1. 以前很怕搜索题，这是第一道深搜题：1959 toj 石油存储,其中每搜一次就能访问一条相邻元素链。
2. Vector的使用，以后得多多用，很实际很强大的东西。Toj 1925。
3. sort()与qsort():int a[100];sort(a,a+100);char数组可以在声明时初始化,过了这行就不能再给数组赋值了！得一个字符一个字符输入并在最后加一个’\0’；假设前面有个compare()

int compare(const void \* ele1,const void \* ele2) //char\*的比较

{

return strcmp((char \*)ele1,(char \*)ele2);

}

对char s[100][7] ,qsort(s,100,sizeof(s[0]),compare);即可得按字典序的char数组。String应该也能类似的排序，但是比较函数可能容易出错，建议不使用.toj 1903.

1. 质数。3种办法：for循环除掉倍数；判断一个数是不是质数：for(i=3;i<sqrt(n);i+=2);

a[0]=2,a[1]=3,for(i=5;i<n;i+=2) for(j=1;j<s;j++) if(i%a[j]==0) break; a[s++]=I;

1. 如果两个同符号浮点数之差的绝对值小于或等于某一个可接受的误差(即精度)，就认为它们是相等的。不要直接用“==”或者“!=”对两个浮点数进行比较，但是可以直接用“<=”和“>=”比较谁大谁小。
2. 今天特别郁闷，处理A+B Preblem 时都是不超时但是WA.觉得不知是哪里有问题，可能是判断两个浮点数时出了问题也可能是别的原因，不过我真的是尽力了！两道题分别是2231 2199 toj.
3. 快速模幂运算，如1336的43433次方对24343取模。算法没看懂以后有时间再继续看。<http://zhidao.baidu.com/question/52017297.html>
4. 广搜题：1132 toj 这题求马从一点到一点的最小距离。马每一步有8个方向，先第一行搜索即走完第一个点的八个方向，再一次从上面的八个点每个点搜索八个方向，每个走到的点都存有从起点到该点的距离，可以用一个结构体数组存搜索到的点，如:

struct cam

{

int x;

int y;

}list[100]; list[0.1.2….] 对每个点都要存储相应距离。一发现目的点就返回函数值。具体见1132 toj.

9.1134 toj给出一个数组，元素个数m,求从中取n个数形成一个组合的所有情况。可以变换为取数组n个下标的所有情况。用递归实现，比如zuhe(int n,int min)表示从m个下标中取n个从min开始取，

void zuhe(int k,int min)//从m个元素中取出k个元素的组合,从min开始取

{

int i,j;

for(i=min;i<=m-k+1;i++) //最后一个位置的元素num[k]可以取1.2...m-k+1;

{

num[k]=i;//存取到的下标

if(k>1) zuhe(k-1,i+1);//递归调用,必须从i+1开始取

else

{

int count=0;

for(j=end;j>=1;j--)

{if(count!=0) cout<<" "; cout<<a[num[j]-1];count=1;}

cout<<endl;

}

}

}

这是从别人代码推出来的，想了很大半天。。。算法思路是，假如10个数，取6个，那么第一步从i=1开始取，循环到5结束，因为最小必须不大于5 否则，6-10不能形成6个数；第二步，i+1开始取5个数;…..直到取满6个数为止。

1. 编程中用到的序数词：st，nd，rd，th。最后一位位是1但后两位不是11的加st，最后一位是2但后两位不是12的加nd，最后一位是3但后两位不是13的加rd，其余都加th在尾缀。

11．1136 toj 这道题与质约数有关。给出一个数N，求第N个数是什么。看似得判断质数，其实容易超时也不好下手，数组打表输出则挺有效。a[n]与(a[n-1]…a[1])与(2,3,5,7)是有密切关系的。

12.今天碰到一个以前没遇到的问题。1176 toj. 此题中，大的数组声明要放在main外面，作为全局变量，否则：第一，程序弹出的黑框一闪而过；第二，交程序的时候会提示”Runtime Error”。以后要小心了。

11.比如给定一个double数，取其小数点后2位，第三位四舍五入。Double n;n\*=100;n+=0.5;n=(int)n;n/=100;就这样类推。。。

12.给出平面上多边形的点的坐标求多边形的面积。公式：S=0.5\*∑Xk\*Yk+1-Xk+1\*Yk +0.5xn\*y1-x1\*yn;point[0..n-1].x,.y由于点的顺时针或逆时针排列问题，可能导致s<0,求其绝对值就行。

13.分解质因式问题。3483 .toj .这题求两个超大数的公约数个数，快速算法是：先求两数最大公约数m，则m的约数个数就是答案；当然m也可能很大，这时，好的算法是：将m分解质因式：temp=temp2;//×î´ó¹«Ô¼Êý

num=1;

//cout<<temp<<endl;

for(i=2;i<=temp;i++) //

{

temp1=0;

while(temp!=1) //这个循环很好，把i除到不能除为止

{

if(temp%i==0)

{

temp1++;

temp/=i;

}

else break;

}

num\*=(temp1+1);

}

然后根据排列组合原理，求约数个数,公式如下：A=http://latex.codecogs.com/gif.latex?P_%7b1%7d%5e%7ba_%7b1%7d%7dP_%7b2%7d%5e%7ba_%7b2%7d%7d....P_%7bn%7d%5e%7ba_%7bn%7d%7d，这个数应该有http://latex.codecogs.com/gif.latex?(a_%7b1%7d+1)(a_%7b2%7d+1)\cdot%20\cdot%20\cdot%20(a_%7bn%7d+1)个正约数(包括冪次为0)，意思是这P1到pn每个数有几种情况，组合原理，当都取0次时，约数为1.最终num就是两个超大数的约数个数。

14.以前只知道sort(a,a+n);不知道还有个这个：int sz = unique(ary, ary + n + m-1) - ary+1;//删除重复，unique()里必须是连续的区域，ary[0]到ary[n+m-1]，返回删除重复元素后的数组最后一个位置。sz是删除后的元素个数。它在srot()后用于求出两集合的交集元素个数。

15.一个数学题。3845 toj 一根木棍，分成很多小木棍，长度都为整数，要求任意3根不构成三角形，求木棍最多为多少根。方法：假设小木棍长度按大小排好序，则连续3根不构成三角形，那么当长度最紧密挨着时，个数最多，则有这么一个数列，1,1,2,3,5,8,13……输入的长度能满足其中多少个木棍总个数就是多少。需要另外一个数组，记录前n根木棍的总长度。

16.3508 toj 注意数据范围问题，有时因为oj的机器位数低，导致出现WA.这题倒不是因为那样，m,n输入时int没问题，但是其乘积会超过int,因此s虽然有：long long sum+=m\*n;但是m\*n会被截断，还是32位，因此有必要：long long m,n;

16.今天张旭叫我写了个找字符串的位置程序。Char \*a,char \*b;strstr(a,b);返回b在a中的地址，可以通过int化为整形。但是我人为写的程序，没调用已有函数却出现了问题。

17.结构体排序+动态规划。1130.toj. 基本思想是：每块石头变成3块（这样就可以得到石头的真正总数了）。block代表所有石头 有3个成员x,y,z，然后将其按照底面积大小从大到小排序。建立一个数组h[],h[i]记录的是当前石头作为顶上石头时候的总高度。于是状态转移方程为 h[i] = max {h[j]+block[i].z)。输出最大的height[i]就可以了。其中block[j]必须能放在block[i]下，照这样的最大值h[j]。

18.C和C++里的strchr和strstr是一样的，都是返回一个字符串在另一个第一次出现的位置，返回的是指针，而str.find()返回的是int数,好像C语言里没有str.find()，目前没找到相关证明，有待以后考证。。

19. 枚举算法。1123 toj. A-L和轻或重共有24种组合，可以用枚举法解决，题目确保用3次称量能找出假币，因此24种组合中只有一种是符合给出的3行输入的。在这个程序里居然出现了：char left[3][7];cin>>left[i];报错，但换个变量名如a[3][7]之后编译通过了，可能是编译器把left这个变量给屏蔽了，很奇怪这个Dev编译器。算法可以有多种形式，这里采用对12个字符逐个判断是否为轻或重；假设为轻或重，则应该在每行的输入左右两边应该出现或不出现这个字符取决于这行的结果是down,up还是even;这里用语句switch(c[i][0])来判断3种情况。

13.和张旭做动规 。poj 1015 ,1163,1159。

1015在toj是1652，在toj 1652和poj 1015里很详细的解释了这道题；

1163比较简单，这里不说了；

1159有点难，此题min[i][j]表示字符串ch的一段ch[i]---ch[j]之间插入的最小个数；递推公式：

if(ch[i]==ch[j]) //如果两个游标所指字符相同，向中间缩小范围

min[i][j]=min[i+1][j-1];

else

min[i][j] = 1 + （min[i+1][j]和min[i][j-1]中的较小值）; //如果不同，典型的状态转换方程，

但是内存会超，可以改这个short min[5001][5001]代替int…;还有一种办法：int min[2][5001]，递推中的min[i][j]中的i改为min[i%2][j];内存减到300kb而且时间缩短一半！！！

14.和张旭做搜索题。刚刚A了一道BFS,poj 2251，这题应该不难，是个3维BFS，每一步方向有6种，这题最好用点结构体保存点的3维坐标，因为忘了把新存储的点的isvisted[i][j][k]的置1，导致几次RE，细节问题；

Poj 2488 这题属于DFS，其中应当注意的是回溯时怎样把一条路径中的相应点标记为未访问，这点其实想的时候很复杂，实现很简单，只要几行代码就可以了：tx=a+dir[t][0];

ty=b+dir[t][1];

if(tx>=0&&tx<n&&ty>=0&&ty<m&&!isvisted[tx][ty])

{

DFS(tx,ty);

isvisted[tx][ty]=0;//»ØËÝ

if(!isfound) countans--;

}

详细见程序。

15.木棍拼接问题。Toj 1009,poj 1011.问从给定的一些短木棍能否拼接成n根长度一致的木棒，输出最小的可能长度。本题参照程序设计导引214页，使用递归函数，有点DFS的思想，对输入木棍的每一种组合判断能否拼接成n根等长木棒；核心函数montage(int ususedSticks,int left) 表示还剩下几根木棍未使用，当前拼接的木棒还剩多少要拼接；剪接的时候主要有两点，虽然效率还不是很高，在toj 60ms，在toj 560 ms - -！;如果当前尝试的是某个原始木棒的第一个位置或最后一个位置，则不用再尝试比这个木棍还小的木棍组合；具体见程序；

16. poj 1321. 此题有DFS的思想，但是每一步方向不明确，更准确的说是标准递归；如要放k个棋子，则要选择不同的k行，每行放一个即可；递归函数void select(int i,int k) 表示从第i行开始选取，要选k行，当第i行可能选不上，因为可能没有”#”,也可能没有有效的”#”；因为题目要求两个棋子不能同行不能同列；注意回溯的时候修改相应条件；

17.关于string 和 char \*的几个常用函数文件头：gets(char \*);和getchar();要加stdio.h；getline(cin,string s)为string.h；qsort(a,n,sizeof(a[0]),comp);为stdlib.h；