CS303E Week 10 Worksheet: More on Lists

Name: _____ EID: _____

Read the questions carefully, and answer each question in the space provided. Use scratch paper to do your work and then copy your answers neatly and legibly onto the test paper. Only answers recorded on the test paper will be graded.

1. (10 points: 1 point each) The following are true/false questions. **Write either T or F in the boxes at the bottom of page 1.** If there's any counterexample, it's false.

   (a) The expression `myList[2][3]` is valid for a nested list with three inner lists, each having at least four elements.

   (b) In a linear search, if the specified item is found, the search algorithm returns the index of the found item plus 1 to account for 0-based indexing.

   (c) Iterating through a nested list requires nested loops to access each individual element.

   (d) Binary search is guaranteed to return the index of the first instance of an item in a list.

   (e) Lists within a list can have a different length from each other.

   (f) In selection sort, the minimum element is repeatedly selected from the unsorted part of the list and swapped with the first element, while in insertion sort, elements are compared and inserted into their correct positions in the sorted part of the list.

   (g) The `len()` function can be used to find the total number of elements in a nested list.

   (h) When using the `.sort()` method on nested lists in Python, the sorting is applied only to the outermost list, leaving the inner lists unaltered.

   (i) In a linear search, one item is checked in the unexplored portion of the list each step, while in binary search, the search space is cut in half with each step.

   (j) In binary search, the number of "searches" required is at most half the length of the list.

a	b	c	d	e	f	g	h	i	j

Page total: _____/10

Questions 2-9 are multiple choice. Each counts 2 points. **Write the letter of the BEST answer in the box on the next page. Please write your answer in UPPERCASE. Each problem has a single answer.**

2. In which scenario would using binary search be more advantageous than linear search?

   A. Searching through a small, unsorted list.

   B. Looking for the first occurrence of an item in a list.

   C. Searching through a large, sorted list.

   D. Finding the maximum value in an unsorted list.

   E. None of the above.

3. How many iterations does a linear search require to find the value 616 in the list `[9, 10, 21, 41, 98, 123, 364, 616, 1218]`?

   A. 8 B. 7 C. 6 D. 9

4. How many iterations does a binary search require to find the value 616 in the list `[9, 10, 21, 41, 98, 123, 364, 616, 1218]`?

   A. 1 B. 2 C. 3 D. 4

5. What will the list `[86485, 42, 1337, 404, 777, 9000, 24601]` look like after three iterations of the selection sort algorithm?

   A. `[42, 404, 777, 1337, 9000, 24601, 86485]`

   B. `[42, 404, 777, 86485, 1337, 9000, 24601]`

   C. `[42, 404, 1337, 86485, 777, 9000, 24601]`

   D. None of the above.

6. What will the list `[86485, 42, 1337, 404, 777, 9000, 24601]` look like after three iterations of the insertion sort algorithm?

   A. `[42, 404, 777, 1337, 9000, 24601, 86485]`

   B. `[42, 1337, 86485, 404, 777, 9000, 24601]`

   C. `[42, 404, 1337, 86485, 777, 9000, 24601]`

   D. None of the above.

7. How do you remove `"dumbo"` from the following nested list?:

```
100acreWood = [["pooh", "tigger", "piglet"], ["eeyore", "rabbit", "owl",
"dumbo"], ["kanga", "roo", "gopher"]]
```

   A. `100acreWood.remove("dumbo")`

   B. `100acreWood.pop(1)`

   C. `100acreWood[1].remove("dumbo")`

   D. `100acreWood[1].pop("dumbo")`

8. How do you check if the string `"mew"` is present in the following nested list?:

```
myTeam = [["eevee", "togepi"], ["mew", "mewtwo"], ["dragonite", "gengar"]]
```

   A. `"mew" in x`

   B. `True if "mew" in [item for item in x] else False`

   C. `"mew" in x[1]`

   D. `"mew" in x[1:1]`

9. On a list that is already mostly sorted, which sorting algorithm is likely to perform worse, selection sort or insertion sort?

   A. Insertion sort, because it will cause unneeded swaps as it sorts through the mostly sorted list.

   B. Selection sort, because it will unnecessarily search for minimum elements, which are likely to be in the correct place already in a mostly sorted list.

   C. The two algorithms will perform exactly the same.

   D. This cannot be determined from the given information.

2	3	4	5	6	7	8	9

Page total: _____/16

The following 8 questions require you to trace the behavior of some Python code and identify the output of that code. For each question, write the output for the code segment on the provided line.

10. (3 points)

```
queensLists = [["ariel"], ["belle", "tiana"], \
               ["mulan", "pocahantas", "rapunzel"]]
result = [homegirl for lst in queensLists for homegirl in lst]
print(result)
```

11. (3 points)

```
inputNums = [13, 17, 25, 28, 10, 30, 21]
myLst = [ [0, 0, 0], [0, 0, 0], [0, 0, 0] ]

for num in inputNums:
    result1 = num % 3
    result2 = (num // 10) % 3
    myLst[result1][result2] += 1
print(myLst)
```

12. (3 points)

```
powerpuff = [["blossom", "mojoJojo"], \
             ["bubbles", "utonium"], \
             ["buttercup", "bunny"]]
powerpuff.sort()
print(powerpuff)
```

13. (3 points)

```
lst1 = [[1, -100], [5, 10]]; lst2 = [[1, -900], [999, 999]]
print(lst1 > lst2)
```

Page total: _____/12

14. (3 points)

```
board = [["H", "X", "Z", "C"], ["A", "O", "T", "J"], \
         ["K", "P", "N", "D"], ["L", "M", "U", "K"]]

for i in range(len(board)):
    for j in range(len(board)):
        if i == j:
            print(board[i][j], end = "")
```

15. (3 points)

```
sanrio = [["keroppi", "helloKitty"], \
          ["pompompurin", "myMelody", "cinnamoroll"], \
          ["chococat", "badtzmaru"]
cuties = [group[0] for group in sanrio]
print(cuties)
```

16. (3 points)

```
looney = [["bugs", "daffy"], [1930, 4], ["tweety", "sylvester", "lola"]]
print(looney[-1][-2][0])
```

17. (3 points)

```
myNums = [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]
total = 0
for i in range(len(myNums)):
    for j in range(4):
        if not j % 2:
            total += myNums[i][j]
print(total)
```

Page total: _____/12