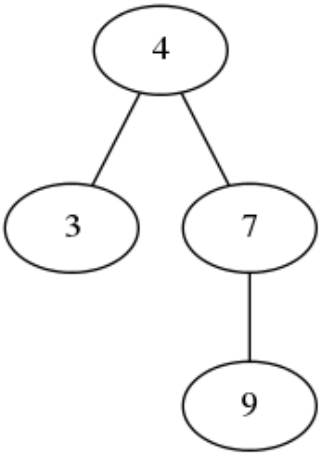
The Stables of Genghis Khan

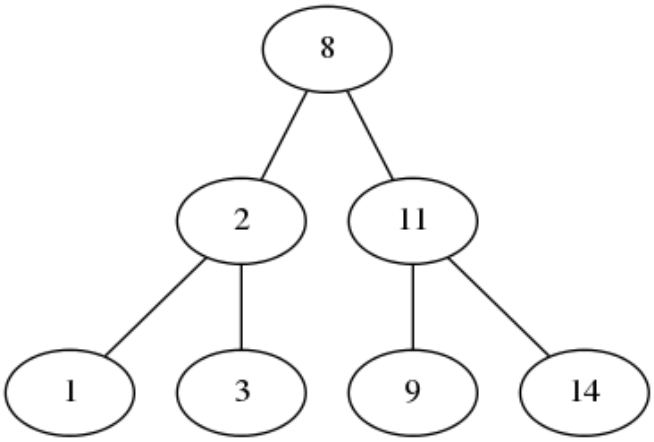
Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Bruce and Carl are having an excellent adventure... through time! They've been trying to teach people throughout history how to use algorithms, but no one has been listening. The Mongols, however, are the exception, and Genghis Khan has given them a chance to prove their worth. He has a number of horses, some of which are currently in his stables. He wants to find out which horses are in his stables and which aren't. Bruce and Carl are going to use binary search trees to help him do this quickly.

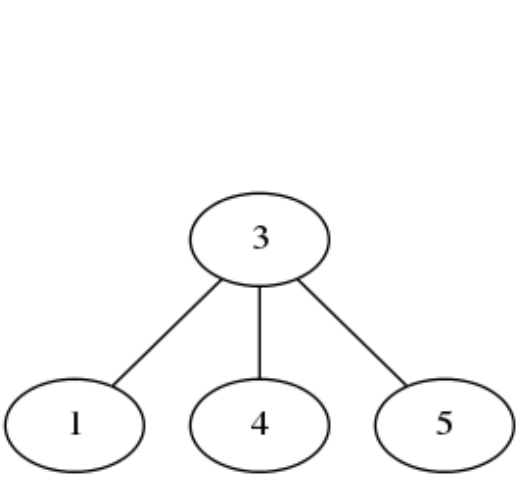
A *binary key tree* consists of a vertex (called the *root*) labelled with a number (called its *key*) and optionally up to two "subtrees" linked to it. These are called the left subtree and the right subtree and are also binary key trees.

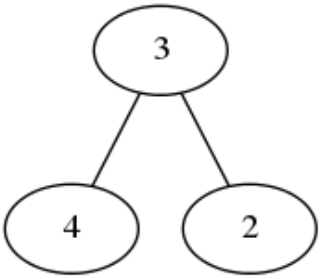
A *binary search tree* is a binary key tree where the left and right subtrees (if they exist) are also binary search trees. Additionally, the key of the root is greater than every key in the left subtree and less than every key in the right subtree. For example, the following are binary search trees:

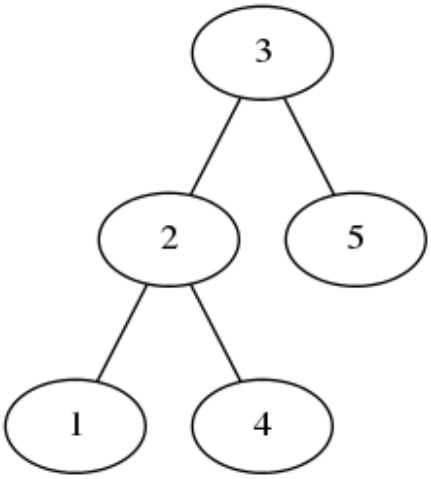




But the following aren't:







Genghis Khan has N horses, numbered 1 to N. K of them are currently in the stables, numbered $X_1, X_2, \ldots, X_K$. Bruce and Carl want to make a binary search tree with vertices numbered with keys $X_1, X_2, \ldots, X_K$ in which they will quickly be able to look up if a horse is in the stable or not.

They use the following algorithm to search the tree for a key A:

1. Start at the root.

2. Compare the value of A to the key X of the current node. If $A = X$ then they have found A and stop, if $A < X$ they repeat step 2 on the left subtree. If $A > X$ they repeat step 2 on the right subtree.

3. If at any stage the subtree they need to perform the algorithm on is not in the tree, then A is not in the tree and they stop.

The cost of looking for A is the number of times step 2 is performed during the execution of the algorithm. Help Bruce and Carl design a binary search tree which minimises the total cost of searching for all of the keys $1, 2, \ldots, N$.

Input

The first line of input contains the integer N ($1 \leq N \leq 10\,000\,000$). The second line contains the integer K ($1 \leq K \leq 300$). The next K lines contain the integers $X_1, X_2, \ldots, X_K$, one per line.

Output

Your program should output a single integer, the minimum total cost of a binary search tree satisfying the requirements.

Scoring

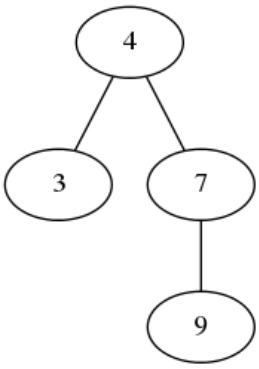
In 30% of the test cases, $N, K \leq 10$.

Example

standard input	standard output
10 4 9 3 7 4	22

Note

Suppose $N = 10$, $K = 4$ and the horses in the stable have numbers $3, 4, 7, 9$. Then the optimal binary search tree has total cost 22, and is the following:



The costs of searching for each number are the following:

Number	Cost	Number	Cost
1	2	6	2
2	2	7	2
3	2	8	3
4	1	9	3
5	2	10	3