CSCI 145 Quiz 5 practice questions on lists and recursion **ANSWER KEY**

1. Suppose `nums = [6, 1, 9, 4]` and `words = ["sky", "a", "tree"]`. Write the value of each expression.

`[num + 1 for num in nums]`	**`[7, 2, 10, 5]`**
`[num for num in nums if num % 2 == 0]`	**`[6, 4]`**
`[len(word) for word in words]`	**`[3, 1, 4]`**
`[num // 2 for num in nums if num > 4]`	**`[3, 4]`**
`[word[-1] for word in words if len(word) > 1]`	**`["y", "e"]`**
`sorted(nums)`	**`[1, 4, 6, 9]`**

2. For each task, write Map, Filter, or Reduce. A task may need two of them, in order.

From a list of prices, add 6% tax to each price	**Map**
From a list of temperatures, keep the ones below 32	**Filter**
From a list of words, find the longest word	**Reduce**
From a list of prices, find the total of the ones over 20	**Filter, then reduce**

3. Consider this code:

```
def ordered(nums):
    return nums.sort()

ages = [19, 7, 12]
print(ordered(ages))
```

a) What does it print? Circle one.

A. `[7, 12, 19]` B. `[19, 7, 12]`

C. None D. An error

b) What is `ages` at the end? **`[7, 12, 19]`**

4. Consider this code:

```
x = [4, 7]
y = x
z = list(x)
y[0] = 1
z.append(2)
y = y + [8]
```

What is the value of each variable at the end?

a) x **`[1, 7]`**

b) y **`[1, 7, 8]`**

c) z **`[4, 7, 2]`**

5. Write the exact output of this code, one printed line per line.

```
def drop_first(nums):
    nums.pop(0)

def without_first(nums):
    return nums[1:]

data = [5, 3, 8]
drop_first(data)
rest = without_first(data)
print(data)
print(rest)
```

Output

```
[3, 8]
[8]
```

6. Consider this code:

```
def shrink(word):
    if len(word) <= 1:
        print("end")
    else:
        print(word[-1])
        shrink(word[:-1])
shrink("stop")
```

a) Write the exact output.

```
p
o
t
end
```

b) How many times is `shrink` called, counting `shrink("stop")`?

4

c) What is `word` in the third call?

"st"

7. Trace `first_big([3, 9, 2, 12], 10)`. Fill in one row per call: its `nums` and what it returns.

```
def first_big(nums, limit):
    if len(nums) == 0:
        return -1
    elif nums[0] > limit:
        return nums[0]
    else:
        return first_big(nums[1:], limit)
```

call	nums	returns
1st	**[3, 9, 2, 12]**	**12**
2nd	**[9, 2, 12]**	**12**
3rd	**[2, 12]**	**12**
4th	**[12]**	**12**

`first_big([4, 1], 5)` returns: **-1**

8. What happens when this code runs? Circle one.

```
def launch(n):
    if n == 0:
        print("Go!")
    else:
        print(n)
        launch(n + 1)
launch(3)
```

A. It prints 3, 2, 1, Go!

B. It prints 3, Go!

C. It prints 3, 4, 5, ... then a `RecursionError`

D. It prints Go!

9. Consider this code:

```
import turtle as t
def box_spiral(length):
    if length >= 20:
        t.forward(length)
        t.right(90)
        box_spiral(length - 30)
box_spiral(100)
```

a) How many times is `box_spiral` called, counting `box_spiral(100)`? **4**

b) How many lines does it draw? **3**

c) What is `length` in the last call? **10**

10. Write a recursive function `all_same(word)`, without a loop. If every letter in `word` is the same, it returns `True`. Otherwise, it returns `False`. A word with fewer than 2 letters returns `True`. Do not include a docstring. For example:

```
>>> all_same("zzz")
True
>>> all_same("zzaz")
False
>>> all_same("")
True
```

```
def all_same(word):
    if len(word) < 2:
        return True
    elif word[0] != word[1]:
        return False
    else:
        return all_same(word[1:])
```