FunctionExercises

August 12, 2016

In [ ]:
```
# Write a function to add two to a number
def addTwo(x):
    pass

addTwo(2)
# answer: 4

addTwo(5)
# answer: 7

addTwo(1)
# answer: 3

addTwo(-10)
# answer: -8

addTwo(-6)
# answer: -4

addTwo(546845313489786)
# answer: 546845313489788
```

In [ ]:
```
# Write a function that multiples a number by three
def multiplyThree(x):
    pass

multiplyThree(2)
# answer: 6

multiplyThree(5)
# answer: 15

multiplyThree(-1)
# answer: -3

multiplyThree(15647186)
# answer: 46941558
```

```
multiplyThree(15)
# answer: 45

multiplyThree(-40)
# answer: -120
```

In [ ]:
```
# Write a function that adds two numbers
def addTwoNumbers(x, y):
    pass

addTwoNumbers(1, 4)
# answer: 5

addTwoNumbers(-20, 7)
# answer: -13

addTwoNumbers(-4, -10)
# answer: -14

addTwoNumbers(-8, -12)
# answer: -20

addTwoNumbers(9, 3)
# answer: 12

addTwoNumbers(6, 14)
# answer: 20

addTwoNumbers(-5, 14)
# answer: 9
```

In [ ]:
```
# Write a function that concatenates two strings
def concatTwoStrings(x, y):
    pass

concatTwoStrings('hello', 'world')
# answer: helloworld

concatTwoStrings('hello ', 'world')
# answer: hello world

concatTwoStrings('mango ', 'pineapple')
# answer: mango pineapple

concatTwoStrings('My name is ', 'Daniel!')
# answer: My name is Daniel!

concatTwoStrings('first string', ' second string')
# answer: first string second string
```

In [ ]:
```
# Write a function that returns the sum of the numbers in a list
def sumNumbersInList(x):
    pass

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

sumNumbersInList([5, 1, 8, 10])
# answer: 24

sumNumbersInList([4, 6, 7, 13, 19])
# answer: 49

sumNumbersInList([-10, -7, -5, 1, 18])
# answer: -3

sumNumbersInList([17, -5, -19, -13, 2])
# answer: -18

sumNumbersInList([14, 9, -7, 15, -5])
# answer: 26

sumNumbersInList([2, -20, 12, 20, 14])
# answer: 28

sumNumbersInList([-2, -10, 19, 13, 12])
# answer: 32

sumNumbersInList([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17
# answer: 171
```

In [ ]:
```
# Write a function which:
# a) if the input is even, returns the input divided by two
# b) if the input is odd, returns the input times two
# The function should use an else statement
def evenOdd(x):
    pass

evenOdd(2)
# answer: 1

evenOdd(7)
# answer: 14

evenOdd(5)
# answer: 10

evenOdd(6)
```

```
        # answer: 3

        evenOdd(-40)
        # answer: -20

        evenOdd(-2)
        # answer: -1

        evenOdd(-3)
        # answer: -6
```

In [ ]:
```
        # Write a function that sums the numbers from 1 to x
        # x will be positive
        # Your function should use a loop
        def sumFromOneToX(x):
            pass

        sumFromOneToX(5)
        # answer: 15

        sumFromOneToX(50)
        # answer: 1275

        sumFromOneToX(15)
        # answer: 120

        sumFromOneToX(3)
        # answer: 6

        sumFromOneToX(50150)
        # answer: 1257536325
```

In [ ]:
```
        # Write a function which returns true if a list is in strictly increasing o
        # Your function should use a loop
        def isIncreasing(x):
            pass

        isIncreasing([1, 2, 3])
        # answer: True

        isIncreasing([1, 2, 2])
        # answer: False

        isIncreasing([0, 3, 6])
        # answer: True

        isIncreasing([0, 3, 6, 9])
        # answer: True
```

```
isIncreasing([0, 3, 2, 9])
# answer: False

isIncreasing([1, -2, 3, 4])
# answer: False
```

In [ ]:
```
# Write a function that returns a list of all the multiples of three betwee
# Your function should use a while loop
def multOfThree(x):
    pass

multOfThree(3)
# answer: [0, 3]

multOfThree(4)
# answer: [0, 3]

multOfThree(6)
# answer: [0, 3, 6]

multOfThree(8)
# answer: [0, 3, 6]

multOfThree(9)
# answer: [0, 3, 6, 9]

multOfThree(30)
# answer: [0, 3, 6, 9, 12, 15, 18, 21, 24, 27, 30]
```

In [ ]:
```
# Write a function which takes a number x and returns True if it is a prime
def isPrime(x):
    pass

isPrime(0)
# answer: False

isPrime(1)
# answer: False

isPrime(2)
# answer: True

isPrime(3)
# answer: True

isPrime(4)
# answer: False
```

```
isPrime(5)
# answer: True

isPrime(67)
# answer: True

isPrime(80)
# answer: False

isPrime(1541)
# answer: False

isPrime(15749)
# answer: True
```

In [ ]:
```
# Write a function which takes a string and returns True if it is a palindr
# Your function should use a loop
def isStringPalindrome(x):
    pass

isStringPalindrome('abcba')
# answer: True

isStringPalindrome('abcbac')
# answer: False

isStringPalindrome('1111111111')
# answer: True

isStringPalindrome('1112111111')
# answer: False

isStringPalindrome('rfjewafefawejfr')
# answer: True

isStringPalindrome('1112112111')
# answer: True

isStringPalindrome('1112112211')
# answer: False
```

In [ ]:
```
# Write a function which takes an integer and returns True if it is a palin
# Can you reuse the code above?
def isNumPalindrome(x):
    pass

isNumPalindrome(111)
```

```
        # answer: True

        isNumPalindrome(112)
        # answer: False

        isNumPalindrome(114564511)
        # answer: False

        isNumPalindrome(11454645411)
        # answer: True

        isNumPalindrome(48611684)
        # answer: True

        isNumPalindrome(48612684)
        # answer: False
```

In [ ]:
```
        # Write a function that sums the numbers from 0 to x, inclusive, if the nu
        # For example, sum 3, 5, 6, 9, 10, 12, 15 for x=17
        # Your function should use a loop.
        # Your function should use the 'or' keyword.
        def sumDivisibleThreeFive(x):
            pass

        sumDivisibleThreeFive(3)
        # answer: 3

        sumDivisibleThreeFive(14)
        # answer: 45

        sumDivisibleThreeFive(15)
        # answer: 60

        sumDivisibleThreeFive(17)
        # answer: 60

        sumDivisibleThreeFive(1548)
        # answer: 559293
```

In [ ]:
```
        # Write a function that returns a list of prime numbers from 0 to x, inclus
        # Your function should use a loop
        # Can you use some code above?
        def primesToX(x):
            pass

        primesToX(2)
        # answer: [2]
```

```
primesToX(10)
# answer: [2, 3, 5, 7]

primesToX(15)
# answer: [2, 3, 5, 7, 11, 13]

primesToX(100)
# answer: [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59,
```

In [ ]:
```
# Write a function that returns the largest element in a list.
# Your function should use a loop
def largestInList(x):
    pass

largestInList([1, 2, 3])
# answer: 3

largestInList([3, 5, 2, 7, 5, 7])
# answer: 7

largestInList([5, 7, 75, 78, 23, 73, 53])
# answer: 78

largestInList([-69, -81, 1, -36, 90, 28, -82, -91, -37, -69])
# answer: 90

largestInList([5, 2, -75, 76, -40, 0, 29, -41, 63, -40])
# answer: 76

largestInList([68, 21, -13, -46, -64, 26, 17, 6, -42, 57])
# answer: 68

largestInList([59, 32, -42, -49, -63, -21, -69, -36, 5, 41])
# answer: 59

largestInList([9, 37, -2, 95, -75, 64, 63, 25, 91, -60])
# answer: 95

largestInList([-129, -413, 793, -47, 609, 585, -843, 965, 294, 877])
# answer: 965
```

In [ ]:
```
# Write a function that reverses a list.
def reverseList(x):
    pass

reverseList([1, 2, 3])
# answer: [3, 2, 1]
```

```
reverseList([-4, 0, 3, 6, 9])
# answer: [9, 6, 3, 0, -4]

reverseList(['lists', 'can', 'have', 'multiple', 'types', True])
# answer: [True, 'types', 'multiple', 'have', 'can', 'lists']

reverseList([True, False, False, True])
# answer: [True, False, False, True]

reverseList(['hello', 123, False])
# answer: [False, 123, 'hello']
```

In [ ]:
```
# Write a function returns the elements in the odd positions of a list.
def oddPositionsInList(x):
    pass

oddPositionsInList([1, 2, 3, 4, 5])
# answer: [2, 4]

oddPositionsInList([1, 2, 3])
# answer: [2]

oddPositionsInList([1])
# answer: []

oddPositionsInList([8, -9, -3, -7, 6])
# answer: [-9, -7]

oddPositionsInList([-61, -15, 78, 41, 82, -62, -89, 19, 27, 3])
# answer: [-15, 41, -62, 19, 3]
```

In [ ]:
```
# Write a function that takes two lists and combines them in alternating or
# e.g. [a,b,c], [1,2,3] → [a,1,b,2,c,3]
# The lists will be of the same size
def combineLists(x, y):
    pass

combineLists(['a', 'b', 'c'], [1, 2, 3])
# answer: ['a', 1, 'b', 2, 'c', 3]

combineLists([1, 2, 3], [1, 2, 3])
# answer: [1, 1, 2, 2, 3, 3]

combineLists([3, 2, 1], [1, 2, 3])
# answer: [3, 1, 2, 2, 1, 3]

combineLists(['hello', 'pineapple', 'mango', 'strawberry'], [4, 7, 3, 5])
# answer: ['hello', 4, 'pineapple', 7, 'mango', 3, 'strawberry', 5]
```

```
In [ ]: # Write a function that takes a string of length 1 and:
        # a) Returns True if it is a vowel
        # b) Returns False if it is a consonant
        def isVowel(x):
            pass

        isVowel('a')
        # answer: True

        isVowel('c')
        # answer: False

        isVowel('d')
        # answer: False

        isVowel('o')
        # answer: True

        isVowel('e')
        # answer: True

        isVowel('p')
        # answer: False

        isVowel('i')
        # answer: True

        isVowel('w')
        # answer: False

        isVowel('u')
        # answer: True
```

```
In [ ]: # Write a function which takes two lists and:
        # a) Returns True if any element of one list is in the other
        # b) Returns False otherwise
        def overlaps(x, y):
            pass

        overlaps([1, 2, 3], [4, 5, 6])
        # answer: False

        overlaps([1, 2, 3], [1, 5, 6])
        # answer: True

        overlaps([-26, -46, -56, 8, -70, -89, -29, -84, -44, -83], [1, 5, 6])
        # answer: False
```

```
overlaps([-37, -58, 91, -51, 79, -87, 44, 97, 59, -78], [17, -87, 31, 52, -
# answer: True

overlaps(['hello', 'pineapple'], ['mango', 'strawberry'])
# answer: False

overlaps(['hello', 'pineapple'], ['mango', 'strawberry', 'mango'])
# answer: False

overlaps(['hello', 'pineapple', 'mango'], ['mango', 'strawberry', 'mango'])
# answer: True
```

In [ ]:
```
# Write a function that takes a list of strings and returns the longest str
# If multiple strings are the longest, return the first one.
def longestString(x):
    pass

longestString(['a', 'ab', 'abc'])
# answer: abc

longestString(['very long string', 'a', 'ab', 'abc', 'hello'])
# answer: very long string

longestString(['mango', 'pineapple', 'strawberry', 'kiwi'])
# answer: strawberry

longestString(['very long string', 'a', 'ab', 'abc', 'hello', 'asd15645t4wa
# answer: asd15645t4wagrafewfa
```