CS 115 Sample Final Exam #1

1. For loop mystery

Consider the following code:

```
for i in range(1, 6):
    print(str(23 + i * 2) + "*", end="")
```

What does it output?

2. Parameter Mystery

In the blank spaces below, write the output produced by each line of the following program, as it would appear on the console.

```
def main():
    x = "happy"
    y = "pumpkin"
    z = "orange"
    pumpkin = "sleepy"
    orange = "vampire"

    orange(y, x, z)
    orange(x, z, y)
    orange(pumpkin, z, "y")
    z = "green"
    orange("x", "pumpkin", z)
    orange(y, z, orange)

def orange(z, y, x):
    print(y, "and", z, "were", x)

main()
```

3. Return Mystery

At the bottom of the page, write the output produced by the following program.

```
def main():
    a = 4
    b = 2
    c = 43
    print(function_b())
    a = function_a()
    f = function_c()
    c = function_b()
    print(a, b, c, f)

def function_b(:
    function_a()
    print("B")
    return 17

def function_c():
    e = function_b()
    print("C")
    d = function_a()
    return d + e

def function_a():
    print("A")
    return 6

main()
```

4. List Mystery 1

Consider the following function:

```
def list_mystery(a):
    a[2] = 3
    a[4] = 7
    a[0] = a(3)
    a[a[1]] = 14
    num = a[3]
```

Indicate in the right-hand column what values would be stored in the list after the subroutine `list_mystery` executes if the passed in list contains the elements in the left column.

Original Contents of List	Final Contents of List
`a1 = [7, 1, 5, 4, 3, 2, 1]` `list_mystery(a1)`	_____
`a2 = [4, 3, 6, 14, 55, 12]` `list_mystery(a2)`	_____
`a3 = [7, 4, 8, 6, 2, 1, 1]` `list_mystery(a3)`	_____
`a4 = [10, 2, 5, 10, 10]` `list_mystery(a4)`	_____
`a5 = [2, 4, -1, 6, -2, 8]` `list_mystery(a5)`	_____

5. List Mystery 2

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)`	_____

6. 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)`	_____

7. Programming

Write a function named `sequence_sum` that prints terms of the following mathematical sequence:

$$1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\frac{1}{6}+\ldots \qquad \text{(also written as } \sum_{i=1}^{\infty}\frac{1}{i}\text{)}$$

Your function should accept a real number as a parameter representing a limit, and should add and print terms of the sequence until the sum of terms meets or exceeds that limit. For example, if your subroutine is passed 2.0, print terms until the sum of those terms is at ≥ 2.0. You should round your answer to 3 digits past the decimal point.

The following is the output from the call `sequence_sum(2.0)`

```
1 + 1/2 + 1/3 + 1/4 = 2.083
```

(Despite the fact that the terms keep getting smaller, the sequence can actually produce an arbitrarily large sum if enough terms are added.) If your function is passed a value less than 1.0, no output should be produced. You must match the output format shown exactly; note the spaces and pluses separating neighboring terms. Other sample calls:

Calls	`sequence_sum(0.0)`	`sequence_sum(1.0)`	`sequence_sum(1.5)`
Output		`1 = 1.000`	`1 + 1/2 = 1.500`

Call	`sequence_sum(2.7)`
Output	`1 + 1/2 + 1/3 + 1/4 + 1/5 + 1/6 + 1/7 + 1/8 = 2.718`

8. Write a function named `count_last_digits` that accepts a list of integers as a parameter and examines its elements to determine how many end in `0`, how many end in `1`, how many end in `2` and so on. Your function will return a list of counters. The count of how many elements end in `0` should be stored in its element at index `0`, how many of the values end in `1` should be stored in its element at index `1`, and so on.

For example, if a list named `list` contains the values `[9, 29, 44, 103, 2, 52, 12, 12, 76, 35, 20]`, the call of `count_last_digits(list)` should return the list `[1, 0, 4, 1, 1, 1, 1, 0, 0, 2]` because 1 element ends with `0` (`20`), no elements end with `1`, 4 elements end with `2` (`2`, `52`, `12`, and `12`), etc.

9. 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
```

Solutions

1.

```
25*27*29*31*33*
```

2.

```
happy and pumpkin were orange
orange and happy were pumpkin
orange and sleepy were y
pumpkin and x were green
green and pumpkin were vampire
```

3.

```
 A
B
17
A
A
B
C
A
A
B
6 2 17 23
```

4.

```
[4, 14, 3, 4, 7, 2, 1]
[14, 3, 3, 14, 7, 12]
[6, 4, 3, 6, 14, 1, 1]
[10, 2, 14, 10, 7]
[6, 4, 3, 6, 14, 8]
```

5.

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

6.

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`

7.

```
def sequence_sum(limit):
     if limit >= 1:
         print("1")
         denominator = 1
         sum = 1.0
         while sum < limit:
             denominator = denominator + 1
             sum = sum + 1.0 / denominator
             print(" + 1/" + str(denominator))
         print(" =", sum)
```

8.

```
def count_last_digits(a):
    result = [0] * 10
    for i in range(0, len(a)):
        value = a[i] % 10
        result[value] = result[value] + 1
    return result
```

9.

```
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))
```