RecursionExercises

August 12, 2016

0.1 Recursion Exercises

1 The answer to every question should use recursion!

2 Level 1!

In [4]:
```
# Define a recursive function called numTimes which takes in a string and a
# the given letter appears in the string
def numTimes(s, letter):
    pass

print numTimes('abc', 'a')
# answer: 1

print numTimes('abcabc', 'b')
# answer: 2

print numTimes('ABCabc', 'C')
# answer: 1
```

In [5]:
```
# Define a recursive function called sum_all which takes in a list and retu
def sumAll(l):
    pass

print sumAll([1, 2, 3])
# answer: 6

print sumAll([1, 2, 3, 1, 2, 3])
# answer: 12

print sumAll([5, 4, 1, 7, 1, 4, 7])
# answer 29
```

In [6]:
```
# Define a recursive function called exp which takes in two ints a and b, a
# Your function hsould not use **
def exp(a, b):
    pass
```

```
print exp(2, 0)
# answer: 1

print exp(2, 5)
# answer: 32

print exp(3, 6)
# answer: 729
```

3 Level 2!

In [ ]:
```
# Define a recursive function called reverse which takes in a string and re
def reverse(s):
    pass

print reverse('abc')
# answer: cba

print reverse('ccc')
# answer: 'ccc'

print reverse('this is a long string')
# answer: gnirts gnol a si siht
```

In [7]:
```
# Define a recursive function called removeLetter which takes in a string a
# and returns the string back without the given letter.
def removeLetter(s, letter):
    pass

print removeLetter('abc', 'a')
# answer: bc

print removeLetter('abcabcabc', 'b')
# answer: acacac

print removeLetter('no instances of this letter', 'z')
# answer: no instances of this letter
```

In [8]:
```
# Define a recursive function called replaceLetter which takes in a string
# The function returns a string that has all of the instances of the first
def replaceLetter(s, a, b):
    pass

print replaceLetter('car', 'r', 'w')
# answer: caw

print replaceLetter('hello', 'o', 'i')
# answer: helli
```

```
print replaceLetter('no instances of letter', 'z', 'a')
# answer: no instances of letter
```

In [ ]:
```
# Define a recursive function called sumDigits which takes in an int and su
# Hint: The % operator can give you the last digit in the number.
def sumDigits(n):
    pass

print sumDigits(123)
# answer: 6

print sumDigits(123456789)
# answer: 45

print sumDigits(369)
# answer: 18
```

In [ ]:
```
# Define a recursive function called cumulativeSum which takes in a list ar
# returns a list with the cumulative sum of the numbers in a list.
# Your function should use the helper function.
def cumulativeSumHelper(lstA, lstB):
    pass

def cumulativeSum(lst, finalLst):
    pass
```

4 Level 3!

In [10]:
```
# Define a recursive function called gcf which takes in an int and returns
# Your code should use the Euclidean algorithm.
def gcf(a, b):
    pass

print gcf(5, 17)
# answer: 1

print gcf(101, 197)
# answer: 1

print gcf(72, 180)
# answer: 36

print gcf(24, 36)
# answer: 12
```

In [ ]:
```
# Define a recursive function called convertToBin which takes an int and
# returns a string which is the binary version of the int
```

```
def convertToBin(a):
    pass

print convertToBin(0)
# answer: 0

print convertToBin(1)
# answer: 1

print convertToBin(2)
# answer: 10

print convertToBin(3)
# answer: 11

print convertToBin(4)
# answer: 100

print convertToBin(50)
# answer: 110010
```

In [ ]:
```
# Define a recursive function called convertToInt which takes a binary numb
# returns the corresponding integer.
def convertToInt(s):
    pass

print convertToInt('10101')
# answer: 21

print convertToInt('1101')
# answer: 13

print convertToInt('1001')
# answer: 9

print convertToInt('1111111')
# answer: 127

print convertToInt('00000000000000')
# answer: 0
```

In [ ]:
```
# Define a recursive function called mergeLists which takes two sorted list
# Your function should use the helper provided.
def mergeListsHelper(lstA, lstB, resultList):
    pass

def mergeLists(lstA, lstB):
    pass
```

```
print mergeLists([1, 2, 3], [4, 5, 6])
# answer: [1, 2, 3, 4, 5, 6]

print mergeLists([1, 3, 5], [2, 4, 6])
# answer: [1, 2, 3, 4, 5, 6]

print mergeLists([1, 7, 10, 15, 22], [3, 5, 9])
# answer: [1, 3, 5, 7, 9, 10, 15, 22]
```