scanf和cin的效率

竞赛中读数据的情况最多的莫过于读一大堆整数了，于是我写了一个程序，生成一千万个随机数到data.txt中，一共55MB。然后我写了个程序主干计算运行时间，代码如下：

#include <ctime>int main()

{

int start = clock();

//DO SOMETHING

printf("%.3lf\n",double(clock()-start)/CLOCKS\_PER\_SEC);

}

最简单的方法就算写一个循环scanf了，代码如下：

const int MAXN = 10000000;

int numbers[MAXN];

void scanf\_read()

{

freopen("data.txt","r",stdin);

for (int i=0;i<MAXN;i++)

scanf("%d",&numbers[i]);

}

可是效率如何呢？在我的电脑Linux平台上测试结果为2.01秒。接下来是cin，代码如下

const int MAXN = 10000000;

int numbers[MAXN];

void cin\_read()

{

freopen("data.txt","r",stdin);

for (int i=0;i<MAXN;i++)

std::cin >> numbers[i];

}

出乎我的意料，cin仅仅用了6.38秒，比我想象的要快。cin慢是有原因的，其实默认的时候，cin与stdin总是保持同步的，也就是说这两种方法可以混用，而不必担心文件指针混乱，同时cout和stdout也一样，两者混用不会输出顺序错乱。正因为这个兼容性的特性，导致cin有许多额外的开销，如何禁用这个特性呢？只需一个语句std::ios::sync\_with\_stdio(false);，这样就可以取消cin于stdin的同步了。程序如下：

const int MAXN = 10000000;

int numbers[MAXN];

void cin\_read\_nosync()

{

freopen("data.txt","r",stdin);

std::ios::sync\_with\_stdio(false);

for (int i=0;i<MAXN;i++)

std::cin >> numbers[i];

}

取消同步后效率究竟如何？经测试运行时间锐减到了2.05秒，与scanf效率相差无几了！有了这个以后可以放心使用cin和cout了。

<https://www.cnblogs.com/limera/p/5405705.html>

<https://blog.csdn.net/u012328159/article/details/50808270>