CSc 110 Midterm 2 Sample Exam #1

1. List Mystery

Consider the following function:

```
def list_mystery(list):
    x = 0
    for i in range(len(list) - 1):
        if list[i] > list[i + 1]:
            x += 1
    return x
```

In the left-hand column below are specific lists of integers. Indicate in the right-hand column what value would be returned by function `list_mystery` if the integer list in the left-hand column is passed as its parameter.

Original Contents of List	Value Returned
`a1 = [8]` `result1 = list_mystery(a1)`	_____
`a2 = [14, 7]` `result2 = list_mystery(a2)`	_____
`a3 = [7, 1, 3, 2, 0, 4]` `result3 = list_mystery(a3)`	_____
`a4 = [10, 8, 9, 5, 6]` `result4 = list_mystery(a4)`	_____
`a5 = [8, 10, 8, 6, 4, 2]` `result5 = list_mystery(a5)`	_____

2. While Loop Mystery

For each call of the function below, write the output that is printed:

```
def mystery(i, j):
    while i != 0 and j != 0:
        i = i // j
        j = (j - 1) // 2
        print(str(i) + " " + str(j) + " ", end='')
    print(i)
```

Function Call	Output
`mystery(5, 0)`	_____
`mystery(3, 2)`	_____
`mystery(16, 5)`	_____
`mystery(80, 9)`	_____
`mystery(1600, 40)`	_____

3. Assertions

For the following function, identify each of the three assertions in the table below as being either ALWAYS true, NEVER true or SOMETIMES true / sometimes false at each labeled point in the code. You may abbreviate these choices as A/N/S respectively.

```
def mystery():
    y = 0
    z = 1
    next = input()

    # Point A
    while next >= 0:
        # Point B
        if y > z:
            # Point C
            z = y
        y += 1
        next = input()
        # Point D

    # Point E
    return z
```

	next < 0	y > z	y == 0
Point A			
Point B			
Point C			
Point D			
Point E			

4. File Processing

Write a function named `word_stats` that accepts as its parameter the name of a file that contains a sequence of words and that reports the total number of words (as an integer) and the average word length (as an un-rounded real number). For example, suppose file contains the following words:

```
To be or not to be, that is the question.
```

For the purposes of this problem, we will use whitespace to separate words. That means that some words include punctuation, as in `"be,"`. For the input above, your function should produce exactly the following output:

```
Total words    = 10
Average length = 3.2
```

6. Programming

Write a function named `longest_name` that reads names typed by the user and prints the longest name (the name that contains the most characters) in the format shown below. Your method should accept an integer n as a parameter and should then prompt for n names.

The longest name should be printed with its first letter capitalized and all subsequent letters in lowercase, regardless of the capitalization the user used when typing in the name.

If there is a tie for longest between two or more names, use the tied name that was typed earliest. Also print a message saying that there was a tie, as in the right log below. It's possible that some shorter names will tie in length, such as ryan and TITO in the left log below; but don't print a message unless the tie is between the longest names.

You may assume that n is at least 1, that each name is at least 1 character long, and that the user will type single-word names consisting of only letters. The following table shows two sample calls and their output.

Call	`longest_name(5)`	`longest_name(7)`

Output	`name #1?` **ryan**
	`name #2?` **TITO**
	`name #3?` **John**
	`name #4?` **lAuRaLyN**
	`name #5?` **SujaN**
	`Lauralyn's name is longest`
		`name #1?` **PeTer**
		`name #2?` **eric**
		`name #3?` **RAFAEL**
		`name #4?` **brian**
		`name #5?` **sarina**
		`name #6?` **LIOR**
		`name #7?` **EmIlIo**
		`Rafael's name is longest`
		`(There was a tie!)`

6. Programming

Write a function called `tally_scores` that takes as a parameter the name of a file that contains a series of student records and that prints a summary of each student record. A student record will begin with a name followed by a sequence of integer test scores. The name is guaranteed to be one word and at the beginning of the line. You may assume that each student has at least one test score. Your function should produce two lines of output for each student: one showing the student's name and test scores and a second line showing the average score.

For example, if a file called `records.txt` contains the following:

```
John 71 83 94
Sally 94 85
Fred 90 95 82 85
```

and the following call is made:

```
tally_scores(records)
```

the following output should be produced:

```
John: 71 83 94
average = 82.66666666666667
Sally: 94 85
average = 89.5
Fred: 90 95 82 85
average = 88.0
```

You are to exactly reproduce the format of this output. You may not construct any extra data structures to solve this problem.

Solutions

1.

Call	Value Returned
`a1 = [8]` `result1 = list_mystery (a1)`	`0`
`a2 = [14, 7]` `result2 = list_mystery (a2)`	`1`
`a3 = [7, 1, 3, 2, 0, 4]` `result3 = list_mystery (a3)`	`3`
`a4 = [10, 8, 9, 5, 6]` `result4 = list_mystery (a4)`	`2`
`a5 = [8, 10, 8, 6, 4, 2]` `result5 = list_mystery (a5)`	`4`

2.

Function Call	Output
`mystery(5, 0)`	`5`
`mystery(3, 2)`	`1 0 1`
`mystery(16, 5)`	`3 2 1 0 1`
`mystery(80, 9)`	`8 4 2 1 2 0 2`
`mystery(1600, 40)`	`40 19 2 9 0 4 0`

3.

	`next < 0`	`y > z`	`y == 0`
Point A	SOMETIMES	NEVER	ALWAYS
Point B	NEVER	SOMETIMES	SOMETIMES
Point C	NEVER	ALWAYS	NEVER
Point D	SOMETIMES	SOMETIMES	NEVER
Point E	ALWAYS	SOMETIMES	SOMETIMES

4.

```
def word_stats(file_name):
    words = open(file_name).read().split()
    count = 0
    sum_length = 0

    for word in words:
        count += 1
        sum_length += len(word)

    average = sum_length / count
    print("Total words    = " + str(count))
    print("Average length = " + str(average))
```

5.

```
def longest_name(names):
   longest = ""
   tie = False
   for i in range(1, names + 1):
      name = input("name #" + str(i) + "? ")
      if len(name) > len(longest):
         longest = name
         tie = False
      elif len(name) == len(longest):
         tie = True
   longest = longest[0].upper() + longest[1:].lower()
   print(longest + "'s name is longest")
   if tie:
      print("(There was a tie!)")
```

6.

```
def tally_scores(file_name):
    file = open(file_name)
    lines = file.readlines()
    for line in lines:
        text = line.split()
        print(text[0] + ":", end='')
        points = 0
        count = 0
        for i in range(1, len(text)):
            next_points = int(text[i])
            points += next_points
            count += 1
            print(" " + str(next_points), end='')
        print()
        average = points / count
        print("average = " + str(average))
```