CPSC 1301, Computer Science I Quiz

Quiz 4b

This is a timed test. You have thirty minutes to complete the test. Your deliverable will be a plain text file, that is, an ASCII file with a `.txt` file extension. When you finish the test, upload your deliverable to Canvas. Do not publish your answer to your `git` repository.

You may work with your study partner for this quiz. In fact, working together is strongly encouraged. If you work with a partner, you must each make a seperate submission for credit, but you must also include the names of both authors in your submission.

1 Things you need to know

elements versus indices When you iterate through a sequence, you almost always have a choice between iterating through the elements of the sequence, or the indices of the sequence. We generally designate the letter e as the iteration variable when we use the elements, and the letter i when we use the indices. The `for` loop at lines 6-7 below illustrate using the index, while lines 10-11 illustrate printing the element.

swapping values Python allows you to swap the values of two variables in one line of code, like this: `var1, var2 = var2, var1`. This assigns the value of `var2` to `var1`, and the value of `var1` to `var2`. See the example program below at line 19.

nested loops You can nest loops, one inside the other. Note that you *must* observe the Python indenting rules. Lines 25-28 below print out a multiplication table using nested loops. The `print()` statement on line 28 breaks between the rows of the table.

range(start, end) As we know, `range(n)` gives a range from 0 to an integer less than n. `range(n, m)` also takes two arguments. The first argument is the start of the range, while the second represents the end of the range. See lines 33-34 below. `range(7)` gives a range of 0 to 6, while `range(22, 28)` gives a range of 22 to 27. Examine the output of part 3 very carefully — the intention is to print all pairs of characters from A to F without repetition. Please pay *very* careful attention to this, and stare at this code until you understand fully what it does and how it works.

using expressions as indices The index of a sequence can be a variable, an integer literal, or an expression. For example, a variable looks like `mySequence[myVar]`, an integer literal looks like `mySequence[42]`, and an expression looks like `mySequence[int(6 / 3)]`. For example, line 42 performs integer division on the iteration variable. This is something you will use often.

2 Example program

```
1  print("this is quiz04b.py")
2  print("--------------------------")
3  print("0. difference between 'i' and 'e'")
4  myList = ['p','q','r','s','t','u']
5  print("'i' means 'index' and represents the index of a sequence")
6  for i in range(len(myList)):
7      print(myList[i], end = "\t")
8  print()
9  print("'e' means 'element' and represents the element in a sequence")
```

```
10 for e in myList:
11     print(e, end = "\t")
12 print()
13
14 print("-------------------------")
15 print("1. swap")
16 varA = 'a'
17 varB = 'z'
18 print("before swap:",varA, varB)
19 varA, varB = varB, varA
20 print("after swap:", varA, varB)
21
22 print("-------------------------")
23 print("2. nested loops")
24 listA = [1,2,3,4,5,6]
25 for e1 in listA:
26     for e2 in listA:
27         print(e1 * e2, end = "\t")
28     print()             #break between rows of multiplication table
29
30 print("-------------------------")
31 print("3. range(i, j)")
32 listB = ['A','B','C','D','E','F']
33 for i in range(0, len(listB) - 1):
34     for j in range(i + 1, len(listB)):
35         print(listB[i], listB[j], end = "\t")
36     print()
37
38 print("-------------------------")
39 print("4. arithmetic operator in list index")
40 listC = [0,1,2,3,4,5,6,7,8,9,10]
41 for i in range(len(listC)):
42    print(listC[i // 3], end = "\t")
43 print()
```

3 Example program output

```
C:\Users\ccc31\cols-st\cpsc1301\python-progs>quiz04b.py
this is quiz04b.py
------------------------
0. difference between 'i' and 'e'
'i' means 'index' and represents the index of a sequence
p       q       r       s       t       u
'e' means 'element' and represents the element in a sequence
p       q       r       s       t       u
-------------------------
1. swap
before swap: a z
after swap: z a
-------------------------
2. nested loops
1       2       3       4       5       6
2       4       6       8       10      12
3       6       9       12      15      18
4       8       12      16      20      24
5       10      15      20      25      30
6       12      18      24      30      36
-------------------------
3. range(i, j)
A B     A C     A D     A E     A F
```

```
B C     B D     B E     B F
C D     C E     C F
D E     D F
E F
------------------------
4. arithmetic operator in list index
0       0       0       1       1       1       2       2       2       3       3
```

4 Quiz

1. Given a list like this: `myList = ['t','r','a','p']`, swap the two middle characters so that the list becomes `myList = ['t','a','r','p']`. Use just one line of code to do this.

2. Given a list like this: `myInts = [1,2,3,4,5]`, use nested loops to produce the quotient of the outer element divided by the inner element. The expected output is shown below. Before you attempt to write the code, write down each arithmetic expression by hand to understand the result. For example, your first row should look like this: 1/1 = 1, 1/2 = .5, 1/3 = .333. 1/4 = .25, 1/5 = .2.

```
1.0     0.5     0.3333333333333333      0.25    0.2
2.0     1.0     0.6666666666666666      0.5     0.4
3.0     1.5     1.0     0.75    0.6
4.0     2.0     1.3333333333333333      1.0     0.8
5.0     2.5     1.6666666666666667      1.25    1.0
```

3. Given a list like this: `myColors = ["red","yellow","blue","black","white"]`, produce a printout of all color combinations. Expected output is shown below

```
red yellow      red blue        red black       red white
yellow blue     yellow black    yellow white
blue black      blue white
black white
```

4. Given a list like this: `yourList = [0,1,2,3,4,5,6,7,8,9,10]`, produce a repeating series of numbers `1, 2, 3`. Here is the expected output:

```
1       2       3       1       2       3       1       2       3       1       2
```