Weekly Coding Session 6

---

1. **Draw Stairs**

Given a number n, draw stairs using the letter "I", n tall and n wide, with the tallest in the top left.

For example, n = 3 results in:

"I\n I\n I"
or printed:

```
I
 I
  I
```

Another example a 7-step stair should be drawn like this:

```
I
 I
  I
   I
    I
     I
      I
```

---

2. **Nth Smallest Element**

Task
Given an array/list of integers, find the Nth smallest element in the array.

Notes
Array/list size is at least 3.
Array/list's numbers could be a mixture of positives , negatives and zeros.
Repetition in array/list's numbers could occur, so don't remove duplications.
Input >> Output Examples
arr=[3,1,2] n=2 ==> return 2
arr=[15,20,7,10,4,3] n=3 ==> return 7
arr=[2,169,13,-5,0,-1] n=4 ==> return 2
arr=[2,1,3,3,1,2], n=3 ==> return 2

---

3. Confuse the Devil

In the churchyard of St Mary's Priory in Monmouth UK, you can see the gravestone of John Renie, beautifully engraved with 285 letters. Intended to confuse the Devil and thus secure Renie's passage to heaven, the epitaph can be read in many different ways. (Exactly how many is the subject of another kata)

```
E I N E R N H O J S J O H N R E N I E
I N E R N H O J S E S J O H N R E N I
N E R N H O J S E I E S J O H N R E N
E R N H O J S E I L I E S J O H N R E
R N H O J S E I L E L I E S J O H N R
N H O J S E I L E R E L I E S J O H N
H O J S E I L E R E R E L I E S J O H
O J S E I L E R E H E R E L I E S J O
H O J S E I L E R E R E L I E S J O H
N H O J S E I L E R E L I E S J O H N
R N H O J S E I L E L I E S J O H N R
E R N H O J S E I L I E S J O H N R E
N E R N H O J S E I E S J O H N R E N
I N E R N H O J S E S J O H N R E N I
E I N E R N H O J S J O H N R E N I E
```

Starting from the central H and reading zig-zag fashion, finishing at any of the four corners, the epitaph reads HERE LIES JOHN RENIE.

Each valid path that spells out the epitaph, starts from the central H, finishes at one of the four corners (E). There are no dead ends and no doubling back. For example, valid paths from the centre to the top-right corner consist only of steps upwards, and to the right.

You would like to use John Renie's idea on your own gravestone, which has enough space for w x h letters. Your chosen epitaph has length l. Luckily for you, w+h is an even number, and (w+h)/2 equals l.

The Task:
Write a function make_grid which returns the grid of letters to be engraved into your gravestone.

Inputs:

s - The message to be engraved on your headstone

w - The available width

h - The available height

Output:
A 2-D array of characters with h rows, and w characters per row.

Just like John Renie's gravestone, your epitaph can be read in many different ways by starting from the central letter and reading zig-zag fashion, finishing at any of the four corners.

Example 1:
make_grid("RIP", 3, 3)
should return:

```
[
['P','I','P'],
['I','R','I'],
['P','I','P']
]
```

Example 2:
make_grid("ABCDE", 3, 5)
should return:

```
[
['E','D','E'],
['D','C','D'],
['C','B','C'],
['B','A','B'],
['C','B','C'],
['D','C','D'],
['E','D','E']
]
```

---

4. **Best travel**

John and Mary want to travel between a few towns A, B, C ... Mary has on a sheet of paper a list of distances between these towns. ls = [50, 55, 57, 58, 60]. John is tired of

driving and he says to Mary that he doesn't want to drive more than t = 174 miles and he will visit only 3 towns.

Which distances, hence which towns, they will choose so that the sum of the distances is the biggest possible to please Mary and John?

Example:
With list ls and 3 towns to visit they can make a choice between:
[50,55,57],[50,55,58],[50,55,60],[50,57,58],[50,57,60],[50,58,60],[55,57,58],[55,57,60],[55,58,60],[57,58,60].

The sums of distances are then: 162, 163, 165, 165, 167, 168, 170, 172, 173, 175.

The biggest possible sum taking a limit of 174 into account is then 173 and the distances of the 3 corresponding towns is [55, 58, 60].

The function chooseBestSum (or choose_best_sum or ... depending on the language) will take as parameters t (maximum sum of distances, integer >= 0), k (number of towns to visit, k >= 1) and ls (list of distances, all distances are positive or zero integers and this list has at least one element). The function returns the "best" sum ie the biggest possible sum of k distances less than or equal to the given limit t, if that sum exists, or otherwise nil, null, None, Nothing, depending on the language. In that case with C, C++, D, Dart, Fortran, F#, Go, Julia, Kotlin, Nim, OCaml, Pascal, Perl, PowerShell, Reason, Rust, Scala, Shell, Swift return -1.

Examples:
ts = [50, 55, 56, 57, 58] choose_best_sum(163, 3, ts) -> 163

xs = [50] choose_best_sum(163, 3, xs) -> nil (or null or ... or -1 (C++, C, D, Rust, Swift, Go, ...)

ys = [91, 74, 73, 85, 73, 81, 87] choose_best_sum(230, 3, ys) -> 228

---

5. **Decode the Morse code, advanced**

Electric telegraph is operated on a 2-wire line with a key that, when pressed, connects the wires together, which can be detected on a remote station. The Morse code encodes every character being transmitted as a sequence of "dots" (short presses on the key) and "dashes" (long presses on the key).

When transmitting the Morse code, the international standard specifies that:

"Dot" – is 1 time unit long.
"Dash" – is 3 time units long.
Pause between dots and dashes in a character – is 1 time unit long.
Pause between characters inside a word – is 3 time units long.
Pause between words – is 7 time units long.
However, the standard does not specify how long that "time unit" is. And in fact different operators would transmit at different speed. An amateur person may need a few seconds to transmit a single character, a skilled professional can transmit 60 words per minute, and robotic transmitters may go way faster.

For this question we assume the message receiving is performed automatically by the hardware that checks the line periodically, and if the line is connected (the key at the remote station is down), 1 is recorded, and if the line is not connected (remote key is up), 0 is recorded. After the message is fully received, it gets to you for decoding as a string containing only symbols 0 and 1.

For example, the message HEY JUDE, that is ···· · −·−− ·−−− ··− −·· · may be received as follows:

11001100110011000000110000001111110011001111110011111100000000000000011001111110011111100000011001100111111000000111111001100110000011

As you may see, this transmission is perfectly accurate according to the standard, and the hardware sampled the line exactly two times per "dot".

That said, your task is to implement two functions:

Function decodeBits(bits), that should find out the transmission rate of the message, correctly decode the message to dots ., dashes - and spaces (one between characters, three between words) and return those as a string. Note that some extra 0's may naturally occur at the beginning and the end of a message, make sure to ignore them. Also if you have trouble discerning if the particular sequence of 1's is a dot or a dash, assume it's a dot.
2. Function decodeMorse(morseCode), that would take the output of the previous function and return a human-readable string.

NOTE: For coding purposes you have to use ASCII characters . and -, not Unicode characters.

The Morse code table is preloaded for you (see the solution setup, to get its identifier in your language).

Eg:
```
morseCodes(".--") //to access the morse translation of ".--"
```
All the test strings would be valid to the point that they could be reliably decoded as described above, so you may skip checking for errors and exceptions, just do your best in figuring out what the message is!

Good luck!

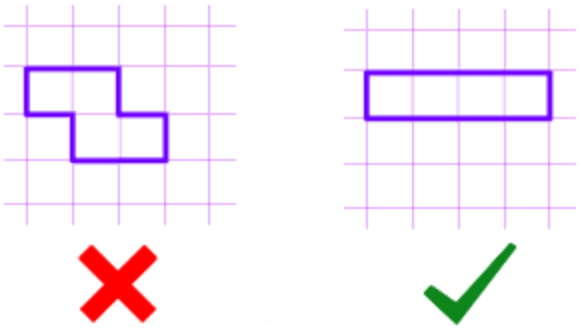
---

6. **Decode the Morse code, advanced**

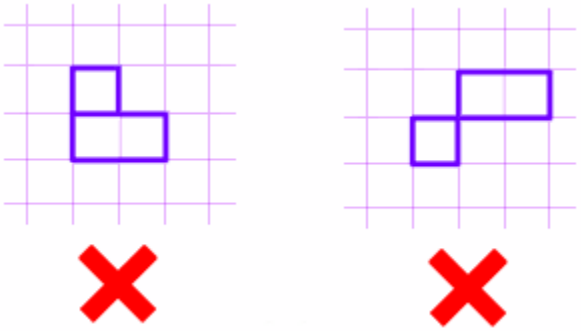
Write a method that takes a field for well-known board game "Battleship" as an argument and returns true if it has a valid disposition of ships, false otherwise. The argument is guaranteed to be 10*10 two-dimension array. Elements in the array are numbers, 0 if the cell is free and 1 if occupied by ship.

Battleship (also Battleships or Sea Battle) is a guessing game for two players. Each player has a 10x10 grid containing several "ships" and objective is to destroy enemy's forces by targetting individual cells on his field. The ship occupies one or more cells in the grid. Size and number of ships may differ from version to version. In this kata we will use Soviet/Russian version of the game.

Before the game begins, players set up the board and place the ships accordingly to the following rules:
There must be single battleship (size of 4 cells), 2 cruisers (size 3), 3 destroyers (size 2) and 4 submarines (size 1). Any additional ships are not allowed, as well as missing ships.
Each ship must be a straight line, except for submarines, which are just single cell.



The ship cannot overlap or be in contact with any other ship, neither by edge nor by corner.



This is all you need to solve this kata. If you're interested in more information about the game, visit this link http://en.wikipedia.org/wiki/Battleship_(game)

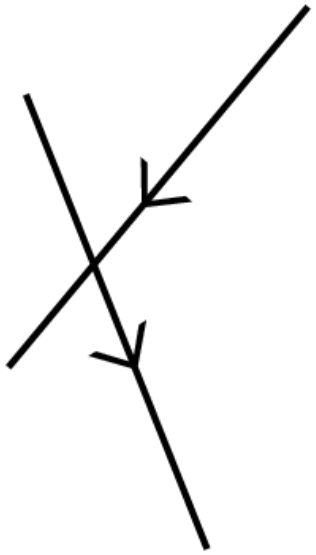
---

7. **Collinearity**

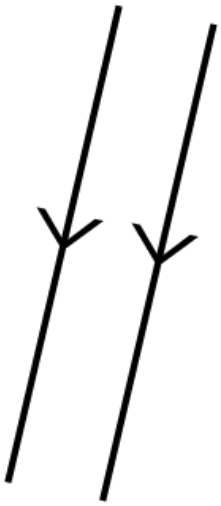
Theoretical Material:

You are given two vectors starting from the origin (x=0, y=0) with coordinates (x1,y1) and (x2,y2). Your task is to find out if these vectors are collinear. Collinear vectors are vectors that lie on the same straight line. They can be directed in the same or opposite directions. One vector can be obtained from another by multiplying it by a certain number. In terms of coordinates, vectors (x1, y1) and (x2, y2) are collinear if (x1, y1) = (k*x2, k*y2) , where k is any number acting as a coefficient.

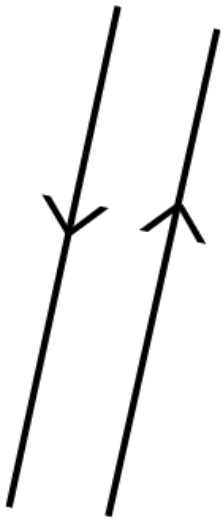
Collinear Vectors



Two non-collinear Vectors



Collinear Vectors in same direction



Collinear Vectors in opposite direction

For more information, check out this article on collinearity.
https://www.cuemath.com/geometry/collinear-vectors/

Problem Description
Write the function collinearity(x1, y1, x2, y2), which returns a Boolean type depending on whether the vectors are collinear or not.

all coordinates are integers
-1000 <= any coordinate <= 1000
Notes
All vectors start from the origin (x=0, y=0).
Be careful when handling cases where x1, x2, y1, or y2 are zero to avoid division by zero errors.
A vector with coordinates (0, 0) is collinear to all vectors.
Examples
(1,1,1,1) → true
(1,2,2,4) → true
(1,1,6,1) → false
(1,2,-1,-2) → true

(1,2,1,-2) → false
(4,0,11,0) → true
(0,1,6,0) → false
(4,4,0,4) → false
(0,0,0,0) → true
(0,0,1,0) → true
(5,7,0,0) → true

---

8. **Statistics for an Athletic Association**

You are the "computer expert" of a local Athletic Association (C.A.A.). Many teams of runners come to compete. Each time you get a string of all race results of every team who has run. For example here is a string showing the individual results of a team of 5 runners:

"01|15|59, 1|47|6, 01|17|20, 1|32|34, 2|3|17"

Each part of the string is of the form: h|m|s where h, m, s (h for hour, m for minutes, s for seconds) are positive or null integer (represented as strings) with one or two digits. Substrings in the input string are separated by , or ,.

To compare the results of the teams you are asked for giving three statistics; range, average and median.

Range : difference between the lowest and highest values. In {4, 6, 9, 3, 7} the lowest value is 3, and the highest is 9, so the range is 9 - 3 = 6.

Mean or Average : To calculate mean, add together all of the numbers and then divide the sum by the total count of numbers.

Median : In statistics, the median is the number separating the higher half of a data sample from the lower half. The median of a finite list of numbers can be found by arranging all the observations from lowest value to highest value and picking the middle one (e.g., the median of {3, 3, 5, 9, 11} is 5) when there is an odd number of observations. If there is an even number of observations, then there is no single middle value; the median is then defined to be the mean of the two middle values (the median of {3, 5, 6, 9} is (5 + 6) / 2 = 5.5).

Your task is to return a string giving these 3 values. For the example given above, the string result will be

"Range: 00|47|18 Average: 01|35|15 Median: 01|32|34"

of the form: "Range: hh|mm|ss Average: hh|mm|ss Median: hh|mm|ss"`

where hh, mm, ss are integers (represented by strings) with each 2 digits.

Remarks:
if a result in seconds is ab.xy... it will be given truncated as ab.
if the given string is "" you will return ""