Really Fast I/O methods for Programming in C++ (Input Methods)..

In programming minimizing the runtime of your program is very necessary as the time limit of the problems is very strict..

Many times I have come across the warning – “Be Careful .. Large Input Output”…

In such problems the optimization in the input/output method really helps in giving a better runtime…

After surfing a lot on the web and reading about the different methods, I made a template for myself, which I use in all my problems and sometimes it help me in getting on the 1st page of best solutions ;)

 You can check all the Fast I/O methods on - [www.spoj.pl/problems/INTEST](http://www.spoj.pl/problems/INTEST) (Input test)  
                                                [www.spoj.pl/problems/INOUTEST](http://www.spoj.pl/problems/INOUTEST) (Input + Output Test)  
  
Due to simplicity most of C++ programmers use cin/cout for the Input/Output...  
But when the testcases are large cin/cout doesn't work.. You can try it on the above problems...  
A solution with cin/cout will time out...  
  
So one of the alternate is the scanf/printf which is nearly 3 times faster than the cin/cout...  
An Accepted solution of the problem INTEST is  
  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
**#include<stdio.h>  
int main()  
{  
 int n,k;  
 scanf("%d%d",&n,&k);  
 int cnt=0;  
 while(n--)  
 {  
  int num;  
  scanf("%d",&num);  
  if(num%k==0)cnt++;   
 }      
 printf("%d",cnt);  
 return 0;  
}**  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
But its runtime is 5.52 seconds which is not very good...  
So we search for other alternates...  
  
Another faster input handling method is the getchar\_unlocked()...  
getchar\_unlocked is thread unsafe as it locks the screen while getting the input.. But using it on the spoj server will not harm your screen :D....  
It is easy to implemet and is very fast...  
We can use getchar\_unlocked to handle integers and strings faster...  
  
An Implementation of getting integer input through getchar\_unlocked() -  
Accepted INTEST Solution   
 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
**#include<iostream>  
#include<cstdio>  
#define gc getchar\_unlocked  
  
void scanint(int &x)  
{  
    register int c = gc();  
    x = 0;  
    for(;(c<48 || c>57);c = gc());  
    for(;c>47 && c<58;c = gc()) {x = (x<<1) + (x<<3) + c - 48;}  
}  
  
int main()  
{     
  
    int n,k;  
    scanint(n);  
    scanint(k);  
    int cnt=0;  
    while(n--)  
    {  
     int num;  
     scanint(num);  
     if(num%k==0)cnt++;   
    }      
    printf("%d",cnt);  
    return 0;               
}**  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
And the runtime of this solution is  0.90 secs.. which is much faster than the scanf...  
  
Registers are faster than memory to access, so the variables which are most frequently used in a C program can be put in registers using *register*keyword...   
  
This implementation was for positive integers.. Similarly you can make one for the +/- integers..  
  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
  
**void scanint(int &x)  
{  
    register int c = gc();  
    x = 0;  
    int neg = 0;  
    for(;((c<48 || c>57) && c != '-');c = gc());  
    if(c=='-') {neg=1;c=gc();}  
    for(;c>47 && c<58;c = gc()) {x = (x<<1) + (x<<3) + c - 48;}  
    if(neg) x=-x;  
}**  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
So this is a very easy to implement fast Input handling method....  
  
We have another fast Input method that is - fread or fread\_unlocked.......  
There isn't much difference between fread and fread\_unlocked as in both the whole input file is first taken in an input buffer and then processed.. Main difference is the locking of screen in fread\_unlocked...  
  
Implementaion of fread in INTEST ...  
 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
**#include<cstdio>  
#include<iostream>  
#include<cstdlib>  
using namespace std;  
#define size 65536  
int main()  
{  
      
    char b[size];  
    int t=0,n,k,cnt=0;  
    int c,j;  
    scanf("%d %d\n",&n,&k);  
    while((c = fread(b,1,size,stdin))>0)  
    {  
               for(j=0;j<c;j++)  
               {  
                if(b[j]=='\n')  
                {  
                              if(t%k==0)cnt++;  
                              t = 0;  
                }  
                else  
                {  
                              t = (t\*10) + (b[j]-'0');                  
                }                
               }  
     }  
    printf("%d\n",cnt);      
    return 0;  
}**  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Now this method is more faster as the runtime of this one is 0.65 secs...  
Using fread\_unlocked increased the runtime to 0.78 secs...  
So better use fread()....  
  
But I found getchar\_unlocked() much easier to manipulate and code.. So that's my choice...  
These were few Input methods that I found useful.. For string handling gets() and fgets() are also good....