Exercise: is in

Exercise: is in

5.0/5.0 points (graded)
ESTIMATED TIME TO COMPLETE: 18 minutes

We can use the idea of **bisection search** to determine if a character is in a string, so long as the string is sorted in alphabetical order.

First, test the middle character of a string against the character you're looking for (the "test character"). If they are the same, we are done - we've found the character we're looking for!

If they're not the same, check if the test character is "smaller" than the middle character. If so, we need only consider the lower half of the string; otherwise, we only consider the upper half of the string. (Note that you can compare characters using Python's `<` function.)

Implement the function `isIn(char, aStr)` which implements the above idea recursively to test if `char` is in `aStr`. `char` will be a single character and `aStr` will be a string that is in alphabetical order. The function should return a boolean value.

As you design the function, think very carefully about what the base cases should be.

```
    aStr: an alphabetized string
5
6   returns: True if char is in aStr; False otherwise
7   '''
8   # Your code here
9   indice = len(aStr)//2
10  if indice == 0:
```

```
    if indice == 0:
        return False
    else:
        if aStr[indice] == char:
            return True
        else:
            if aStr[indice] > char:
                return isIn(char, aStr[:-(indice-1)])
            else:
                return isIn(char, aStr[-indice:])
```

Press ESC then TAB or click outside of the code editor to exit

Correta

```
def isIn(char, aStr):
    '''
    char: a single character
    aStr: an alphabetized string

    returns: True if char is in aStr; False otherwise
    '''
    # Base case: If aStr is empty, we did not find the char.
    if aStr == '':
        return False

    # Base case: if aStr is of length 1, just see if the chars are equal
    if len(aStr) == 1:
        return aStr == char

    # Base case: See if the character in the middle of aStr equals the
    #   test character
    midIndex = len(aStr)//2
    midChar = aStr[midIndex]
    if char == midChar:
        # We found the character!
        return True

    # Recursive case: If the test character is smaller than the middle
    #  character, recursively search on the first half of aStr
    elif char < midChar:
        return isIn(char, aStr[:midIndex])

    # Otherwise the test character is larger than the middle character,
    #  so recursively search on the last half of aStr
    else:
        return isIn(char, aStr[midIndex+1:])
```

Test results

Hide output

CORRECT

Test: isIn('a', '')

Output:

```
False
```

Test: isIn('t', 'bcddhlotuxxz')

Output:

```
True
```

Test: isIn('p', 'bcemnppswx')

Output:

```
True
```

Test: isIn('z', 'eghhhuxz')

Output:

```
True
```

Test: isIn('x', 'adloppqtwzz')

Output:

```
False
```

Test: isIn('v', 'acchijjoppqrrx')

Output:

```
False
```

Test: isIn('u', 'cdginopquvw')

Output:

```
True
```

Test: isIn('t', 'fimoppz')

Output:

```
False
```

Test: isIn('n', 'dgiklmmnnsx')

Output:

```
True
```

Test: isIn('d', 'u')

Output:

```
False
```

Hide output

Note: In programming there are many ways to solve a problem. For your code to check correctly here, though, you must write your recursive function such that you make a recursive call directly to the function `isIn`. Thank you for understanding.

Hints

Basic function structuring

Be very careful about how you slice the string in the recursive cases! Before you execute the recursive cases, you test the middle character - so if you reach the recursive cases, you know the middle character cannot be a match, right? So be careful to *not* include this character when you make your recursive call!

What should your base case be?

You should be thinking about 3 situations:

- What happens when the string is empty?
- What happens when the string is of length 1?
- What happens when the test character equals the middle character?

What should your recursive case be?

You should be thinking about 2 situations:

- What happens when the test character is smaller than the middle character?
- What happens when it is larger?

If you are getting the error stating that "Your code should be recursive" when you already make a call to `isIn` : check your indention -- specifically, a common mistake is that your function and docstring do not start at the same indentation level.

Enviar

Answers are displayed within the problem

Exercise: is in

Ocultar discussão

Topic: Lecture 4 / Exercise: is in

Add a Post

Show all posts por atividade recente

? I get 4,5 out of 5, and I don't know why? 3

Hi all! This is my code, and to be honest I was very pleased with it, not getting recursion that well ...

pythontutor.com was really helpful

	hi! if any of you are struggling to understand what your code is doing and why, I found pythontut...	1
?	Question about the string Hi, i'm a bit confused with the instructions. So basically, if the string is not in alphabetical order th...	2
	Tested Code in Spyder Worked but Got 0/5! I wrote the below code, and it seems that it worked, but why I'm getting a wrong answer here! ple...	2
	Extremely Tough but satisfying to resolve This was definitely the hardest problem yet. I've found it quite hard to get a grasp on the recursiv...	4
	Spoiler How Many Lines did it Take You? I was just curious to know how many lines it took everyone to accomplish this task? I feel like min...	37
	Unrelated realization I've just noticed that if you type spacebar behind a character, spacebar essentially acts as a back ...	3
	Messy code Hi. It took me 30 lines to get the right answer. I would appreciate some guidance on how to clean ...	2
?	SPOILER AHEAD, question about invalid syntax I have a question that relates to my code so do not look at this if you do not want to see my code!...	6
	This exercise did not take 18 minutes to complete. I spent a good 2 hours on it, and I'm sure there's a way more elegant way to do it.	3
	[SPOILER solved problem] I'm not sure if my code is efficient or elegant. Hi, I just want to know is my code on this level of class optimal or do I over code? I spent more the...	11
	no index error when slicing but yielding empty string Curiously I found out by chance that slicing a string in the form "a"[:0] and "a"[1:] do not produce ...	3
	The autograder doesn't check Bisection Search Note that i have completed the question and wonder why my solution seems easier than the des...	2