

题目名 DNA 排序题

问题描述

有若干个 DNA，每个 DNA 可能出现一次或者多次。
统计每个 DNA 出现的次数，然后再按照 DNA 出现次数的升序来输出这些 DNA。对于出现次数相同的 DNA，按照字典顺序排列输出（升序）。

输入

第一行是一个整数 n 。
接下来 n 行，每行一个 DNA。每个 DNA 是由 A, C, G, T 字符组成的，长度不超过 20。
• 对于 40% 的数据： $1 \leq n \leq 20$ ；
• 对于 70% 的数据： $1 \leq n \leq 2000$ ；
• 对于所有的数据： $1 \leq n \leq 200\,000$ 。

输出

输出经去重和排序后的 DNA，每个 DNA 一行。

输入输出样例

Examples

input	Click to copy
6 CTGGTGGGGAGATGAG TTGCCACCAACGACGAT TTGCCACCAACGACGAT TATGCAG TATGCAG TATGCAG	
output	
CTGGTGGGGAGATGAG TTGCCACCAACGACGAT TATGCAG	

问题分析

因为既要记住字符串也要记住长度，就想到使用结构体来读取。首先用 char 来逐个读入字符串，然后用一个结构体数组来比较并记录字符串及其出现次数。然后用 qsort，自己写一个基于出现次数和字典序排序的 cmp 函数。

程序代码

```
1.  #include<bits/stdc++.h>
2.  using namespace std;
3.  typedef struct
4.  {
5.      char name[21];
6.      int cnt;
7.  }dna;
8.  int cmp(const void *a,const void *b)
9.  {
10.     dna *p=(dna *)a;
11.     dna *q=(dna *)b;
12.     if(p->cnt!=q->cnt){
13.         if(p->cnt>q->cnt)
14.             return 1;
15.         else
16.             return -1;
17.     }
18.     else return strcmp(p->name,q->name);
19. }
20. int main()
21. {
22.     int T;
23.     cin>>T;
24.     dna store[T];
25.     char name[25];
26.     int count=0,flag=1;
27.     for(int i=0;i<T;i++)
28.     {
29.         cin>>name;
30.         flag=1;
31.         if(count==0)
32.         {
33.             strcpy(store[0].name,name);
34.             store[0].cnt=1;
35.             count=1;
```

```

36.         }
37.         else
38.         {
39.             for(int j=0;j<count;j++)
40.             {
41.                 if(strcmp(name,store[j].name)==0)
42.                 {
43.                     store[j].cnt++;
44.                     flag=0;
45.                     break;
46.                 }
47.             }
48.             if(flag)
49.             {
50.                 strcpy(store[count].name,name);
51.                 store[count].cnt=1;
52.                 count++;
53.             }
54.         }
55.     }
56.     qsort(store,count,sizeof(store[0]),cmp);
57.     for(int i=0;i<count;i++)
58.     {
59.         cout<<store[i].name<<endl;
60.         //cout<<store[i].cnt<<endl;
61.     }
62. }

```

解题结果

#	Problem	Language	Sent	Judged	Verdict	Percent	CPU	Judge
1243285	75. DNA 排序题	C++11	2018-05-31 13:32:03	2018-05-31 13:32:18	 Time Limit Exceeded	90.0 %	2.996	Goat
								

解题备注

技巧是利用一个 flag 来判断是否已经存储过这个字符串，但是第八个案例过不了还是没有解决

题目名 成绩排序

问题描述

有 n ($1 \leq n \leq 100$) 个学生的成绩记录，其中包含学号和成绩两项。按照成绩从高到低顺序输出成绩及格 (≥ 60) 学生的学号和成绩。成绩相同时按照学号从小到大顺序输出。

输入

- 第 1 行：输入一个整数 n ，表示学生记录数。
- 第 2 行 ~ $n+1$ 行：每行是学号（11 位数字）及成绩（0 到 100 之间的整数）。学号和成绩之间有一个空格。

输出

每行输出成绩及格学生按要求排序后以一个空格分隔的学号及成绩。

输入输出样例

Examples
<div>input</div> <div>5 10002130201 90 10002130230 80 10002130231 85 10002130148 48 10002130167 90</div>
<div>output</div> <div>10002130167 90 10002130201 90 10002130231 85 10002130230 80</div>


问题分析

既要储存学号也要储存成绩，所以是结构体排序。用 char 存储学号，用 int 存储成绩。
然后自己写一个 cmp 函数

程序代码

```
1.  #include<bits/stdc++.h>
2.  using namespace std;
3.  typedef struct
4.  {
5.      char num[11];
6.      int grade;
7.  }store;
8.  int cmp(const void *a,const void *b)
9.  {
10.     store *p=(store *)a;
11.     store *q=(store *)b;
12.     if(p->grade!=q->grade)
13.     {
14.         if(p->grade>q->grade)
15.             return -1;
16.         else
17.             return 1;
18.     }
19.     else
20.         return strcmp(p->num,q->num);
21. }
22. int main()
23. {
24.     int n;
25.     cin>>n;
26.     store student[n];
27.     for(int i=0;i<n;i++)
28.         scanf("%s %d",student[i].num,&student[i].grade);
29.     qsort(student,n,sizeof(student[0]),cmp);
30.     for(int i=0;i<n;i++)
31.     {
32.         if(student[i].grade>=60)
33.             printf("%s %d\n",student[i].num,student[i].grade);
34.     }
35.     return 0;
36. }
```

解题结果

	2849. 成绩排序	C++11	2018-06-05 18:44:21	2018-06-05 18:44:23	 Accepted	0.004	Cow
1245726	<div><div><div><div><div></div></div></div><div><div><div></div></div><div><div></div></div><div><div></div></div><div><div></div></div><div><div></div></div><div><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

注意读取时可以就是 scanf (“%s %d” , xxx, xxxx)

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 ~ $T+1$ 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

input
3 0 10 1000
output
1 23 7049112

Click to copy

问题分析

很直接能想到是嵌套循环，但是会担心超时的问题，所以要优化一下判断条件减少循环个数

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

在写循环的时候判断语句注意是可以取等的

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：

- $F(0) = 0, F(1) = F(2) = 1$;
- $F(n) = F(n - 1) + F(n - 2)$ (当 $n > 2$ 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

Click to copy

input

3
0
5
119

output

case #0:
0
case #1:
5
case #2:
3311648143516982017180081

问题分析

看到斐波那契先是想好了用递归，发现要求的范围很大，尝试了一下迭代法发现确实超出了 `long long` 的范围，然后马上转为大整数来做，

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

C++11

Submit

#	Language	Sent	Judged	Verdict	CPU	Judge
1244146	C++11	2018-06-02 10:00:30	2018-06-02 10:00:32	✓ Accepted	0.004	Goat
	✓					

PAST SUBMISSIONS

#	Sent	Judged	Lang	Verdict	Time
1244146	2018-06-02 10:00:30	2018-06-02 10:00:32	C++11	Accepted	0.004
1244139	2018-06-02 09:34:34	2018-06-02 09:34:36	C++11	Wrong Answer on test 1	0.000

解题备注

基础的大整数的题，注意范围即可

题目名 三元斐波那契数列

问题描述

一个三元斐波纳奇数列定义为如下递归式：

$$A[i] = A[i-1] + A[i-2] + A[i-3] \ (i \geq 3)$$

给你一个数列 A，其中包含一个且只有一个-1，你必须把这个-1 替换成一个正数 N 使得 A 数列成为一个三元斐波纳奇数列。

如果不存在合法的 N，输出-1。

输入

第 1 行：整数 $T \ (1 \leq T \leq 10)$ 为问题数

第 2 ~ T+1 行：每行有若干个数字，第一个数表示 A 数列的大小 $M(4 \leq M \leq 20)$ ，后面紧接着 M 个数，表示 A 数列，其每项的值在 $1 \sim 1000000$ 之间（除唯一的那个-1 之外）。

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），然后对于每组数据，在一行中输出 N，如果不存在合法的 N，输出-1。

输入输出样例

input
3 4 1 2 3 -1 6 10 20 30 60 -1 200 5 1 2 3 5 -1
output
case #0: 6 case #1: 110 case #2: -1

问题分析

首先存入所有数字，寻找-1 的位置，按照公式处理
根据题目模拟写出判断式 $f(n) = f(n-1) + f(n-2) + f(n-3)$
循环判断整个数列是否满足

程序代码

```
1.  #include<bits/stdc++.h>
2.  using namespace std;
3.  typedef long long ll;
4.  int main()
5.  {
6.      int T;
7.      cin>>T;
8.      for(int i=0;i<T;i++)
9.      {
10.         int M;
11.         cin>>M;
12.         int position=0;
13.         ll store[M];
14.         int flag=1;
15.         printf("case #%d:\n",i);
16.         for(int j=0;j<M;j++)
17.         {
18.             cin>>store[j];
19.             if(store[j]==-1)
20.                 position=j;
21.         }
22.         if(position==0)
23.             store[0]=store[3]-store[2]-store[1];
24.         else if(position==1)
25.             store[1]=store[3]-store[2]-store[0];
26.         else if (position==2)
27.             store[2]=store[3]-store[1]-store[0];
28.         else
29.             store[position]=store[position-1]+store[position-2]+store[position-3];
30.         for(int j=3;j<M;j++)
31.             if(store[j]!=store[j-1]+store[j-2]+store[j-3]||store[j]==0||store[j-1]==0||store[j-2]==0||store[j-3]==0)
32.             {
33.                 flag=0;break;}
34.         if(flag)
35.             printf("%d\n",store[position]);
36.         else
37.             printf("-1\n");
38.     }
39. }
```

解题结果

C++11

Submit

#	Language	Sent	Judged	Verdict	CPU	Judge
1244185	C++11	2018-06-02 12:19:45	2018-06-02 12:19:46	✔ Accepted	0.004	Goat
	✔					

PAST SUBMISSIONS					
#	Sent	Judged	Lang	Verdict	Time
1244185	2018-06-02 12:19:45	2018-06-02 12:19:46	C++11	Accepted	0.004
1244168	2018-06-02 11:24:16	2018-06-02 11:24:17	C++11	Wrong Answer on test 1	0.000

解题备注

第一次没过后分析题意，发现是 $1 \sim 10000$ ，之前的代码 0 被忽略了，所以再加一个判断

题目名 多次函数

问题描述

在一个平面坐标系统中显示一个多次函数的图像。

为简化起见，设多次函数的最大次数为 3 ，系数最多为 2 位数。函数表示为： $f(x) = c_3x^3 + c_2x^2 + c_1x + c_0$ 。给出的函数是最简形式。 c_i 为 1 时可省略， c_3 为正数时可省略 $+$ ，次方为 1 时可省略 1 。

坐标系统的 x 轴和 y 轴范围限制在 $[-20, 20]$ ，具体格式见样例。

输入

不多于 20 行，每行一个最多为 3 次的函数。形式： $f(x) = c_3x^3 + c_2x^2 + c_1x + c_0$ 。

输出

显示每个函数的图像。两个函数之间用一个空行分隔，最后一个函数之后没有空行。

输入输出样例

$$f(x) = 3x^2$$
$$f(x) = -5x$$
$$f(x)=1$$

```
output
```

.....^.....

Age Group	Percentage
18-24	~1%
25-34	~28%
35-44	~1%
45-54	~1%
55-64	~1%
65-74	~1%
75-84	~1%
85+	~1%

_____ *

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....*|*.....

.....|.....

.....|.....

-----*----->

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....*.....^.....

..... |

..... |

..... |

..... |

..... * .. |

..... |

..... |

..... |

..... |

..... * .. |

..... |

..... |

..... |

..... |

..... * |

..... |

..... |

..... |

..... |

-----*----->

..... |

..... |

..... |

..... |

..... | *

..... |

..... |

..... |

..... |

..... | *

..... |

..... |

..... |

..... |

..... |*

..... |

..... |

..... |

..... |

..... |*

..... |

..... |

..... |

..... |

..... |

..... |

..... |

..... |

..... |

..... |

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

.....|.....

- - - - - + - - - - - >

.....|.....

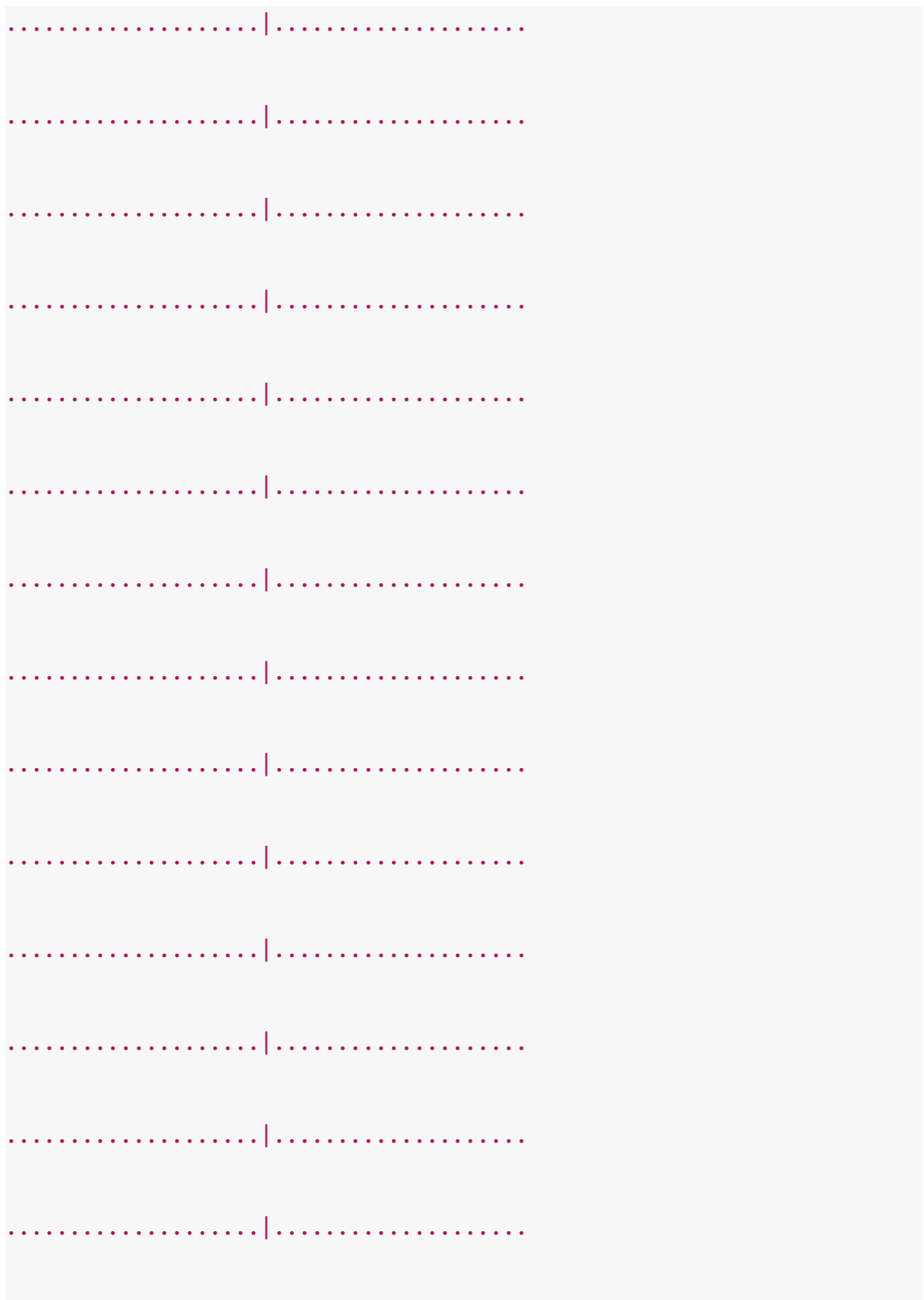
.....|.....

.....|.....

.....|.....

.....|.....

.....|.....



Note

如果在表示坐标轴的 、、、 字符和表示原点的  字符位置需要显示  时，用  去覆盖。

问题分析

首先题目的难点是怎么读入系数和次数，这个利用 `strstr` 来解决。至于打表则利用函数作图。

整个图像的储存都利用二维数组来处理

程序代码

```
1.  #include<stdio.h>
2.  #include<string.h>
3.  #include<math.h>
4.  char zuobiao[41][41];
5.  int i,j;
6.  int c3,c2,c1,c0;
7.  void initial()
8.  {
9.      c3=c2=c1=c0=0;
10.     for(j=0; j<41; j++)
11.         for(i=0; i<41; i++)
12.             {
13.                 if(i!=20&&j==20) zuobiao[i][j]='-';
14.                 else if(i==20&&j!=20) zuobiao[i][j]='|';
15.                 else zuobiao[i][j]='.';
16.             }
17.     zuobiao[20][20]='+';
18.     zuobiao[20][40]='^';
19.     zuobiao[40][20]='>';
20. }
21. void draw()
22. {
23.     for(j=40; j>=0; j--)
24.     {
25.         for(i=0; i<41; i++)
26.             printf("%c",zuobiao[i][j]);
27.         printf("\n");
28.     }
29.     printf("\n");
30. }
31. int getnum1(char *p1,char *p2)
32. {
33.     int sign,c=0;
34.     if(*p1=='=')
35.     {
```

```

36.         if(*(p1+1)=='-') sign=-1,p1+=1;
37.         else sign=1;
38.     }
39.     else
40.     {
41.         if(*(p1)=='-') sign=-1;
42.         else sign=1;
43.     }
44.     int k;
45.     if(p2-p1-1==0) c=1;
46.     else
47.         for(k=0; k<(p2-p1-1); k++)
48.             c=c*10+(*(p1+1+k)-'0');
49.     c*=sign;
50.     return c;
51. }
52. int main()
53. {
54.     char f[30];
55.     char *p1,*p2,*p3,*p4,*p5;
56.     while(scanf("%s",f)!=EOF)
57.     {
58.         initial();
59.         p1=strstr(f,"=");
60.         p2=strstr(p1,"x^3");
61.         if(p2!=NULL)
62.             c3=getnum1(p1,p2),p2=p2+3;
63.         else p2=p1;
64.         p3=strstr(p2,"x^2");
65.         if(p3!=NULL)
66.             c2=getnum1(p2,p3),p3=p3+3;
67.         else p3=p2;
68.         p4=strchr(p3,'x');
69.         if(p4!=NULL)
70.             c1=getnum1(p3,p4),p4+=1;
71.         else p4=p3;
72.         p5=f+strlen(f);
73.         c0=getnum1(p4,p5);
74.
75.         int l,m;
76.         for(l=-20; l<=20; l++)
77.         {
78.             m=c3*l*l*l+c2*l*l+c1*l+c0;
79.             if(abs(m)>20) continue;

```

```
80.         else
81.             zuobiao[l+20][m+20]='*';
82.         }
83.         draw();
84.     }
85. }
```

解题结果

| # | Problem | Language | Sent | Judged | Verdict | CPU | Judge |
|---------|----------------------------|----------|---------------------|---------------------|------------|-------|-------|
| 1246439 | 2891. 多次函数 | C | 2018-06-06 22:40:18 | 2018-06-06 22:40:18 | ✔ Accepted | 0.000 | Cow |
| | ✔ | | | | | | |

解题备注

使用 strstr 时注意指针