Question 1:

YouTube currently displays a like and a dislike button, allowing you to express your opinions about particular content. It's set up in such a way that you cannot like and dislike a video at the same time.

There are two other interesting rules to be noted about the interface:

1. Pressing a button, which is already active, will undo your press.
2. If you press the like button after pressing the dislike button, the like button overwrites the previous "dislike" state. The same is true for the other way round.

Create a function that takes in an array of button inputs and returns the final state.

Examples:

likeOrDislike(["Dislike"]) ⟶ "Dislike"

likeOrDislike(["Like", "Like"]) ⟶ "Nothing"

likeOrDislike(["Dislike", "Like"]) ⟶ "Like"

likeOrDislike(["Like", "Dislike", "Dislike"]) ⟶ "Nothing"

Extra Information:

- if no button is currently active, return "Nothing".
- If the array is empty, return "Nothing".

Question 2:
Some characters do not change after a rotation of 180 degrees. They can be read, although sometimes in a different way. For example, uppercase letters "H", "I", "N", "O", "S", "X", "Z" after rotation are not changed, the letter "M" becomes a "W", and vice versa.

So, the word "WOW" turns into the word "MOM". On the other hand, the word "HOME" cannot be rotated.

Find the number of unique Rotated Words in the input string txt (without duplicates).

Examples
rotatedWords("HSSN") ⟶ 1
// String can be rotated to a valid string

rotatedWords("OS IS UPDATED") ⟶ 2
// "OS" and "IS" can be rotated to a valid string

rotatedWords("MUBASHIR") ⟶ 0

Extra Information

String contains only uppercase letters.

Question 3:
An employee working at a very bizzare company earns one penny on their first day. However, for every day that passes, their base amount **doubles**, so they earn two pennies on the second day and four pennies on the third day (totalling 7 pennies). Given a number of days, return how many pennies the employee accumulates.

Examples:
doubledPay(1) ⟶ 1

doubledPay(2) ⟶ 3

doubledPay(3) ⟶ 7

Extra Information
You will only get tests for valid positive integers.

Question 4:
I'm trying to watch some lectures to study for my next exam but I keep getting distracted by meme compilations, vine compilations, anime, and more on my favorite video platform.

Your job is to help me create a function that takes a string and checks to see if it contains the following words or phrases:

- "anime"
- "meme"
- "vines"
- "roasts"
- "Danny DeVito"

If it does, return "NO!". Otherwise, return "Safe watching!".

Examples:
preventDistractions("vines that butter my eggroll") ⟶ "NO!"

preventDistractions("Hot pictures of Danny DeVito") ⟶ "NO!"

preventDistractions("How to ace BC Calculus in 5 Easy Steps") ⟶ "Safe watching!"

Question 5:
Abigail and Benson are playing Rock, Paper, Scissors.

Each game is represented by an array of length 2, where the first element represents what Abigail played and the second element represents what Benson played.

Given a sequence of games, determine who wins the most number of matches. If they tie, output "Tie".

- R stands for Rock
- P stands for Paper
- S stands for Scissors

Examples:
calculateScore([["R", "P"], ["R", "S"], ["S", "P"]]) → "Abigail"

// Benson wins the first game (Paper beats Rock).
// Abigail wins the second game (Rock beats Scissors).
// Abigail wins the third game (Scissors beats Paper).
// Abigail wins 2/3.

calculateScore([["R", "R"], ["S", "S"]]) → "Tie"

calculateScore([["S", "R"], ["R", "S"], ["R", "R"]]) → "Tie"

Question 6:
Create a function to find only the root value of x in any quadratic equation ax^2 + bx + c. The function will take three arguments:

- a as the coefficient of x^2
- b as the coefficient of x
- c as the constant term

Examples
quadraticEquation(1, 2, -3) ⟶ 1

quadraticEquation(2, -7, 3) ⟶ 3

quadraticEquation(1, -12, -28) ⟶ 14

Question 7:
Mona has created a method to sort an array in ascending order.

Starting from the left of the array, she compares numbers by pairs. If the first pair is ordered [smaller number, larger number], she moves on. If the first pair is ordered [larger number, smaller number], she swaps the two integers before moving on to the next pair. She repeats this process until she reaches the end of the array.

Then, she moves back to the beginning of the array and repeats this process until the entire array is sorted.

If the unsorted array is: [3, 9, 7, 4], she will perform the following steps (note *Swaps* here refers to cumulative swaps):

1. She starts with the first pair.
2. [3, 9] is in order, move on. Array: [3, 9, 7, 4]. Swaps: 0.
3. [9, 7] is not. Swap. Array: [3, 7, 9, 4]. Swaps: 1
4. [9, 4] is not. Swap. Array: [3, 7, 4, 9]. Swaps: 2
5. Check if array is sorted. It is not, so start over at first pair.
6. [3, 7] is in order, move on. Array: [3, 7, 4, 9]. Swaps: 2
7. [7, 4] is not. Swap. Array: [3, 4, 7, 9]. Swaps: 3.
8. [7, 9] is in order, move on. Array: [3, 4, 7, 9]. Swaps: 3.
9. Check if array is sorted. It is. End.

Sorting the array [3, 9, 7, 4] takes her *3 swaps*. Write a function that takes in an unsorted array and returns the number of swaps it takes for the array to be sorted according to Mona's algorithm.

Examples:
numberOfSwaps([5, 4, 3]) ⟶ 3

numberOfSwaps([1, 3, 4, 5]) ⟶ 0

numberOfSwaps([5, 4, 3, 2]) ⟶ 6

Question 8:
Write a function that sorts the **positive numbers** in **ascending order**, and keeps the **negative numbers** untouched.

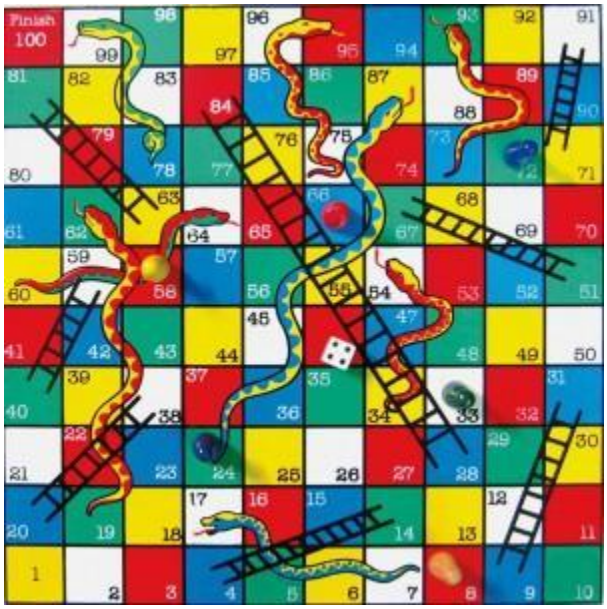
Examples:
posNegSort([6, 3, -2, 5, -8, 2, -2]) ➞ [2, 3, -2, 5, -8, 6, -2]

posNegSort([6, 5, 4, -1, 3, 2, -1, 1]) ➞ [1, 2, 3, -1, 4, 5, -1, 6]

posNegSort([-5, -5, -5, -5, 7, -5]) ➞ [-5, -5, -5, -5, 7, -5]

posNegSort([]) ➞ []

Question 9: Snakes and Ladders



Snakes and ladders, known originally as Moksha Patam, is an ancient Indian board game for two or more players regarded today as a worldwide classic. It is played on a game board with numbered, gridded squares. A number of "ladders" and "snakes" are pictured on the board, each connecting two specific board squares. The object of the game is to navigate one's game piece, according to die rolls, from the start (bottom square) to the finish (top square), helped by climbing ladders but hindered by falling down snakes.

Following are the requirements for the game.

1. Make a **Grid()** function that will take the current score (location on which the player is) and then display the 10x10 grid using cout command.

   Requirement: Grid should be stored in 2D array

   a. Numbers will be displayed on the grid
   b. "*" will represent the 1st player
   c. "x" will represent the 2nd player
2. Make a **RollDice()** function that will roll a random number between 1 to 6 and return that number.

3. Make a **Score()** function that will take the current score and the dice number and then return the final number on which the player will land after checking whether the player has landed on a ladder or on a snake or on a simple location.

Keep asking the player to roll the dice and then keep displaying the updated game board until the 1st player or 2nd player reaches 100 (the score of the player becomes 100). (Remember not

to exceed 100. i.e. if the 1st player is on 99 then he/she needs 1 to win. He/she will remain on 99 until the dice rolls 1 then he wins.)

***** Also add the feature of loading and storing the board configuration from the file.

Question 10:

A text file named "matter.txt" contains some text, which needs to be displayed such that every next character is separated by a symbol "#". Write a function definition for hash_display() in c++ that would display the entire content of the file matter.txt in the desired format.

Example:

If the file matter.txt has the following content stored in it :

THE WORLD IS ROUND

The function hash_display() should display the following content :

T#H#E# #W#O#R#L#D# #I#S# #R#O#U#N#D#

Question 11:

Aditi has used a text editing software to type some text. After saving the article as WORDS.TXT, she realised that she has wrongly typed alphabet J in place of alphabet I everywhere in the article.

Write a function definition for JTOI() in c++ that would display the corrected version of entire content of the file WORDS.TXT with all the alphabets "J" to be displayed as an alphabet "I" on screen.

Note: Assuming that WORD.TXT does not contain any J alphabet otherwise.

Example:

If Aditi has stored the following content in the file WORDS.TXT:

WELL, THJS JS A WORD BY JTSELF. YOU COULD STRETCH THJS TO BE A SENTENCE

The function JTOI() should display the following content:

WELL, THIS IS A WORD BY ITSELF. YOU COULD STRETCH THIS TO BE A SENTENCE

Problem # 12

I'm trying to write a function to flatten a 2D array into one array. In other words, I want to transform this: [[1, 2], [3, 4]] into [1, 2, 3, 4].

Here is my code:

```
void flatten(int 2D arr[][2], int rowSize, int colSize) {
  int arr2[rowSize * colSize];
  for (int i = 0; i < rowSize; i++) {
    arr2[i] = arr[i];
  }
}
```

But...it doesn't seem to be working! Fix my code so that it correctly flattens the array.

Examples
flatten([[1, 2], [3, 4]]) ⟶
// Expected: [1, 2, 3, 4]

flatten([[6, 7], [2, 4]]) ⟶
// Expected: [6, 7, 2, 4]

flatten([[0, 8], [4, 10]]) ⟶
// Expected: [0, 8, 4, 10]

Problem # 13:

Given an array of women and an array of men, either:

- Print "sizes don't match" if the two arrays have different sizes.
- If the sizes match, print an array of pairs, with the first woman paired with the first man, second woman paired with the second man, etc.

Examples

zipIt(["Elise", "Mary"], ["John", "Rick"])
→ [["Elise", "John"], ["Mary", "Rick"]]

zipIt(["Ana", "Amy", "Lisa"], ["Bob", "Josh"])
→ "sizes don't match"

zipIt(["Ana", "Amy", "Lisa"], ["Bob", "Josh", "Tim"])
→ [["Ana", "Bob"], ["Amy", "Josh"],["Lisa", "Tim"]]

Problem # 14

Implement the method largestColumnFirst(). This method takes a two-dimensional array of integers M, finds the column with the largest sum, and switches it with the first column in M.

For example:

$$\begin{matrix} 6 & 7 & \mathbf{9} & 4 & 8 \\ 3 & 2 & \mathbf{7} & 4 & 1 \\ 9 & 4 & \mathbf{5} & 8 & 3 \end{matrix} \Rightarrow \begin{matrix} \mathbf{9} & 7 & 6 & 4 & 8 \\ \mathbf{7} & 2 & 3 & 4 & 1 \\ \mathbf{5} & 4 & 9 & 8 & 3 \end{matrix}$$

Hint: Once you find the largest-sum column, you will need to swap its entries with the first column, element-by-element. Notice that this is different from the way rows can be swapped, as was shown in the previous example. Can you see why?

Problem # 15

A Sudoku is a 9x9 grid that is completed when every 3x3 square, row and column consists of the numbers 1-9.

For this task, you will be given a completed 3x3 square, in the form of a two-dimensional array. Create a function that checks to make sure this 3x3 square contains each number from 1-9 exactly once. Make sure there are no duplicates, and no numbers outside this range.

Test Cases:

isMiniSudoku([[1, 3, 2], [9, 7, 8], [4, 5, 6]]) ⟶ true

isMiniSudoku([[1, 1, 3], [6, 5, 4], [8, 7, 9]]) ⟶ false
// The 1 is repeated twice

isMiniSudoku([[0, 1, 2], [6, 4, 5], [9, 8, 7]]) ⟶ false
// The 0 is included (outside range)

isMiniSudoku([[8, 9, 2], [5, 6, 1], [3, 7, 4]]) ⟶ true