# NingBo University of Technology ACM Programming Contest

# 题解报告

**Problem A: 斯丢彼得饽役**

题意：签到题，判断两人的魅力大小

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. #define ll long long
5. **int** main(){
6. **int** t;
7. **int** a,b,c,d;
9. cin>>t;
10. **while**(t--){
11. cin>>a>>b>>c>>d;
12. **if**(a\*b>c\*d) cout<<"Idiot"<<endl;
13. **else** cout<<"Foolish"<<endl;
14. }
16. **return** 0;
17. }

## Problem B: Wild growth

题意：三个人三个身高，有一种药水有两种功能

1. 使一人长高2cm
2. 使两人分别长高1cm

求最少用多少瓶药水使得三人一样高

题解：将三人按身高从小到大排序，先将最矮的用数次药水的第一种功能长高到中间那个人的高度，如果比第二个人高1cm，则给第二个人和第三个人用一次药水的第二种功能，最后给第一个人和第二个人用数次药水的第二种的功能直到三人的身高一样高。

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. #define ll long long
5. **int** main(){
6. **int** t;
7. **int** a[3];
8. **int** ans;
10. cin>>t;
11. **while**(t--){
12. ans=0;
13. cin>>a[0]>>a[1]>>a[2];
14. sort(a,a+3);
15. **if**((a[1]-a[0])%2==1){
16. ans++;
17. a[1]++;
18. a[2]++;
19. }
20. ans+=(a[1]-a[0])/2;
21. ans+=a[2]-a[1];
22. cout<<ans<<endl;
23. }
24. **return** 0;
25. }

## Problem C: WC有趣的暑假时光

题解：先走到最短的边的点上，然后来回走最短的边

1. #include<iostream>
2. **using** **namespace** std;
4. **int** main(){
5. **int** t,n,a,b,c,ans;
7. cin>>t;
8. **while**(t--){
9. cin>>n>>a>>b>>c;
10. ans=0;
11. **if**(n==1) ans=0;
12. **else**{
13. **int** MIN=min(a,min(b,c));
14. **if**(a==MIN) ans=(n-1)\*a;
15. **else** **if**(b==MIN) ans=(n-1)\*b;
16. **else** ans=min(a,b)+(n-2)\*c;
17. }
18. cout<<ans<<endl;
19. }
20. }

## Problem D: 龙王的围棋！

题解：我们容易得到，在所有棋子经过翻转后，最边上一圈的棋子均翻了偶数次，所以还是黑色，而中间的棋子均翻了奇数次（9次），所以变成白色，最后只用统计棋盘中间的白子有多少个即可。

Ps：特判棋盘长或宽只有1的情况。

1. #include<iostream>
2. **using** **namespace** std;
4. **int** main(){
5. **int** t,n,m,ans;
7. cin>>t;
8. **while**(t--){
9. cin>>n>>m;
10. ans=0;
11. **if**(n==1&&m==1) ans=0;
12. **else** **if**(n==1) ans=m-2;
13. **else** **if**(m==1) ans=n-2;
14. **else**{
15. ans=(n-2)\*(m-2);
16. }
17. **if**(ans%2==1) cout<<"そら ぎんこ"<<endl;
18. **else** cout<<"ひなつる あい"<<endl;
19. }
20. **return** 0;
21. }

## Problem E: 淘金币系列A

## Problem F: 德克萨斯扑克

题解：模拟各种牌型即可

1. #include<iostream>
2. #include<algorithm>
3. **using** **namespace** std;
5. **int** type(**int** a[]){
6. **int** num[15]={0};
7. **int** \_4,\_3,\_2;
8. \_4=\_3=\_2=0;
10. **for**(**int** i=1;i<=5;i++) num[a[i]]++;
12. **for**(**int** i=2;i<=14;i++){
13. **if**(num[i]==4) \_4++;
14. **if**(num[i]==3) \_3++;
15. **if**(num[i]==2) \_2++;
16. }
18. **if**(\_4==1) **return** 7;
19. **if**(\_3==1&&\_2==1) **return** 6;
20. **if**(\_3==1) **return** 4;
21. **if**(\_2==2) **return** 3;
22. **if**(\_2==1) **return** 2;
24. **for**(**int** i=2;i<=5;i++){
25. **if**(a[i-1]+1!=a[i]) **return** 1;
26. }
28. **return** 5;
29. }
31. **int** check1(**int** a[],**int** b[]){
32. **int** a21,a22,a1;
33. **int** b21,b22,b1;
34. a21=a22=a1=b21=b22=b1;
36. **if**(a[1]==a[2]||a[2]==a[3]) a21=a[1];
37. **if**(a[3]==a[4]||a[4]==a[5]) a22=a[4];
38. **for**(**int** i=1;i<=5;i++){
39. **if**(a[i]!=a21&&a[i]!=a22) a1=a[i];
40. }
42. **if**(b[1]==b[2]||b[2]==b[3]) b21=b[1];
43. **if**(b[3]==b[4]||b[4]==b[5]) b22=b[4];
44. **for**(**int** i=1;i<=5;i++){
45. **if**(b[i]!=b21&&b[i]!=b22) b1=b[i];
46. }
48. **if**(a22>b22) **return** 1;
49. **if**(a22<b22) **return** -1;
50. **if**(a21>b21) **return** 1;
51. **if**(a21<b21) **return** -1;
52. **if**(a1>b1) **return** 1;
53. **if**(a1<b1) **return** -1;
54. **return** 0;
56. }
58. **int** check2(**int** a[],**int** b[]){
59. **int** a2,b2;
60. **for**(**int** i=1;i<=4;i++){
61. **if**(a[i]==a[i+1]){
62. a2=a[i];
63. a[i]=a[i+1]=15;
64. }
65. }
67. **for**(**int** i=1;i<=4;i++){
68. **if**(b[i]==b[i+1]){
69. b2=b[i];
70. b[i]=b[i+1]=15;
71. }
72. }
74. sort(a+1,a+1+5);
75. sort(b+1,b+1+5);
77. **if**(a2>b2) **return** 1;
78. **if**(a2<b2) **return** -1;
79. **for**(**int** i=1;i<=3;i++){
80. **if**(a[i]>b[i]) **return** 1;
81. **if**(a[i]<b[i]) **return** -1;
82. }
83. **return** 0;
84. }
86. **int** check3(**int** a[],**int** b[]){
87. **for**(**int** i=5;i>=1;i--){
88. **if**(a[i]>b[i]) **return** 1;
89. **if**(a[i]<b[i]) **return** -1;
90. }
91. **return** 0;
92. }
94. **int** main(){
95. **int** t;
96. **int** a[6];
97. **int** b[6];
98. cin>>t;
99. **while**(t--){
100. **for**(**int** i=1;i<=5;i++) cin>>a[i];
101. **for**(**int** i=1;i<=5;i++) cin>>b[i];
102. sort(a+1,a+1+5);
103. sort(b+1,b+1+5);
104. **int** typeA=type(a);
105. **int** typeB=type(b);
106. **int** flag=0;
107. **if**(typeA>typeB) flag=1;
108. **if**(typeA<typeB) flag=-1;
109. **if**(typeA==typeB){
110. **if**(typeA==7||typeA==6||typeA==4) flag=(a[3]>b[3]?1:-1);
111. **if**(typeA==5){
112. **if**(a[1]==b[1]) flag=0;
113. **if**(a[1]>b[1]) flag=1;
114. **if**(a[1]>b[1]) flag=-1;
115. }
116. **if**(typeA==3) flag=check1(a,b);
117. **if**(typeA==2) flag=check2(a,b);
118. **if**(typeA==1) flag=check3(a,b);
119. }
120. **if**(flag==1) cout<<"WIN"<<endl;
121. **if**(flag==-1) cout<<"LOSE"<<endl;
122. **if**(flag==0) cout<<"DRAW"<<endl;
123. }
124. }

## Problem G: Product subsequence

题意：有一个长度为n的数组，求一段连续序列使得其乘积最大

题解：

对于暴力解法，两个for求所有情况，时间复杂度为，n的范围为10^5，会TLE

所以我们用DP的做法

MAX[i]表示以a[i]为结尾的连续序列的最大值

MIN[i]表示以a[i]为结尾的连续序列的最小值

转移方程为

MAX[i]=max(a[i],max(MAX[i-1]\*a[i],MIN[i-1]\*a[i]));

MIN[i]=min(a[i],min(MAX[i-1]\*a[i],MIN[i-1]\*a[i]));

时间复杂度为

1. #include<bits/stdc++.h>
2. **using** **namespace** std;
3. #define ll long long
5. **int** t;
6. **int** n;
7. ll a[100000+5];
8. ll MAX[100000+5];
9. ll MIN[100000+5];
10. ll ans;
12. **int** main(){
13. cin>>t;
14. **while**(t--){
15. cin>>n;
16. **for**(**int** i=1;i<=n;i++) cin>>a[i];
17. ans=MAX[1]=MIN[1]=a[1];
19. **for**(**int** i=2;i<=n;i++){
20. MAX[i]=max(a[i],max(MAX[i-1]\*a[i],MIN[i-1]\*a[i]));
21. MIN[i]=min(a[i],min(MAX[i-1]\*a[i],MIN[i-1]\*a[i]));
22. ans=max(ans,MAX[i]);
23. }
24. cout<<ans<<endl;
25. }
27. **return** 0;
28. }

## Problem H: 达达和他的朋友们

## Problem I: 生化危机

题解：dfs或bfs均可，每次找一组有多少人，再判断这一组有多少人被感染，如果原人数等于被感染的人数，则ans++；

1. #include<iostream>
2. #include<string.h>
3. **using** **namespace** std;
5. **int** t;
6. **int** n;
7. **char** g[105][105];
8. **int** vis[105][105];
9. **int** dx[]={1,-1,0,0};
10. **int** dy[]={0,0,1,-1};
11. **int** good,bad;
13. **bool** check(**int** x,**int** y){
14. **if**(g[x-1][y]=='.'||g[x][y-1]=='.'||g[x+1][y]=='.'||g[x][y+1]=='.') **return** **true**;
15. **return** **false**;
16. }
18. **void** dfs(**int** x,**int** y){
19. vis[x][y]=1;
20. **if**(g[x][y]=='#') good++;
21. **if**(check(x,y)) bad++;
22. **for**(**int** i=0;i<4;i++){
23. **int** xx=x+dx[i];
24. **int** yy=y+dy[i];
25. **if**(1<=xx&&xx<=n&&1<=yy&&yy<=n
26. &&!vis[xx][yy]&&g[xx][yy]=='#') dfs(xx,yy);
27. }
28. }
30. **int** main(){
31. cin>>t;
32. **while**(t--){
33. cin>>n;
34. memset(vis,0,**sizeof**(vis));
35. **for**(**int** i=1;i<=n;i++){
36. **for**(**int** j=1;j<=n;j++){
37. cin>>g[i][j];
38. }
39. }
41. **int** ans=0;
42. **for**(**int** i=1;i<=n;i++){
43. **for**(**int** j=1;j<=n;j++){
44. **if**(g[i][j]=='#'&&vis[i][j]==0){
45. good=bad=0;
46. dfs(i,j);
47. **if**(good==bad) ans++;
48. }
49. }
50. }
51. cout<<ans<<endl;
52. }
53. }

## Problem J: 淘金币系列B