  aa--->17  at--->16  gt--->5 | tt--->34  ta--->17  aa--->15  at--->12  gt--->8 | tt--->38  ta--->14  at--->11  aa--->7  gt--->5 | tt--->31  ta--->15  at--->14  aa--->9  ag--->8 | tt--->53  ta--->15  aa--->15  at--->14  gt--->4 | tt子串出现次数最多，ta，at，aa其次，其余较少。 |
| A类  3字串 | gga--->10  agg--->8  acg--->7  cgg--->7  ggc--->5 | gga--->11  cgg--->10  agg--->5  tgg--->5  ggc--->5 | cgg--->16  gga--->15  acg--->10  ggc--->8  agg--->6 | aaa--->12  ata--->5  tat--->5  ttt--->5  aac--->4 | cgg--->11  gga--->11  ggc--->10  agg--->7  gag--->6 | gga，cgg出现次数较多，其余三字串较少 |
| B类  3字串 | ttt--->19  tta--->13  att--->10  tat--->8  aat--->7 | ttt--->18  tta--->15  taa--->10  att--->9  aat--->7 | ttt--->23  tta--->13  att--->9  tat--->7  taa--->4 | ttt--->17  tta--->11  att--->8  tct--->5  tat--->5 | ttt--->43  aaa--->9  tta--->9  att--->9  tat--->9 | ttt，tta出现次数较多，其余较少 |

源代码：

#include<stdio.h>

#include<string.h>

int calc(char [],char []); //函数原型声明

int main()

{

char str[1000];

char substr[1000];//声明两个字符数组，一个存放DNA序列，一个存放子串

printf("请输入DNA序列:"); //输入序列及子串

gets(str);

printf("请输入要统计的子串:");

gets(substr);

int res=calc(str,substr); //调用函数获取统计结果

printf("该子串共出现%d次\n",res); //输出统计结果

return 0;

}

int calc(char str[], char substr[])

{

int i,j;

int sum=0; //设置计数变量sum进行累加统计

int n=strlen(str);

int subn=strlen(substr);//使用strlen函数获取DNA序列及子串的长度

for(i=0;i<n;i++) //嵌套循环进行枚举判断

{

for(j=0;j<subn;j++)

{

if(\*(str+j+i)!=\*(substr+j))

/\*如果以第i个字符为起始的DNA序列的第j个字符与子串第j个字符不匹配则跳出循环\*/

{

break;

}

}

if(j==subn)

/\*如果跳出循环后循环变量j的值等于子串长度则证明存在这样的一个子串\*/

{

sum++; //进行计数

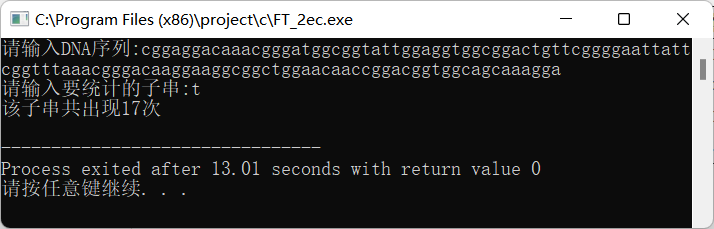
}

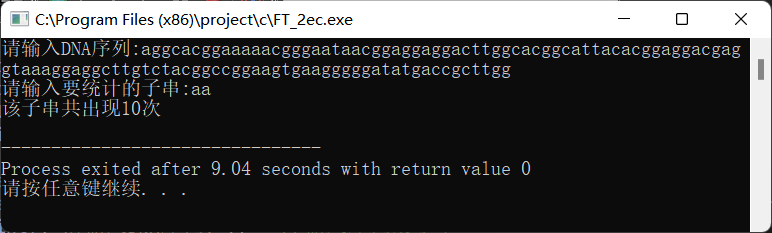
}

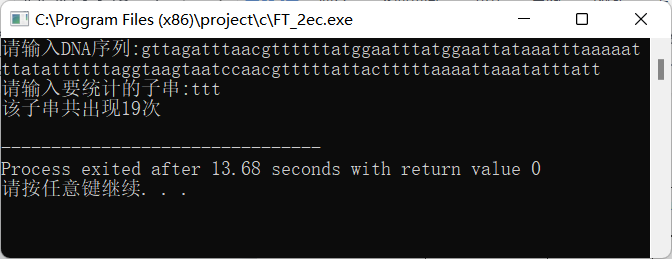
return sum; //返回统计结果

}

测试：







五、实验分析及实验小结

1. 总结解决问题的方法

第一题：

应用知识点：

1. 函数的递归调用
2. 二维数组的使用与通过嵌套循环进行赋值
3. 函数的定义与声明
4. %lg占位符的使用
5. math.h头文件及其库函数pow的使用

解决问题的思路：

首先在主函数中定义一个二维数组用来存放行列式，并通过输入自定义阶数并循环输入为二维数组赋值，调用detCalc函数，在函数中定义b二维数组用来存放余子式，函数共有三层循环，最外层循环起到按行展开作用，内部两层循环起到构建余子式的作用并存放到二维数组b中，通过pow函数来对余子式的正负号进行判断并与展开的元进行相乘计算并累加到res变量中，进行递归调用直至余子式为1阶，返回res的值并在主函数中输出，%lg占位符用来自动除去多余小数点后位数。

第二题：

使用知识点：

1. 使用指针操作数组
2. 枚举法
3. string.h头文件的使用及使用库函数strlen统计字符串长度
4. 使用gets函数输入字符串
5. 字符数组的定义
6. 函数的定义与调用
7. 数组名作为函数参数

解决问题的思路：

定义两个字符数组用于存放DNA序列和子串，允许用户自行输入，并将字符数组名作为函数参数传递到计算函数中，函数中使用strlen函数获取字符串长度，嵌套循环，外部循环用来指定DNA序列起始字符，内部循环用来将DNA序列的起始字符后相应第j个字符与子串的第j个字符进行判等，若不相等则跳出循环，最后判断循环变量j的值是否与子串长度1相等，相等则证明存在该字串，计数加一，最后返回总数并输出。

附录：

A类样本DNA序列片段：

1.aggcacggaaaaacgggaataacggaggaggacttggcacggcattacacggaggacgaggtaaaggaggcttgtctacggccggaagtgaagggggatatgaccgcttgg

2.cggaggacaaacgggatggcggtattggaggtggcggactgttcggggaattattcggtttaaacgggacaaggaaggcggctggaacaaccggacggtggcagcaaagga

3.gggacggatacggattctggccacggacggaaaggaggacacggcggacatacacggcggcaacggacggaacggaggaaggagggcggcaatcggtacggaggcggcgga

4.atggataacggaaacaaaccagacaaacttcggtagaaatacagaagcttagatgcatatgttttttaaataaaatttgtattattatggtatcataaaaaaaggttgcga

5.cggctggcggacaacggactggcggattccaaaaacggaggaggcggacggaggctacaccaccgtttcggcggaaaggcggagggctggcaggaggctcattacggggag

B类样本DNA序列片段：

6.gttagatttaacgttttttatggaatttatggaattataaatttaaaaatttatattttttaggtaagtaatccaacgtttttattactttttaaaattaaatatttatt

7.gtttaattactttatcatttaatttaggttttaattttaaatttaatttaggtaagatgaatttggttttttttaaggtagttatttaattatcgttaaggaaagttaaa

8.gtattacaggcagaccttatttaggttattattattatttggattttttttttttttttttttaagttaaccgaattattttctttaaagacgttacttaatgtcaatgc

9.gttagtcttttttagattaaattattagattatgcagtttttttacataagaaaatttttttttcggagttcatattctaatctgtctttattaaatcttagagatatta

10.gtattatatttttttatttttattattttagaatataatttgaggtatgtgtttaaaaaaaatttttttttttttttttttttttttttttttaaaatttataaatttaa