蛟龙五班

区间DP

Mas

区间DP

区间类动态规划是线性动态规划的扩展

它在分阶段地划分问题时,与阶段中元素出现的顺序和由前一阶段的哪些元素合并而来有很大的关系

若状态$dp[l][r]$表示将$l \sim r$下标位置的所有元素合并能获得的价值的最大值

$$dp[l][r] = \max_{l \le k < r}\{dp[l][k] + dp[k+1][r] + cost\}$$

其中$cost$为将这两组元素合并起来的代价。

计算完所有 dp 值时, $dp[1][n]$ 就是最终的答案。

不仅仅是 $dp[1][n]$,其实每个连续子区间 $[l,r]$ 的答案也都保存在 $dp[l][r]$ 里了。

利用这个性质,如果我们的区间动态规划代码没有通过一组较大的数据,可以尝试把所有子区间与它们的 dp 值都打印出来,肉眼找到一个最短的算错的区间。然后再以这段区间当做读入去调试,会方便很多。

区间DP转移顺序

状态$dp[l][r]$表示将$l \sim r$下标位置的所有元素合并能获得的价值的最大值

$$dp[l][r] = \max_{l \le k < r}\{dp[l][k] + dp[k+1][r] + cost\}$$

计算 $dp[l][r]$ 时需要知道所有$dp[l][k], dp[k+1][r]$,

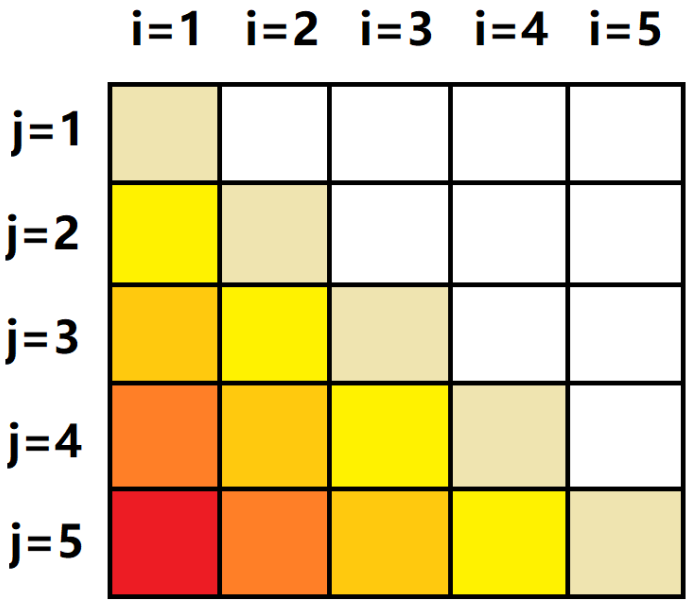
如何保证这种拓扑性？使用记忆化搜索的方式能实现,但不推荐。

$dp[l][r]$ 转移所需用到的区间长度一定都短于它本身的长度。

不妨以$len = r - l + 1$作为阶段。

首先从小到大枚举 len,然后枚举 l 的值,根据 len 和 l 计算出 r。

所有区间类动态规划都推荐用这种枚举方式。



#1484、合并石子

题目描述

在一个操场上一排地摆放着 N 堆石子

现要将石子有次序地合并成一堆。规定每次只能选相邻的 2 堆石子合并成新的一堆，并将新的一堆石子数记为该次合并的得分

计算出将 N 堆石子合并成一堆的最小得分

输入格式

第一行为一个正整数 $N(2 \leq N \leq 100)$

以下 N 行,每行一个正整数，小于 10000 ，分别表示第 i 堆石子的个数 $(1 \leq i \leq N)$

输出格式

一个正整数,即最小得分

输入样例1

```
3
1
2
3
```

输出样例1

```
9
```

#1484、合并石子

设$dp[l][r]$为将区间$[l,r]$合并成一堆的最大/小代价。现在我们**考虑最后一次合并**,即 $[l,r]$ 这一堆是由哪两堆最后合并得到。

因为是在区间里合并石子,枚举分界线 k, 只可能 $[l,k]$ 和 $[k+1,r]$ 这两个区间先各自合并成一堆,然后再合成 $[l,r]$。

$[l,k]$ 和 $[k,r]$ 内部的合并方案直接调用 $dp[l][k], dp[k+1][r]$ 即可,而最后一次合并时的代价则是 $\sum_{i=l}^{r} a[i]$,用前缀和快速计算。

区间 dp 往往还要考虑初始化。本题的初始条件是每堆石子本身,此时还没花费代价,即初始化 $dp[i][i]=0$

时间复杂度$O(n^3)$

```
for (int len = 1; len <= n; len++)
    for (int l = 1; l + len - 1 <= n; l++)
    {
        int r = l + len - 1;
        for (int k = l; k < r; k++)
            dp[l][r] = min(dp[l][r], dp[l][k] + dp[k + 1][r] + sum[r] - sum[l - 1]);
    }
```

#732、石子合并

题目描述

将 n 堆石子绕圆形操场排放，现要将石子有序地合并成一堆。规定每次只能选相邻的两堆合并成新的一堆，并将新的一堆的石子数记做该次合并的得分

请编写一个程序，读入堆数 n 及每堆的石子数，并进行如下计算：

选择一种合并石子的方案，使得做 $n-1$ 次合并得分总和最大

选择一种合并石子的方案，使得做 $n-1$ 次合并得分总和最小

输入格式

输入第一行一个整数 n，表示有 n 堆石子

第二行 n 个整数，表示每堆石子的数量

输出格式

输出共两行：

第一行为合并得分总和最小值

第二行为合并得分总和最大值

数据范围与提示

对于 100% 的数据，有 $1 \leq n \leq 200$

样例输入

```
4
4 5 9 4
```

样例输出

```
43
54
```

#732、石子合并

设$dp[l][r]$为将区间$[l,r]$合并成一堆的最大/小代价

由于区间成环,尝试枚举一条边破开,那么破环成链存在n种情况,将每种情况合并成一堆取最值

时间复杂度$O(n^4)$

不难发现上述做法中对一个区间有多次重复计算

考虑将链延长两倍变成 $2\times n$ 堆,其中第i堆与第 $n+i$ 堆相同

用动态规划求解后,取 $dp[1][n],dp[2][n+1],\cdots,dp[i][n+i-1],\cdots$中的最优值即为答案

时间复杂度为$O(n^3)$

#1475、山区建小学

题目描述

政府在某山区修建了一条道路,恰好穿越总共 m 个村庄的每个村庄一次,没有回路或交叉,任意两个村庄只能通过这条路来往

已知任意两个相邻的村庄之间的距离为 d_i (为正整数),其中 $0 < i < m$

为了提高山区的文化素质,政府又决定从 m 个村中选择 n 个村建小学(设 $0 < n \leq m \leq 500$)

请根据给定的 m, n 以及所有相邻村庄的距离,选择在哪些村庄建小学,才使得所有村到最近小学的距离总和最小,计算最小值

输入格式

第 1 行为 m 和 n ,其间用空格间隔

第 2 行为 $m-1$ 个整数,依次表示从一端到另一端的相邻村庄的距离,整数之间以空格间隔

例如:

```
10 3
2 4 6 5 2 4 3 1 3
```

表示在 10 个村庄建 3 所学校。第 1 个村庄与第 2 个村庄距离为 2 ,第 2 个村庄与第 3 个村庄距离为 4 ,第 3 个村庄与第 4 个村庄距离为 6 ,...,第 9 个村庄到第 10 个村庄的距离为 3

输出格式

各村庄到最近学校的距离之和的最小值

输入样例

```
10 2
3 1 3 1 1 1 1 1 3
```

输出样例

```
18
```

#1475、山区建小学

若区间 $[l,r]$ 要建立一所学校, $r-l+1$ 是奇数时建立在 $\frac{l+r}{2}$ 处距离和最小,偶数时建立在最中间两点距离和最小。

令 $dis[l][r]$ 表示在$[l,r]$范围内建一所学校的最小距离和

设 $dp[i][j]$ 为考虑完前i个村庄建立j个学校的最小距离和,答案为$dp[n][m]$

$$dp[i][j]=\min_{1\le k<i}\{dp[k][j-1]+dis[k+1][i]\}$$

时间复杂度$O(n^2m)$

严格意义上来讲,这题不算是区间动态规划,因为只是在 $[1,i]$ 这个前缀上表示状态。

今后我们拿到一道疑似区间动态规划的题时,需要首先考虑一件事：假设状态表示只包含前缀 $[1,i]$,转移 $dp[i]=dp[j]+cost(j+1,i)$ 中 $cost$ 函数能否**独立**地计算？如果能,用前缀区间代替所有区间可以省空间和时间。

#735、括号配对

题目描述

给定一个由 `(` , `)` , `[` , `]` 括号组成的字符串 S .

我们可以在任何位置添加 `(` , `)` , `[` , `]` ，以使得到的括号字符串有效。

从形式上讲，只有满足下面几点之一，括号字符串才是有效的：

> *空串*
> *如果一个串 S 是合法的，那么 (S) 、 $[S]$ 也是合法的*
> *如果 a 和 b 是合法的，那么 ab 也是合法的*

下面几个串是合法的

```
()
[]
(())
()[]
()[()]
```

这几个不是合法的

```
(
]
)(
([)]
([(]
```

给定一个括号字符串，返回为使结果字符串变成合法括号串而必须添加的最少括号数。

输入格式

输入仅一行，为字符串

输出格式

输出仅一个整数，表示增加的最少字符数

样例输入1

```
[])
```

样例输出1

```
1
```

样例输入2

```
()())(
```

样例输出2

```
2
```

数据范围与提示

对于 100% 的数据，输入的字符串长度小于 100

#735、括号配对

设$dp[l][r]$为将区间$[l,r]$变为合法的最小代价

初始时

若$l=r$,$dp[l][r]=1$

若$l>r$,$dp[l][r]=0$

若$l\le r$,$dp[l][r]=\infty$

大区间可由小区间合并得到

$$dp[l][r]=\min_{l\le k<r}\{dp[l][k]+dp[k+1][r]\}$$

若有$s[l]$、$s[r]$匹配成功

$$dp[l][r]=\min(dp[l][r],dp[i+1][r-1])$$

时间复杂度$O(n^3)$

括号配对变式

给出一个包含 (,),[,] 的字符串,找到一个最长的子序列使其是合法的括号序列。问最大的长度是多少?

设 $dp[l][r]$ 为区间 $[l,r]$ 里能找到的最长合法括号子序列长度。

现在我们来考虑 $dp[l][r]$ 两端的两个字符发挥了什么作用:

- $s[l]$、$s[r]$ 都有用,且它们正好匹配成功

$$dp[l][r] = \max(dp[l][r], dp[l+1][r-1] + 2)$$

- $s[l]$、$s[r]$ 都有用,而且 $[l,r]$ 是由两段拼成的

$$dp[l][r] = \max(dp[l][r], \max_{l \le k < r}(dp[l][k] + dp[k+1][r]))$$

- $s[l]$、$s[r]$ 至少有一个没有用

$$dp[l][r] = \max(dp[l][r], \max(dp[l][r-1], dp[l+1][r]))$$

时间复杂度 $O(n^3)$

#743、加分二叉树

题目描述

设一个 n 个节点的二叉树 tree 的中序遍历为 $(1,2,3,\cdots,n)$,其中数字 $1,2,3,\cdots,n$ 为节点编号。

每个节点都有一个分数（均为正整数）,记第 i 个节点的分数为 d_i，tree 及它的每个子树都有一个加分,任一棵子树 subtree（也包含 tree 本身）的加分计算方法如下:

记 subtree 的左子树加分为 l,右子树加分为 r，subtree 的根的分数为 a,则 subtree 的加分为:

$l \times r + a$

若某个子树为空,规定其加分为 1,叶子的加分就是叶节点本身的分数。不考虑它的空子树。

试求一棵符合中序遍历为 $(1,2,3,\cdots,n)$ 且加分最高的二叉树 tree。

要求输出:

tree 的最高加分； tree 的前序遍历结果

输入格式

第一行一个整数 n 表示节点个数;

第二行 n 个空格隔开的整数,表示各节点的分数

输出格式

第一行一个整数,为最高加分 b；

第二行 n 个用空格隔开的整数,为该树的前序遍历

样例输入

```
5
5 7 1 2 10
```

样例输出

```
145
3 1 2 4 5
```

数据范围与提示

对于 100% 的数据, $n < 30, b < 100$,结果不超过 4×10^9

#743、加分二叉树

设$dp[l][r]$为区间$[l,r]$的最高加分,答案为$dp[1][n]$

每次枚举区间$[l,r]$内的一个点作为根

令$idx[l][r]$为$[l,r]$内根节点编号

$$dp[l][r]=\max_{l\le k\le r}\{dp[l][k-1]\times dp[k+1][r]+w_k\}$$

需要特判左/右子树为空的情况

当$dp[l][r]$取到最大值时,记录根节点,递归输出即可

时间复杂度$O(n^3)$

#736、分离与合体

题目描述

经过在机房里数日的切磋,LYD 从杜神牛那里学会了分离与合体,出关前,杜神牛给了他一个测试……

杜神牛造了 n 个区域,他们紧邻着排成一行,编号 $1 \sim n$

在每个区域里都放着一把 OI 界的金钥匙,每一把都有一定的价值,LYD 当然想得到他们了。然而杜神牛规定 LYD 不能一下子把他们全部拿走,而是每次只可以拿一把。为了尽快得到所有金钥匙,LYD 自然就用上了刚学的分离与合体特技

一开始 LYD 可以选择 $1 \sim n-1$ 中的任何一个区域进入,我们不妨把这个区域记为 k。进入后 LYD 会在 k 区域发生分离,从而分离成两个小 LYD。分离完成的同时会有一面墙在 k 区域和 $k+1$ 区域间升起,从而把 $1..k$ 和 $k+1..n$ 阻断成两个独立的区间,并在各自区间内任选除区间末尾之外(即从 $1 \sim k-1$ 和 $k+1 \sim n-1$ 中选取)的任意一个区域再次发生分离,这样就有了四个小小 LYD……重复以上所叙述的分离,直到每个小 LYD 发现自己所在的区间只剩下了一个区域,那么他们就可以抱起自己梦寐以求的 OI 金钥匙

但是 LYD 不能就分成这么多个个体存在于世界上,这些小 LYD 还会再合体,合体的小 LYD 所在区间中间的墙会消失。合体会获得 $($ 合并后所在区间左右端区域里金钥匙价值之和 $) \times ($ 之前分离的时候所在区域的金钥匙价值 $)$

例如,LYD 曾在 $1 \sim 3$ 区间中的 2 号区域分离成为 $1 \sim 2$ 和 $3 \sim 3$ 两个区间,合并时获得的价值就是 $($ 1 号金钥匙价值 $+$ 3 号金钥匙价值 $) \times ($ 2 号金钥匙价值 $)$

输入格式

第一行一个正整数 n 第二行 n 个用空格分开的正整数 a_i ,表示 $1 \sim n$ 区域里每把金钥匙的价值

输出格式

第一行一个数,表示获得的最大价值
第二行按照分离阶段从前到后,区域从左到右的顺序,输出发生分离区域编号。
若有多种方案,选择分离区域尽量靠左的方案(也可以理解为输出字典序最小的)

样例输入

```
7
1 2 3 4 5 6 7
```

样例输出

```
238
1 2 3 4 5 6
```

数据范围与提示

对于 20% 的数据, $n \leq 10$
对于 40% 的数据, $n \leq 50$
对于 100% 的数据, $n, a_i \leq 300$,保证运算过程和结果不超过 32 位正整数范围

#736、分离与合体

设$dp[l][r]$为区间$[l,r]$的最高得分,答案为$dp[1][n]$

每次枚举区间$[l,r]$内的一个分离点

令$idx[l][r]$为$[l,r]$内分离点编号

$$dp[l][r]=\max_{l\le k\le r}\{dp[l][k]+\ dp[k+1][r]+(w_l+w_r)\times w_k\}$$

对于输出方案需要分层次输出,直接BFS输出即可

时间复杂度$O(n^3)$

谢谢观看