1. Create a checker board generator, which takes as inputs n and 2 elements to generate an n x n checkerboard with those two elements as alternating squares.

Examples

```
checker_board(2, 7, 6)    [
  [7, 6],
  [6, 7]
]

checker_board(3, "A", "B")    [
  ["A", "B", "A"],
  ["B", "A", "B"],
  ["A", "B", "A"]
]

checker_board(4, "c", "d")    [
  ["c", "d", "c", "d"],
  ["d", "c", "d", "c"],
  ["c", "d", "c", "d"],
  ["d", "c", "d", "c"]
]

checker_board(4, "c", "c")    "invalid"
```

2. A string is an almost-palindrome if, by changing only one character, you can make it a palindrome. Create a function that returns True if a string is an almost-palindrome and False otherwise.

Examples

```
almost_palindrome("abcdcbg")    True
# Transformed to "abcdcba" by changing "g" to "a".

almost_palindrome("abccia")    True
# Transformed to "abccba" by changing "i" to "b".

almost_palindrome("abcdaaa")    False
# Can't be transformed to a palindrome in exactly 1 turn.

almost_palindrome("1234312")    False
```

3. Create a function that finds how many prime numbers there are, up to the given integer.

Examples

prime_numbers(10)   4
# 2, 3, 5 and 7

prime_numbers(20)   8
# 2, 3, 5, 7, 11, 13, 17 and 19

prime_numbers(30)   10
# 2, 3, 5, 7, 11, 13, 17, 19, 23 and 29

4. If today was Monday, in two days, it would be Wednesday.

Create a function that takes in a list of days as input and the number of days to increment by. Return a list of days after n number of days has passed.

Examples

after_n_days(["Thursday", "Monday"], 4)   ["Monday", "Friday"]

after_n_days(["Sunday", "Sunday", "Sunday"], 1)   ["Monday", "Monday", "Monday"]

after_n_days(["Monday", "Tuesday", "Friday"], 1)   ["Tuesday", "Wednesday", "Saturday"]

5. You are in the process of creating a chat application and want to add an anonymous name feature. This anonymous name feature will create an alias that consists of two capitalized words beginning with the same letter as the users first name.

Create a function that determines if the list of users is mapped to a list of anonymous names correctly.

Examples

is_correct_aliases(["Adrian M.", "Harriet S.", "Mandy T."], ["Amazing Artichoke", "Hopeful Hedgehog", "Marvelous Mouse"])   True

is_correct_aliases(["Rachel F.", "Pam G.", "Fred Z.", "Nancy K."], ["Reassuring Rat", "Peaceful Panda", "Fantastic Frog", "Notable Nickel"])   True

is_correct_aliases(["Beth T."], ["Brandishing Mimosa"])   False

# Both words in "Brandishing Mimosa" should begin with a "B" - "Brandishing Beaver" would do the trick.