CSP/NOIP 入门级模拟赛 Day6 - Solution

第 1 题：购物问题

【算法分析】

简单模拟，将每行第一个数除以十再乘以第二个数，再把每行的结果加起来保留两位小数即可（全程要开 double）。

【核心代码】

```
ans += (double)a / 10.0 * (double)b;
```

【参考程序】

```
#include<bits/stdc++.h>
#define ll long long
#define re register
using namespace std;
double ans=0;//一定要用 double 统计答案
inline int read()
{
    int x=0,f=1;char c=getchar();
    while(c>'9'||c<'0'){if(c=='-')f=-f;c=getchar();}
    while(c<='9'&&c>='0'){x=x*10+c-'0';c=getchar();}
    return x*f;
}
int main()
{
    freopen("money.in","r",stdin);
    freopen("money.out","w",stdout);
    int n=read();
    for(re int i=1;i<=n;i++)
    {
        int a=read(),b=read();
        ans+=(double)a/10.0*(double)b;//强转 double 计算
    }
    printf("%.2lf\n",ans);//保留两位小数用格式化输出
    return 0;
}
```

第 2 题：买票问题

【算法分析】

运用队列，设头尾指针，头指针不动，尾指针尽量往外扩。一直到扩不动，在把头指针往后移。这样能保证每一个元素只进队出队一次，复杂度是 O（N）的。

【核心代码】

```
for (int i = 1; i <= n; ++ i){
        while (m <= n && s[m] - s[i - 1] <= f)  m ++;
```

```
        Max=max(Max, m - i);
        if (m == n + 1) break;
}
```

【参考程序】

```
#include <bits/stdc++.h>
#define N 1000005
using namespace std;
int n, m, p, Max;
int a[N], s[N];
int main() {
    scanf("%d%d", &n, &m);
    for (int i = 1; i <= n; ++i) scanf("%d", &a[i]), s[i] = s[i - 1] +
a[i];
    P = 1;
    for (int i = 1; i <= n; ++i) {
        while (p<= n && s[p] - s[i - 1] <= m) p ++;
        Max = max(Max, p - i);
        if (p == n + 1)
            break;
    }
    printf("%d", Max);
}
```

第 3 题：奇怪函数

【算法分析】

一道类数学题。前置知识：x 的位数为 log10（x）+1.

由提示得 log10（x^x）=x*log10(x)，

则可二分 x 的大小，判断 x*log10（x）是否大于等于 n，

二分结果即为答案。

【核心代码】

```
inline int check(int x, int n) //判断 x^x 的位数是否有 n 位
{
        double cheng = (double)x * log10(x) + 1;
        if(cheng - n >= 0)      return 1;
        else return 0;
}
while(l <= r) //二分枚举 x
 {
         int mid = l + r >> 1;
         int flag = check(mid,n);
         if(flag)  r = mid - 1, ans = mid;
         else l = mid + 1;
 }
```

【参考程序】

```
#include<bits/stdc++.h>
#define ll long long
#define re register
using namespace std;
inline int read()
{
    int x=0,f=1;char c=getchar();
    while(c<'0'||c>'9'){if(c=='-')f=-f;c=getchar();}
    while(c>='0'&&c<='9'){x=x*10+c-'0';c=getchar();}
    return x*f;
}
inline int check(int x,int n)//判断 x^x 的位数是否有 n 位
{
    double cheng=(double)x*log10(x)+1;
    if(cheng-n>=0)return 1;
    else return 0;
}
int main()
{
    freopen("xx.in","r",stdin);
    freopen("xx.out","w",stdout);
    int n=read();
    int l=1,r=9e8,ans;
    while(l<=r)//二分枚举 x
    {
        int mid=l+r>>1;
        int flag=check(mid,n);
        if(flag)r=mid-1,ans=mid;
        else l=mid+1;
    }
    printf("%d\n",ans);
    return 0;
}
```

第 4 题：地形调查

【算法分析】

bfs 把所有高度相同且相邻的点合并成一个块，如果一个块周围没有比它高的点，就是山峰，如果一个块周围没有比它低的点，就是湖泊。合并信息可用 bfs 实现。

【核心代码】

```
inline void bfs(int x, int y)  //套 bfs 模板
{
```

```
    int h = 0, s = 0;
    vis[x][y] = 1;
    h = max(higher[x][y], h);
    s = max(s, shorter[x][y]);
    queue<pair<int, int> > q;
    q.push(make_pair(x, y));
    while (!q.empty()) {
        int a = q.front().first, b = q.front().second;
        q.pop();
        if (!vis[a - 1][b] && a > 1 && mapp[a - 1][b] == mapp[a][b]) {
            vis[a - 1][b] = 1;
            h = max(h, higher[a - 1][b]);
            s = max(s, shorter[a - 1][b]);
            q.push(make_pair(a - 1, b));
        }
        if (!vis[a][b - 1] && b > 1 && mapp[a][b - 1] == mapp[a][b]) {
            vis[a][b - 1] = 1;
            h = max(h, higher[a][b - 1]);
            s = max(s, shorter[a][b - 1]);
            q.push(make_pair(a, b - 1));
        }
        if (!vis[a + 1][b] && a < n && mapp[a + 1][b] == mapp[a][b]) {
            vis[a + 1][b] = 1;
            h = max(h, higher[a + 1][b]);
            s = max(s, shorter[a + 1][b]);
            q.push(make_pair(a + 1, b));
        }
        if (!vis[a][b + 1] && b < m && mapp[a][b + 1] == mapp[a][b]) {
            vis[a][b + 1] = 1;
            h = max(h, higher[a][b + 1]);
            s = max(s, shorter[a][b + 1]);
            q.push(make_pair(a, b + 1));
        }
    }
    if (s == 0)
        ansl++;  //周围没有点比他低，是湖泊
    if (h == 0)
        ansm++;  //周围没有点比他高，是山峰
    return;
}
```

【参考程序】

```
#include<bits/stdc++.h>
#define ll long long
```

```
#define re register
using namespace std;
inline int read()
{
    int x=0,f=1;char c=getchar();
    while(c<'0'||c>'9'){if(c=='-')f=-f;c=getchar();}
    while(c>='0'&&c<='9'){x=x*10+c-'0';c=getchar();}
    return x*f;
}
int mapp[1010][1010],vis[1010][1010];
int ansm,ansl;               //ansl 表示湖泊，ansm 表示山峰
int higher[1010][1010],shorter[1010][1010],n,m;

inline void bfs(int x,int y) //套 bfs 模板
{
    int h=0,s=0;
    vis[x][y]=1;
    h=max(higher[x][y],h);
    s=max(s,shorter[x][y]);
    queue<pair<int,int > >q;
    q.push(make_pair(x,y));
    while(!q.empty())
    {
        int a=q.front().first,b=q.front().second;
        q.pop();
        if(!vis[a-1][b]&&a>1&&mapp[a-1][b]==mapp[a][b])
        {
            vis[a-1][b]=1;
            h=max(h,higher[a-1][b]);
            s=max(s,shorter[a-1][b]);
            q.push(make_pair(a-1,b));
        }
        if(!vis[a][b-1]&&b>1&&mapp[a][b-1]==mapp[a][b])
        {
            vis[a][b-1]=1;
            h=max(h,higher[a][b-1]);
            s=max(s,shorter[a][b-1]);
            q.push(make_pair(a,b-1));
        }
        if(!vis[a+1][b]&&a<n&&mapp[a+1][b]==mapp[a][b])
        {
            vis[a+1][b]=1;
            h=max(h,higher[a+1][b]);
            s=max(s,shorter[a+1][b]);
```

```
            q.push(make_pair(a+1,b));
        }
        if(!vis[a][b+1]&&b<m&&mapp[a][b+1]==mapp[a][b])
        {
            vis[a][b+1]=1;
            h=max(h,higher[a][b+1]);
            s=max(s,shorter[a][b+1]);
            q.push(make_pair(a,b+1));
        }
    }
    if(s==0)ansl++;//周围没有点比他低，是湖泊
    if(h==0)ansm++;//周围没有点比他高，是山峰
    return;
}

int main()
{
    freopen("seek.in","r",stdin);
    freopen("seek.out","w",stdout);
    n=read(),m=read();
    for(re int i=1;i<=n;i++)
        for(re int j=1;j<=m;j++)
            mapp[i][j]=read();
    for(re int i=1;i<=n;i++)
        for(re int j=1;j<=m;j++)
        {
            if(i>1)
            {
                if(mapp[i-1][j]>mapp[i][j])higher[i][j]=1;
                if(mapp[i-1][j]<mapp[i][j])shorter[i][j]=1;
            }
            if(i<n)
            {
                if(mapp[i+1][j]>mapp[i][j])higher[i][j]=1;
                if(mapp[i+1][j]<mapp[i][j])shorter[i][j]=1;
            }
            if(j>1)
            {
                if(mapp[i][j-1]>mapp[i][j])higher[i][j]=1;
                if(mapp[i][j-1]<mapp[i][j])shorter[i][j]=1;
            }
            if(j<m)
            {
                if(mapp[i][j+1]>mapp[i][j])higher[i][j]=1;
```

```
                if(mapp[i][j+1]<mapp[i][j])shorter[i][j]=1;
            }
        }  //预处理出每个点四周是否有比他高（低）的点
    for(re int i=1;i<=n;i++)
        for(re int j=1;j<=m;j++)
        {
            if(vis[i][j])continue;
            bfs(i,j);
        }
    cout<<ansl<<' '<<ansm<<endl;
    return 0;
}
```