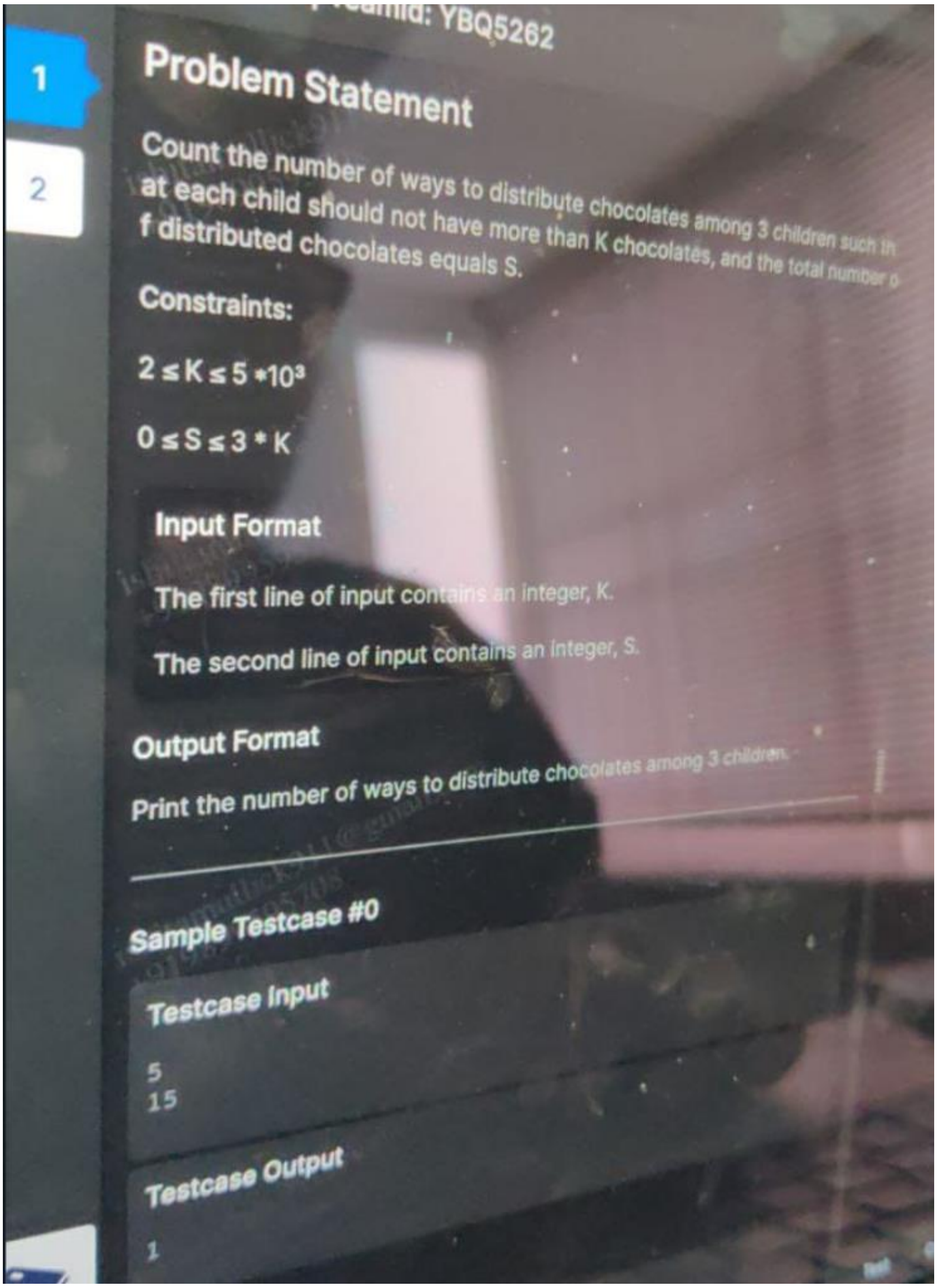
Random

1)



amid: YBQ5262

Problem Statement

Count the number of ways to distribute chocolates among 3 children such that each child should not have more than K chocolates, and the total number of distributed chocolates equals S.

Constraints:

$2 \le K \le 5 * 10^3$

$0 \le S \le 3 * K$

Input Format

The first line of input contains an integer, K.

The second line of input contains an integer, S.

Output Format

Print the number of ways to distribute chocolates among 3 children.

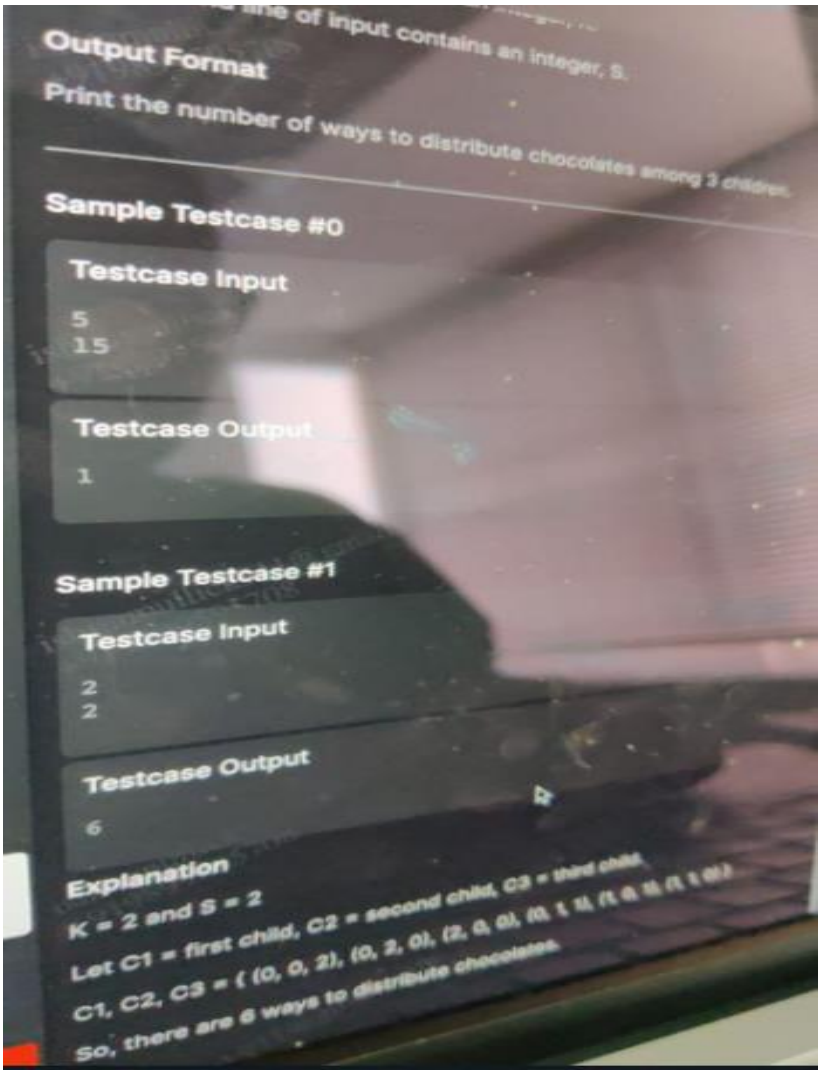
Sample Testcase #0

Testcase Input

```
5
15
```

Testcase Output

```
1
```



line of input contains an integer, S.

Output Format

Print the number of ways to distribute chocolates among 3 children.

Sample Testcase #0

Testcase Input

```
5
15
```

Testcase Output

```
1
```

Sample Testcase #1

Testcase Input

```
2
2
```

Testcase Output

```
6
```

Explanation

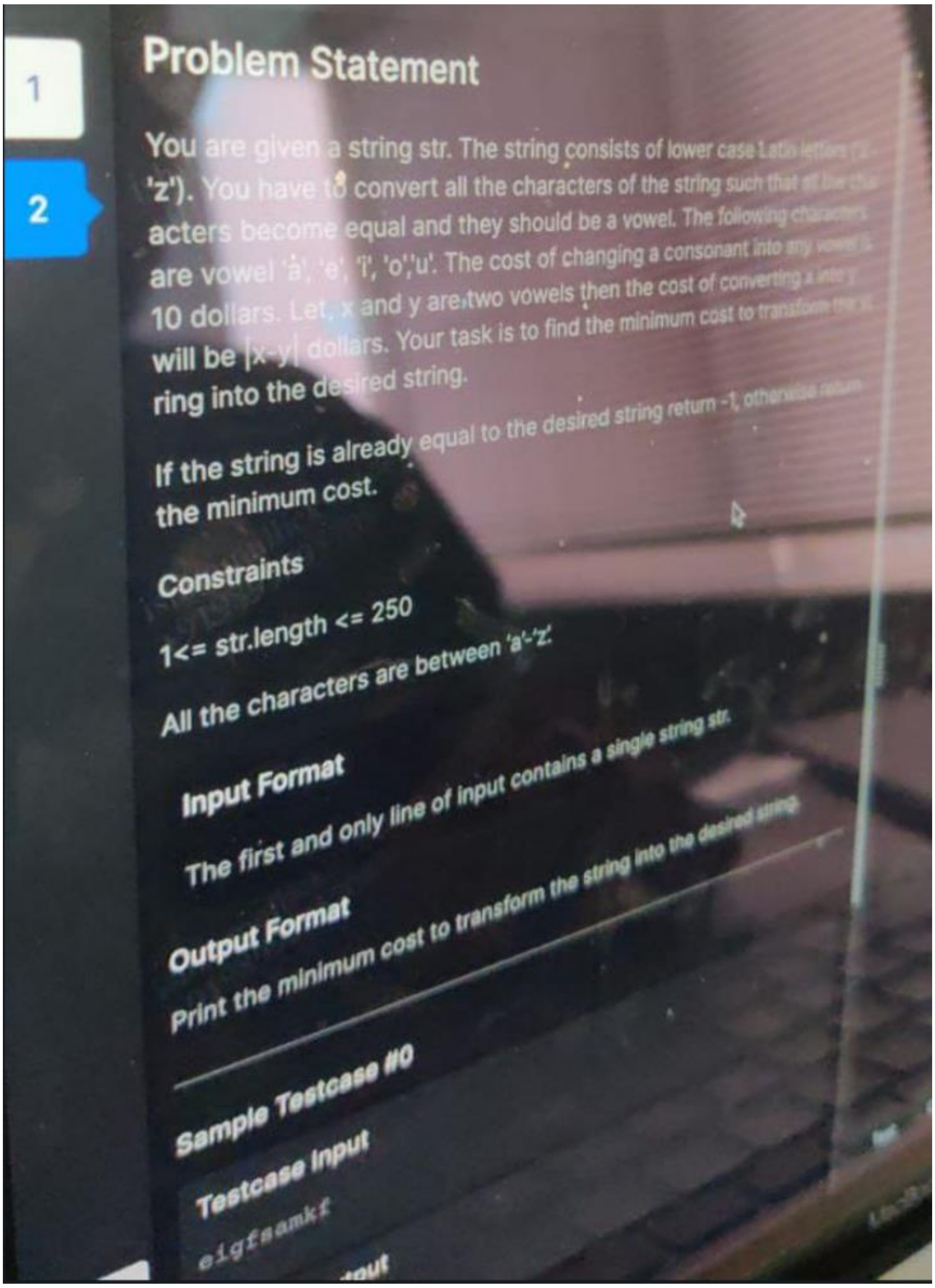
K = 2 and S = 2

Let C1 = first child, C2 = second child, C3 = third child.

C1, C2, C3 = { (0, 0, 2), (0, 2, 0), (2, 0, 0), (0, 1, 1), (1, 0, 1), (1, 1, 0) }

So, there are 6 ways to distribute chocolates.

2)



Problem Statement

You are given a string str. The string consists of lower case Latin letters ('a'- 'z'). You have to convert all the characters of the string such that all the characters become equal and they should be a vowel. The following characters are vowel 'a', 'e', 'i', 'o','u'. The cost of changing a consonant into any vowel is 10 dollars. Let, x and y are two vowels then the cost of converting x into y will be |x-y| dollars. Your task is to find the minimum cost to transform the string into the desired string.

If the string is already equal to the desired string return -1, otherwise return the minimum cost.

Constraints

1<= str.length <= 250

All the characters are between 'a'-'z'.

Input Format

The first and only line of input contains a single string str.

Output Format

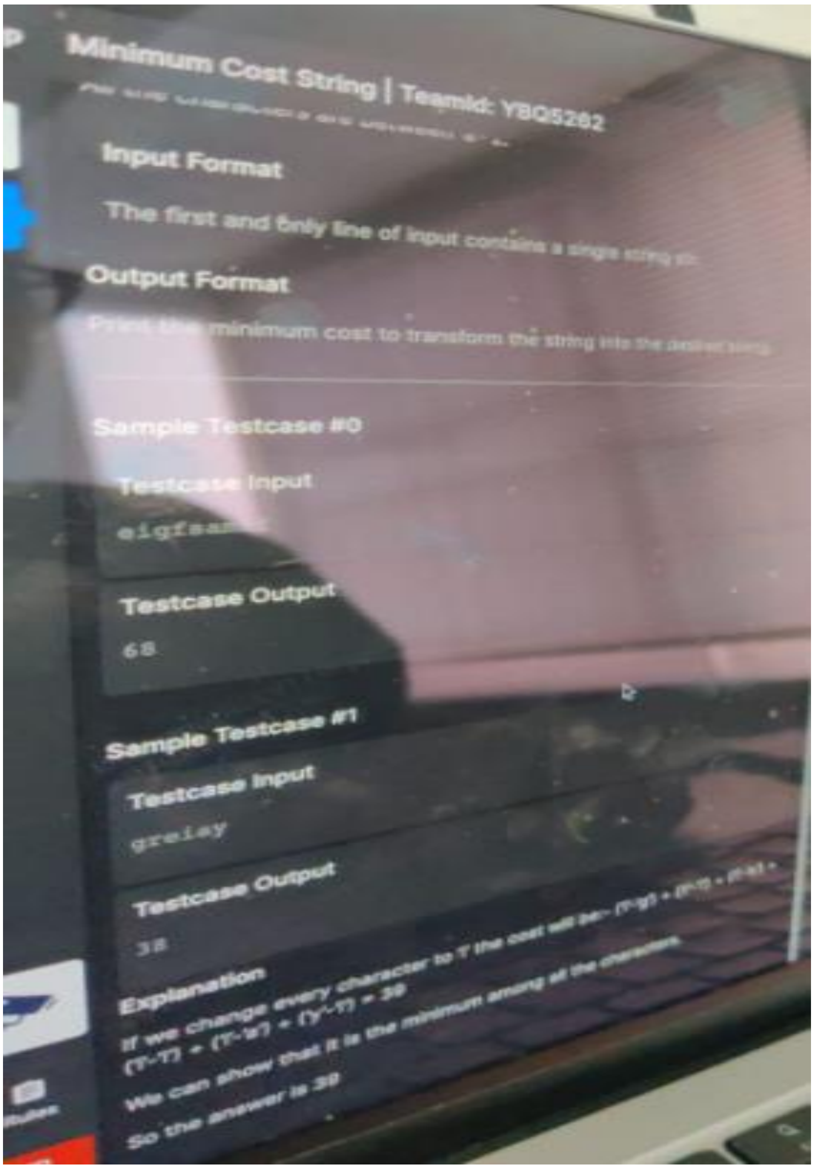
Print the minimum cost to transform the string into the desired string

Sample Testcase #0

Testcase Input

```
eigfsamkf
```

utput



Minimum Cost String | Teamid: YBQ5282

[illegible]

Input Format

The first and only line of input contains a single string str.

Output Format

Print the minimum cost to transform the string into the desired string.

Sample Testcase #0

Testcase Input

```
eigfaa
```

Testcase Output

```
68
```

Sample Testcase #1

Testcase Input

```
greiay
```

Testcase Output

```
38
```

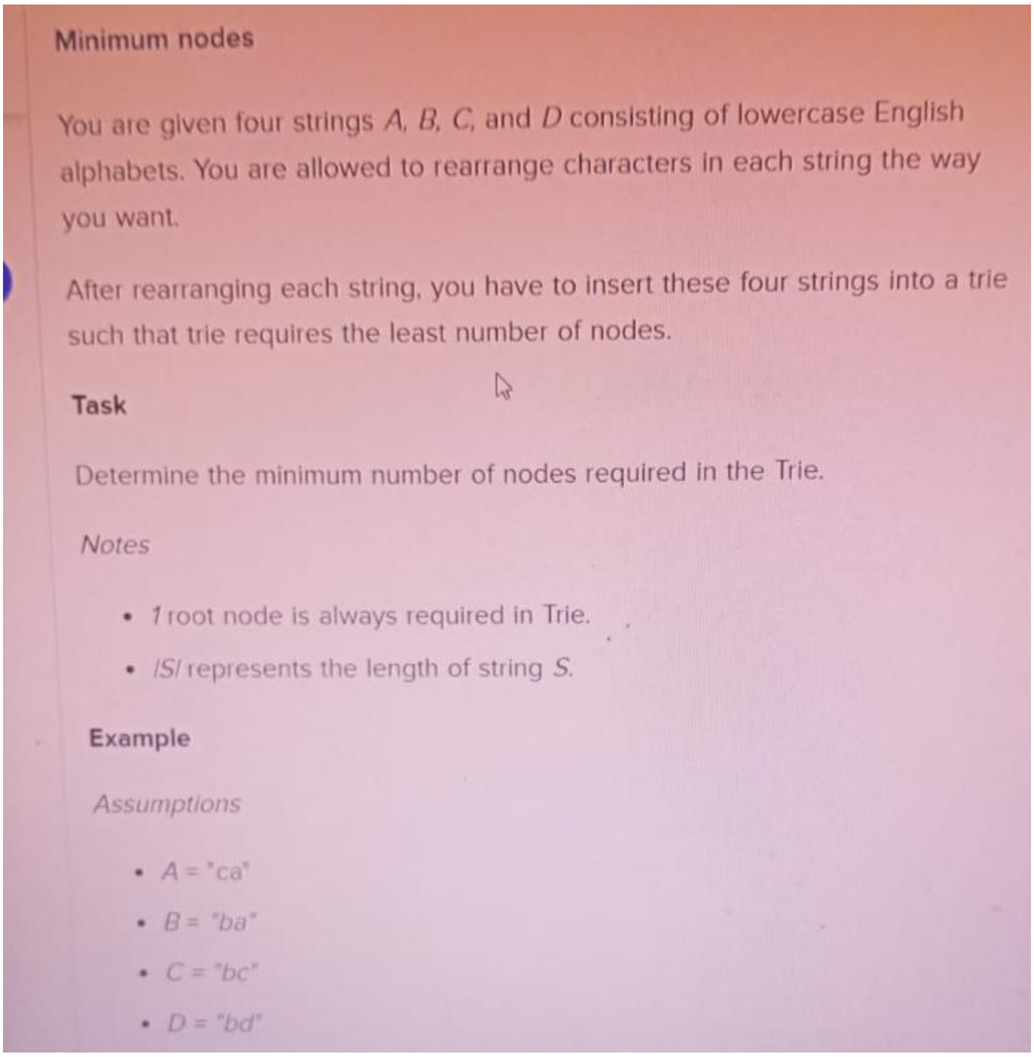
Explanation

If we change every character to 'i' the cost will be:- ('i'-'g') + ('r'-'i') + ('i'-'e') + [illegible] + ('i'-'i') + ('i'-'a') + ('y'-'i') = 39

We can show that it is the minimum among all the characters.

So the answer is 39

3)



Minimum nodes

You are given four strings *A*, *B*, *C*, and *D* consisting of lowercase English alphabets. You are allowed to rearrange characters in each string the way you want.

After rearranging each string, you have to insert these four strings into a trie such that trie requires the least number of nodes.

Task

Determine the minimum number of nodes required in the Trie.

Notes

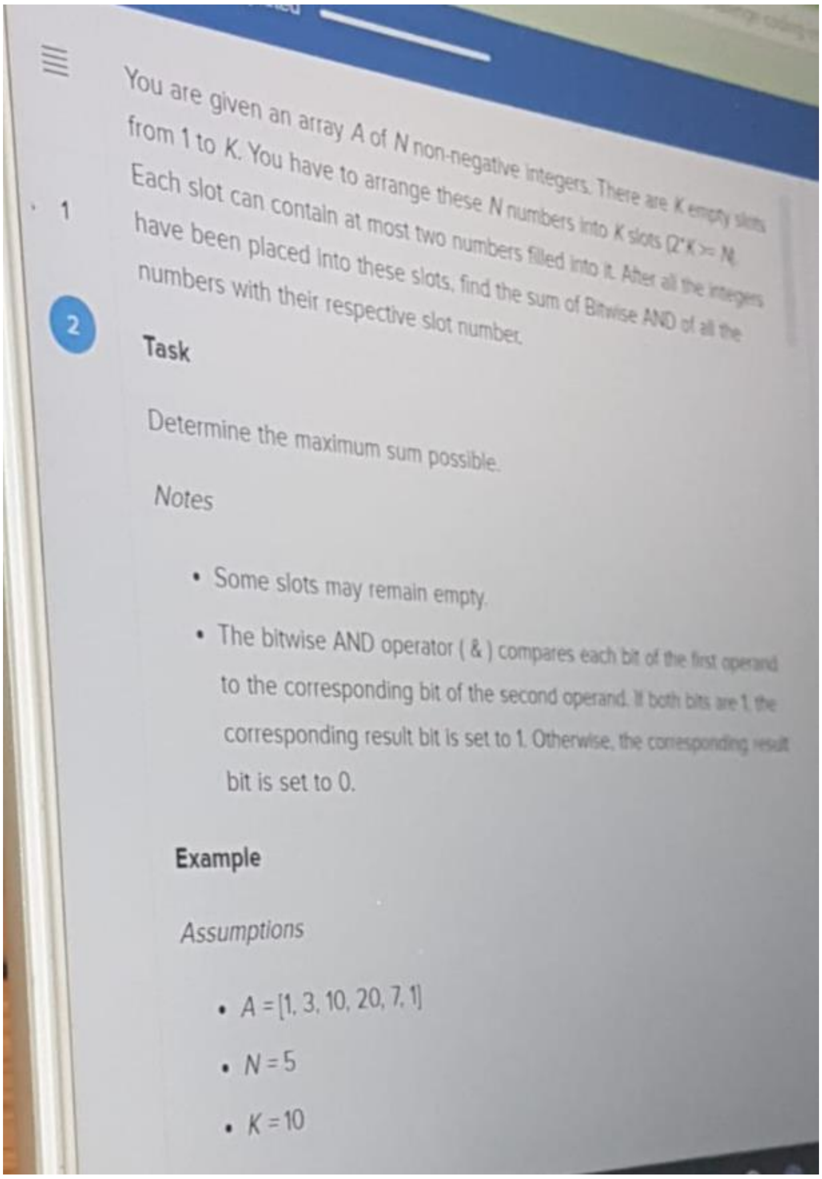
- *1* root node is always required in Trie.
- *|S|* represents the length of string *S*.

Example

Assumptions

- *A* = "ca"
- *B* = "ba"
- *C* = "bc"
- *D* = "bd"

4)



You are given an array *A* of *N* non-negative integers. There are *K* empty slots from 1 to *K*. You have to arrange these *N* numbers into *K* slots (2*K >= N). Each slot can contain at most two numbers filled into it. After all the integers have been placed into these slots, find the sum of Bitwise AND of all the numbers with their respective slot number.

Task

Determine the maximum sum possible.

Notes

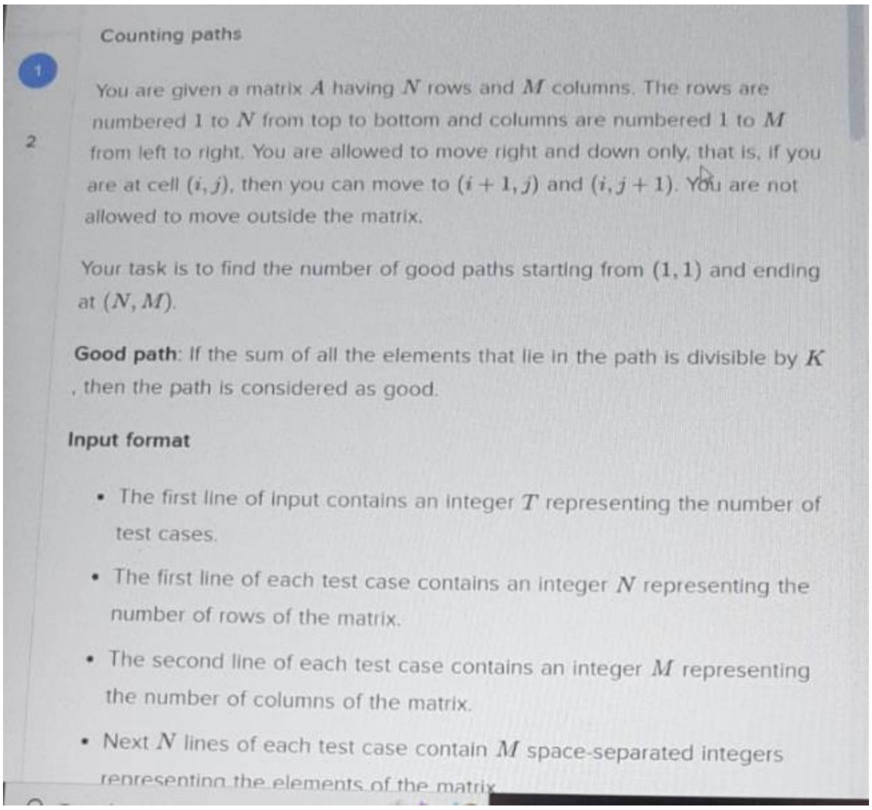
- Some slots may remain empty.
- The bitwise AND operator (&) compares each bit of the first operand to the corresponding bit of the second operand. If both bits are 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.

Example

Assumptions

- *A* = [1, 3, 10, 20, 7, 1]
- *N* = 5
- *K* = 10

5)



Counting paths

You are given a matrix A having N rows and M columns. The rows are numbered 1 to N from top to bottom and columns are numbered 1 to M from left to right. You are allowed to move right and down only, that is, if you are at cell (i, j), then you can move to $(i+1, j)$ and $(i, j+1)$. You are not allowed to move outside the matrix.

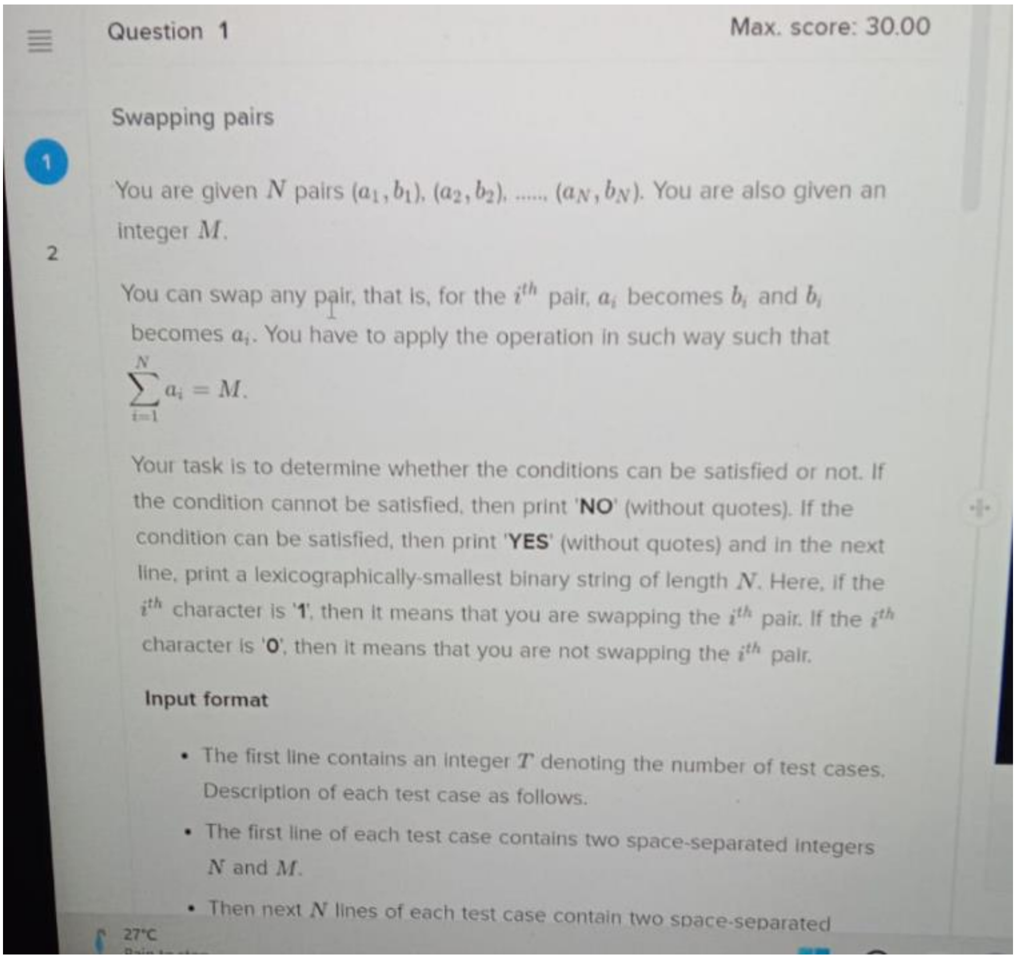
Your task is to find the number of good paths starting from $(1, 1)$ and ending at (N, M).

Good path: If the sum of all the elements that lie in the path is divisible by K , then the path is considered as good.

Input format

- The first line of input contains an integer T representing the number of test cases.
- The first line of each test case contains an integer N representing the number of rows of the matrix.
- The second line of each test case contains an integer M representing the number of columns of the matrix.
- Next N lines of each test case contain M space-separated integers representing the elements of the matrix

6)



Question 1

Max. score: 30.00

Swapping pairs

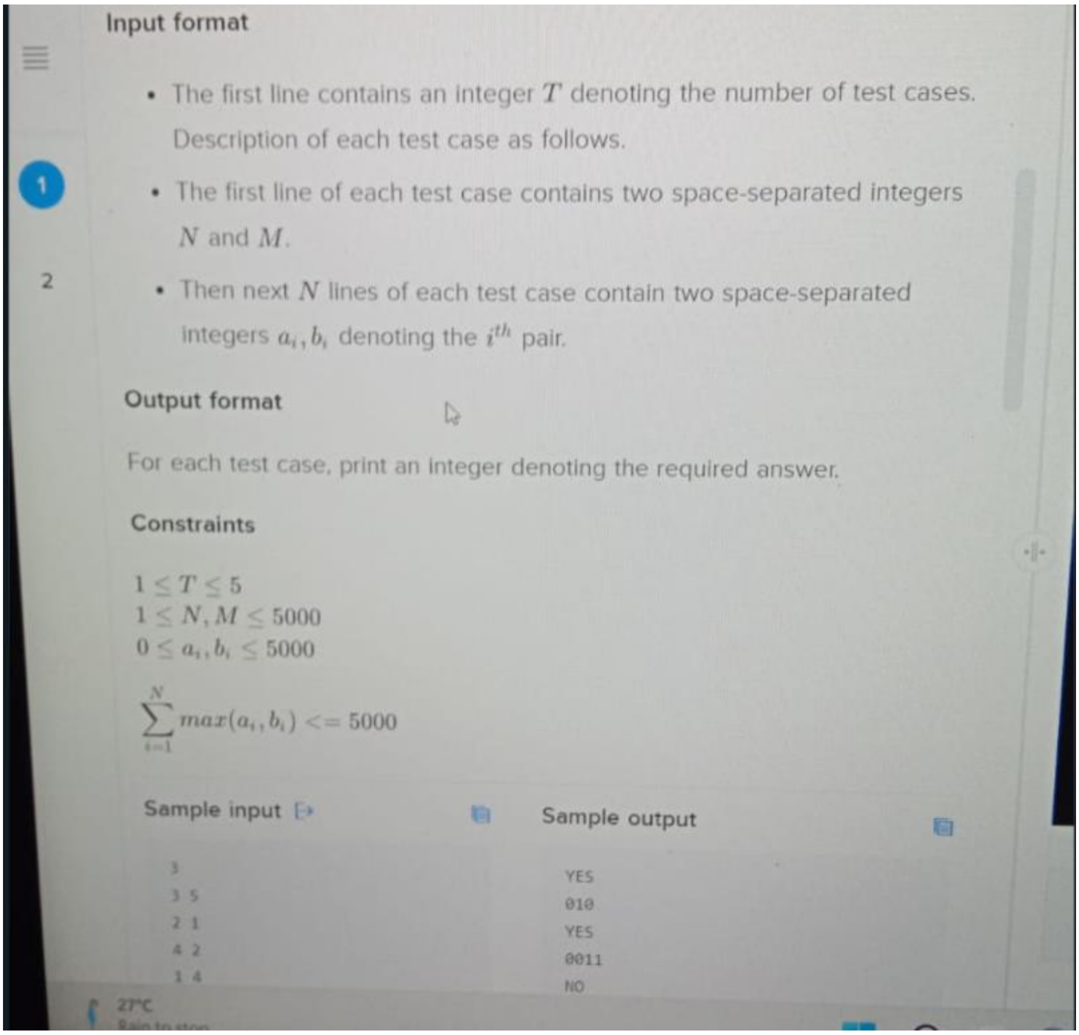
You are given N pairs (a_1, b_1), (a_2, b_2), ... (a_N, b_N). You are also given an integer M.

You can swap any pair, that is, for the i^{th} pair, a_i becomes b_i and b_i becomes a_i. You have to apply the operation in such way such that $\sum_{i=1}^{N} a_i = M$.

Your task is to determine whether the conditions can be satisfied or not. If the condition cannot be satisfied, then print '**NO**' (without quotes). If the condition can be satisfied, then print '**YES**' (without quotes) and in the next line, print a lexicographically-smallest binary string of length N. Here, if the i^{th} character is '**1**', then it means that you are swapping the i^{th} pair. If the i^{th} character is '**0**', then it means that you are not swapping the i^{th} pair.

Input format

- The first line contains an integer T denoting the number of test cases. Description of each test case as follows.
- The first line of each test case contains two space-separated integers N and M.
- Then next N lines of each test case contain two space-separated



Input format

- The first line contains an integer T denoting the number of test cases. Description of each test case as follows.
- The first line of each test case contains two space-separated integers N and M.
- Then next N lines of each test case contain two space-separated integers a_i, b_i denoting the i^{th} pair.

Output format

For each test case, print an integer denoting the required answer.

Constraints

$1 \leq T \leq 5$
$1 \leq N, M \leq 5000$
$0 \leq a_i, b_i \leq 5000$

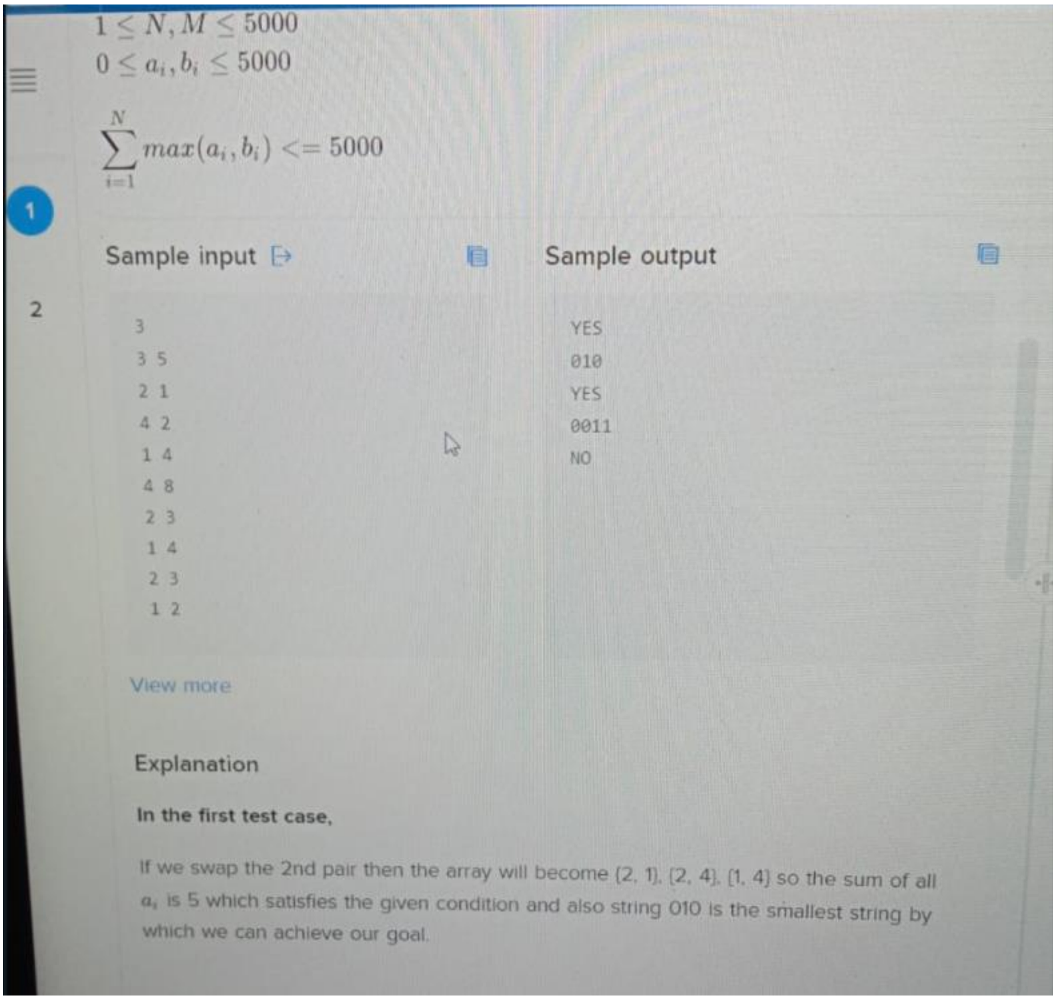
$$\sum_{i=1}^{N} max(a_i, b_i) <= 5000$$

Sample input

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

Sample output

```
YES
010
YES
0011
NO
```



$1 \leq N, M \leq 5000$

$0 \leq a_i, b_i \leq 5000$

$$\sum_{i=1}^{N} max(a_i, b_i) <= 5000$$

Sample input

```
3
3 5
2 1
4 2
1 4
4 8
2 3
1 4
2 3
1 2
```

View more

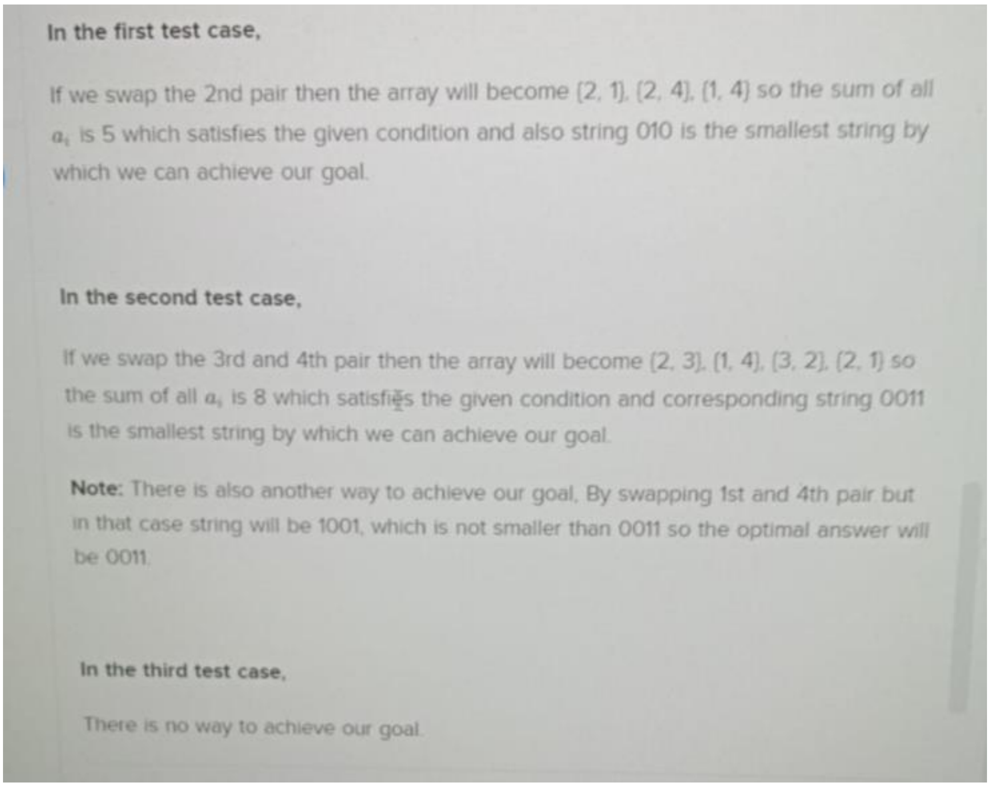
Sample output

```
YES
010
YES
0011
NO
```

Explanation

In the first test case,

If we swap the 2nd pair then the array will become (2, 1), (2, 4), (1, 4) so the sum of all a_i is 5 which satisfies the given condition and also string 010 is the smallest string by which we can achieve our goal.



In the first test case,

If we swap the 2nd pair then the array will become (2, 1), (2, 4), (1, 4) so the sum of all a_i is 5 which satisfies the given condition and also string 010 is the smallest string by which we can achieve our goal.

In the second test case,

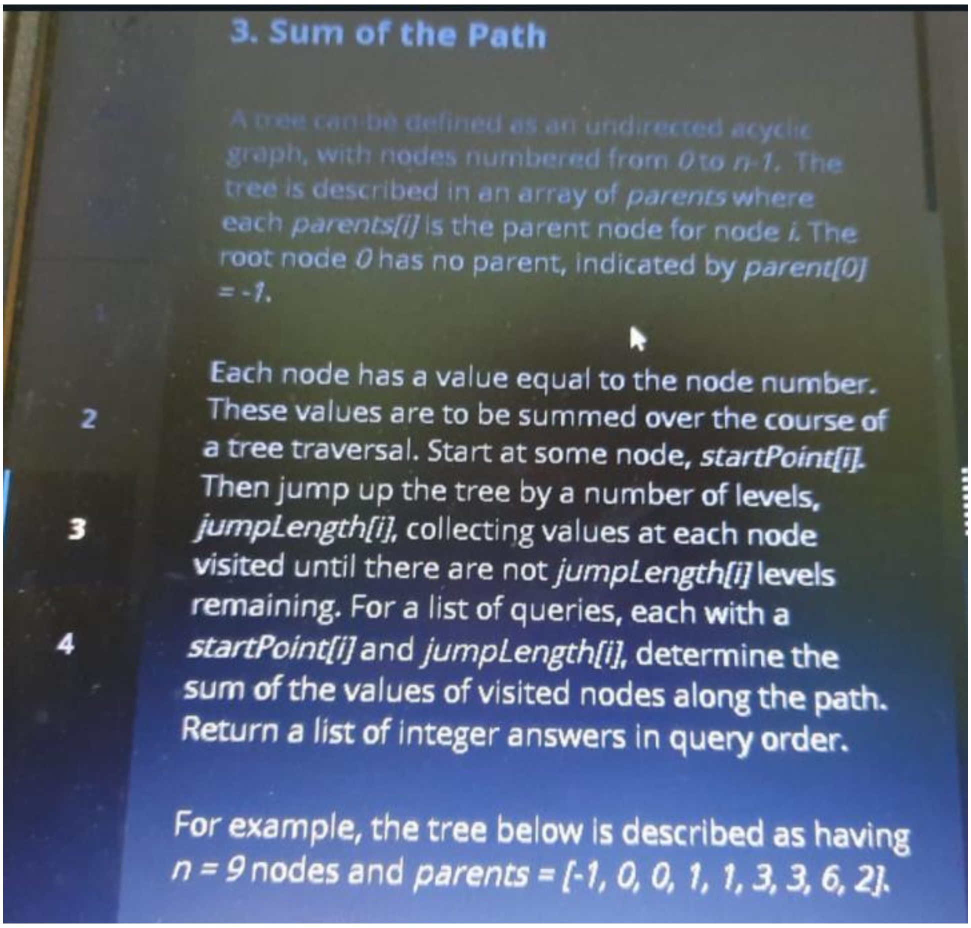
If we swap the 3rd and 4th pair then the array will become (2, 3), (1, 4), (3, 2), (2, 1) so the sum of all a_i is 8 which satisfies the given condition and corresponding string 0011 is the smallest string by which we can achieve our goal.

Note: There is also another way to achieve our goal, By swapping 1st and 4th pair but in that case string will be 1001, which is not smaller than 0011 so the optimal answer will be 0011.

In the third test case,

There is no way to achieve our goal.

7)

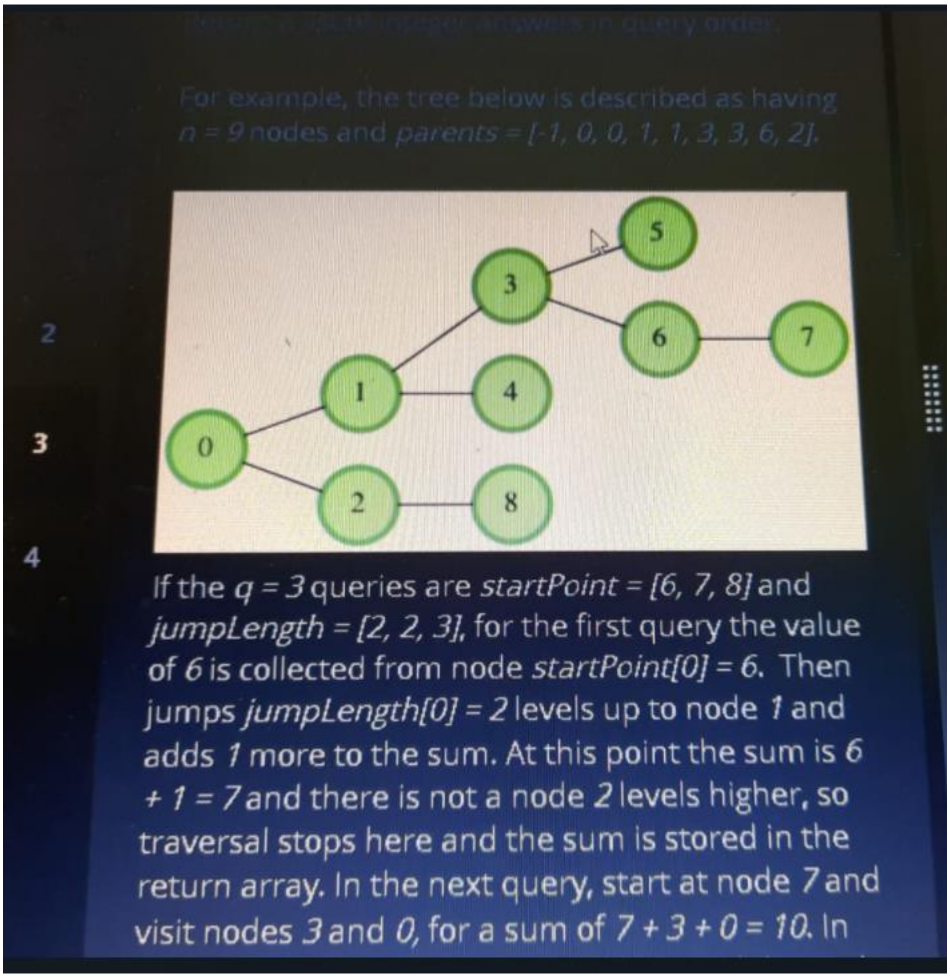


3. Sum of the Path

A tree can be defined as an undirected acyclic graph, with nodes numbered from *0* to *n-1*. The tree is described in an array of *parents* where each *parents[i]* is the parent node for node *i*. The root node *0* has no parent, indicated by *parent[0] = -1*.

Each node has a value equal to the node number. These values are to be summed over the course of a tree traversal. Start at some node, *startPoint[i]*. Then jump up the tree by a number of levels, *jumpLength[i]*, collecting values at each node visited until there are not *jumpLength[i]* levels remaining. For a list of queries, each with a *startPoint[i]* and *jumpLength[i]*, determine the sum of the values of visited nodes along the path. Return a list of integer answers in query order.

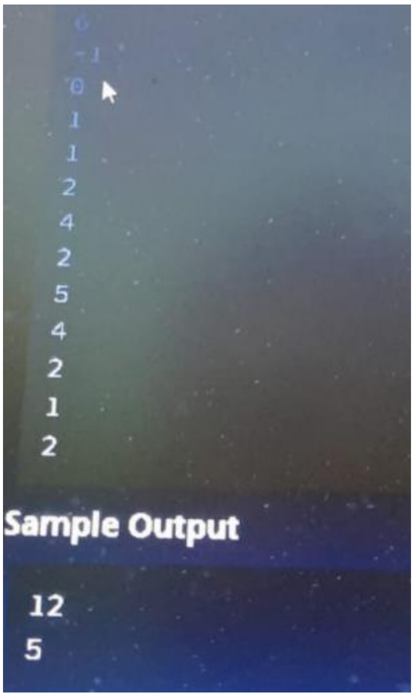
For example, the tree below is described as having *n = 9* nodes and *parents = [-1, 0, 0, 1, 1, 3, 3, 6, 2]*.



Return an array of integer answers in query order.

For example, the tree below is described as having $n = 9$ nodes and $parents = [-1, 0, 0, 1, 1, 3, 3, 6, 2]$.

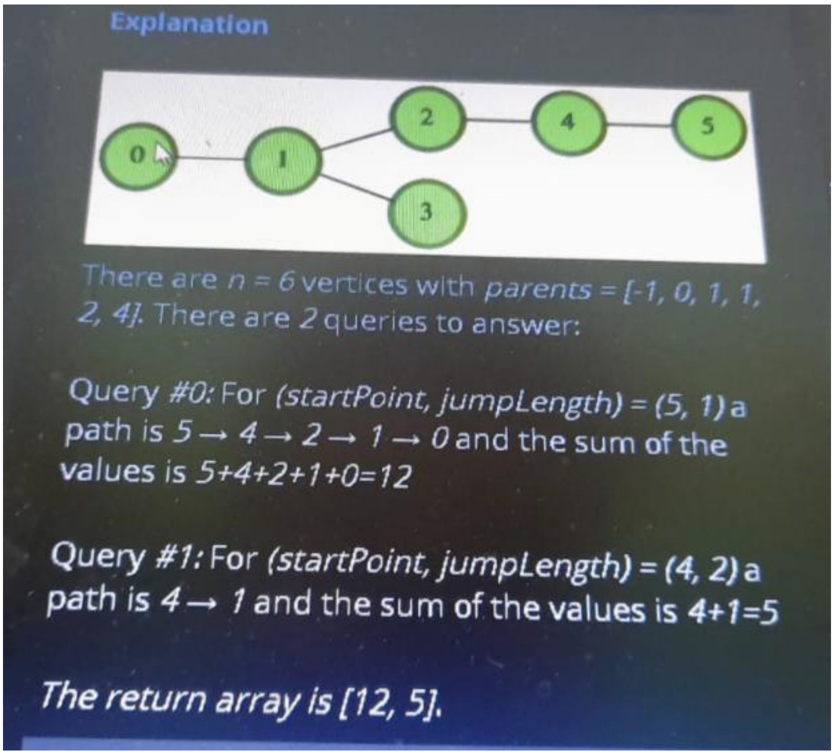
If the $q = 3$ queries are $startPoint = [6, 7, 8]$ and $jumpLength = [2, 2, 3]$, for the first query the value of 6 is collected from node $startPoint[0] = 6$. Then jumps $jumpLength[0] = 2$ levels up to node 1 and adds 1 more to the sum. At this point the sum is $6 + 1 = 7$ and there is not a node 2 levels higher, so traversal stops here and the sum is stored in the return array. In the next query, start at node 7 and visit nodes 3 and 0, for a sum of $7 + 3 + 0 = 10$. In



```
6
-1
0
1
1
2
4
2
5
4
2
1
2
```

Sample Output

```
12
5
```



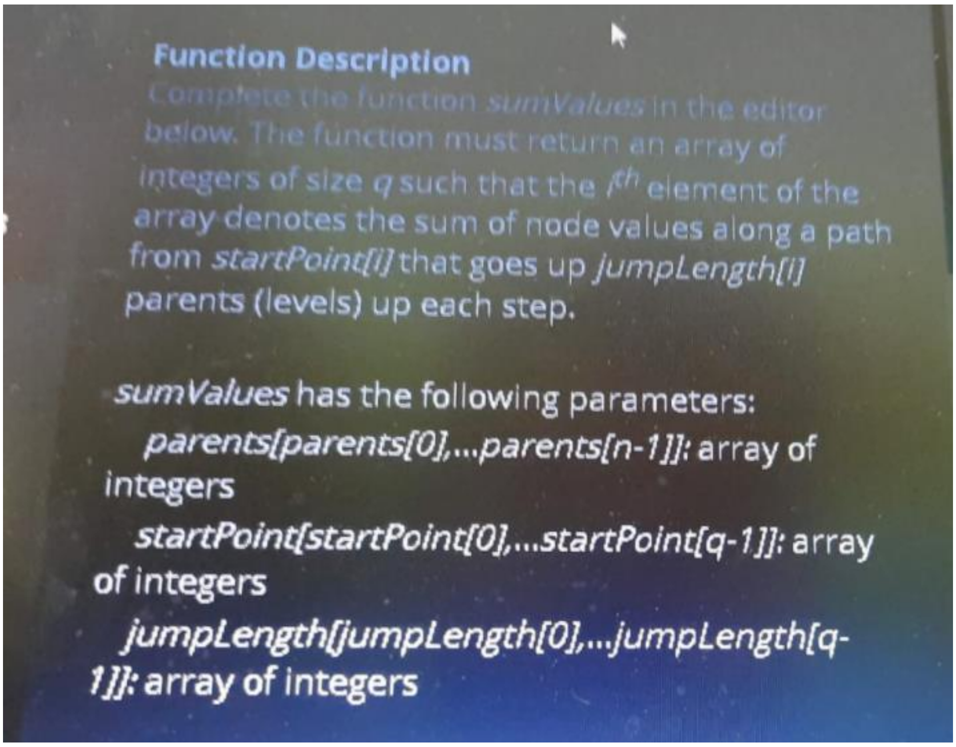
Explanation

There are *n = 6* vertices with *parents = [-1, 0, 1, 1, 2, 4]*. There are *2* queries to answer:

Query *#0:* For *(startPoint, jumpLength) = (5, 1)* a path is *5 → 4 → 2 → 1 → 0* and the sum of the values is *5+4+2+1+0=12*

Query *#1:* For *(startPoint, jumpLength) = (4, 2)* a path is *4 → 1* and the sum of the values is 4+1=5

The return array is [12, 5].



Function Description

Complete the function *sumValues* in the editor below. The function must return an array of integers of size *q* such that the *i*th element of the array denotes the sum of node values along a path from *startPoint[i]* that goes up *jumpLength[i]* parents (levels) up each step.

sumValues has the following parameters:

parents[parents[0],...parents[n-1]]: array of integers

startPoint[startPoint[0],...startPoint[q-1]]: array of integers

jumpLength[jumpLength[0],...jumpLength[q-1]]: array of integers