前缀和

前缀和主要用于原始数据不被修改的情况下，频繁查询某个区间的累加和。

```
class prefixsum{
public:
    // 传入一个数组构造前缀和
    void PrefixSum(std::vector<int> &nums){
        prefsum = std::vector<int>(nums.size()+1, 0);
        // 计算累加和
        for(int i = 1;i< nums.size();i++) {
            prefsum[i] = prefsum[i-1] + nums[i-1];
        }

    }

    int query(int i, int j) {
        // 查询闭区间[i,j]的和
        return prefsum[j+1] - prefsum[i];
    }

private:
    std::vector<int> prefsum;
};
```

差分数组：频繁对原始数组的某个区间的元素进行删减。 对nums构造一个差分数据diff:

```
std::vector<int> diff = std::vector<int>(nums.size(), 0);
// 构造差分数组
diff[0] = nums[0];
for(int i = 1; i < nums.size();i++) {
    diff[i] = nums[i] - nums[i-1];
}
```

通过差分数据可反推原数组：

```
std::vector<int> res = std::vector<int> (diff.size());
// 根据差分数组构造原数组
res[0] = nums[0];
for(int i = 1;i<diff.size(); i++) {
    res[i] = res[i-1] + diff[i-1];
}
```

通过差分数组可以方便的进行区间操作，如对nums[i...j]之间的元素都+3，可以表示为diff[i] += 3,这样nums[i...nums.size()-1]之间的元素都+3, 再将diff[j+1]-=3,表示nums[j+1...nums.size()-1]之间的元素都-3;此时完成了nums[i...j]的元素都+3的操作。

```cpp
#include <vector>

class Difference {
public:
  // 构造差分数组
  void diffSum(std::vector<int> nums) {
    int len = nums.size();
    diff[0] = nums[0];
    for (int i = 0; i < len; i++) {
      diff[i] = nums[i] - nums[i - 1];
    }
  }

  // 区间操作
  void increment(int i, int j, int val) {
    diff[i] += val;
    if (j + 1 < diff.size()) {
      diff[j + 1] -= val;
    }
  }

  // 还原数组
  std::vector<int> result() {
    int              len = diff.size();
    std::vector<int> res(len);
    res[0] = diff[0];
    for (int i = 1; i < len; i++) {
      res[i] = res[i - 1] + diff[i];
    }

    return res;
  }

private:
  std::vector<int> diff;  // 差分数组
};
```

算法实践

leetcode 1109航班预订

```cpp
class Solution {
public:
  std::vector<int> corpFlightBookings(std::vector<std::vector<int>>&
bookings,
                                      int                            n) {
    int              len = bookings.size();
    std::vector<int> diff(n, 0);

    for (int i = 0; i < len; i++) {
```

```
        std::vector<int> tmp   = bookings[i];
        int              start = tmp[0] - 1;
        int              end   = tmp[1] - 1;
        int              book  = tmp[2];
        diff[start] += book;
        if (end + 1 < n) {
          diff[end + 1] -= book;
        }
      }
      // 构造结果数组
      std::vector<int> res(n, 0);
      res[0] = diff[0];
      for (int i = 1; i < n; i++) {
        res[i] = res[i - 1] + diff[i];
      }

      return res;
    }
};
```