Quiz 4a

9 questions

1
point

1.

One of the tasks that you will engage in when learning a new programming language is locating the name of a built-in function that performs a common, simple operation. While you might be tempted to write your own code that performs this operation, locating a built-in function is usually preferable since the built-in version is automatically correct and others that read your code will immediately recognize what your code is doing.

Python has a built-in function that adds up the numbers in a list. For example, given the list $[1, 2, 5, 4]$, this function returns $1 + 2 + 5 + 4 = 12$. Use your search skills to find the name of this built-in function. Enter the name of the built-in function below, without any parentheses or arguments.

(Note that we could just tell you the name of this function. However, the point of this problem is for you to start learning how to locate useful language features on your own.)

Enter answer here

1
point

2.

Let `my_list` be the list [`"This"`, `"course"`, `"is"`, `"great"`].

- What is `len(my_list)`?
- What non-negative number is the index of `"great"`? I.e., how would you replace the question marks in `my_list[???]` so that the resulting value is `"great"`?

Submit two numbers, one for each of these two questions, separated by spaces.

Enter answer here

1
point

3.
Let `my_list` be the list [`"This"`, `"course"`, `"is"`, `"great"`].

We can use Python's slice notation to get part of this list. What non-negative numbers can be used to get the slice [`"course"`, `"is"`]? I.e., what two non-negative numbers should we put in `my_list[??? : ???]` to get that result?

Submit the two numbers in order, separated only by spaces.

Enter answer here

1
point

4.
If we want to split a list `my_list` into two halves, which of the following uses slices to do so correctly?

More precisely, if the length of `my_list` is 2n, i.e., even, then the two parts should each have length n. If its length is 2n+1, i.e., odd, then the two parts should have lengths n and n+1.

□ my_list[0 : len(my_list) // 2] and
my_list[len(my_list) // 2 + 1 : len(my_list)]

□ my_list[: len(my_list) // 2 - 1] and
my_list[len(my_list) // 2 :]

□ my_list[: len(my_list) // 2] and
my_list[len(my_list) // 2 :]

□ my_list[0 : len(my_list) // 2] and
my_list[len(my_list) // 2 : len(my_list)]

1
point

5.

What is the distance between point $[4, 7]$ and the nearest point on the circle centered at $[2, 9]$ with radius 2? Provide at least 4 digits of accuracy.

Hint: The distance between a point and a circle is the distance between the point and the center of the circle minus the radius of the circle. You can use the point-to-point distance code described in this week's videos.

Enter answer here

1
point

6.

A ball with velocity $[4, 2]$ reflects off a vertical wall. What is its new velocity?

○ [4, -2]

○ [-4, 2]

○ [4, 2]

○ [-4, -2]

1
point

7.
Which of the following illustrate how to properly structure a keydown or keyup event handler? (For more advanced Python programmers, assume that you have just imported simplegui and haven't used `from`.)

□

```
def keydown_handler(key):
    if key == simplegui.KEY_MAP["left"]:
        …
```

□

```
def keydown_handler(key):
    if "left" == KEY_MAP[key]:
        …
```

□

```
def keydown_handler(key):
    if "left" == simplegui.KEY_MAP[key]:
        …
```

□

```
def keydown_handler(key):
    if key == KEY_MAP["left"]:
        …
```

1
point

8.
Assume you have a program with a keydown handler. You run it, and press a single key and *hold it down continuously*. How many times does the keydown handler get called?

Experiment in CodeSkulptor to find out.

○ Unlimited — i.e., repeatedly until you finally release the key

○ 1

○ 2 — once at the beginning and once when you release the key

○ 2 — once at the beginning and once when you release the key

1 point

9.

Several keys on the keyboard, such as Shift, CapsLock, and Ctrl, typically act to modify what happens when you press other keys, rather than doing anything on their own. When using the SimpleGUI keydown handler, how are such keys treated?

Experiment in CodeSkulptor to find out.

○ Modify other key presses — e.g., pressing the 'a' key creates an event with a different value than pressing Shift and 'a' together.

○ Independent key press events — e.g., pressing Shift by itself creates an event

○ No effect — e.g., pressing the Shift key does not create or modify the behavior of any event.

7 questions unanswered

Upgrade to submit