一.前缀和

1.一维前缀和

> 对于一个给定的数列A，它的前缀和数列S是通过递推能求出的基本信息之一：
>
> $$S[i]=\sum_{j=1}^{i}A[j]$$

> 一个部分和，即数列A某个下标区间内的数的和，可表示为前缀和相减的形式：
>
> $$sum(l,r)=(\sum_{i=l}^{r}A[i])=S[r]-S[l-1]$$

例一：51nod 1081 子段求和

> 给出一个长度为N的数组，进行Q次查询，查询从第i个元素开始长度为l的子段所有元素之和。
> 例如，1 3 7 9 -1，查询第2个元素开始长度为3的子段和，1 {3 7 9} -1。3 + 7 + 9 = 19，输出19。

2.二维前缀和

> 在二维矩阵中，可类似地求出二维前缀和，进一步计算出二维部分和

例二：Acwing 99. 激光炸弹

一种新型的激光炸弹，可以摧毁一个边长为 R 的正方形内的所有的目标。

现在地图上有 N 个目标，用整数X_i,Y_i表示目标在地图上的位置，每个目标都有一个价值W_i。

激光炸弹的投放是通过卫星定位的，但其有一个缺点，就是其爆炸范围，即那个边长为 R 的正方形的边必须和 x，y轴平行。

若目标位于爆破正方形的边上，该目标不会被摧毁。

求一颗炸弹最多能炸掉地图上总价值为多少的目标。

二.差分

> 对于一个给定的数列A，它的差分数列B定义为：
>
> $$B[1]=A[1],B[i]=A[i]-A[i-1](2\le i\le n)$$
>
> 容易发现，“前缀和”与”差分“是一对互逆运算，差分序列B的前缀和序列就是原序列A，前缀和序列S的差分序列也是原序列A。
>
> 把序列A的区间[l,r]加d（即把A[i]，A[i+1]，…，A[r]都加上d），其差分序列B的变化为B[l]加上d，B[r+1]减去d，其他位置不变。即把原序列上的”区间操作“转化为差分序列上的”单点操作“进行计算，降低求解难度。

三.二分

1.二分查找

> 二分搜索，也称折半搜索、二分查找，是用来在一个有序数组中查找某一元素的算法。
>
> 以在一个升序数组中查找一个数为例。
>
> 它每次考察数组当前部分的中间元素，如果中间元素刚好是要找的，就结束搜索过程；如果中间元素小于所查找的值，那么左侧的只会更小，不会有所查找的元素，只需要到右侧去找就好了；如果中间元素大于所查找的值，同理，右侧的只会更大而不会有所查找的元素，所以只需要到左侧去找。
>
> 在二分搜索过程中，每次都把查询的区间减半，因此对于一个长度为 n的数组，至多会进行o(log n)次查找。

在单调递增序列a中查找≥X的数中最小的一个（即X或者X的后继）

```
int binary_search(int l, int r, int x)
{
    while(l < r)
    {
        int mid = (l + r) >> 1;
        if(a[mid] >= x)
            l = mid;
        else
            r = mid - 1;
    }
    return a[l];
}
```

若a[mid]≥X，则根据序列a的单调性，mid之后的数会更大，所以≥X的最小的数不可能在m i d之后，可行区间应该缩小为左半段。因为mid也可能是答案，故此时应取r=mid。同理，若a[mid]<X，取l=mid+1。

在单调递增序列a中查找≤X的数中最大的一个（即X或者X的前驱）

```
int binary_search(int l, int r, int x)
{
    while(l < r)
    {
        int mid = (l + r + 1) >> 1;
        //不能用（l+r）>>1  r-l=1时，进入l=mid，陷入死循环  进入 r=mid-1  会造成 l>r
        if(a[mid] <= x)
            l = mid;
        else
            r = mid - 1;
    }
    return a[l];
}
```

若a[mid]≤X，则根据序列a的单调性，mid之后的数会更小，所以≤X的最大的数不可能在mid之前，可行区间应该缩小为右半段。因为mid也可能是答案，故此时应取l=mid。同理，若a[mid]>X，取r=mid-1。

STL的二分查找

对于一个有序的 array 你可以使用 `lower_bound()` 来找到第一个大于等于你的值的数，upper_bound()` 来找到第一个大于你的值的数。

请注意，必须是有序数组，否则答案是错误的。

2.二分答案

> 解题的时候往往会考虑枚举答案然后检验枚举的值是否正确。如果我们把这里的枚举换成二分，就变成了“二分答案”。

例题三：poj 1064 **Cable master**

有N条绳子，它们的长度分别为L_i。如果从它们中切割出K条长度相同的绳子的话，这K条绳子每条最长能有多长？答案保留到小数点后2位。$1 \le N \le 10000$。$1 \le K \le 10000$。$1 \le L_i \le 100000$

例题四：最大化平均值

有N个物品的重量和价值分别是W_i和V_i。从中选出K个物品使得单位重量的价值最大

限制条件：$1 \le K \le N \le 1e4$。$1 \le W_i$，$V_i \le 1e6$。

例题五：POJ 3273 **Monthly Expense最大化最小值**

ZZY非常牛逼，这一天他看到了这样一道题：

给你一个长度为N的序列，现在需要把他们切割成M个子序列（要求子序列是连续的，如 abcd 切割成 ac 和 bd 就是不合法的），而且要求，每个子序列的和均不超过某个值X。

ZZY觉得这个题太小儿科了，于是他去写模电作业了，静静的等着你们AC(WA)。

四.贪心

> 贪心算法顾名思义就是用计算机来模拟一个“贪心”的人做出决策的过程。
>
> 这个人每一步行动总是按某种指标选取最优的操作，他总是 **只看眼前，并不考虑以后可能造成的影响** 。
>
> 可想而知，并不是所有的时候贪心法都能获得最优解，所以一般使用贪心法的时候，都要确保自己能证明其正确性。

> 常见的证明手段有（了解一下）：
>
> 1.微扰（邻项交换）
>
> 证明在任意局面下，任何对局部最优策略的微小改变都会造成整体结果变差。经常用于以“排序”为贪心策略的证明。
>
> 2.范围缩放
>
> 证明任何对局部最优策略作用范围的扩展都不会造成整体结果变差。
>
> 3.决策包容性
>
> 证明在任意局面下，作出局部最优决策以后，在问题状态空间中的可达集合包含了作出其他任何决策后的可达集合。换言之，这个局部最优策略提供的可能性包含其他所有策略提供的可能性。
>
> 3.反证法
>
> 5.数学归纳法

例题六：硬币问题

有1元、5元、10元、50元、100元、500元的硬币各C_1、C_5、C_10、C_50、C_100、C_500枚。现在要用这些硬币来支付A元，最少需要多少枚硬币？假设本题至少存在一种支付方案。限制条件：$0 \leq$ 硬币数量$C \leq 1e9.0 \leq A \leq 1e9$

例题七：洛谷 P1969 积木大赛

春春幼儿园举办了一年一度的“积木大赛”。今年比赛的内容是搭建一座宽度为n的大厦，大厦可以看成由n块宽度为1的积木组成，第i块积木的最终高度需要是h_i。

在搭建开始之前，没有任何积木（可以看成n块高度为0的积木）。接下来每次操作，小朋友们可以选择一段连续区间$[l, r]$，然后将第第L块到第 R 块之间（含第L 块和第 R块）所有积木的高度分别增加1。

小M是个聪明的小朋友，她很快想出了建造大厦的最佳策略，使得建造所需的操作次数最少。但她不是一个勤于动手的孩子，所以想请你帮忙实现这个策略，并求出最少的操作次数。

例题八：区间问题

有N项工作，每项工作分别在S_i时间开始，在T_i时间结束。对于每项工作，你都可以选择是否参与。如果选择了参与，那么自始至终都必须全程参与。此外，参与工作的时间段不能重叠（即是开始的瞬间和结束的瞬间的重叠也是不允许的）

限制条件：$1 \leq N \leq 1e5$。$1 \leq S[i] \leq T[i] \leq 1e9.$

例题九：P1080 国王游戏

恰逢 H国国庆，国王邀请n 位大臣来玩一个有奖游戏。首先，他让每个大臣在左、右手上面分别写下一个整数，国王自己也在左、右手上各写一个整数。然后，让这 n 位大臣排成一排，国王站在队伍的最前面。排好队后，所有的大臣都会获得国王奖赏的若干金币，每位大臣获得的金币数分别是：排在该大臣前面的所有人的左手上的数的乘积除以他自己右手上的数，然后向下取整得到的结果。

国王不希望某一个大臣获得特别多的奖赏，所以他想请你帮他重新安排一下队伍的顺序，使得获得奖赏最多的大臣，所获奖赏尽可能的少。注意，国王的位置始终在队伍的最前面。

五.高精度

> 高精度算法，属于处理大数字的数学计算方法。在一般的科学计算中，会经常算到小数点后几百位或者更多，当然也可能是几千亿几百亿的大数字。一般这类数字我们统称为高精度数，高精度算法是用计算机对于超大数据的一种模拟加，减，乘，除，乘方，阶乘，开方等运算。对于非常庞大的数字无法在计算机中正常存储，于是，将这个数字拆开，拆成一位一位的，或者是四位四位的存储到一个数组中， 用一个数组去表示一个数字，这样这个数字就被称为是高精度数。高精度算法就是能处理高精度数各种运算的算法，但又因其特殊性，故从普通数的算法中分离，自成一家。

例题十：高精度加法

> 第一行输入一个T，表示测试样例的个数，接下来每行输入一个A,B(小于1000位)

例题十一：洛谷 P2142 高精度减法

> ## 题目描述
>
> 高精度减法
>
> ## 输入格式
>
> 两个整数a,b（第二个可能比第一个大）
>
> ## 输出格式
>
> 结果（是负数要输出负号）