CaucusRace

You are invited to play the following game. There is a sort of a circle (*"the exact shape doesn't matter",* as [the Dodo](https://books.google.com/books?id=xabjCwAAQBAJ&lpg=PT25&ots=BZGz14LnHA&dq=the%20exact%20shape%20doesn't%20matter%20dodo&pg=PT25#v=onepage&q&f=false) once said) with integer values written around it one by one. The player starts the game with a balance of zero on her *account*.

At the beginning of the game, she chooses one of the positions on the circle i. The number values[i] written there is added to her *account*. Each move the player makes shifts her to the next position clockwise. Each time the player makes a move, her *account* changes by the number written on the new position. If, after a move, her balance becomes zero or negative, the game is over. The player wins if she manages to go through the full circle.

Given the list of values, return the sorted list of winning start positions (0-based).

Example

For values = [-1, 4, -1, 3, -2, 2, 2, -3, 1, 3, -2], the output should be

caucusRace(values) = [1, 3, 5, 8].

Here are the games outcomes for each possible starting position:

* start = 0: account = [-1], game over

xstart = 1: account = [4, 3, 6, 4, 6, 8, 5, 6, 9, 7, 6], win!

* start = 2: account = [-1], game over
* start = 3: account = [3, 1, 3, 5, 2, 3, 6, 4, 3, 7, 6], win!
* start = 4: account = [-2], game over
* start = 5: account = [2, 4, 1, 2, 5, 3, 2, 6, 5, 8, 6], win!
* start = 6: account = [2, -1], game over
* start = 7: account = [-3], game over
* start = 8: account = [1, 4, 2, 1, 5, 4, 7, 5, 7, 9, 6], win!
* start = 9: account = [3, 1, 0], game over
* start = 10: account = [-2], game over

Input/Output

* **[execution time limit] 4 seconds (js)**
* **[input] array.integer values** The sequence of integers placed around the circle.  
    
   *Guaranteed constraints:* 0 < values.length < 5×104,  
   -100 < values[i] < 100.
* **[output] array.integer**

|  |  |
| --- | --- |
| **input** | **output** |
| values: [-1, 4, -1, 3, -2, 2, 2, -3, 1, 3, -2] | [1, 3, 5, 8] |
| see file - 2 | [120, 125, 158, 159, 274, 329] |
| values: [-2, -1, -7, -10, 3, -3, -6, 13, -10, -6, -10, 15, -5, 14, 9, 3, 8, 15, -8, 2, -3, -8, -7, 9, 7, 1, -9, 3, -1, 3, 10, 14, -8, 15, -4, 14, 9, 6, 5, 12, 9, 4, 6, -10, 8, 0, 10, -9, 4, 2, 0, 8, 8, -6, 15, -3, 13, 8, -4, 6, -3, 14, 4, 13, -9, -4, 12, 9, -2, -10, 7, 8, -1, -1, -3, 10, 9, -3, -5, -1, 10, 8, -2, -8, 11, -7, -9, -7, -9, 8, -1, 10, 1, 7, -7, 13, 2, 0, 4, 14] | [11, 13, 14, 15, 23, 27, 29, 30, 31, 33, 35, 36, 37, 38, 39, 40, 44, 46, 48, 49, 51, 52, 54, 56, 57, 59, 61, 62, 66, 70, 89, 91] |

|  |  |
| --- | --- |
| see file - 4 | [43] |
| see file - 5 | [20, 21, 22, 25, 29, 30, 31, 148, 149, 191, 192, 221, 224, 225, 226, 227, 255, 257, 278, 279, 280, 281, 282, 293, 375, 376, 377, 378, 379, 385, 387, 390, 391, 405, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 422, 423, 424, 426, 431, 432, 436, 439, 473, 474, 475, 477, 481] |
| see file - 6 | [58] |
| see file - 7 | [] |
| see file - 8 | [0, 2, 3, 4, 5, 14, 15, 16, 18, 26, 27, 29, 30, 33, 34, 35, 36, 37, 38, 39, 43, 44, 45, 47, 49, 50, 51, 52, 54, 58, 66, 67, 69, 70, 71, 75, 76, 87, 93, 95, 96, 97, 98, 99, 100, 107, 110, 114, 115, 116, 117, 118, 119, 120, 121, 122, 125, 128, 129, 131, 132, 134, 135, 136, 137, 138, 140, 142, 143, 146, 147, 148, 154, 156, 157, 161, 162, 163, 164, 165, 166, 170, 176, 177, 178, 179, 180, 186, 193, 194, 195, 196, 197, 198, 215, 216, 217, 218, 219, 220] |

|  |  |
| --- | --- |
| see file - 9 | [40995] |
| see file - 10 | [173, 174, 175, 176, 177, 178, 179, 197, 198, 199, 204, 205, 206, 227, 230, 231, 233, 234, 236, 238, 239, 242, 244, 259, 262, 284, 337, 345, 346, 350, 361, 363, 364, 366, 371, 372, 373, 374, 376, 377, 398, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 417, 418, 421, 422, 423, 431, 432, 435, 436, 437, 439, 440, 482, 492, 497, 498, 502, 503, 504, 505, 652, 685, 686, 687, 688, 690, 692, 693, 698, 764, 767, 780, 790, 791, 798, 999, 1004, 1005, 1010, 1011, 1033, 1034, 1035, 1036, 1038] |
|  |  |