C4-2021级航类-第四次练习赛题解

A 二进制转十进制(不是水题)

难度	考点
2	数组,位运算

问题分析

开数组储存二进制01字符串,然后用移位和位或拼出需要的数,数据范围是unsigned long long

参考代码

```
#include <stdio.h>
int main(){
char buffer[100] = { 0 };
    scanf("%s", buffer);
    int n = strlen(buffer), i;
    unsigned long long ans = 0, p = 0;
    for(i = n - 1; i >= 0; i--){
        ans |= ((unsigned long long)(buffer[i] - '0') << (unsigned long long )p);
        p++;
    }
    printf("%llu\n", ans);
return 0;
}
```

B WWH函数

难度	考点
2	递归

问题分析

按照题目要求调用函数即可，具体实现过程见代码。

参考代码

```
#include<stdio.h>
#include<string.h>
#include<math.h>
int solve(int a,int b,int c)
{
    if(a>8||b>8||c>8)
    {
```

```
8           return solve(8,8,8);
9       }
10      else if(a<=0||b<=0||c<=0)
11      {
12          return 1;
13      }
14      else if(a<b&&b<c)
15      {
16          return  solve(a, b, c-1) + solve(a, b-1, c-1) - solve(a, b-1, c);
17      }
18      else return  solve(a-1, b, c) + solve(a-1, b-1, c) + solve(a-1, b, c-1) -
    solve(a-1, b-1, c-1);
19  }
20  int main()
21  {
22      int a,b,c;
23      scanf("%d%d%d",&a,&b,&c);
24      printf("%d",solve(a,b,c));
25   }
```

C 简单的数学题

难度	考点
3	数学

题目分析

结论：输出1~n内的所有完全平方数。

分析：一盏灯的开关状态转换的次数为它编号所有约数的个数，例如编号为6的灯，它的状态会被第1、2、3、6次操作所影响。一开始所有灯的状态为**关**，而题目要求输出最后状态为**开**的灯，那么就是在求约数个数为**奇数**的数。

对于一个数x，设它一共有n个约数，且从小到大分别为：
p_1、p_2、p_3、……、p_{n-2}、p_{n-1}、p_n

那么我们很容易可以知道：$p_1 \times p_n = p_2 \times p_{n-1} = p_i \times p_{n-i+1} = x$.

若n为奇数，令$i = \frac{n}{2} + 1$（$\frac{n}{2}$为n整除2），则:
$n - i + 1 = (\frac{n}{2} + \frac{n}{2} + 1) - (\frac{n}{2} + 1) + 1 = \frac{n}{2} + 2 = i$

也就是说，$x = p_i \times p_{n-i+1} = p_i \times p_i$，那么x必定是完全平方数。

示例代码

```c
#include<stdio.h>

int n;

int main(){
    int i;

    scanf("%d", &n);

    for(i=1; i*i<=n; ++i)
        printf("%d ", i*i);
    return 0;
}
```

D 千年后的回响

难度	考点
2	进制转换

问题分析

这道题首先要理解题意，题目将三位二进制数与八卦之间一一建立了映射关系，并使用八卦充当了一个简易的八进制编码系统。求八进制位数与第零位八进制数都是非常基础的操作，而输出阳爻和阴爻的代码会被大量复用，因此可以抽象成一个函数减少冗余代码。
很多同学枚举了八种情况来表示八卦，但如果卦的阶数进一步变大，枚举的数量将会以指数量级上涨，更一般地做法是分别去判断二进制的第0，1，2位来输出对应爻。

参考代码

```c

#include<stdio.h>
void yin(int count);//输出阴爻函数
void yang(int count);//输出阳爻函数
int main(){
    int a=0;
    scanf("%d",&a);
    int k=a%8;
    int count=0;
    while(a){
        count++;//计算八进制位数
        a/=8;
    }
    if(k&(1<<2)){//取出第2位
        yang(count);
        printf("\n");
    }else{
        yin(count);
        printf("\n");
    }
    if(k&(1<<1)){//取出第1位
        yang(count);
        printf("\n");
```

```
    }else{
        yin(count);
        printf("\n");
    }
    if(k&(1<<0)){//取出第0位
        yang(count);
    }else{
        yin(count);
    }
    return 0;
}
void yin(int count){
    for(int i=1;i<=count;i++)
        printf("-");
    for(int i=1;i<=count;i++)
        printf(" ");
    for(int i=1;i<=count;i++)
        printf("-");
}
void yang(int count){
    for(int i=1;i<=3*count;i++)
        printf("-");
}

```

E LJF算可靠度

难度	考点
2	概率统计

问题分析

利用rand()函数生成伪随机数计算积分即可。

参考代码

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main()
{
    double t, x, y;
    int cnt = 0;
    srand((unsigned)time(NULL));
    scanf("%lf", &t);
    for (int i = 0; i < 1000000; i++)
    {
        x = 1.0 * (rand() % 32768) / 32768 * t;
        y = 1.0 * (rand() % 32768) / 32768 * exp(-0.25);
        if (y < exp(-pow(x + 0.5, 2)))
            cnt++;
    }
```

```
    // printf("%d\n", cnt);
    printf("%f", exp(-cnt / 1000000.0 * t * exp(-0.25)));
}
```

F 532排名制

难度	考点
4	函数调用

问题分析

没有什么考察思维的地方，主要需要足够细心。

参考代码

```
//
// Created by moc85 on 2022/3/26.
//

#include <stdio.h>
#include <math.h>
/*
 * 表示学生成绩的全局数组
 */
int stu_id[50];     //学号
double gpa[50];         //GPA
double w_average[50];  //加权平均
double credit[50];     //总学分
double total_score[50];//最终得分

/*
 * 表示单个学生成绩的全局数组（由于不需复用，可定义为全局）
 */
int c_scores[30];   //得分
double c_credit[30];   //学分
int c_honor[30];    //是否荣誉课

/*
 * 三项指标最高值
 */
double max_gpa;
double max_wAverage;
double max_credit;

void fillScores(int n);
double getSingleGPA(int score);
double getGPA(int m);
double getAvg(int m);
double getCredits(int m);
void setMax(int n);
void setTotalScore(int n);
void outputScore(int n);

```

```c
int main()
{
    int n = 0;
    scanf("%d", &n);
    fillScores(n);
    setMax(n);
    setTotalScore(n);
    outputScore(n);
    return 0;
}

/*
 * 输入成绩
 */
void fillScores(int n)
{
    int i = 0, j = 0, m = 0;
    for (i = 0; i < n; i++) {
        scanf("%d %d", &stu_id[i], &m);

        for (j = 0; j < m; j++) {
            scanf("%lf %d %d", &c_credit[j], &c_scores[j], &c_honor[j]);
        }

        gpa[i] = getGPA(m);
        w_average[i] = getAvg(m);
        credit[i] = getCredits(m);
    }
}

/*
 * 计算单个课程GPA
 */
double getSingleGPA(int score)
{
    if (score < 60) {
        return 0;
    }
    return 4.0 - 3.0 * pow(100 - score, 2) / 1600;
}

/*
 * 计算GPA
 */
double getGPA(int m)
{
    int i = 0;
    double totalGPA = 0, totalCD = 0;
    for (i = 0; i < m; i++) {
        double c_gpa = getSingleGPA(c_scores[i]);
        totalGPA += c_gpa * c_credit[i];
        totalCD += c_credit[i];
    }

    return totalGPA / totalCD;
}

/*
```

```
 * 计算荣誉课加权平均
 */
double getAvg(int m)
{
    int i = 0;
    double totalScore = 0, totalCD = 0;
    for (i = 0; i < m; i++) {
        if (c_honor[i]) {
            totalScore += c_scores[i] * c_credit[i];
            totalCD += c_credit[i];
        }
    }

    return totalScore / totalCD;
}

/*
 * 计算总学分数
 */
double getCredits(int m)
{
    int i = 0;
    double totalCD = 0;
    for (i = 0; i < m; i++) {
        totalCD += c_credit[i];
    }

    return totalCD;
}

/*
 * 计算三项指标最高值
 */
void setMax(int n)
{
    int i = 0;
    for (i = 0; i < n; i++) {
        if (gpa[i] > max_gpa) {
            max_gpa = gpa[i];
        }

        if (w_average[i] > max_wAverage) {
            max_wAverage = w_average[i];
        }

        if (credit[i] > max_credit) {
            max_credit = credit[i];
        }
    }
}

/*
 * 计算总成绩
 */
void setTotalScore(int n)
{
    int i = 0;
    for (i = 0; i < n; i++) {
```

```
            total_score[i] = 50 * gpa[i] / max_gpa + 30 * w_average[i] / max_wAverage + 20 * credit[i] / max_credit;
        }
    }

    /*
     * 输出成绩
     */
    void outputScore(int n)
    {
        int i = 0;
        for (i = 0; i < n; i++) {
            printf("%d %.4f\n", stu_id[i], total_score[i]);
        }
    }
```

G 开心的华胖胖

难度	考点
3	动态规划

问题分析

一共有n个活动，活动为h_0~h_{n-1}。

用$f[i]$表示以第i个数起始的连续活动的快乐值最大和，我们需要求的答案即为$\max_{0\le i\le n-1}(f[i])$

对于每一个$f[i]$，我们考虑$f[i+1]$和$f[i]$的关系，有$f[i]=max(f[i+1]+h_i,h_i)$

参考代码

```
#include<stdio.h>

const int N=1e5+10;
int h[N],f[N];

int main() {
    int n;
    scanf("%d",&n);
    int i;
    for(i=0;i<n;i++) {
        scanf("%d",&h[i]);
    }

    f[n-1]=h[n-1];//初始化f[n-1]

    for(i=n-2;i>=0;i--) {
        //以下是递推公式的部分
        if(f[i+1]+h[i]>h[i]) {
            f[i] = f[i+1]+h[i];
        }else {
            f[i] = h[i];
```

```
            }
        }

        int res = f[0];
        for(i=1;i<n;i++) {
            if (f[i]>res) {
                res = f[i];
            }
        }

        printf("%d\n",f[i]);

        return 0;
    }
```

H zhn の 字符串

难度	考点
6	数学推导

如果直接计算的话，很明显算不了，无论是时间复杂度还是枚举方式，都不是**优雅**的。

我们可以单独考虑每一位a_i对于总答案的贡献，如何计算呢?

我们考虑每一个a_i分别作为个位、十位、百位、千位的贡献，具体计算如下:

1. a_i作为个位，那么必然有一个加号放在a_i后面，还剩下$k-1$个加号，可以有$n-2$个位置填，所以他作为个位的贡献为：$a_i * C_{n-2}^{k-1}$
2. a_i作为十位，那么必然有一个加号放到a_{i+1}后面，那么剩下的$k-1$个加号还有$n-3$个位置填，所以他作为十位的贡献为：$a_i * C_{n-3}^{k-1} * 10$

   ....
3. 以此类推，在a_i后面离a_i最近的加号位于$a_{i+n-k-1}$后面，那么剩下的$k-1$个加号可以有$n-2-(n-k-1)==k-1$个位置填，所以贡献为：$a_i * 10^{n-k-1} * C_{k-1}^{k-1}$种方式可以填。
4. 我们以上讨论的情况都是a_i后面有加号的情况，如果a_i后面没有加号，那么所有的加号都得放在前面的$i-1$个位置，这种情况的贡献是$a_i * 10^{n-i} * C_{i-1}^{k}$
5. 我们把上面这些东西求和，经过简单的合并同类项，可以得到如下的最终公式：

$$\sum_{i=1}^{n-k} 10^{i-1} * (\sum_{j=1}^{n-i} a_j * C_{n-i-1}^{k-1} + a_{n-i+1} * C_{n-i}^{k})$$

   其中，j的部分可以用前缀和优化，以便得到较好的时间复杂度，否则会TLE。

```
#include <stdio.h>
#define M 1000000007
#define N 100005
#define ll long long
int n,k;
char d[N];
ll s[N],fac[N],inv_fac[N],num[N];
ll ksm(ll a,ll b){
    ll temp=1ll;
    while(b){
```

```
        if(b&1) temp=temp*a%M;
        a=a*a%M;
        b>>=1;
    }
    return temp;
}
ll inv(ll x){
    return (ksm(x,M-2)%M+M)%M;
}
void init(){
    fac[0]=1;
    for(int i=1;i<=n; i++) fac[i]=fac[i-1]*i%M;
    for(int i=0;i<=n;i++) inv_fac[i]=inv(fac[i]);
}
ll C(ll a,ll b){
    if(a<b) return 0;
    return fac[a]*inv_fac[b]%M *inv_fac[a-b]%M;
}
int main(){
    scanf("%d%d ",&n,&k);
    scanf("%s",d+1);
    for(int i=1;i<=n;i++) num[i]=d[i]-'0';
    for(int i=1;i<=n;i++) s[i]=s[i-1]+num[i];
    init();
    ll ans=0;
    ll base=1;
    for(ll i=1;n-i>=k;i++){
        ans+=(((num[n-i+1])*C(n-i,k)%M*base%M)%M+M)%M;
        ans+=((s[n-i]*C(n-i-1,k-1)%M*base%M)%M+M)%M;
        base*=10;
        base=base%M;
    }
    printf("%lld",(ans%M+M)%M);
    return 0;
}

```

I 方阵数局简单版

难度	考点
2	二维数组

题目分析

二维数组的预习题，会用二维数组就会做

参考代码

```c
#include <stdio.h>
#include <math.h>
int a[1002][1002];

int main()
{
    int n,m,i,j,x,y,t;
    scanf("%d%d",&n,&m);
    for(i=1;i<=n;i++)
    {
        for(j=1;j<=m;j++)
        {
            scanf("%d",&a[i][j]);
        }
    }
    scanf("%d%d%d",&x,&y,&t);
    while(t--)
    {
        switch (a[x][y])
        {
        case 1:x--;break;
        case 2:y++;break;
        case 3:x++;break;
        case 4:y--;break;
        }
    }
    printf("(%d,%d)",x,y);
}
```

J 阿水排排坐

难度	考点
2	冒泡排序

问题分析

冒泡排序，参考 **这个**

参考代码

```c
#include <stdio.h>
#define M 10005
double a[M];
int main()
{
    int n,i,j;
    double tmp;
    scanf("%d",&n);
    for(i=0;i<n;i++)
```

```
        scanf("%lf",&a[i]);
    for(i=0;i<n-1;i++)
    {
        for(j=0;j<n-1-i;j++)
        {
            if(a[j]>a[j+1])
            {
                tmp=a[j];
                a[j]=a[j+1];
                a[j+1]=tmp;
            }
        }
    }
    for(i=0;i<n;i++)printf("%.5f\n",a[i]);

    return 0;
}
```

K Pure Brightness

难度	考点
5	字符串处理

函数功能模块设计。本题最难的地方在于输入输出控制。以下标程给一种逐字符读入的办法。

```
#include<stdio.h>
#include<string.h>
#include<ctype.h>
#include<math.h>

const double PI=acos(-1);

double toradian(double degree)
{
    return degree/180*PI;
}

double todegree(double radian)
{
    return radian/PI*180;
}

char input[110];
int inputtop;

int parse()
{
    if(strcmp(input,"springequinox")==0)
    {
        return 0;
    }
    else if(strcmp(input,"purebrightness")==0)
```

```
28      {
29          return 1;
30      }
31      else if(strcmp(input,"grainrain")==0)
32      {
33          return 2;
34      }
35      else if(strcmp(input,"beginningofsummer")==0)
36      {
37          return 3;
38      }
39      else if(strcmp(input,"grainbuds")==0)
40      {
41          return 4;
42      }
43      else if(strcmp(input,"graininear")==0)
44      {
45          return 5;
46      }
47      else if(strcmp(input,"summersolstice")==0)
48      {
49          return 6;
50      }
51      else if(strcmp(input,"minorheat")==0)
52      {
53          return 7;
54      }
55      else if(strcmp(input,"majorheat")==0)
56      {
57          return 8;
58      }
59      else if(strcmp(input,"beginningofautumn")==0)
60      {
61          return 9;
62      }
63      else if(strcmp(input,"endofheat")==0)
64      {
65          return 10;
66      }
67      else if(strcmp(input,"whitedew")==0)
68      {
69          return 11;
70      }
71      else if(strcmp(input,"autumnequinox")==0)
72      {
73          return 12;
74      }
75      else if(strcmp(input,"colddew")==0)
76      {
77          return 13;
78      }
79      else if(strcmp(input,"frost\'sdescent")==0)
80      {
81          return 14;
82      }
83      else if(strcmp(input,"beginningofwinter")==0)
84      {
85          return 15;
```

```
    }
    else if(strcmp(input,"minorsnow")==0)
    {
        return 16;
    }
    else if(strcmp(input,"majorsnow")==0)
    {
        return 17;
    }
    else if(strcmp(input,"wintersolstice")==0)
    {
        return 18;
    }
    else if(strcmp(input,"minorcold")==0)
    {
        return 19;
    }
    else if(strcmp(input,"majorcold")==0)
    {
        return 20;
    }
    else if(strcmp(input,"beginningofspring")==0)
    {
        return 21;
    }
    else if(strcmp(input,"rainwater")==0)
    {
        return 22;
    }
    else if(strcmp(input,"awakeningofinsects")==0)
    {
        return 23;
    }
    else
    {
        return -1;
    }
}

char readline()
{
    inputtop=0;
    while(1)
    {
        char c=getchar();
        if(c=='\n')
        {
            input[inputtop]='\0';
            inputtop++;
            return '\n';
        }
        else if(c==EOF)
        {
            input[inputtop]='\0';
            inputtop++;
            if(inputtop!=1)
            {
                return '\n';
```

```
            }
            else
            {
                return EOF;
            }
        }
        else if(isspace(c))
        {
            continue;
        }
        else if(isupper(c))
        {
            input[inputtop]=tolower(c);
            inputtop++;
        }
        else
        {
            input[inputtop]=c;
            inputtop++;
        }
    }
}

int main()
{
    double epsilon=23+26.0/60.0;
    epsilon=toradian(epsilon);
    while(readline()!=EOF)
    {
        int ans=parse();
        if(ans==-1)
        {
            printf("This day is not one of the 24 Solar Terms.\n");
            continue;
        }
        double lambda=toradian(ans*15);
        double delta=asin(sin(epsilon)*sin(lambda));
        double alpha;
        if(ans==0)
        {
            alpha=0;
        }
        else if(ans>0&&ans<6)
        {
            alpha=atan(cos(epsilon)*tan(lambda));
        }
        else if(ans==6)
        {
            alpha=PI/2;
        }
        else if(ans>6&&ans<12||ans>12&&ans<18)
        {
            alpha=atan(cos(epsilon)*tan(lambda))+PI;
        }
        else if(ans==12)
        {
            alpha=PI;
        }
```

```
202             else if(ans==18)
203             {
204                 alpha=PI/2*3;
205             }
206             else if(ans>18&&ans<23)
207             {
208                 alpha=atan(cos(epsilon)*tan(lambda))+2*PI;
209             }
210             alpha=todegree(alpha);
211             delta=todegree(delta);
212             printf("(%.3lf, %.3lf)\n",alpha,delta);
213         }
214     }
```