

学号：_____ 姓名：_____ 年级：_____

题目名 进制数位和均值

问题描述

输入一个十进制整数 n ，将 n 分别转换为二进制，三进制，……， $n-1$ 进制的数，计算 n 转换成的所有进制数的每位数字之和的平均值。计算结果用十进制表示为最简分数。

输入

第一行输入一个整数 $T(1 \leq T \leq 20)$ ，表示数据组数。

接下来 T 行，每行一个整数 $n(2 < n \leq 106)$ 。

对于 50% 的数据： $2 < n \leq 200$ 。

对于 100% 的数据： $2 < n \leq 106$ 。

输出

对于每组数据，在一行中输出一个形如 n/m 的最简分数表示答案。

输入输出样例

Examples	
input	
2	
7	
6	
output	
3/1	
9/4	



问题分析

因为是进制问题，而且进制是在不断增加的，就可以想到用循环。数的大小是 10^6 ，可以用 long long 来解决。而进制数的每位则用取余和整除来求，最简形式则除以一个最大公约数。

程序代码

```
1.  #include<bits/stdc++.h>
2.  using namespace std;
3.  typedef long long ll;
4.  ll gcd(ll a,ll b)
5.  {
6.      return b?gcd(b,a%b):a;
7.  }
8.  int main()
9.  {
10.     int T;
11.     cin>>T;
12.     for(int i=0; i<T; i++)
13.     {
14.         ll num;
15.         cin>>num;
16.         ll sum=0;
17.         for(int j=2; j<num; j++)
18.         {
19.             ll tmp=num;
20.             while(tmp>0)
21.             {
22.                 sum+=tmp%j;
23.                 tmp/=j;
24.             }
25.         }
26.         ll t=gcd(sum,num-2);
27.         printf("%lld/%lld\n",sum/t,(num-2)/t);
28.     }
29.     return 0;
30. }
```

解题结果

#	Problem	Language	Sent	Judged	Verdict	CPU	Judge
1249362	1233. 进制数位和均值	C++11	2018-06-13 19:15:19	2018-06-13 19:15:21	 Accepted	0.228	Cow
							

解题备注

注意用一个 tmp 来保存每次进制之下的原数据。

题目名 二进制倒置

问题描述

给定一个整数 $n(0 \leq n \leq 10100)$ 、将 n 的 334 位二进制表示形式（不包括开头可能的值为 0 的位， $n=0$ 表示为 1 位 0）前后倒置，输出倒置后的二进制数对应的整数。

例如： $n=10$ ，其二进制表示为（330 个 0）1010，倒置后为 0101，对应输出就是 5。

输入

第 1 行：一个整数 T ($1 \leq T \leq 10$) 为问题数。
接下来共 T 行整数，对应每个问题有 1 行，表示 n 。

输出

对于每个问题，输出一行问题的编号（0 开始编号，格式：case #0: 等）。
然后对应每个问题在一行中输出结果。

输入输出样例

Examples

[illegible]

问题分析

利用进制转换模板，先将十进制转换为二进制，再倒置，再转换为十进制

程序代码

```
1. #include<bits/stdc++.h>
2. using namespace std;
3. const int maxn = 10000;
4. using namespace std;
5. int t[maxn], A[maxn],c,pro;
6. char str[maxn], str1[maxn], str2[maxn], str3[maxn];
7. void res(char * s)
8. {
9.     int i = 0;
10.    int len = strlen(s);
11.    for (i=0; i<strlen(s)/2; i++)
12.    {
13.        char temp = s[i];
14.        s[i] = s[strlen(s) - i - 1];
15.        s[strlen(s) - i - 1] = temp;
16.    }
17. }
```

```

18. void solve(char *str1, char *str2, int n, int m)
19. {
20.     int i, len=strlen(str1), k;
21.     for (i = len-1; i >= 0; i--)
22.         t[len-1-i] = isdigit(str1[i]) ? str1[i] - '0' : i
supper(str1[i]) ? str1[i] - 'A' + 10 : str1[i] - 'a' + 32;/
/倒置
23.     for (k = 0; len;)
24.     {
25.         for (i=len-1; i >= 1; i--)
26.         {
27.             t[i - 1] += t[i] % m*n;
28.             t[i] /= m;
29.         }
30.         A[k++] = t[0] % m;
31.         t[0] /= m;
32.         while (len>0 && !t[len - 1])
33.             len--;
34.     }
35.     str2[k] = '\0';
36.     for (i = 0; i<k; i++)
37.         str2[k - 1 - i] = A[i] + (A[i]<10 ? 48 : A[i]<36 ?
55 : 61);
38. }
39. int main()
40. {
41.     char input[100];
42.     char binary[334];
43.     char output[100];
44.     int T;
45.     cin>>T;
46.     for(int i=0; i<T; i++)
47.     {
48.         scanf("%s",input);
49.         solve(input,binary,10,2);
50.         res(binary);
51.         solve(binary,output,2,10);
52.         printf("case %d:\n%s\n",i,output);
53.     }
54.     return 0;
55. }

```

解题结果

#	Problem	Language	Sent	Judged	Verdict	CPU	Judge
1249721	3031. 二进制倒置	C++11	2018-06-14 14:20:56	2018-06-14 14:20:57	✔ Accepted	0.004	Goat
	✔						

解题备注

注意模板使用时要倒置一次

题目名 平衡三进制

问题描述

衡三进制分别使用字符 ‘-’， ‘0’， ‘1’ 表示 -1, 0, 1。下表表示从 0 到 10 的十进制数对应的平衡三进制的值。

十进制	平衡三进制
0	0
1	1
2	1-
3	10
4	11
5	1--
6	1-0
7	1-1
8	10-
9	100
10	101

例如 $7=1\times 3^2+(-1)\times 3^1+1\times 3^0=9-3+1=7$ 。

现在给你一个关于平衡三进制的串，请将其转成对应的十进制数。

输入

第 1 行：整数 T ($1 \leq T \leq 10$) 为测试数据组数。

第 2 ~ $T+1$ 行：每个问题一行，每行输入一个平衡三进制的字符串，保证其转换成的十进制整数小于 $1e9$ ，且全为非负整数。

输出

对于每个问题，输出一行问题的编号（0 开始编号，格式：case #0: 等），然后输出对应问题的结果。

输入输出样例

Examples

input
3 1--- 1000 1-0-1-1-1-1
output
case #0: 14 case #1: 27 case #2: 37726

问题分析

将三进制里的三种情况分别转换为-1,0,1，再乘以该位的三的 n 次方即可，

程序代码

```
1.  #include<bits/stdc++.h>
2.  using namespace std;
3.  int main()
4.  {
5.      int T;
6.      cin>>T;
7.      for(int i=0; i<T; i++)
8.      {
9.          char s[20];
10.         scanf("%s",s);
```

```

11.         long long num=0;
12.         int k=0;
13.         for(int j=0;j<strlen(s);j++)
14.         {
15.             if(s[j]=='-')
16.                 k=-1;
17.             else
18.                 k=s[j]-'0';
19.             num+=pow(3,strlen(s)-j-1)*k;
20.         }
21.         printf("case #d:\n%lld\n",i,num);
22.     }
23. }

```

解题结果

#	Problem	Language	Sent	Judged	Verdict	CPU	Judge
1249519	3190. 平衡三进制	C++11	2018-06-13 20:50:17	2018-06-13 20:50:19	✔ Accepted	0.000	Goat
	✔						

解题备注

注意三的 n 次方时是 $size-i-1$

题目名 一元多项式乘法

问题描述

计算两个一元多项式的乘积。

输入

每行两个多项式，以一个空格分隔，多项式格式为 $a_nx^n+\cdots+a_1x+a_0$ 。

每行长度不超过 100， $0<n<50$ 。

输出

每组数据一行，根据次数由高到低顺序输出两个多项式乘积的非零项系数，两个系数之间由一个空格分隔。

输入输出样例

Examples

input
<pre>x+1 x-1 x^2-3x+1 x^3 x+2 2</pre>
output
<pre>1 -1 1 -3 1 2 4</pre>

问题分析

问题的主要难点是读取麻烦，通过一个读取函数，编写多个判断语句，注意常数项，第一项，一次项的特殊情况

程序代码

```
1. #include<bits/stdc++.h>
```

```

2.  using namespace std;
3.  void readpoly(char *s, int *xishu)
4.  {
5.      while(*s)
6.      {
7.          int sign=1,a=0,i=0;
8.          if(*s=='+')
9.              s++;
10.         else if (*s=='-')
11.         {
12.             sign=-1;
13.             s++;
14.         }
15.         while(isdigit(*s))
16.         {
17.             a=a*10+*s-'0';
18.             s++;
19.         }
20.         if(a==0)
21.             a=1;
22.         if(*s!='x')
23.         {
24.             xishu[0]=a*sign;
25.             return ;
26.         }
27.         else s++;
28.         if(*s=='^')
29.             s++;
30.         while(isdigit(*s))
31.         {
32.             i=i*10+*s-'0';
33.             s++;
34.         }
35.         if(i==0)
36.             i=1;
37.         xishu[i]=a*sign;
38.     }
39. }
40. void multiply(char *s1, char *s2, int *xishu)
41. {
42.     int xishu1[100]={0},xishu2[100]={0};
43.     readpoly(s1,xishu1);
44.     readpoly(s2,xishu2);
45.     for(int i=0;i<50;i++)

```

```

46.         for(int j=0;j<50;j++)
47.             xishu[i+j]=xishu[i+j]+xishu1[i]*xishu2[j];
48.     }
49.     int main()
50.     {
51.         char s1[101],s2[101];
52.         while(scanf("%s %s",s1,s2)!=EOF)
53.         {
54.             int xishu[100]={0},out[100],n=0;
55.             multiply(s1,s2,xishu);
56.             for(int i=0;i<100;i++)
57.                 if(xishu[i])
58.                     out[n++]=xishu[i];
59.             for(int i=n-1;i>=0;i--)
60.             {
61.                 printf("%d",out[i]);
62.                 if(i>0)
63.                     printf(" ");
64.                 else
65.                     printf("\n");
66.             }
67.         }
68.         return 0;
69.     }

```

解题结果

#	Problem	Language	Sent	Judged	Verdict	CPU	Judge
1249565	2. 一元多项式乘法	C++11	2018-06-13 21:44:59	2018-06-13 21:45:01	✔ Accepted	0.004	Goat
	<div> <div>✔</div> <div>✔</div> <div>✔</div> <div>✔</div> <div>✔</div> <div>✔</div> <div>✔</div> <div>✔</div> <div>✔</div> <div>✔</div> </div>						

解题备注

注意特殊情况要考虑完全，字符串涉及符号时多用一个 sign 来判断，还要删去不存在的幂次。

题目名 四元一次方程

问题描述

对于一个非负整数 n ，四元一次方程：

$$4w+3x+2y+z=n$$

的非负整数解是不唯一的。

编程计算不同解的个数。

例如： $n=0$ 时有 1 个解 $(0, 0, 0, 0)$ ； $n=2$ 时有 2 个解 $(0, 0, 1, 0)$ 和 $(0, 0, 0, 2)$

输入

第 1 行：整数 T ($1 \leq T \leq 10$) 为问题数

第 2 \sim $T+1$ 行：每一个问题中的 n ， $0 \leq n \leq 1000$ 。

输出

对于每个问题，在一行中输出解的个数。

输入输出样例

Examples	
input	Click to copy
<pre>3 0 10 1000</pre>	
output	
<pre>1 23 7049112</pre>	

问题分析

直接模拟很简单，但是四次循环会导致超时，所以在最后优化循环条件

程序代码

```
1.  #include<bits/stdc++.h>
2.  using namespace std;
3.  int main()
4.  {
5.      int T;
6.      cin>>T;
7.      for(int i=0;i<T;i++)
8.      {
9.          int n,cnt=0;
10.         cin>>n;
11.         if(n>0)
12.             for(int w = 0;w <= n/4;w++)
13.                 for(int x = 0;x <= (n-4*w)/3;x++)
14.                     for(int y = 0;y <= (n-4*w-3*x)/2;y++)
15.                         if(4*w+3*x+2*y<=n)
16.                             cnt++;
17.         else
18.             cnt=1;
19.         cout<<cnt<<endl;
20.     }
21. }
```

解题结果

#	Problem	Language	Sent	Judged	Verdict	CPU	Judge
1245763	2944. 四元一次方程	C++11	2018-06-05 19:13:03	2018-06-05 19:13:05	✔ Accepted	0.028	Goat
	✔						

解题备注

注意为 0 的情况

题目名 斐波那契数列

问题描述

斐波那契数列的递归定义如下：

$$F(0)=0, F(1)=F(2)=1;$$
$$F(n)=F(n-1)+F(n-2) \text{ (当 } n>2 \text{ 时)}。$$

给定一个整数 n ($0\leq n\leq 120$)，求 $F(n)$ 的值。

输入

第 1 行：一个整数 T ($1\leq T\leq 10$) 为问题数。
第 2~T+1 行，一个整数 n ($0\leq n\leq 120$)。

输出

对每个测试数据，首先输出一行问题的编号 (0 开始编号，格式：case #0: 等)。
在接下来一行中输出 $F(n)$ 的值。

输入输出样例

Examples
<div><div>input</div><div>3 0 5 119</div></div> <div><div>output</div><div>case #0: 0 case #1: 5 case #2: 3311648143516982017180081</div></div>

问题分析

根据样例发现这个是一个隐藏的大数计算，则写一个循环 120 次的语句来记录所有的斐波那契数列，然后按需读取

程序代码

```
1. #include<bits/stdc++.h>
```

```

2.  using namespace std;
3.  typedef long long ll;
4.  int main(void)
5.  {
6.      int T;
7.      scanf("%d",&T);
8.      int i, j, fib[121][27]={0}, n;
9.      fib[1][0]=1;
10.     for(i=2; i<121; i++)
11.     {
12.         for(j=0; j<26; j++)
13.         {
14.             fib[i][j]+=(fib[i-1][j]+fib[i-2][j]);
15.             fib[i][j+1]=fib[i][j]/10;
16.             fib[i][j]%=10;
17.         }
18.     }
19.     for(i=0; i<T; i++)
20.     {
21.         scanf("%d",&n);
22.         printf("case #%d:\n",i);
23.         for(j=26;j+1 ; j--)
24.             if(fib[n][j]) break;
25.         for(;j+1 ; j--)
26.             printf("%d",fib[n][j]);
27.         if(n==0) printf("0");
28.         printf("\n");
29.     }
30.     return 0;
31. }

```

解题结果

#	Problem	Language	Sent	Judged	Verdict	CPU	Judge
1244147	3076. 斐波那契数列	C++11	2018-06-02 10:00:40	2018-06-02 10:00:41	✔ Accepted	0.000	Goat
	✔						

解题备注

注意输出时要把无效的都删去，用一个循环来找到有效的下标

