Ng
January 7, 2015

期末救星

往年期末考试题解题报告

Running Test

Description

80000除以k，结果向上取正

Hint

常用 cmath 函数

ceil ———————— 向上取正

round ———————— 四舍五入

floor ———————— 向下取正

Code

```
#include <iostream>
#include <cmath>

using namespace std ;

int main()
{
     int x ;
     cin >> x ;
     double r = 800 * 100.0 ;
     cout << (int)ceil(r/x) << endl ;
     return 0 ;
}
```

Cross ?

Description

一个 2 X 2 的矩阵，由两个1和两个2组成，将相同的数连起来。

若形成一个 X 则输出Yes，反之输出No 。

Code

```
#include <iostream>

using namespace std ;

int N,a,b,c,d ;

int main()
{
     cin >> N ;
     for (int i = 1 ; i <= N ; ++ i)
     {
          cin >> a >> b >> c >> d ;
          if(a == d && b == c)
               cout << "Yes" << endl ;
          else cout << "No" << endl ;
     }
     return 0 ;
}
```

Bouncing Sequence

Description

一个长度为 n (1<=n<=4000) 的序列 a ， 求它的 bouncing 序列 b ， 序列 b 满足:

b_0 为 a 中的最小值

b_1 为 a 中除了 {b_0} 的最大值

b_2 为 a 中除了 {b_0 b_1} 的最小值

b_3 为 a 中除了 {b_0 b_1 b_2} 的最大值

……

Hint

i & 1 等价于 i % 2 ， 然而两者的运算优先级不同，小心使用

在语句 ：

if (表达式)

while (表达式)

……

之中，只要表达式的结果不为 0 ， 则代表 true ， 反之为 false 。

语句 i++ 表示先调用i，再执行 i = i + 1 ;

语句 ++i 表示先执行 i = i + 1 ; 再调用 i

本题只需在排序后，按照要求的顺序输出

Code

```
#include <cstdio>
#include <iostream>
#include <algorithm>

using namespace std ;

int a[4000] ;

int N ;

int main()
{
    cin >> N ;
    for (int i = 0 ; i < N ; ++ i)  scanf("%d",&a[i]) ;

    sort(a , a + N) ;

    int l = 0 , r = N - 1 ;
    for (int i = 0 ; i < N ; ++ i)  if(i & 1)
        printf("%d\n",a[r--]) ;
    else
        printf("%d\n",a[l++]) ;

    return 0 ;
}
```

Love Letter

Description

给出字符串 S_1 S_2

已知 $S_n = S_{n-2} + S_{n-1}$

求 S_k 中字串 “AC” 的出现次数是否为 x 次

输出 Accepted (是) 或 Forever alone (否)

Hint

直接求出S_k，暴力枚举AC数

Code

```
#include <iostream>
#include <string>
#include <cstdio>

using namespace std ;

string s2 , s1 ;

int k,x ;

int main()
{
    cin >> k >> x ;
    cin >> s1 >> s2 ;

    for (int i = 3 ; i <= k ; ++ i)
    {
```

```
        swap(s1 , s2) ;
        s2 = s2 + s1 ;
    }

    int u = (int) s2.size() ; int ans = 0 ;
    for(int i = 0 ; i < u - 1 ; i ++)
         if('A' == s2[i] && 'C' == s2[i + 1])
              ans ++ ;

    if(ans == x)
            printf("Accepted\n") ;
    else    printf("Forever alone\n") ;

    return 0 ;
}
```

Code

```
//看看就好的高智商做法
//如果给定的字符串长度非常大，只有这种方法可以过

#include <iostream>
#include <string>
#include <cstdio>

using namespace std ;

string s2 , s1 ;

int f[20] ;

int k,x ;

int main()
{
    cin >> k >> x ;
    cin >> s1 >> s2 ;

    int len = (int) s1.size() ;
    for (int i = 0 ; i < len - 1 ; ++ i)
        if('A' == s1[i] && 'C' == s1[i + 1])
            f[1] ++ ;
```

```
    len = (int) s2.size() ;
    for (int i = 0 ; i < len - 1 ; ++ i)
        if('A' == s2[i] && 'C' == s2[i + 1])
            f[2] ++ ;

    char z[2] ; z[0] = * s1.begin() ; z[1] = *
s1.rbegin() ;
    char y[2] ; y[0] = * s2.begin() ; y[1] = *
s2.rbegin() ;

    for (int i = 3 ; i <= k ; i ++)
    {
        f[i] = f[i - 2] + f[i - 1] ;
        if('A' == z[1] && 'C' == y[0])
            f[i] ++ ;
        swap(z , y) ; y[1] = z[1] ;
    }

    cout << ((f[k] == x) ? ("Accepted") : ("Forever
alone")) << endl ;

    return 0 ;
}
```

Circle Line

Description

给出，每两个相邻车站之间的距离，求任意两车站间的距离

Hint

这里使用了前缀和优化，sum[i]表示从车站1到车站i+1的距离之和

Code

```
#include <iostream>

using namespace std ;

int sum[110] ;

int n,s,t ;

int main()
{
    cin >> n ;
    for (int i = 1 ; i <= n ; ++ i)
    {
        cin >> sum[i] ;
        sum[i] = sum[i] + sum[i - 1] ;
    }
    cin >> s >> t ; if(s > t)   swap(s , t) ;
    cout << min(sum[t - 1] - sum[s - 1] , sum[s - 1] +
sum[n] - sum[t - 1]) << endl ;
    return 0 ;
}
```

Encode the string

Description

缩短字符串，当出现重复字符时，用该字符连续出现的次数取代其后续的相同字符。

Hint

putchar函数可以输出单个字符

Code

```
#include <cstdio>
#include <cstring>

int main()
{
    char a[100010] ;
    while(scanf("%s\n",a) != EOF)
    {
        int len = (int) strlen(a) ;
        putchar(a[0]) ; int cnt = 1 ;
        for(int i = 1 ; i < len ; i ++)
            if(a[i] != a[i - 1])
            {
                if(cnt > 1) printf("%d",cnt) ;
                cnt = 1 ; putchar(a[i]) ;
            }else   cnt ++ ;
        if(cnt > 1) printf("%d",cnt) ;
        putchar('\n');
    }
    return 0 ;
}
```

LOL

Description

有三个上几届的校队，他们想约两个妹子玩LOL，给出他们一天中的休闲时间和多个妹子的休闲时间，求他们能不能空出时间一起玩。

Hint

将时间转换，以秒为单位。标记每一秒有多少个汉子和妹子。如果汉子全体都在，妹子多于两个，LOL就成了。

Code

```
#include <cstdio>
#include <cstring>
#include <iostream>

using namespace std ;

const int TMax = 24 * 3600 + 10 ;

int Girl[TMax] ;
int Cow[TMax] ;
int p[TMax] ;

void Add_Person(int x,int a[])
{
    memset(p,0,sizeof(p)) ; for(int i = 1 ; i <= x ; i +
+)
    {
        int a1,a2,a3,b1,b2,b3 ; scanf("%d:%d:%d %d:%d:
%d",&a1,&a2,&a3,&b1,&b2,&b3) ;
        int st = a1 * 3600 + a2 * 60 + a3 ;
        int ed = b1 * 3600 + b2 * 60 + b3 ;
```

```
        if(st <= ed)    {++ p[st] ; -- p[ed] ;}
    }

    int count = 0 ; for(int i = 0 ; i < TMax ; i ++)
    {
        count += p[i] ; if(count > 0)
            a[i] += 1 ;
    }
}

bool Three_Cow()
{
    int a,b,c ;
    if(scanf("%d %d %d",&a,&b,&c) == EOF)
        return false ;
    memset(Cow,0,sizeof(Cow)) ;
    memset(Girl,0,sizeof(Girl)) ;
    Add_Person(a,Cow) ; Add_Person(b,Cow) ;
Add_Person(c,Cow) ;
    return true ;
}

void Out24(int x)
{
    int a1 = x / 3600 ;
    int a2 = (x - a1 * 3600) / 60 ;
    int a3 = x - a1 * 3600 - a2 * 60 ;
    printf("%02d:%02d:%02d",a1,a2,a3) ;
}

bool Common_Time_Check()
{
    bool tag = false ; int last = -1 ;

    for(int i = 0 ; i < TMax ; ++ i)
    {
        if(Girl[i] < 2 || Cow[i] < 3)
        {
            if(last != -1)
            {
                Out24(last) ; putchar(' ') ; Out24(i) ;
putchar('\n') ;
                last = -1 ; tag = true ;
```

```
            }
        }
        else    if(-1 == last)  last = i ;
    }

    return tag ;
}

int main()
{
    bool first_test_data = true ;

    while(Three_Cow())
    {
        if(first_test_data)
            first_test_data = false ;
        else    puts("") ;

        int n ; scanf("%d",&n) ; for (int i = 1 ; i <=
n ; ++ i)
        {
            int m ; scanf("%d",&m) ;
            Add_Person(m,Girl) ;
        }

        if(!Common_Time_Check())
            puts("You cannot start a game!") ;
    }

    return 0 ;
}
```

Star and Matrix

Description

给定 n X m 矩阵，每次操作可将矩阵中某一元素加或减 d

求最少几次操作后，矩阵的每一个元素都相同。

Code

```
#include <iostream>
#include <cstdio>

using namespace std ;

const int N = 110 ;

int a[N][N] ;

int n,m,d ;

int my_abs(int x)   {return (x > 0) ? (x) : (-x) ;}

int main()
{
    cin >> n >> m >> d ;

    for (int i = 1 ; i <= n ; ++ i)
        for (int j = 1 ; j <= m ; ++ j)
                scanf("%d",&a[i][j]) ;

    bool succ = true ; int ans = 2147483647 ;

    for (int i = 1 ; i <= n ; ++ i)
        for (int j = 1 ; j <= m ; ++ j)
            if((a[i][j] % d) != (a[1][1] % d))
                succ = false ;
```

```
    if(!succ)   ans = -1 ;
    else
    {
        for(int i = 1 ; i <= 10000 ; ++ i) if((i % d) ==
(a[1][1] % d))
        {
            int cnt = 0 ;
            for (int x1 = 1 ; x1 <= n ; ++ x1)
                for (int y1 = 1 ; y1 <= m ; ++ y1)
                {
                    cnt += my_abs(a[x1][y1] - i) / d ;
                    if(cnt > ans)   break ;
                }
            ans = min(ans , cnt) ;
        }
    }

    cout << ans << endl ;

    return 0 ;
}
```

Fractal

Description

这题上次做过了

Code

```
//感谢我的组员，他们指出了过去的代码中出现的错误@王晨阳 @刘聪

#include <cstdio>
#include <cstring>

char s[2187][2187] ;

int p[10] ;

void A(int x,int y,int dep)
{
    for(int i = 0 ; i < p[dep - 1] ; ++ i)
        for(int j = 0 ; j < p[dep - 1] ; ++ j)
            s[x + i][y + j] = '.' ;
}

void B(int x,int y,int dep)
{
    if(dep == 1)
    {
        s[x][y] = 'X' ;
        return ;
    }
    dep -= 1 ;
    B(x , y , dep) ;
    A(x + p[dep - 1] , y , dep) ;
    B(x + p[dep - 1] * 2 , y , dep) ;
    A(x , y + p[dep - 1] , dep) ;
    B(x + p[dep - 1] , y + p[dep - 1] , dep) ;
```

```
    A(x + p[dep - 1] * 2 , y + p[dep - 1] , dep) ;
    B(x , y + p[dep - 1] * 2 , dep) ;
    A(x + p[dep - 1] , y + p[dep - 1] * 2 , dep) ;
    B(x + p[dep - 1] * 2 , y + p[dep - 1] * 2 , dep) ;
}

int main()
{
    p[0] = 1 ;
    for(int i = 1 ; i <= 9 ; ++ i)
        p[i] = p[i - 1] * 3 ;

    int n ; while(scanf("%d",&n) , n != -1)
    {
        memset(s,0,sizeof(s)) ;
        B(0 , 0 , n) ;
        for(int i = 0 ; i < p[n - 1] ; ++ i)
            printf("%s\n",s[i]) ;
        puts("-") ;
    }

    return 0 ;
}
```

Line

Description

找出每个数左侧第一个比它大的数的编号，不存在则算作0

输出这些编号的和。

Hint

这里使用了数据结构————栈 stack

从后向前枚举每一个元素

当枚举一个新的元素，若新元素比栈顶的元素大，这时，将让栈中所有比它小的元素出栈，每出一个向结果中加一次新元素的编号，最后再将新元素放入栈中。

正确性证明如下：

新元素较后入栈，所以它一定在栈内元素的左侧，而且它也是第一个比这些元素大的。因为如果先前的元素无法令栈中元素出栈，那么，栈中元素一定更大。更深一步，这个栈具有单调性。不可能存在一个元素比它底层的元素更大，因为在它入栈前，这些比它小的元素已经完全出栈了。

Code

```
#include <iostream>
#include <cstdio>
#include <stack>

using namespace std ;
```

```
const int N = 1000010 ;

int a[N] ;

int n ;

stack <int> s ;

int main()
{
    int T ; cin >> T ;
    for (int _T = 1 ; _T <= T ; ++ _T)
    {
        cin >> n ;
        for (int i = 1 ; i <= n ; ++ i)
            scanf("%d",&a[i]) ;
        long long ans = 0 ;
        for(int i = n ; i > 0 ; -- i)
        {
            while( (int) s.size() > 0 && (s.top() <
a[i]))
            {
                s.pop() ;
                ans += i ;
            }
            s.push(a[i]) ;
        }
        while ( (int) s.size() > 0)
            s.pop() ;
        cout << ans << endl ;
    }
    return 0 ;
}
```