**按照分数排序**

**结构体sort排序问题**

#include<stdio.h>

#include<string.h>

#include<cstring>

#include<iostream>

#include<cmath>

#include<algorithm>

using namespace std;

struct stu{

string name;

int score[4];

};

stu S[101];

int cmp(stu a,stu b)

{

if(a.score[0]==b.score[0])

{

for(int i=1;i<=3;i++)

{

if(a.score[i]!=b.score[i]) return a.score[i]>b.score[i];

}

}

else return a.score[0]>b.score[0];

}

int main()

{

int n;cin>>n;

for(int i=0;i<n;i++)

cin>>S[i].name>>S[i].score[0]>>S[i].score[1]>>S[i].score[2]>>S[i].score[3];

sort(S,S+n,cmp);

for(int i=0;i<n-1;i++)

cout<<S[i].name<<' '<<S[i].score[0]<<' '<<S[i].score[1]<<' '<<S[i].score[2]<<' '<<S[i].score[3]<<endl;

cout<<S[n-1].name<<' '<<S[n-1].score[0]<<' '<<S[n-1].score[1]<<' '<<S[n-1].score[2]<<' '<<S[n-1].score[3];

}

**圆周排列+vector运用**

/\*

有3颗红珊瑚，4颗白珊瑚，5颗黄玛瑙，用它们串成一圈作为手链。

如果考虑手链可以随意转动或翻转，一共可以有多少不同的组合样式呢？

\*/

//可翻转其实就是，能reverse，不信试试

/\*

暴力的思路：不过很棒！

这个题，主要运用stl中的

string.find(str),

next\_permutation(str.begin(),str.end()),

reverse(str.begin,str.end())。

在一个vector中存下已经出现过的排列，每一个排列先在vector中查找是否已存在该排列，

若不存在，则res++，并将str+str(可任意转动)和reverse(str.begin(),str.end()) (可任意翻转) 压入vector。

这样的话，比较花时间，但是可以接受。

\*/

#include<bits/stdc++.h>

using namespace std;

vector<string> vec;

int main()

{

string str="AAABBBBCCCCC";

int res=0;

do

{

int flag=1;

for(int i=0;i<vec.size();i++)

{

if(vec[i].find(str)!=string::npos) //查找 a是否包含子串b的写法

{

flag=0;

break;

}

}

if(flag)

{

res++;

string tmp=str+str;//因为是环。。所以这样相当于连起来了

vec.push\_back(tmp);

reverse(tmp.begin(),tmp.end());

vec.push\_back(tmp);

}

}while(next\_permutation(str.begin(),str.end()));

cout<<res<<endl;

return 0;

}

/\*

使用用数学方法计算

12！/（3！\*4！\*5！\*12）=2310种，这是不考虑可以翻转的答案。

若考虑翻转需要求出那些左右对称的情况，

将1个A，1个C两边都隔5个，剩下2个A，4个B，4个C，两边对称，

即将ABBCC排列，共5！/（2\*2）=30种。

所以最终结果是30+（2310-30）/2=1170。

\*/

**糖果组合问题**

**//for循环 穷举**

#include<bits/stdc++.h>

#include<vector>

using namespace std;

vector<int>v;

int main()

{

int a,b,n=0;

cin>>a>>b;

if(a>b)

{

int t=a;a=b;b=t;

}

for(int i=b;i<=a\*b;i++)

{

for(int j=0,k=0; ;j+=b)

{

k=0;

if(j+k>i) break;

for(;;k+=a)

{

if(j+k>i) break;

if(j+k==i)

{

v.push\_back(i);

break;

}

}

if(j+k==i) break;

}

}

for(int i=v.size()-1;i>=0;i--)

{

if(v[i]-v[i-1]>1)

{

cout<<v[i]-1;

break;

}

}

}

**//DFS**

#include<bits/stdc++.h>

using namespace std;

int n,cnt=0,Flag;

int a[101];

void DFS(int x,int y,int n)

{

if(a\*x+b\*y>n) return ;

if(a\*x+b\*y==n)

{

Flag=0;

return ;

}

DFS(x+1,y,n);

DFS(x,y+1,n);

}

int main()

{

int n;cin>>n;

for(int i=1;i<=n;i++)

{

cin>>a[i];

}

for(int i=1;i<=a\*b;i++)

{

Flag=1;

DFS(0,0,i);

cout<<i<<' '<<Flag<<endl;

cnt+=Flag;

}

cout<<cnt;

}

**//回溯法 穷举**

#include<iostream>

using namespace std;

int fun(int num,int i,int j)

{

if(num < 0)

return 0;

else if(num == 0)

return 1;

else

{

return fun(num-i,i,j) + fun(num-j,i,j);

}

}

int main()

{

int i,j;

int count = 0;

cin>>i>>j;

for(int k=i;k<=i\*j;k++)

{

if(!fun(k,i,j))

{

count = k;

}

}

cout<<count<<endl;

}

**数学法**:a\*b-a-b就OK了

**最大宝藏**

/\*

在下图所示例子中,

按照5->7->8->3->7的顺序,将得到最大值30

输入：

第一行正整数N(100>=N>1),表示山的高度。

接下来有N行非负整数,第i行有i个整数(1<=i<=N),表示山的第i层上从左到右每条路上的珠宝数目

输出：

一个整数,表示从山底到山顶的所能得到的珠宝的最大数目.

样例输入

5

7

3 8

8 1 0

2 7 4 4

4 5 2 6 5

样例输出

30

\*/

#include<bits/stdc++.h>

using namespace std;

int n,a[105][105];

int d[105][105];

int DP(int x,int y)

{

if(x==n)

{

return a[x][y];

}

else

return a[x][y]+max(DP(x+1,y),DP(x+1,y+1));

}

//DP2为记忆化搜索

int DP2(int x,int y)

{

if(d[x][y]>=0)

{

return d[x][y];

}

if(x==n)

{

return d[x][y]=a[x][y];

}

else

return d[x][y]=a[x][y]+max(DP2(x+1,y),DP2(x+1,y+1));

}

int main()

{

cin>>n;

for(int i=1;i<=n;i++)

{

for(int j=1;j<=i;j++)

cin>>a[i][j];

}

memset(d,-1,sizeof(d));

cout<<DP2(1,1);

}