

312. Burst Balloons

Hard 8826 243 Add to List Share

You are given `n` balloons, indexed from `0` to `n - 1`. Each balloon is painted with a number on it represented by an array `nums`. You are asked to burst all the balloons.

If you burst the i^{th} balloon, you will get $\text{nums}[i - 1] * \text{nums}[i] * \text{nums}[i + 1]$ coins. If $i - 1$ or $i + 1$ goes out of bounds of the array, then treat it as if there is a balloon with a 1 painted on it.

Return the *maximum* coins you can collect by bursting the balloons wisely.

Example 1:

Input: nums = [3,1,5,8]
Output: 167

3,1,5,8

275 35 ✓ (3 11 5)

375 105

31158

15
15
3
180

(i) 2 3 5

$2 \Rightarrow 1 \times 2 \times 3 = 6$

[3 5]

$1 \times 3 \times 5 = 15$

[5] 5

26

(ii) 2 3 5

$2 = 1 \times 2 \times 3 = 6$

[3 5] = $3 \times 5 = 15$

$2 \Rightarrow 1 \times 3 \times 1 = 3$

24

(iii) 1 2 5

$2 \Rightarrow 1 \times 2 \times 5 = 10$

[2 5] = $1 \times 2 \times 5 = 10$

[5] = $1 \times 5 \times 1 = 5$

45

(iv) 3 5 2

$3 = 2 \times 3 \times 5 = 30$

[3 5] = $2 \times 3 \times 1 = 6$

[2] = 2

42

5 3 2

$5 = 3 \times 5 \times 1 = 15$

[2 3] = $2 \times 3 \times 1 = 6$

[2] = 2

23

5 2 3

$5 = 3 \times 5 \times 1 = 15$

[2 3] = $1 \times 2 \times 3 = 6$

3 - 3

24

(v) 0 1 2 3

[3, 1, 5, 8]

[3 5 8]

2 8

15

120

24

8

[illegible]

Diagram illustrating the recursive steps for calculating the binomial coefficient $C(5, 2)$ using Pascal's Triangle:

- Initial state: $C(5, 2)$ (circled in green).
- Step 1: $C(5, 2)$ depends on $C(4, 1)$ and $C(4, 3)$ (circled in green).
- Step 2: $C(4, 1)$ depends on $C(3, 0)$ and $C(3, 2)$ (circled in green).
- Step 3: $C(3, 2)$ depends on $C(2, 1)$ and $C(2, 3)$ (circled in green).
- Step 4: $C(2, 1)$ depends on $C(1, 0)$ and $C(1, 2)$ (circled in green).
- Step 5: $C(1, 0)$ and $C(1, 2)$ are base cases (circled in green).

Final result: $C(5, 2) = 10$ (circled in green).

```
public static int MaxCoins(int[] arr, int si, int ei) {
    if (si+1==ei) {
        return 0;
    }
    int ans=Integer.MIN_VALUE;
    for (int k = si+1; k < ei; k++) {
        int left = MaxCoins(arr, si, k);
        int right =MaxCoins(arr, k, ei);
        int self=arr[si]*arr[k]*arr[ei];
        ans = Math.max(ans, left+right+self);
    }
    return ans;
}
```

Handwritten notes on a whiteboard:

- Top left: A bracketed vertical line.
- Top center: Two horizontal arrows pointing right, one above the other.
- Center: A small icon of a picture with a red 'x' below it.
- Bottom left: Two checkmarks.
- Bottom center: A horizontal line.
- Bottom right: A series of handwritten symbols and boxes: $f(x) = (8)$, a circled 0 , a circled 1 , and a circled 2 .

Handwritten notes and diagrams illustrating the Huffman tree construction process:

- Top left: A small diagram showing a box with a picture icon and a star, labeled with a circled 'x'.
- Top right: A partial Huffman tree diagram showing nodes $(1,0)$ and $(0,1)$ with associated values $6-1=5$ and $2-1=1$.
- Center: A large Huffman tree diagram showing the merging of nodes. The root node is $(3,0)$. The left child is $(2,0)$ and the right child is $(1,1)$. The $(2,0)$ node has children $(3,0)$ and $(2,1)$. The $(1,1)$ node has children $(2,1)$ and $(1,2)$. The $(2,1)$ node has children $(2,1)$ and $(2,2)$. The $(1,2)$ node has children $(1,2)$ and $(1,3)$. The $(2,2)$ node has children $(2,2)$ and $(2,3)$. The $(1,3)$ node has children $(1,3)$ and $(1,4)$. The $(2,3)$ node has children $(2,3)$ and $(2,4)$. The $(1,4)$ node has children $(1,4)$ and $(1,5)$. The $(2,4)$ node has children $(2,4)$ and $(2,5)$. The $(1,5)$ node has children $(1,5)$ and $(1,6)$. The $(2,5)$ node has children $(2,5)$ and $(2,6)$. The $(1,6)$ node has children $(1,6)$ and $(1,7)$. The $(2,6)$ node has children $(2,6)$ and $(2,7)$. The $(1,7)$ node has children $(1,7)$ and $(1,8)$. The $(2,7)$ node has children $(2,7)$ and $(2,8)$. The $(1,8)$ node has children $(1,8)$ and $(1,9)$. The $(2,8)$ node has children $(2,8)$ and $(2,9)$. The $(1,9)$ node has children $(1,9)$ and $(1,10)$. The $(2,9)$ node has children $(2,9)$ and $(2,10)$. The $(1,10)$ node has children $(1,10)$ and $(1,11)$. The $(2,10)$ node has children $(2,10)$ and $(2,11)$. The $(1,11)$ node has children $(1,11)$ and $(1,12)$. The $(2,11)$ node has children $(2,11)$ and $(2,12)$. The $(1,12)$ node has children $(1,12)$ and $(1,13)$. The $(2,12)$ node has children $(2,12)$ and $(2,13)$. The $(1,13)$ node has children $(1,13)$ and $(1,14)$. The $(2,13)$ node has children $(2,13)$ and $(2,14)$. The $(1,14)$ node has children $(1,14)$ and $(1,15)$. The $(2,14)$ node has children $(2,14)$ and $(2,15)$. The $(1,15)$ node has children $(1,15)$ and $(1,16)$. The $(2,15)$ node has children $(2,15)$ and $(2,16)$. The $(1,16)$ node has children $(1,16)$ and $(1,17)$. The $(2,16)$ node has children $(2,16)$ and $(2,17)$. The $(1,17)$ node has children $(1,17)$ and $(1,18)$. The $(2,17)$ node has children $(2,17)$ and $(2,18)$. The $(1,18)$ node has children $(1,18)$ and $(1,19)$. The $(2,18)$ node has children $(2,18)$ and $(2,19)$. The $(1,19)$ node has children $(1,19)$ and $(1,20)$. The $(2,19)$ node has children $(2,19)$ and $(2,20)$. The $(1,20)$ node has children $(1,20)$ and $(1,21)$. The $(2,20)$ node has children $(2,20)$ and $(2,21)$. The $(1,21)$ node has children $(1,21)$ and $(1,22)$. The $(2,21)$ node has children $(2,21)$ and $(2,22)$. The $(1,22)$ node has children $(1,22)$ and $(1,23)$. The $(2,22)$ node has children $(2,22)$ and $(2,23)$. The $(1,23)$ node has children $(1,23)$ and $(1,24)$. The $(2,23)$ node has children $(2,23)$ and $(2,24)$. The $(1,24)$ node has children $(1,24)$ and $(1,25)$. The $(2,24)$ node has children $(2,24)$ and $(2,25)$. The $(1,25)$ node has children $(1,25)$ and $(1,26)$. The $(2,25)$ node has children $(2,25)$ and $(2,26)$. The $(1,26)$ node has children $(1,26)$ and $(1,27)$. The $(2,26)$ node has children $(2,26)$ and $(2,27)$. The $(1,27)$ node has children $(1,27)$ and $(1,28)$. The $(2,27)$ node has children $(2,27)$ and $(2,28)$. The $(1,28)$ node has children $(1,28)$ and $(1,29)$. The $(2,28)$ node has children $(2,28)$ and $(2,29)$. The $(1,29)$ node has children $(1,29)$ and $(1,30)$. The $(2,29)$ node has children $(2,29)$ and $(2,30)$. The $(1,30)$ node has children $(1,30)$ and $(1,31)$. The $(2,30)$ node has children $(2,30)$ and $(2,31)$. The $(1,31)$ node has children $(1,31)$ and $(1,32)$. The $(2,31)$ node has children $(2,31)$ and $(2,32)$. The $(1,32)$ node has children $(1,32)$ and $(1,33)$. The $(2,32)$ node has children $(2,32)$ and $(2,33)$. The $(1,33)$ node has children $(1,33)$ and $(1,34)$. The $(2,33)$ node has children $(2,33)$ and $(2,34)$. The $(1,34)$ node has children $(1,34)$ and $(1,35)$. The $(2,34)$ node has children $(2,34)$ and $(2,35)$. The $(1,35)$ node has children $(1,35)$ and $(1,36)$. The $(2,35)$ node has children $(2,35)$ and $(2,36)$. The $(1,36)$ node has children $(1,36)$ and $(1,37)$. The $(2,36)$ node has children $(2,36)$ and $(2,37)$. The $(1,37)$ node has children $(1,37)$ and $(1,38)$. The $(2,37)$ node has children $(2,37)$ and $(2,38)$. The $(1,38)$ node has children $(1,38)$ and $(1,39)$. The $(2,38)$ node has children $(2,38)$ and $(2,39)$. The $(1,39)$ node has children $(1,39)$ and $(1,40)$. The $(2,39)$ node has children $(2,39)$ and $(2,40)$. The $(1,40)$ node has children $(1,40)$ and $(1,41)$. The $(2,40)$ node has children $(2,40)$ and $(2,41)$. The $(1,41)$ node has children $(1,41)$ and $(1,42)$. The $(2,41)$ node has children $(2,41)$ and $(2,42)$. The $(1,42)$ node has children $(1,42)$ and $(1,43)$. The $(2,42)$ node has children $(2,42)$ and $(2,43)$. The $(1,43)$ node has children $(1,43)$ and $(1,44)$. The $(2,43)$ node has children $(2,43)$ and $(2,44)$. The $(1,44)$ node has children $(1,44)$ and $(1,45)$. The $(2,44)$ node has children $(2,44)$ and $(2,45)$. The $(1,45)$ node has children $(1,45)$ and $(1,46)$. The $(2,45)$ node has children $(2,45)$ and $(2,46)$. The $(1,46)$ node has children $(1,46)$ and $(1,47)$. The $(2,46)$ node has children $(2,46)$ and $(2,47)$. The $(1,47)$ node has children $(1,47)$ and $(1,48)$. The $(2,47)$ node has children $(2,47)$ and $(2,48)$. The $(1,48)$ node has children $(1,48)$ and $(1,49)$. The $(2,48)$ node has children $(2,48)$ and $(2,49)$. The $(1,49)$ node has children $(1,49)$ and $(1,50)$. The $(2,49)$ node has children $(2,49)$ and $(2,50)$. The $(1,50)$ node has children $(1,50)$ and $(1,51)$. The $(2,50)$ node has children $(2,50)$ and $(2,51)$. The $(1,51)$ node has children $(1,51)$ and $(1,52)$. The $(2,51)$ node has children $(2,51)$ and $(2,52)$. The $(1,52)$ node has children $(1,52)$ and $(1,53)$. The $(2,52)$ node has children $(2,52)$ and $(2,53)$. The $(1,53)$ node has children $(1,53)$ and $(1,54)$. The $(2,53)$ node has children $(2,53)$ and $(2,54)$. The $(1,54)$ node has children $(1,54)$ and $(1,55)$. The $(2,54)$ node has children $(2,54)$ and $(2,55)$. The $(1,55)$ node has children $(1,55)$ and $(1,56)$. The $(2,55)$ node has children $(2,55)$ and $(2,56)$. The $(1,56)$ node has children $(1,56)$ and $(1,57)$. The $(2,56)$ node has children $(2,56)$ and $(2,57)$. The $(1,57)$ node has children $(1,57)$ and $(1,58)$

```

public static int calculateMinimum(int[][] arr, int cr, int cc) {
    if (cr == arr.length || cc == arr[0].length) {
        return Integer.MAX_VALUE;
    }
    if (cr == arr.length - 1 && cc == arr[0].length - 1) {
        return arr[cr][cc] > 0 ? 1 : 1 - arr[cr][cc];
    }
    int left = calculateMinimum(arr, cr, cc + 1);
    int right = calculateMinimum(arr, cr + 1, cc);
    int p = Math.min(left, right) - arr[cr][cc];

    return p > 0 ? p : 1;
}

```

[illegible]