数据结构 A
作业 7 参考答案

作业情况

教材：数据结构教程（C++ 语言描述）李春葆等
题目范围：数组与递归
邮箱：wjyyy1@126.com
授课教师：彭蓉 教授
助教：王骏峣

习题 1 单选题

设有数组 $A[i,j]$，数组中的每个元素长度为 3 个字节，i 的值为 1 到 8，j 的值为 1 到 10，数组从内存首地址 BA 开始以列为主序顺序存放，元素 $A[5,8]$ 的存储首地址为（　　）。

A. $BA+141$
B. $BA+180$
C. $BA+222$
D. $BA+225$

答案：B

解析：参考教材 5.1.2 数组的存储结构，题目说“**以列为主序**”进行存放，因此 $A[5,8]$ 前面还有 $1\sim7$ 列的各个元素，以及 $A[1,8]\sim A[4,8]$。共有 $7\times8+4=60$ 个元素，每个元素占据 3 个字节，共为 180 字节的偏移量。

在做题的时候，你可以认为内存首地址 BA 存储的是 $A[1,1]$，偏移量为 0，内存地址为 $BA+0$。所以计算出 $A[5,8]$ 为 $BA+180$。

习题 2 单选题

设某二维数组 $a[10][20]$ 采用顺序存储方式，每个数组元素占用 1 个存储单元，$a[0][0]$ 的存储地址为 200，$a[6][2]$ 的存储地址是 322，则该数组（　　）。

A. 只能按行优先存储
B. 只能按列优先存储
C. 按行优先存储或按列优先存储均可
D. 以上都不对

答案：A

解析：每个数组元素占用 1 个存储单元，所以 $a[6][2]$ 前面应该有 122 个元素。假设按行优先存储，$a[6][2]$ 前面有 $6 \times 20 + 2 = 122$ 个元素；假设按列优先存储，$a[6][2]$ 前面有 $2 \times 10 + 6 = 26$ 个元素。

按列存储与题目所给条件矛盾，因此只能按行优先存储。

习题 3 填空题

设有一个带宽为 5 的 10 阶对角阵 A，采用压缩存储方式，$A[1][1]$ 的存储地址为 1，每个元素占一个地址空间，若以行序为主序，$A[8][6]$ 的存储地址是（　　）；若以列序为主序，$A[8][6]$ 的存储地址是（　　）。

答案：33，27

解析：参考教材 5.2.3，带宽为 5 的 10 阶对角阵 A 应该如下所示，半带宽 $b = 2$。

$$\begin{bmatrix}
a_{1,1} & a_{1,2} & a_{1,3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & 0 & 0 & 0 & 0 & 0 & 0 \\
a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} & 0 & 0 & 0 & 0 & 0 \\
0 & a_{4,2} & a_{4,3} & a_{4,4} & a_{4,5} & a_{4,6} & 0 & 0 & 0 & 0 \\
0 & 0 & a_{5,3} & a_{5,4} & a_{5,5} & a_{5,6} & a_{5,7} & 0 & 0 & 0 \\
0 & 0 & 0 & a_{6,4} & a_{6,5} & a_{6,6} & a_{6,7} & a_{6,8} & 0 & 0 \\
0 & 0 & 0 & 0 & a_{7,5} & a_{7,6} & a_{7,7} & a_{7,8} & a_{7,9} & 0 \\
0 & 0 & 0 & 0 & 0 & a_{8,6} & a_{8,7} & a_{8,8} & a_{8,9} & a_{8,10} \\
0 & 0 & 0 & 0 & 0 & 0 & a_{9,7} & a_{9,8} & a_{9,9} & a_{9,10} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & a_{10,8} & a_{10,9} & a_{10,10}
\end{bmatrix}$$

其特征为有 5 条斜向右下的线上是有元素的，其他为 0。压缩存储时只考虑非零元素，按照顺序存储在（一维）压缩数组里。

如果只是填空题，挨个数出 $A[8][6]$ 前面有多少个元素就可以了。当以行为主序，就一行一行数（从第一行到最后一行左到右）；当以列为主序，就一列一列数（从第一列到最后一列上到下）。

如果本题是编程题，可以考察给定 b 和下标 (i, j)，求出 $A[i][j]$ 在压缩矩阵中的的存储地址。前 b 行中，第 i 行有 $b + i$ 个元素，后 b 行中，第 i 行有 $b - (n - i) + 1$ 个元素。其余中间各行有 $2b + 1$ 个元素。

因此第 i 行第 j 列以行为主序存储在压缩数组中的位置为：

$$\begin{cases} \sum_{k=1}^{i-1}(b+k)+j, & i \le b \\ \frac{(3b+1)b}{2}+\sum_{k=1}^{i-b-1}(2b+1)+j-(i-b-1), & b < i \le n-b \\ \frac{(3b+1)b}{2}+(n-2b)(2b+1)+\sum_{k=1}^{i-n+b}(b-(n-k)+1)+j-(i-b-1), & i > n-b \end{cases}$$

计算方式主要是计算前面的行共有多少非零元素，加上当前行前面已经出现过的非零元素个数。（不作详细解释，感兴趣可以单独询问助教）。

习题 4 N 皇后 II

【问题描述】

n 皇后问题研究的是如何将 n 个皇后放置在 $n \times n$ 的棋盘上，并且使皇后彼此之间不能相互攻击。给定一个整数 n，返回 n 皇后不同的解决方案的数量。要求设计如下函数：

```
class Solution{
public:
    int totalNQueens(int n)
    { ... }
};
```

【输入形式】

输入一个整数 n。

【输出形式】

输出 n 皇后问题的解决方案的数量。

【样例输入】

6

【样例输出】

4

【样例说明】

6 皇后问题有 4 个不同的求解方案。测试数据存放在 `in.txt` 文件中。

【评分标准】

共 10 个测试用例，每通过一个测试得 10 分。

答案：

```cpp
#include<cmath>
#include<cstdio>
#include<cstring>
#include<iostream>
#include<fstream>
using namespace std;
class Solution {
    int cnt;        //累计解个数
    int q[50];      //q[i]存放第i行放置皇后的列号
public:
    int totalNQueens(int n)
    {
        cnt=0;//初始化并清空
        memset(q,0,sizeof(q));
        queen(1,n);
        return cnt;
    }
    bool place(int i,int j)     //测试(i,j)位置能否摆放皇后
    {
        for(int k=1;k<i;k++)       //k=1~i-1是已放置了皇后的行
        //如果前面有人在j列放了皇后，就false，结束当前测试
            if(q[k]==j||(abs(q[k]-j)==abs(i-k)))//同一列或同一对角线
                return false;
        //没有遇到异常情况就返回true，可以放置
        return true;
    }
    void queen(int i,int n)//已放置1~i-1的皇后，正在放置第i行
    {
        if(i>n)        //所有皇后放置结束
        {
            cnt++;       //统计一个解
            return;
        }
            else
            {
                for(int j=1;j<=n;j++)    //在第i行上试探每一个列j
                if(place(i,j))     //如果(i,j)可以放置皇后
                {
                    q[i]=j;          //修改值，表示第i行的皇后放在了第j列
                    queen(i+1,n);    //每一层有多个放皇后的可能
                }//所以循环会多次进入queen(i+1,n)
            }
    }
};
int main()
{
    ifstream in("in.txt");
```

```
    int n;
    in>>n;
    Solution obj;
    cout<<obj.totalNQueens(n);
    return 0;
    }
```

解析：本题需要计算 n 皇后的方案数。在搜索过程中，需要时刻注意不要和前面放置过的皇后产生冲突，为了记录这样的信息，我们把每一行已经放置的皇后存储在数组 q 中。

每次尝试将第 i 行的皇后放在第 j 列，需要保证前面没有其他的 $q[k]=j$，且没有对角线冲突，即 $|q[k]-j|=|k-i|$。如果满足条件，就可以继续放置下一行的皇后，否则就需要回溯到上一层重新放置上一层的皇后进行测试。

习题 5 母牛的故事

【问题描述】
有一头母牛，它每年年初生一头小母牛。每头小母牛从第四个年头开始，每年年初也生一头小母牛。请编程实现在第 n 年的时候，共有多少头母牛?
【输入形式】
输入一个整数 n（$0<n<55$），n 的含义如题目中描述。
【输出形式】
输出在第 n 年的时候母牛的数量。
【样例输入】
5
【样例输出】
6
【样例说明】
第 5 年时共有 6 头母牛。测试数据存放在 `in.txt` 文件中。
【评分标准】
该题目有 10 个测试用例，每通过一个测试用例，得 10 分。

答案：

```
#include<iostream>
#include<fstream>
using namespace std;
long long func(int n)
{
    //第n年的牛数量为上一年（n-1年）的数量
    //加上这一年已经可以生产小牛的母牛数量
    //return func(n-1)+func(n-3);
```

```
9       //问题在于，n不能一直缩小，所以要在n足够小的时候定义递归边界
10      if(n<=3)//前三年，只有第一头牛能生产
11          return n;
12      return func(n-1)+func(n-3);
13  }
14  int main()
15  {
16      ifstream in("in.txt");
17      int n;
18      in>>n;
19      cout<<func(n);
20      return 0;
21      }
```

解析：详见代码注释。推导过程为 $f_i = f_{i-3} + f_{i-1}$。考虑到本章内容为递归，所以以递归的形式实现，即 $f(n) = f(n-3) + f(n-1)$。

为了防止递归越界，我们要在 n 足够小的时候特判递归边界，让其停止。

个人认为题目描述不够清楚，实际上第一头牛在 $n = 2$ 时才开始生小牛。所以有 $f_1 = 1, f_2 = 2, f_3 = 3$。从 $n = 4$ 开始就满足上面的递推式子了。

建议大家在期末考试遇到这种看到样例解释仍然不明白的问题及时提问。

总结

题目：数组和递归

日期：2024 年 5 月 9 日

批改人：王骏峣

邮箱：wjyyy1@126.com

习题 1、2：概念题，注意数组存储的方式和行列优先级。

习题 3：主要考察对对角矩阵压缩存储的掌握，建议对其定义进行进一步的了解和分析。

习题 4：基础的递归搜索，也是深度优先搜索（dfs）的基础。注意不同层搜索之间的制约、产生的变化和影响。

习题 5：考察简单的递归，类似“斐波那契数列”。本题题目描述较为难懂，建议优先使用样例解释进行输入输出数据的核对，思路没问题后再开始代码编写。

本次作业经查重，有两位同学的代码有相似情况，考虑到部分同学已转入 B 班，仅扣除 20% 的分数并提出提醒。

各位同学如有问题欢迎及时在群里提出，或者通过邮件/QQ 联系我。