CS303E Week 6 Worksheet: Functions

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 the page.** If there's any counterexample, it's false.

   (a) A function must always have a return statement.

   (b) A variable defined within a function is limited to that function and is not accessible outside of it.

   (c) You can call a function without passing any arguments, even if it has required parameters.

   (d) When a function lacks a return statement or includes a return statement without specifying a value, it implicitly returns 0.

   (e) A function can have multiple parameters with the same name as long as they have different default values.

   (f) Keyword arguments are passed to a function based on their order in the function's parameters, while positional arguments are explicitly specified by name in the function call.

   (g) When you pass a mutable object as an argument to a function in Python, the function can modify the object.

   (h) A break statement and a return statement can be used interchangeably.

   (i) Functions can have multiple return statements, but only one of them will be executed during the function's execution.

   (j) Functions can return multiple values using the return statement followed by a comma-separated list of values.

a	b	c	d	e	f	g	h	i	j
F	T	F	F	F	F	T	F	T	T

Page total: _____/10

Questions 2-6 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. What is the primary difference between using `return` and `print()`?

   A. Both `return` and `print()` serve the same purpose, so you can use either one interchangeably.

   B. `return` is used to exit a function and pass a value back to the caller, while `print()` is used to display output on the console.

   C. `print()` is used to pass values to the caller, while `return` is used to display output on the console.

   D. `return` and `print()` are used for the same purpose, but `return` is used when working with numerical values, and `print()` is used for strings.

3. Which of the following is not an advantage of using a function?

   A. Reusability of code.

   B. Easier debugging and maintenance.

   C. Increased program execution speed.

   D. Improved code organization and readability.

4. What is a requirement for a function?

   A. A function must have at least one parameter.

   B. A function must contain at least one return statement.

   C. A function must be defined using the `def` keyword.

   D. A function must have default values for its parameters.

5. What is the significance of proper indentation in relation to functions?

   A. Indentation within a function is optional and doesn't affect the function's behavior.

   B. Indentation is used to group code together and define the body of the function.

   C. Indentation is used to define the end of a function.

   D. Indentation is necessary to declare function parameters.

6. In which of the following scenarios would you not want to use default arguments for a function? Alright. This is an… odd question. I could've made this clearer LOL. Think of this as, which situation is the WORST to use default arguments

   A. When you want to simplify function calls by allowing some arguments to be omitted. this is reason TO use default arguments (they assign values to omitted arguments)

   B. When you want to minimize potential confusion and ensure that function behavior is explicit. our function should be explicit regardless of default arguments, won't it? Plus, users are still able to pass their own values, which muddies our function still. This seems like a case where you wouldn't want parameters at all.

   C. When you need the function to handle different cases with varying argument values. This is vague and suspicious — functions should already be able to handle different cases. Using (or not using) default arguments wouldn't change anything. This maybe seems like a case where you'd want if statements (conditionals).

   D. When you want to ensure that all function arguments are explicitly provided by the caller.
   Yes!! If we want to ensure that all parameters are provided when the function is used, then we would want to AVOID default arguments. because default arguments let you call a function without providing all parameters. So D is the only scenario where you would 100% want to avoid default arguments

2	3	4	5	6
B	C	C	B	D

Page total: _____/10

The following 7 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.

7. (3 points)

A function call using keyword arguments! (passing in parameter values out of order, but specifying the parameter names)

Just be careful w/ PEMDAS here and float division. our return is 2 + 6 / 3 which is equal to 2 + 2.0 = 4.0

```
def lulu(buttercup, coco, jasper):
    return jasper + buttercup / coco

peanut = lulu(coco = 3, jasper = 2, buttercup = 6)
print(peanut)
```

4.0

8. (3 points)

A function with a default value!

coraline() essentially multiplies our parameter by 4. So, coraline("mister!") gets us back "mister!" times four (we can multiply strings). This is stored in taterTot. coraline(coraline()) simply nests one coraline call within another. But note that coraline() uses the default parameter (7) because we did not specify one. so coraline() = coraline(7) which returns 28 (7 * 4). So, coraline(coraline()) is just coraline(28), which returns 112 (28 * 4). This is stored in pumpkin. Then we print taterTot and pumpkin.

```
def coraline(wybie = 7):
    wybie *= 2
    return wybie * 2

taterTot = coraline("mister!")
pumpkin = coraline(coraline())
print(taterTot, pumpkin)
```

mister!mister!mister!mister! 112

9. (3 points)

Note that if basil is False, then 'not basil' is True. Our while loop thus has condition 'True and True', so it executes. But it immediately runs into a break. A break ends the loop *and* skips us to the code after the loop. So we do not hit 'return grey + 5'. After the loop, our function finishes, so we implicitly return None.

```
def kookie(grey):
    kitkat = True
    basil = False
    while not basil and kitkat:
        break
        return grey + 5

print(kookie(10))
```

None

Page total: _____/9

10. (3 points)

```
def josh(chica, apollo, coco, peaches):
    return chica + apollo * coco * peaches

print(josh(chica = 5, peaches = 2, apollo = 2, 3))
```

Our function call in our print statement is invalid, because we try sending a positional argument (the 3) after keyword arguments. If you mix positionals and keywords, the positionals *must* come first.

Error

11. (3 points)

```
def elmo():
    cookieMonster = "chunky"
    BIGbird = "squawky"
    return BIGbird
    return cookieMonster

myFavoriteStreet = elmo() # GO TEAM SESAME!!!
print(myFavoriteStreet)
```

Once we hit a return, we're out of the function. So, when we hit 'return BIGbird', we send back BIGbird ("squawky") and our function ends. So return cookieMonster never does anything.

squawky

12. (3 points) A function that returns multiple values!

```
def marriage():
    return "Bert", "Ernie"

spouse1, spouse2 = marriage()
print(spouse1, "<3", spouse2)
```

We create two variables, one for each value returned from marriage(). spouse1 is "Bert" and spouse2 is "Ernie". And we simply print these strings with a heart in between.

Bert <3 Ernie

13. (3 points)

```
def countDracula():
    count = 0
    for num in range(10):
        count += 1

print(count)
```

Note that our print(count) is not indented as part of our function countDracula(). But the variable count exists only within this function. So when we try printing count outside the scope of our function, our program crashes.

Error

Page total: _____/12