签到题合集【8.10】

这次的题目～

比较简单～

比较签到～

除了第四题，其它代码都少得可怜～

就决定开学去吃日料了！

A# P5015: 标题统计

```
/*
    做题思路：签到题，可以简单的 getline()，也可以更简单的 while(cin>>s) 然后长度累计即可
    但是你会 while(cin>>?) 吗?
    反正大一上的姐姐不会...
    然后当时期中考就没拿满分...
    时间复杂度：O(?)
*/

#include <iostream>
#include <algorithm>
#define ll long long
using namespace std;

int main() {
    int ans=0;
    string s;
    while (cin >> s) ans += s.length(); // 文件EOF自动结束
    cout << ans << endl;
}
```

B# P1781: 宇宙总统

```
/*
    做题思路：字符串排序，显然 int 和 long long 都是存不下的，因此通过字符串读取再选出票数最多的
    时间复杂度：O(n)
*/

#include <iostream>
```

```cpp
#include <algorithm>
#define ll long long
using namespace std;

int main() {
    int i,j,n;
    cin >> n;
    string s,ans = "";
    for (i=0; i<n; i++) {
        cin >> s;
        if (s.length() > ans.length() or (s.length() == ans.length() and s > ans)) { // 字符串比大小
            j = i+1; // 下标从1开始
            ans = s;
        }
    }
    cout << j << endl << ans << endl;
}
```

C# P1208: 混合牛奶 Mixing Milk

```cpp
/*
    做题思路：非常简单的贪心，挑便宜的买
    时间复杂度：O(mlogm)
*/

#include <iostream>
#include <algorithm>
#define ll long long
using namespace std;

struct node {
    int x,y;
};

int cmp(node a,node b) {
    return a.x<b.x;
}

int main() {
    int i,j,n,m,ans = 0;
    cin >> n >> m;
    node a[m];
    for (i=0; i<m; i++) cin >> a[i].x >> a[i].y;
    sort(a,a+m,cmp); // 价格升序
    for (i=0; i<n; i++) {
        ans += a[i].x*min(n,a[i].y);
```

```
        n -= min(n,a[i].y);
        if (n == 0) break; // 买够走人～
    }
    cout << ans << endl;
}
```

D# P1443: 马的遍历

```
/*
    做题思路：BFS，这道题是极其罕见的只能用BFS。从起点开始，八个方向没遍历过的入队，同时标记走了多少步
    时间复杂度：O(nm)
*/

#include <iostream>
#include <algorithm>
#include <queue>
#include <iomanip>
#define ll long long
using namespace std;

int dx[8] = {1,1,-1,-1,2,2,-2,-2},dy[8] = {2,-2,2,-2,1,-1,1,-1};
int main() {
    int i,j,n,m,x,y;
    cin >> n >> m >> x >> y;
    int a[n][m];
    for (i=0; i<n; i++)
        for (j=0; j<m; j++) a[i][j] = -1;
    queue<int> q1,q2,q3;
    q1.push(x-1);
    q2.push(y-1);
    q3.push(0);
    a[x-1][y-1] = 0;
    while (q1.size()) { // BFS板子
        x = q1.front();
        y = q2.front();
        q1.pop();
        q2.pop();
        for (i=0; i<8; i++) {
            if (x+dx[i] >= 0 and x+dx[i] < n and y+dy[i] >= 0 and y+dy[i] < m and a[x+dx[i]][y+dy[i]] == -1) { // 8个方向没遍历过的入队
                a[x+dx[i]][y+dy[i]] = q3.front()+1; // 必须马上标记已遍历，不然会TLE
                q1.push(x+dx[i]);
                q2.push(y+dy[i]);
                q3.push(q3.front()+1);
            }
```

```
        }
        q3.pop();
    }
    for (i=0; i<n; i++) {
        for (j=0; j<m; j++) cout << left << setw(5) << a[i][j];
        cout << endl;
    }
}
```

E# P1049: 装箱问题

```
/*
    做题思路：就如同寒假的学习任务【1.31】所说：表面上是个搜索，实际上是个01
    但是不知道DFS+回溯能不能过，可能会超时～
    时间复杂度：O(nV)
*/

#include <iostream>
#include <algorithm>
#define ll long long
using namespace std;

int dp[20001];
int main() {
    int i,j,n,m;
    cin >> n >> m;
    int v[m];
    for (i=0; i<m; i++) cin >> v[i]; // 体积 = 价值
    for (i=0; i<m; i++)
        for (j=n; j>=v[i]; j--)
            dp[j] = max(dp[j],dp[j-v[i]]+v[i]);
    cout << n-dp[n] << endl; // 求最小空余体积 = 原体积-最大价值
}
```

F# P1115: 最大子段和

```
/*
    做题思路：贪心 or dp。如果有印象的话，这是数据结构课本第一道题。
    时间复杂度：O(n)
*/

// 贪心
#include <iostream>
#include <algorithm>
```

```cpp
#define ll long long
using namespace std;

int main() {
    int i,j,n,x,num=0,ans = -99999999;
    cin >> n;
    for (i=0; i<n; i++) {
        cin >> x;
        num += x; // 当前子段和
        ans = max(ans,num); // 更新最大值
        if (num < 0) num = 0; // 如果当前子段和小于零，则最终子段必不包括该子段，因此重新计数
    }
    cout << ans << endl;
}

// dp：设dp[i]为子段尾部下标为i的所有子段的最大子段和，这些子段有两种：子段尾部下标为i-1的子段+a[i]、只有a[i]。因此有转移方程 dp[i] = max(dp[i-1]+a[i],a[i])
#include <iostream>
#include <algorithm>
#define ll long long
using namespace std;

int main() {
    int i,j,n,ans = -99999999;
    cin >> n;
    int dp[n];
    for (i=0; i<n; i++) cin >> dp[i]; // dp[i]先初始化为a[i]
    for (i=1; i<n; i++) dp[i] += max(0,dp[i-1]); // 与转移方程等价
    for (i=0; i<n; i++) ans = max(ans,dp[i]); // 找最大～
    cout << ans << endl;
}
```

G# P1507: NASA的食物计划

```cpp
/*
    做题思路：二维01背包，为原01背包变种，处理方式只需把原一维背包体积变为二维背包体积即可。
    时间复杂度：O(M)
*/

#include <iostream>
#include <algorithm>
#define ll long long
using namespace std;

int dp[401][401];
int main() {
```

```
    int i,j,k,n,m,t;
    cin >> n >> m >> t;
    int w1[t],w2[t],v[t];
    for (i=0; i<t; i++) cin >> w1[i] >> w2[i] >> v[i];
    for (i=0; i<t; i++)
        for (j=n; j>=w1[i]; j--)
            for (k=m; k>=w2[i]; k--)
                dp[j][k] = max(dp[j][k],dp[j-w1[i]][k-w2[i]]+v[i]);
    cout << dp[n][m] << endl;
}
```