A - Googol String

       题意:

              目前只考虑包含'0','1'的字符串，有两个操作Switch与Reverse...其中Switch将其中的'0'变成'1','1'变成'0', Reverse将字符串反转

              定义:

                     S0=""

                     Sn=Sn-1+'0'+Switch(Reverse(Sn-1))

                     现在已知n=10的100次方，问第k(小于10的18次方)位是0还是1

       题解:

              其实看到k的范围就知道n这么大的范围只是一个幌子，当n到62的时候，位数已经突破10的18次方位了，所以我觉得两个思路来做，第一个先打表，然后download数据后跑结果，第二个就是我用的在线递归，注意到Sn的长度总是2的n次方-1...就不难写出递归的方法了...

       Code:

**[cpp]** [view plaincopy](http://blog.csdn.net/kk303/article/details/47907519)

1. #include <iostream>
2. #include <cstdio>
3. #include <cmath>
4. #include <algorithm>
5. #include <cstring>
6. #include <queue>
7. #include <stack>
8. #include <map>
9. **using** **namespace** std;
10. **int** dfs(**long** **long** k,**long** **long** h){
11. **if** (k==((h-1)/2))
12. **return** 0;
13. **if** (k<(h/2-1))
14. **return** dfs(k,h/2);
15. **else**{
16. k=h-2-k;
17. **return** 1-dfs(k,h/2);
18. }
19. }
20. **int** main(){
21. freopen("A-small-attempt0.in.txt","r",stdin);
22. freopen("output.txt","w",stdout);
23. **int** T;
24. cin>>T;
25. **long** **long** c=1;
26. **for** (**int** i=1;i<=62;i++) c=c\*2;
27. **for** (**int** cases=1;cases<=T;cases++){
28. **long** **long** k;
29. cin>>k;
30. k--;
31. cout<<"Case #"<<cases<<": "<<dfs(k,c)<<endl;
32. }
33. **return** 0;
34. }

B - gCube

       题意:

              直接点说就是求D个数之积开D次方的结果。

       题解:

              很容易想到用log来处理，因为用了log后原本的连乘变成了连加，最后再把结果还原就是。

       Code:

**[cpp]** [view plaincopy](http://blog.csdn.net/kk303/article/details/47907519)

1. #include <iostream>
2. #include <cstdio>
3. #include <cmath>
4. #include <algorithm>
5. #include <cstring>
6. #include <queue>
7. #include <stack>
8. #include <map>
9. **using** **namespace** std;
10. **const** **int** MAXN=100005;
11. **int** a[MAXN];
12. **int** main(){
13. freopen("B-large.in.txt","r",stdin);
14. freopen("output.txt","w",stdout);
15. **int** T;
16. scanf("%d",&T);
17. **for** (**int** cases=1;cases<=T;cases++){
18. **int** n,m;
19. scanf("%d%d",&n,&m);
20. **for** (**int** i=0;i<n;i++) scanf("%d",&a[i]);
21. printf("Case #%d:\n",cases);
22. **while** (m--){
23. **int** L,R,D;
24. scanf("%d%d",&L,&R);
25. D=(R-L+1);
26. **double** ans=0;
27. **for** (**int** i=L;i<=R;i++) ans+=log2(a[i]);
28. ans/=D;
29. printf("%.10lf\n",pow(2,ans));
30. }
31. }
32. **return** 0;
33. }

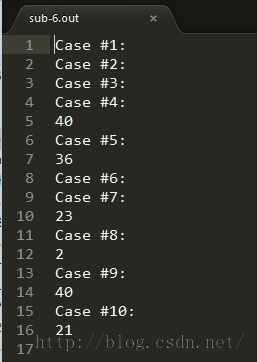
C - gCampus

       题意:

              给一个图(点的个数小于100，边的个数小于10000)，问其中有哪些边是不在任意一个两两间的最短路上的。

       题解:

              这个之前做过[类似的](http://blog.csdn.net/kk303/article/details/25834683" \t "_blank)，跑个floyd然后比较下即可..比较坑的是小数据..跑出结果后不敢提交有么有...



       Code:

**[cpp]** [view plaincopy](http://blog.csdn.net/kk303/article/details/47907519)

1. #include <iostream>
2. #include <cstdio>
3. #include <cmath>
4. #include <algorithm>
5. #include <cstring>
6. #include <queue>
7. #include <stack>
8. #include <map>
9. **using** **namespace** std;
10. **const** **int** MAXN=20005;
11. **struct** Edge{
12. **int** u,v,c;
13. }edge[MAXN];
14. **int** dis[105][105];
15. **int** ABS(**int** x){
16. **if** (x<0) **return** -x;
17. **return** x;
18. }
19. **void** Floyd(**int** n){
20. **for** (**int** k=0;k<n;k++)
21. **for** (**int** u=0;u<n;u++)
22. **for** (**int** v=0;v<n;v++)
23. **if** (dis[u][v]-dis[u][k]>dis[k][v])
24. dis[u][v]=dis[u][k]+dis[k][v];
25. }
26. **bool** f[MAXN];
27. **int** main(){
28. freopen("C-small-attempt0.in.txt","r",stdin);
29. freopen("output.txt","w",stdout);
30. **int** T;
31. scanf("%d",&T);
32. **for** (**int** cases=1;cases<=T;cases++){
33. **int** n,m;
34. scanf("%d%d",&n,&m);
35. memset(dis,0x7f,**sizeof**(dis));
36. **for** (**int** i=0;i<n;i++) dis[i][i]=0;
37. **for** (**int** i=0;i<m;i++){
38. **int** u,v,c;
39. scanf("%d%d%d",&u,&v,&c);
40. edge[i].u=u,edge[i].v=v,edge[i].c=c;
41. dis[u][v]=min(dis[u][v],c);
42. dis[v][u]=min(dis[v][u],c);
43. }
44. printf("Case #%d:\n",cases);
45. Floyd(n);
46. memset(f,**false**,**sizeof**(f));
47. **for** (**int** s=0;s<n;s++)
48. **for** (**int** t=0;t<n;t++)
49. **for** (**int** k=0;k<m;k++)
50. **if** (dis[s][edge[k].u]+edge[k].c+dis[edge[k].v][t]==dis[s][t])
51. f[k]=**true**;
52. **for** (**int** i=0;i<m;i++)
53. **if** (!f[i]) printf("%d\n",i); //
54. }
55. **return** 0;
56. }