CS303E Mock Exam 3
Dr. Bill Young § Fall 2025

Name: _____ EID: _____

Read the questions carefully, and answer each question in the space provided. If you like, you can use scratch paper to do your work, but copy your answers neatly and legibly onto the test paper. Only answers recorded on the test paper will be graded. Don't write in the spaces marked "Page Total" at the bottom of each page. Note: points as listed on the exam sum to 100, but your mock exam grade will be scaled to a 10-point scale.

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) Binary search will still work correctly even if the input list is unsorted, it just won't be as efficient as compared to running on a sorted list.

   (b) If the `__str__` method isn't explicitly defined for a class, python will raise an error if you try to print an object of that class.

   (c) Within a class, `self.__score` is a private variable.

   (d) `"Hello"` is an object in python.

   (e) If we execute the assignment `s = "call"`, then afterward we can change the string to `"tall"` by adding the line `s[0] = 't'`.

   (f) For any given string `s`, the function calls `s.split()` and `s.split(" ")` will return the same list.

   (g) Dictionaries cannot have repeated keys.

   (h) The expression `bool([])` is equivalent to `False`.

   (i) Any function which calls itself is considered recursive.

   (j) For a given sorted list, it always takes less comparisons/probes to find a given element by binary searching over the list as opposed to searching over the list linearly.

a	b	c	d	e	f	g	h	i	j

Page total: _____/10

2. (10 points: 1 point each) The following are short answer questions. Fill in the word or phrase that *best* matches the description provided. If there's a small line in the question, the answer is the word you might write on that line to complete the sentence.

   (a) _____ Unlike strings which cannot be changed after they're constructed, list objects are _____.

   (b) _____ This value is returned implicitly from functions with no explicit `return` statement.

   (c) _____ The part(s) of a recursive function which return an answer directly, without needing to make a recursive call.

   (d) _____ In the function call `f(2, 5, c=4, d=4)`, while the values for `c` and `d` are passed as keyword arguments, the values `2` and `5` are this type of argument.

   (e) _____ Object attributes beginning with two underscores are _____ attributes.

   (f) _____ This function is called whenever a new object is created.

   (g) _____ Comparisons between characters are done by comparing their underlying _____ codes.

   (h) _____ In contrast to an absolute path name, a _____ path name specifies the location of a file based on the current working directory.

   (i) _____ Strings are sequences of _____.

   (j) _____ This technique for sorting a list consists of repeatedly finding the smallest element in the unsorted part of the list and swapping it to the beginning.

Page total: _____/10

Questions 3–11 are multiple choice. Each counts 1 point. **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.**

3. Which of the following best describes what will happen if we call `open("file.txt", "w")` but a file named `file.txt` already exists?

   A. The old `file.txt` will be overwritten with any new lines we write to the file.
   B. Any new lines we write to the file will be appended to the end of the old file.
   C. Python will raise an error due to the name conflict.
   D. Python will automatically change the name of the new output file to `file-1.txt` to avoid the name conflict.

4. Which of the following is NOT a reason to explicitly close a file after you're finished writing to it?

   A. The file will become corrupted if the program terminates and `close` wasn't called.
   B. It's good programming practice to release resources once finished with them.
   C. The output lines may be held in a buffer and not actually written to the file until it's closed.
   D. The file cannot be accessed by other programs while it's still open.

5. Suppose `L` is a non-empty list of integers. When using the linear search function from the lectures on `L` to try and find an element that isn't present in the list, how many comparisons/probes are made?

   A. `len(L)`
   B. `len(L) + 1`
   C. `int(math.log2(len(L)))`
   D. `int(math.log2(len(L))) + 1`

6. Which of the following is a reason to prefer a recursive implementation to an iterative one for the same task?

   A. Recursive solutions are always more efficient because they avoid duplicated work.
   B. Recursive functions are often simpler to implement and understand.
   C. Modern computer hardware is more optimized for many function calls rather than many loop iterations.
   D. The size of the runtime stack allows for more complex calculations to be done recursively which couldn't be done iteratively.

7. Suppose `c` is an object of some type and `i` is a positive integer. For which of the following possible types for `c` is `c[i]` never a valid operation?

   A. `list` B. `dict` C. `string` D. `set`

8. Suppose `s1` and `s2` are sets and `s1.issubset(s2)` is `True`. Which of the following must also be true?

   A. `s1.intersection(s2) != s2`

   B. `s1.intersection(s2) == s2`

   C. `s1.difference(s2) == set()`

   D. `s2.difference(s1) == set()`

9. Which type of error, if any, is present in the following code to sum adjacent products in a list of integers? For example, `sumAdjacentProducts([1, 6, 3, 8])` should return $1 \cdot 6 + 6 \cdot 3 + 3 \cdot 8 = 48$.

```
def sumAdjacentProducts(L):
    tot = 0
    for i in range(len(L)):
        tot += L[i - 1] * L[i]
    return tot
```

   A. Syntax Error B. Runtime Error C. Logic Error D. No Error

10. Suppose `L` is a non-empty, two-dimensional list of integers. Which of the following best describes the type of `L[0]`?

    A. a 2D list of integers
    B. a list of integers
    C. an integer
    D. the value `None`

11. Suppose you're using the `Student` class from homework 7 and you'd like to print the information corresponding to a particular `Student` object `s`. Which of the following is NOT a valid way to display the student?

    A. `print(s)`

    B. `print(str(s))`

    C. `print(s.__str__())`

    D. all of these are valid

3.	4.	5.	6.	7.	8.	9.	10.	11.	12.

Page total: _____/10

Questions 12–18 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 in the provided box. Note: when displaying a dictionary or set, any ordering of the elements is acceptable.

12. (3 points)

```
s = "what a beautiful day"
a_count = 0
for ch in range(len(s)):
    if ch == "a":
        a_count += 1
print(a_count)
```

13. (3 points)

```
nums = [2, 2, 1, 4, 1, 4, 3, 4, 2, 2]
s = set()
for x in nums:
    if x not in s:
        s.add(x)
    else:
        s.remove(x)
print(s)
```

Page total: _____/9

14. (3 points)

```
words = ["banana", "BANANA", "ban", "bandana", "Alabama"]
words.sort()
print(words)
```

15. (3 points)

```
def f(lst, j):
    val = lst[j]
    i = j + 1
    while (i < len(lst)) and (lst[i] < val):
        lst[i - 1] = lst[i]
        i += 1
    lst[i - 1] = val
    return lst

print(f([7, 2, 14, 6, 9, 13, 15, 19], 2))
```

16. (3 points)

```
def f(s):
    if s == "":
        return {}
    res = f(s[1:])
    if s[0] not in res:
        res[s[0]] = 0
    res[s[0]] += 1
    return res

print(f("an assassin sins"))
```

Page total: _____/9

17. (3 points)

```
class Pokemon:
    def __init__(self, number, name):
        self.number = number
        self.name = name

    def __str__(self):
        return "This is " + self.name + " with Pokedex id " \
            + str(self.number)


p1 = Pokemon(7, "Squirtle")
p2 = Pokemon(7, "Squirtle")

print(p1 == p2, end=" ")
print(p1.number == p2.number, end=" ")
print(str(p1) == str(p2), end=" ")
```

18. (3 points)

```
def f(lst):
    if lst[-1] > 5:
        lst.append(8)
    print(lst, end=" ")

nums = [1, 9]
f(nums)
print(nums, end=" ")
f(nums)
```

Page total: _____/6

19. (10 points: 1 point each) The following questions require you to write a *Python expression* that returns the indicated value. You can assume that any modules you need have been imported. Note that this asks for a single, one-line expression for each question, not a longer program fragment. So, you should not have any assignments, loops, or if statements. Hint: if you think you need a loop, try a list comprehension instead.

(a) _____
a list of the squares of all integers 1 through 100 (i.e. [1, 4, 9, ..., 9801, 10000])

(b) _____
for a given non-empty list of integers `L`, a boolean indicating whether all integers in `L` are unique

(c) _____
for a given string `s`, a set containing all lowercase vowels in `s` (vowels are 'a', 'e', 'i', 'o', and 'u')

(d) _____
the first half of the even-length, non-empty string `s`

(e) _____
for a given non-empty list of integers `L`, the number of even integers in `L`

(f) _____
for a given non-empty list of floats `L`, the average of the values in `L`

(g) _____
for a given string `s`, `s` repeated 100 times

(h) _____
for a given non-integer float `num`, a two-element tuple containing the integer immediately below and above num; e.g. if `num` is 3.54, the expression would give `(3, 4)`

(i) _____
the value of pi to five decimal places, as a string, displayed right-justified in a field of width 10

(j) _____
for a given dictionary `d`, the number of unique values (as opposed to keys) in `d`

Page total: _____/10

20. The following 10 questions (worth 1 point each) require you to evaluate a Python expression in the left hand column. For each question, write what the expression evaluates to on the provided line. If evaluation results in an error, write "error"; you don't have to identity the specific type of error.

    You must show a value of the appropriate type. For example, `7.0` **rather than** `7` **for a float and** `"7"` **instead of** `7` **for a string. Answers that do not indicate the data type correctly are wrong.**

    (a) `max({"A":3, "Z":9, "W":-5}.keys())` _____

    (b) `"texas longhorns"[-10::2]` _____

    (c) `"a!!b!cde!".split("!")` _____

    (d) `" \nxyz abc ".lstrip()` _____

    (e) `"aababcabcd".rindex("b")` _____

    (f) `{1,2,5} ^ {2, 4}` _____

    (g) `"wow" >= "wow wow"` _____

    (h) `[2, 5, 1].append(7)` _____

    (i) `{"A":3, "Z":9, "W":-5}["Z"]` _____

    (j) `[(x*x - 1) for x in range(5)]` _____

Page total: _____/10

21. (9 points) Complete the `visitorDictionary` function below which accepts a string `filename` denoting the name of a file containing traveler data. Each line in the file contains a name followed by a colon and then one or more countries the person has visited, separated by commas. The function should read in the contents of the file and then construct and return a dictionary mapping each country to the set of people who have visited it. All traveler names will be unique. See the sample file contents and function call below.

```
>>> visitorDictionary("data.txt")
{"US": {"Alice", "Bob", "Charlie"},
 "Canada": {"Alice", "Charlie"},
 "Mexico": {"Alice", "Bob"},
 "Peru": {"Bob"},
 "Cuba": {"Bob", "Charlie"}}
```

contents of `data.txt`:

```
Alice: US, Canada, Mexico
Bob: Peru, Cuba, Mexico, US
Charlie: Cuba, Canada, US
```

```
def visitorDictionary(filename):
```

Page total: _____/9

22. (10 points) You've recently started playing a new mobile game with your friends, and you've decided to write a Python class to help you keep track of how many in-game coins each of them has. Complete the `CoinCounts` class on the next page by filling in the methods. The `__init__` method takes a list of tuples, one for each of your friends. Each tuple contains two elements, the friend's name (a string) and the number of coins they have in the game (an integer). All fields must be private, however it's up to you exactly what data to store in your class. In addition to `__init__`, you'll also need to complete the following methods:

    - `giveCoins` which accepts a string denoting the friend you're gifting coins to and a positive integer denoting how many coins you're giving them; this function should update that friend's coin count and not return anything; it's possible the name given doesn't match one we've already seen, in which case we'll be adding that friend to the collection with the given number of coins as the initial count
    - `getRichestFriends` which takes no parameters and returns a set of the names of your richest friends; if one of your friends has more coins than all the others, then this will return a set with just that person's name; if multiple friends are tied for having the most coins, then this will return a set with all of their names
    - `__str__` which takes no parameters and returns a string denoting how many friends you have and what the total coin count is among all your friends (see the sample output for examples of this)

    Here is some sample output to demonstrate the class behavior:

```
>>> cc = CoinCounts([("Amy", 15), ("Jeremy", 23), ("Vanessa", 12)])
>>> cc.getRichestFriends()
{"Jeremy"}
>>> str(cc)
"Friend Count: 3, Total Coin Count: 50"
>>> cc.giveCoins("Amy", 8)
>>> cc.getRichestFriends()
{"Amy", "Jeremy"}
>>> str(cc)
"Friend Count: 3, Total Coin Count: 58"
>>> cc.giveCoins("Juan", 33)
>>> cc.getRichestFriends()
{"Juan"}
>>> str(cc)
"Friend Count: 4, Total Coin Count: 91"
```

```
class CoinCounts:
    def __init__(self, friends):




    def giveCoins(self, friend, coins):




    def getRichestFriends(self):




    def __str__(self):
```

Page total: _____/10

23. (7 points) A certain competition consists of a number of tasks, and each participant's performance on each task is judged on an A/B/C/D/F scale, with A being the best and F being the worst. Each team consists of three members, and the team's performance for a task is simply the grade of whichever team member performed best. Complete the `teamPerformance` function that accepts three strings of equal length denoting the performance of each team member on the tasks. For example, the string "BFAD" means the participant got a B for the first task, an F for the second task, an A for the third task, and a D for the fourth task. The function should recursively create and return a string denoting the team's performance for the tasks. Your solution MUST BE RECURSIVE and you MAY NOT USE LOOPS. You MAY NOT DEFINE A RECURSIVE HELPER FUNCTION.

    Here are some example calls to the function:

    ```
    >>> teamPerformance("ADFCD", "CABCC", "BFACF")
    "AAACC"
    >>> teamPerformance("", "", "")
    ""
    >>> teamPerformance("BCDCB", "BDDCD", "FFFAF")
    "BCDAB"
    ```

```
def teamPerformance(str1, str2, str3):
```

Page total: _____/7