Quiz on Content

Aliases & Recursion

Review (Week 11.1)

Does any of the following statements produce an error? which one? why? (solve first reasoning about this, later you can execute)

What is printed at the end (assuming that you comment out any of the statements which produce an error).

```
la = [1,2,3]                     #a
la[0] = la[0] + 100              #b
la = la + [500]                  #c
la = la + 600                    #d
la.append(700)                   #a
la = la.append(800)              #b
print("la is now...", la)        #c
```

Review (Week 11.1)

Do any of the following statements produce an error? which one? why? (solve first reasoning about this, later you can execute)

What is printed at the end (assuming that you comment out any of the statements which produce an error).

```
st = "abc"                    #a

st = st + "def"               #b

st[0] = "X"                   #c

st.append("!")                #d

print("st is now...", st)
```

Theory and Understanding

Question 1: Matching

- **base case**
- **recursion**
- **recursive call**
- **recursive definition**
- **infinite recursion**

- The statement that calls an already executing function. - recursive call
- A definition which defines something in terms of itself. To be useful it must include base cases which are not recursive. - recursive definition
- A branch of the conditional statement in a recursive function that does not give rise to further recursive calls. - base case
- A function that calls itself recursively without ever reaching the base case. - infinite recursion
- The process of calling the function that is already executing. - recursion

Question 2: True or False

```
def mystery(numList):
    value = 0
    for num in numList:
        value = value + num
    return value
print(mystery([1,3,5,7,9]))
```

True or false? The code above can be rewritten recursively.

Coding

Q3. Reverse a List

Write a recursive function called **reverseList(lst)** that reverses a list.

https://runestone.academy/runestone/books/published/thinkcspy/IntroRecursion/ProgrammingExercises.html

Q4. Vowel Counter

Write a function called **count_vowels(st)** that returns the number of vowels in the string **st**, using recursion.

Q5. Palindrome Checker

Write a function **is_palindrome(word)** that uses **recursion** to check whether **word** is a palindrome. It should return True if the word is a palindrome and False if it is not.

E.g.

- A and ABA should return True
- AB and ABC should return False