The Coin Change Problem

HackerRank

Given an amount and the denominations of coins available, determine how many ways change can be made for amount. There is a limitless supply of each coin type.

Example
$n = 3$
$c = [8, 3, 1, 2]$

There are 3 ways to make change for $n = 3$: $\{1, 1, 1\}$, $\{1, 2\}$, and $\{3\}$.

Function Description

Complete the *getWays* function in the editor below.

getWays has the following parameter(s):

- *int n:* the amount to make change for
- *int c[m]:* the available coin denominations

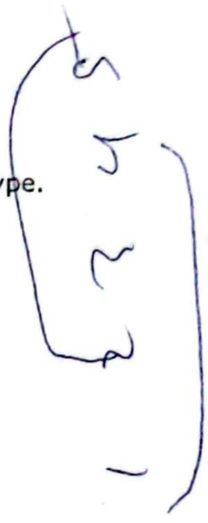
3 3 4 7

Returns

- *int:* the number of ways to make change

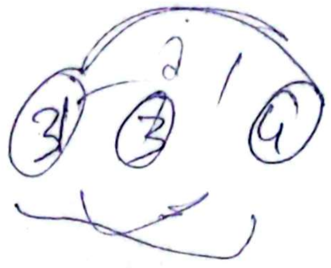
Input Format

The first line contains two space-separated integers n and m, where:
n is the amount to change
m is the number of coin types
The second line contains m space-separated integers that describe the values of each coin type.



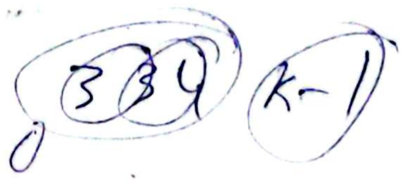
Constraints

- $1 \leq c[i] \leq 50$
- $1 \leq n \leq 250$
- $1 \leq m \leq 50$
- Each $c[i]$ is guaranteed to be distinct.



Hints

Solve overlapping subproblems using Dynamic Programming (DP):
You can solve this problem recursively but will not pass all the test cases without optimizing to eliminate the overlapping subproblems. Think of a way to store and reference previously computed solutions to avoid solving the same subproblem multiple times. * Consider the degenerate cases:
- How many ways can you make change for 0 cents? - How many ways can you make change for > 0



cents if you have no coins? • If you're having trouble defining your solutions store, then think about it in terms of the base case $(n = 0)$. • The answer may be larger than a 32-bit integer.

Sample Input 0

```
4 3
1 2 3
```

Sample Output 0

```
4
```

Explanation 0

There are four ways to make change for $n = 4$ using coins with values given by $C = [1, 2, 3]$:

1. $\{1, 1, 1, 1\}$
2. $\{1, 1, 2\}$
3. $\{2, 2\}$
4. $\{1, 3\}$

Sample Input 1

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

Sample Output 1

```
5
```

Explanation 1

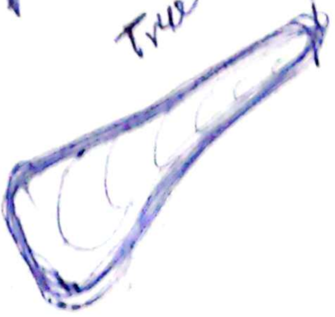
There are five ways to make change for $n = 10$ units using coins [with values given by $C = [2, 5, 3, 6]$:

1 ←

1. $\{2, 2, 2, 2, 2\}$
2. $\{2, 2, 3, 3\}$
3. $\{2, 2, 6\}$
4. $\{2, 3, 5\}$
5. $\{5, 5\}$

True

True



| | | $

[1] :

Mars Exploration

HackerRank

A space explorer's ship crashed on Mars! They send a series of `SOS` messages to Earth for help.

Letters in some of the `SOS` messages are altered by cosmic radiation during transmission. Given the signal received by Earth as a string, s, determine how many letters of the `SOS` message have been changed by radiation.

Example

$s = \text{'SOSTOT'}$

The original message was `SOSSOS`. Two of the message's characters were changed in transit.

Function Description

Complete the *marsExploration* function in the editor below.

marsExploration has the following parameter(s):

- *string s:* the string as received on Earth

| | | | |

Returns

- *int:* the number of letters changed during transmission

Input Format

There is one line of input: a single string, s.

Constraints

- $1 \leq \text{length of } s \leq 99$
- $\text{length of } s \text{ modulo } 3 = 0$
- s will contain only uppercase English letters, ascii[A-Z].

Sample Input 0

```
SOSSPSSQSSOR
```

Sample Output 0

```
3
```

Explanation 0

s = **SOSSPSSQSSOR**, and signal length $|s| = 12$. They sent 4 `SOS` messages (i.e.: $12/3 = 4$).

```
Expected signal: SOSSOSSOSSOS
Recieved signal: SOSSPSSQSSOR
Difference:          X  X   X
```

Sample Input 1

```
SOSSOT
```

Sample Output 1

```
1
```

Explanation 1

s = **SOSSOT**, and signal length $|s| = 6$. They sent 2 `SOS` messages (i.e.: $6/3 = 2$).

```
Expected Signal: SOSSOS
Received Signal: SOSSOT
Difference:           X
```

Sample Input 2

```
SOSSOSSOS
```

Sample Output 2

```
0
```

Explanation 2

Since no character is altered, return 0.

Angry Children 2

HackerRank

Bill Gates is on one of his philanthropic journeys to a village in Utopia. He has brought a box of packets of candies and would like to distribute one packet to each of the children. Each of the packets contains a number of candies. He wants to minimize the cumulative difference in the number of candies in the packets he hands out. This is called the *unfairness sum*. Determine the minimum unfairness sum achievable.

2 3 0

For example, he brings $n = 7$ packets where the number of candies is $packets = [3, 3, 4, 5, 7, 9, 10]$. There are $k = 3$ children. The minimum difference between all packets can be had with $3, 3, 4$ from indices $0, 1$ and 2. We must get the difference in the following pairs: $\{(0,1), (0,2), (1,2)\}$. We calculate the *unfairness sum* as:

```
packets candies
0       3               indices          difference       result
1       3               (0,1),(0,2)      |3-3| + |3-4|    1
2       4               (1,2)            |3-4|            1

Total = 2
```

3 - 3 - 4

-2

Function Description

Complete the *angryChildren* function in the editor below. It should return an integer that represents the minimum unfairness sum achievable.

angryChildren has the following parameter(s):

- *k*: an integer that represents the number of children
- *packets*: an array of integers that represent the number of candies in each packet

3

~~6~~

Input Format

The first line contains an integer n.
The second line contains an integer k.
Each of the next n lines contains an integer $packets[i]$.

Constraints

$2 \le n \le 10^5$
$2 \le k \le n$
$0 \le packets[i] \le 10^9$

X 0 3 3 4 5 7 9 10

0

6 0 1 13

Output Format

A single integer representing the minimum achievable unfairness sum.

Sample Input 0

8

4 2

```
7
3
```

/2

```
10
100
300
200
1000
20
30
```

10 20 30

Sample Output 0

```
40
```

Explanation 0

Bill Gates will choose packets having 10, 20 and 30 candies. The unfairness sum is
$|10 - 20| + |20 - 30| + |10 - 30| = 40$.

Sample Input 1

```
10
4
1
2
3
4
10
20
30
40
100
200
```

$\underline{10\ 20\ 30}$ 100
200 300
1000

Sample Output 1

```
10
```

Explanation 1

Bill Gates will choose 4 packets having 1,2,3 and 4 candies. The unfairness sum i
$|1 - 2| + |1 - 3| + |1 - 4| + |2 - 3| + |2 - 4| + |3 - 4| = 10$.

~~1~~ 2 3 4 $\bigcirc\!\!\!\!5$

Crossword Puzzle

HackerRank

A 10×10 Crossword grid is provided to you, along with a set of words (or names of places) which need to be filled into the grid. Cells are marked either `+` or `-`. Cells marked with a `-` are to be filled with the word list.

The following shows an example crossword from the input $crossword$ grid and the list of words to fit, $words = [POLAND, LHASA, SPAIN, INDIA]$:

```
Input                   Output

++++++++++              ++++++++++
+------+++              +POLAND+++
+++-++++++              +++H++++++
+++-++++++              +++A++++++
+++-----++              +++SPAIN++
+++-++-+++              +++A++N+++
++++++-+++              ++++++D+++
++++++-+++              ++++++I+++
++++++-+++              ++++++A+++
++++++++++              ++++++++++
POLAND;LHASA;SPAIN;INDIA
```

Function Description

Complete the *crosswordPuzzle* function in the editor below. It should return an array of strings, each representing a row of the finished puzzle.

crosswordPuzzle has the following parameter(s):

- *crossword*: an array of 10 strings of length 10 representing the empty grid
- *words:* a string consisting of semicolon delimited strings to fit into $crossword$

Input Format

Each of the first 10 lines represents $crossword[i]$, each of which has 10 characters, $crossword[i][j]$.

The last line contains a string consisting of semicolon delimited $words[i]$ to fit.

Constraints

$1 \le |words| \le 10$
$crossword[i][j] \in \{+, -\}$
$words[i][j] \in ascii[A - Z]$

Output Format

Position the words appropriately in the 10×10 grid, then return your array of strings for printing.

Sample Input 0

/3

```
+-++++++++
+-++++++++
+-++++++++
+-----++++
+-+++-++++
+-+++-++++
+++++-++++
++------++
+++++-++++
+++++-++++
LONDON;DELHI;ICELAND;ANKARA
```

Sample Output 0

```
+L++++++++
+O++++++++
+N++++++++
+DELHI++++
+O+++C++++
+N+++E++++
+++++L++++
++ANKARA++
+++++N++++
+++++D++++
```

Sample Input 1

```
+-++++++++
+-++++++++
+-------++
+-++++++++
+-++++++++
+------+++
+-+++-++++
+++++-++++
+++++-++++
++++++++++
AGRA;NORWAY;ENGLAND;GWALIOR
```

Sample Output 1

```
+E++++++++
+N++++++++
+GWALIOR++
+L++++++++
+A++++++++
+NORWAY+++
+D+++G++++
+++++R++++
+++++A++++
++++++++++
```

Sample Input 2

```
++++++-+++
++------++
++++++-+++
++++++-+++
+++------+
++++++-+-+
++++++-+-+
++++++++-+
++++++++-+
++++++++-+
ICELAND;MEXICO;PANAMA;ALMATY
```

Sample Output 2

```
++++++I+++
++MEXICO++
++++++E+++
++++++L+++
+++PANAMA+
++++++N+L+
++++++D+M+
++++++++A+
++++++++T+
++++++++Y+
```

2 5 3 6

4 3

1 2 3

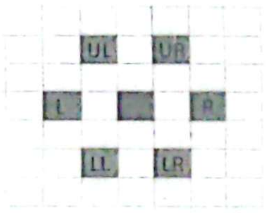
$$\begin{bmatrix} \cancel{0} & 0 & 0 & 0 & 0 \\ \cancel{0} & 0 & 0 & 0 & 0 \\ \cancel{0} & 0 & 0 & 0 & 0 \end{bmatrix}$$

Red Knight's Shortest Path

HackerRank

In ordinary chess, the pieces are only of two colors, black and white. In our version of chess, we are including new pieces with unique movements. One of the most powerful pieces in this version is the *red knight*.

The red knight can move to six different positions based on its current position (UpperLeft, UpperRight, Right, LowerRight, LowerLeft, Left) as shown in the figure below.



The board is a grid of size $n \times n$. Each cell is identified with a pair of coordinates (i, j), where i is the row number and j is the column number, both zero-indexed. Thus, $(0, 0)$ is the upper-left corner and $(n-1, n-1)$ is the bottom-right corner.

Complete the function `printShortestPath`, which takes as input the grid size n, and the coordinates of the starting and ending position (i_{start}, j_{start}) and (i_{end}, j_{end}) respectively, as input. The function does not return anything.

Given the coordinates of the starting position of the red knight and the coordinates of the destination, print the minimum number of moves that the red knight has to make in order to reach the destination and after that, print the order of the moves that must be followed to reach the destination in the shortest way. If the destination cannot be reached, print only the word "Impossible".

Note: There may be multiple shortest paths leading to the destination. Hence, assume that the red knight considers its possible neighbor locations in the following order of priority: *UL, UR, R, LR, LL, L*. In other words, if there are multiple possible options, the red knight prioritizes the first move in this list, as long as the shortest path is still achievable. Check sample input 2 for an illustration.

Input Format

The first line of input contains a single integer n. The second line contains four space-separated integers $i_{start}, j_{start}, i_{end}, j_{end}$. (i_{start}, j_{start}) denotes the coordinates of the starting position and (i_{end}, j_{end}) denotes the coordinates of the final position.

Constraints

- $5 \leq n \leq 200$
- $0 \leq i_{start}, j_{start}, i_{end}, j_{end} < n$
- the starting and the ending positions are different

/3

Output Format

If the destination can be reached, print two lines. In the first line, print a single integer denoting the minimum number of moves that the red knight has to make in order to reach the destination. In the second line, print the space-separated sequence of moves.

If the destination cannot be reached, print a single line containing only the word `Impossible`.

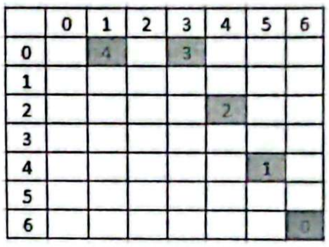
Sample Input 0

```
7
6 6 0 1
```

Sample Output 0

```
4
UL UL UL L
```

Explanation 0



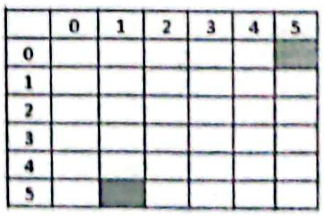
Sample Input 1

```
6
5 1 0 5
```

Sample Output 1

```
Impossible
```

Explanation 1



Sample Input 2

```
7
0 3 4 3
```

Sample Output 2

```
2
LR LL
```

Explanation 2

	0	1	2	3	4	5	6
0				0			
1							
2			1		1		
3							
4				2			
5							
6							

Jesse and Cookies

HackerRank

Jesse loves cookies and wants the sweetness of some cookies to be greater than value k. To do this, two cookies with the least sweetness are repeatedly mixed. This creates a special combined cookie with:

sweetness $= (1\times$ *Least sweet cookie* $+ 2\times$ *2nd least sweet cookie*$)$.

This occurs until all the cookies have a sweetness $\geq k$.

Given the sweetness of a number of cookies, determine the minimum number of operations required. If it is not possible, return -1.

Example
$k = 9$
$A = [2, 7, 3, 6, 4, 6]$

The smallest values are $2, 3$.
Remove them then return $2 + 2 \times 3 = 8$ to the array. Now $A = [8, 7, 6, 4, 6]$.
Remove $4, 6$ and return $4 + 6 \times 2 = 16$ to the array. Now $A = [16, 8, 7, 6]$.
Remove $6, 7$, return $6 + 2 \times 7 = 20$ and $A = [20, 16, 8, 7]$.
Finally, remove $8, 7$ and return $7 + 2 \times 8 = 23$ to A. Now $A = [23, 20, 16]$.
All values are $\geq k = 9$ so the process stops after 4 iterations. Return 4.

Function Description
Complete the *cookies* function in the editor below.

ok.

cookies has the following parameters:

- *int k:* the threshold value
- *int A[n]:* an array of sweetness values

Returns

- *int:* the number of iterations required or -1

Input Format

The first line has two space-separated integers, n and k, the size of $A[]$ and the minimum required sweetness respectively.

The next line contains n space-separated integers, $A[i]$.

Constraints

$1 \leq n \leq 10^6$
$0 \leq k \leq 10^9$
$0 \leq A[i] \leq 10^6$

$\boxed{(-1)}$

Sample Input

```
STDIN               Function
-----               --------
6 7                 A[] size n = 6, k = 7
1 2 3 9 10 12       A = [1, 2, 3, 9, 10, 12]
```

Sample Output

```
2
```

Explanation

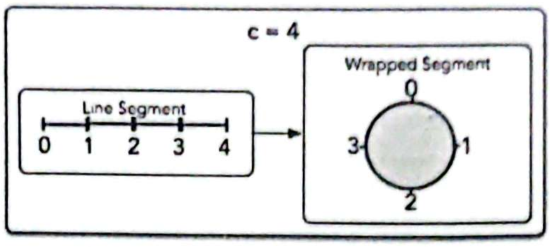
Combine the first two cookies to create a cookie with *sweetness* $= 1 \times 1 + 2 \times 2 = 5$
After this operation, the cookies are $3, 5, 9, 10, 12$.
Then, combine the cookies with sweetness 3 and sweetness 5, to create a cookie with resulting *sweetness* $= 1 \times 3 + 2 \times 5 = 13$
Now, the cookies are $9, 10, 12, 13$.
All the cookies have a sweetness ≥ 7.

Thus, 2 operations are required to increase the sweetness.

Distant Pairs

HackerRank

We take a line segment of length c on a one-dimensional plane and bend it to create a circle with circumference c that's indexed from 0 to $c-1$. For example, if $c = 4$:

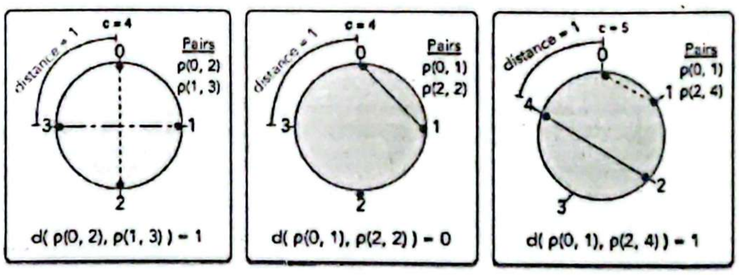


We denote a *pair* of points, a and b, as $\rho(a, b)$. We then plot n pairs of points (meaning a total of $2 \cdot n$ individual points) at various indices along the circle's circumference. We define the distance $d(a, b)$ between points a and b in pair $\rho(a, b)$ as $min(|a-b|, c-|a-b|)$.

Next, let's consider two pairs: $\rho(a_i, b_i)$ and $\rho(a_j, b_j)$. We define distance $d(\rho(a_i, b_i), \rho(a_j, b_j))$ as the *minimum* of the six distances between any two points among points a_i, b_i, a_j, and b_j. In other words:

$$d(\rho_i, \rho_j) = min(d(a_i, a_j), d(a_i, b_i), d(a_i, b_j), d(b_i, b_j), d(a_j, b_i), d(a_j, b_j))$$

For example, consider the following diagram in which the relationship between points in pairs at non-overlapping indices is shown by a connecting line:



Given n pairs of points and the value of c, find and print the *maximum* value of $d(\rho_i, \rho_j)$, where $i \neq j$, among all pairs of points.

Input Format

The first line contains two space-separated integers describing the respective values of n (the number of pairs of points) and c (the circumference of the circle).
Each line i of the n subsequent lines contains two space-separated integers describing the values of a_i and b_i (i.e., the locations of the points in pair i).

Constraints

- $1 \leq c \leq 10^6$
- $2 \leq n \leq 10^5$
- $0 \leq a, b < c$

/3

Output Format

Print a single integer denoting the maximum $d(\rho_i, \rho_j)$, where $i \neq j$.

Sample Input 0

```
5 8
0 4
2 6
1 5
3 7
4 4
```

0 1 2 3

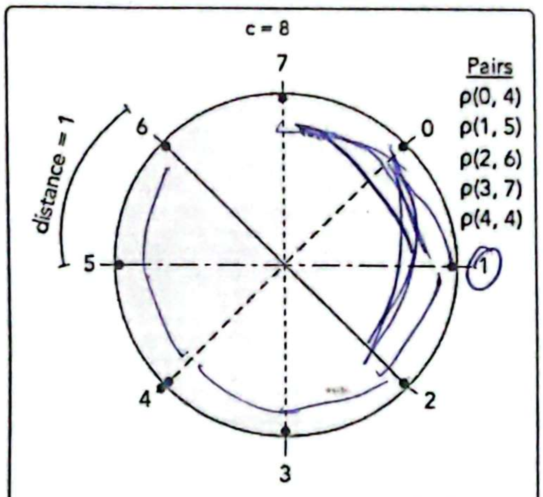
0 4 → 1
1 5 → 2
2 6 → 1
3 7 → 1
4 4.

Sample Output 0

```
2
```

Explanation 0

In the diagram below, the relationship between points in pairs at non-overlapping indices is shown by a connecting line:



d(ρ(0, 4), ρ(1, 5)) = 1
d(ρ(0, 4), ρ(2, 6)) = 2
d(ρ(0, 4), ρ(3, 7)) = 1
d(ρ(0, 4), ρ(4, 4)) = 0
d(ρ(1, 5), ρ(2, 6)) = 1
d(ρ(1, 5), ρ(3, 7)) = 2
d(ρ(1, 5), ρ(4, 4)) = 0
d(ρ(2, 6), ρ(3, 7)) = 1
d(ρ(2, 6), ρ(4, 4)) = 0
d(ρ(3, 7), ρ(4, 4)) = 0

As you can see, the maximum distance between any two pairs of points is 2, so we print 2 as our answer.

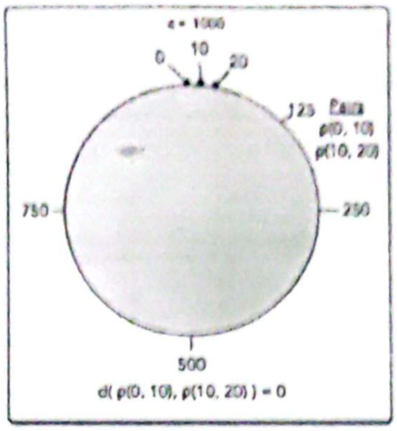
Sample Input 1

```
2 1000
0 10
10 20
```

Sample Output 1

Explanation 1

In the diagram below, we have four individual points located at three indices:



Because two of the points overlap, the minimum distance between the two pairs of points is 0. Thus, we print 0 as our answer.