NOI2010 超级钢琴

> 求前 k 大长度为 $L \sim R$ 的字段和的和 。

算法 : ST表 + 堆 + 贪心 。

首先考虑因为区间连续 ， 则可以预处理前缀和 。

子任务 *1* ： $k = 1$

所以枚举 i : $[1, n]$ 。

对于每一次访问 ， 查询 $\max(s[\min(i + k, n)] - s[i - 1]), k \in [l, r]$ 。

用 ST表 $nlogn$ 预处理即可 。

正解 :

通过 $k = 1$ 我们可以受到启发 。

即查询必须 $O(1)$ 解决 。

对于查询区间 $[i, [i + l, i + r]]$ [1] ， 若当前区间的最优位置为 t 。

则当前区间的次优解 p , 必定属于 $[i + l, t - 1]$ 或 $[t + 1, i + r]$ 。

于是将问题转换成了形式相同的子问题 。所以 ， 当扩展到全局时 ， **贪心**地取 k 次 ， 便是所求的答案 。

实现难点 :

与一般的 ST 表不同 ， 由于我们需要知道最优解的位置 t ， 故用 ST 表存最优解位置而不是其区间和 。

关键代码 :

```
namespace ST{
    inline void makeST(){
        for(int i = 1 ; i <= n ; ++i) f[i][0] = i;
        for(int j = 1 ; j <= 19; ++j){
            for(int i = 1 ; i + (1 << j) - 1 <= n ; ++i){
                int x = f[i][j - 1];int y = f[i + (1 << j - 1)][j - 1];
                f[i][j] = (s[x] > s[y] ? x : y);
            }
        }
        return;
    }
    inline int ask(int l,int r){
        int k = (int)log2(r - l + 1);
        int x = f[l][k] ;int y = f[r - (1<<k) + 1][k];
        return (s[x] > s[y] ? x : y);
    }
};
```

---

1. $[i + l, i + r]$ 表示区间 。 ↩