0基础ACM暴力法与模拟教程

by吴柯

什么是暴力法？

暴力法（brute force），又称暴力枚举法，穷举法，是众多算法中最基本的一种，也是最常见的一种之一。

换句话说，对于要利用计算机程序解决的问题，把所有可能的情况或结果一一进行尝试，排除，选择，从中找出问题的解，或者是实现所需要的功能，这就是暴力法。

那这到底是个什么鬼呢？

If else if else if else if else if if if if …

其实后面还有好多呢。。。

```
if(tmp2<1||(tmp2==1&&(n%tmp==0)))
{
    puts("Captionyuan");
}
else if(tmp2<2||(tmp2==2&&(n%tmp==0)))
{
    puts("Alina");
}
else if(tmp2<3||(tmp2==3&&(n%tmp==0)))
{
    puts("Bigbrother");
}
else if(tmp2<4||(tmp2==4&&(n%tmp==0)))
{
    puts("Kopyh");
}
else if(tmp2<5||(tmp2==5&&(n%tmp==0)))
{
    puts("Watson");
}
```

还有for for for for for。。。

```
for(int i=1;i<=n;++i)
{
    int a1=i;
    for(int j=0;j<v[a1].size();++j)
    {
        int b1=v[i][j];
        for(int k=0;k<v[b1].size();++k)
        {
            int c1=v[b1][k];
            for(int d=0;d<v[c1].size();++d)
            {
                //.....
            }
        }
    }
}
```

以及

If else 和 for。。。

```
46  for(int i=1;i<=n;++i)
47  {
48    if(ccmd[i][0]=='E')
49    {
50      if(pos[num].r0==acmd[i][1]&&pos[num].c0==acmd[i][2])
51      {
52        pos[num].r0=acmd[i][3];pos[num].c0=acmd[i][4];
53      }
54      else if(pos[num].r0==acmd[i][3]&&pos[num].c0==acmd[i][4])
55      {
56        pos[num].r0=acmd[i][1];pos[num].c0=acmd[i][2];
57      }
58    }
59    else
60    {
61    if(ccmd[i][0]=='D')
62    {
63      if(ccmd[i][1]=='R')
64        for(int j=1;j<=acmd[i][0];++j)
65        {
66          if(pos[num].r0==acmd[i][j])
67            return 0;
68          if(acmd[i][j]<pos[num].r0)
69            k++;
70        }
71      pos[num].r0-=k,k=0;
72
73      if(ccmd[i][1]=='C')
74        for(int j=1;j<=acmd[i][0];++j)
75        {
76          if(pos[num].c0==acmd[i][j])
77            return 0;
78          if(acmd[i][j]<pos[num].c0)
79            k++;
80        }
81       pos[num].c0-=k,k=0;
82    }
83    else if(ccmd[i][0]=='I')
84    {
85      if(ccmd[i][1]=='R')
86        for(int j=1;j<=acmd[i][0];++j)
87          if(acmd[i][j]<=pos[num].r0)
```

以上代码均出自本人以往程序

如有巧合，你来咬我啊~

好吧，虽然极端了一点，但这的确就是暴力算法的本来面目。。。。。就代码而言的确有点“暴力”呢。。。

然而，正是这种简单，粗暴，易于是实现的算法广泛存在于各类计算机编程以及应用中，同时也是许多经典算法的基础！

对于acm-icpc来说，暴力法主要应用于基本题（水题）中。（没错，我就是水题收割机）

- 下面，我们就来举几个栗子吧

第一道题：kopyh买水果

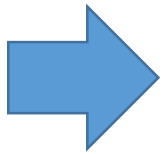
大致题意：kopyh有100元，他要花光这100元买6splus，6s，6plus三种水果，

6splus 5元，6s 3元，3个6plus 1元。每种水果都要买，且每种水果最多只能买100个，求出所有的购买方案。

虽然某学姐在第一天出过了类似的题，但我还是想再讲一遍

解题步骤1：分析问题，理解题目的要求并将问题转化成数学形式

X个6splus加Y个6s加Z个6plus要100元
X，Y，Z大于等于1，小于等于100



X*5+Y*3+Z/3=100
1<=X<=100
1<=Y<=100
1<=Z<=100

解题步骤2：思考解决对应问题的算法，数据结构

- 由于今天的主题是暴力法，所以算法自然就是暴力法咯
- 对应每一种水果，依次暴力枚举数量（决定所要暴力枚举的对象以及初始条件）
- 并通过判断所需价格之和是否为100以及每种的数量是否达到要求来判断当前枚举到的情况是否符合解的条件（决定解的形式及判定条件）
- 符合则按照所要求的输入输出格式输出结果，不符合则跳过
- 继续枚举直至所有可能即符合要求的结果均被枚举到（决定暴力枚举的边间条件）

解题步骤3：用代码来实现

```
99    for(i=1;i<=20;++i)
00    {
01       for(j=1;j<=33;++j)
02       {
03          for(k=1;k<=100;++k)
04          {
05             if(k%3)
06             {
07                continue;
08             }
09             if(5*i+3*j+k/3==100)
10             {
11                printf("%d %d %d\n",i,j,k);
12             }
13          }
14       }
15    }
```

3层for循环，依次枚举三个变量，2个if判断分别用来排除不可能的状态和选择满足答案的状态。

编程重在实践，自己动手写才是最能提高能力的，别人的代码可以参考学习，**但严禁复制粘贴！**

解题步骤4：debug，测试样例，提交代码

- Debug技巧：在代码的关键部位添加printf语句，输出中间变量，根据中间变量来debug
- 测试样例注意技巧：有时题中所给的样例可能过于简单，无法测出代码中的bug，程序也不一定正确，不一定能ac。所以，测试样例时，多想一下极端的情况，比如0，1，负数，非常大的数，之类的。

虽然本题不需要输入样例，但以上技巧却常用于acm比赛及平常做题练习中

暴力算法优缺点及性能分析

- 优点：思维简单，实现容易且容易证明其算法正确性。因此，常用于检测数据，破解密码以及许多对效率要求不高的程序等等。

缺点：效率非常低，非常低，非常非常低
除一些特别的问题没有比较好的高效算法之外（如矩阵乘法，全图最短路等），往往有更好的算法来代替暴力法

暴力算法的效率

- 算法效率详细定义请参考《算法导论》(记得人手一本哟）。
- 这里只做简单的介绍，且只适合暴力算法。（毕竟误人子弟还是不太好的）

简而言之————主程序中最大有x层循环，算法时间复杂度为O(n^x)（注意是n的x次幂不是异或哟）

所以刚才那道题时间复杂度为O(n^3)

好吧，确实在误人子弟>_<

暴力算法的优化

- 优化可以减少暴力枚举的次数，降低算法时间复杂度

- 对于大多数题，如果题目所给的输入数据n的最大范围，超过时间复杂度所能承受的极限，那么代码就必须优化或者换算法了

优化的方式

- 1 减少循环层数，直接降低一档时间复杂度，如O(n^3)—>O(n^2)

```
for(i=1;i<=20;++i)
{
    for(j=1;j<=33;++j)
    {
        if(3*(100-5*i-3*j)<=100&&(3*(100-5*i-3*j)>0))
        {
            printf("%d %d %d\n",i,j,3*(100-5*i-3*j));
        }
    }
}
```

3元1次不定方程，只需枚举2个变元即可暴力枚举出所有解（刚考完没多久的线代别忘了哟，很有用的），因此可以少1层循环，但需要修改if语句

- 2 分析题目条件，适当减少枚举的上限

- 我快做不下去ppt了。。。。T_T

- 简而言之，就是尽可能的降低暴力枚举的次数，可以通过if else 语句 尽可能排除不合理或不满足要求的情况（在下周的搜索算法中这可是十分重要的，这里不赘述了，到时候让kopyh丰神来教）

模拟

- 模拟（simulation），又叫做仿真，或模拟仿真。和暴力法一样，也是最基本的算法之一。不需要过多的思考，算法的过程往往也非常直接，简单，但实现起来却有时相当困难，是属于需要大量代码量的算法。

- 换句话说，将现实中的过程，通过计算机，利用各种计算机数据类型（往往以数组，字符串数组，结构体，类为主），以编程的方式进行实现，就是模拟。
- 所以，模拟非常考察代码实现能力，所以做模拟题也是用来练习代码实现能力的好办法

对于模拟算法，多说无用，多写代码吧！

- Talk is cheap, show me the code

第二道题：大数加法

- C和C++中，最大的整形为无符号长整形unsigned long long，可表示的最大整数为2^64-1,换算成十进制大概是1e18左右。且没有更大的整型数据类型了
- 如果需要对更大的整数进行运算该怎么办呢？

- 答案是用字符串进行模拟

1234 可以用字符串表示为 “1234”

- 但字符串显然没有加减法（C中），所以我们需要自己实现加法的过程，这个过程即是模拟整数加法。

- 1234 没错，就是小学数学，只不过你需要用循环来实现它
- +1839

—1-0-1—

3073

最后说几句吧

- 今天的题，极水无比，就是来让学弟学妹们来锻炼代码能力的，
- 编程重在实践，听人讲100遍不如动手写一遍
- 学长水平有限，如果对讲课内容中有不明白的地方，可以课件或下午做题时问我（当然，别直接问下午开的题怎么做），或者加我QQ私聊，只要有时间都会回答。号码 ：2447159273
- 最后，感谢袁茂询同学的友情出题。

今天讲的暴力法与模拟，都属于朴素算法，也就是naive算法，是最简单最水的算法了

Acm与算法之路，或者说是计算机科学的学习之路，道阻且长，后面难得东西多着呢。在真正厉害的算法或技术面前，我们现有的姿势水平真的只能说是naive。
正因如此，只有不断学习，坚持训练，才能有成效，才能提升自己的能力，这也是acm，以及计算机科学独有的魅力。望大家能坚持下去，在之后的训练中不断提升自己水平。