前缀和 差分 简单贪心

By — Nil

I. 前缀和(prefix-sum)

前缀和： 可以理解为「数列的前N项和」，是一种重要的预处理方式。

一维前缀和：对于一个长度为n的序列$\{a_i\}$，如果需要多次查询数组里$[l,r]$位置中序列数字的和(即$\sum_{i=l}^{r} a_i$)，那么就可以考虑使用前缀和。

我们在高中学习数列的时候，都学过数列的前N项和，即

$$S_k = \sum_{i=1}^{k} a_i$$

这个时候我们知道：

$$S_0 = 0, S_i = S_{i-1} + a_i$$

于是我们可以得到另一个式子:

$$\sum a_{[l,r]} = S_r - S_{l-1}$$

然后我们来考虑一下时间复杂度，看看前缀和究竟带给我们什么了～
让我们先来看看下面这个问题：

对于一个长度为N的数列A，我们有Q次询问
对于每一次询问，我们给出一个区间$[L,R]$，你需要给出

$$\sum_{i=L}^{R} a_i$$

solution 1: *Step1*: 读入数列 A $\Longrightarrow$ *Step2*: Q次循环，每次读入L,R
$\Longrightarrow$ *Step3:* $R-L$次循环，求和并输出
很显然的是，读入是$O(N)$的，查询的外层循环Q次，内层循环$R-L$次，
那么最坏情况就是每一次都枚举$1\sim N$所有的数字，时间复杂度是
$O(N+QN)$的.
solution 2: *Step1*: 读入数列 A $\Longrightarrow$ *Step2*: N次循环预处理出$S_{i\in[1,N]}$前缀和
$\Longrightarrow$ *Step3:* Q次循环，每次读入L,R $\Longrightarrow$ *Step4* : 直接输出S_R-S_{L-1}
读入是$O(N)$的，预处理循环Q次，查询的循环Q次，对于每一次查询，直接$O(1)$输出，所以复杂度是$O(N+N+Q)$的.

让我们来看看代码吧

```C++
int main() {
  int n,q,i; scanf("%d%d",&n,&q);
  int a[n+1]; for(i=1;i<=n;i++) scanf("%d",a+i);
  for(;q--;){
    int l,r,ans=0; scanf("%d%d",&l,&r);
    for(i=l;i<=r;i++) ans+=a[i];
    printf("%d",ans);
  }
}
```

```C++
 int main() {
   int n,q,i; scanf("%d%d",&n,&q);
   int a[n+1]; for(i=1;i<=n;i++) scanf("%d",a+i);
   int pre[n+1]; pre[0]=0;
   for(i=1;i<=n;i++) pre[i]=pre[i-1]+a[i];
   for(;q--;){
     int l,r; scanf("%d%d",&l,&r);
     printf("%d",pre[r]-pre[l-1]);
   }
 }
```