Task Descriptions

Tutorial Task

Follow the tutorial instructions here.

Instructions for a Task Sequence

Before a Task Sequence

Take 5 minutes to review the tutorial and the source code it's linked to. You will edit the tutorial soon, so familiarize yourself with the purpose of the tutorial, the source code on the left, the snippets, and the outputs. You can execute the source code if you'd like using the code editor's terminal (if you need help finding it, ask the experimenter).

To Complete a Task Sequence...

You will be given a list of 3 tasks. Complete as many as you can in 10 minutes.

For each task in the sequence, you will be asked to change the code in a tutorial.

For each task, modify the code to make the change described. You must make sure all of the outputs and text have been updated to reflect this change to the code. Once you're done with a task, say "I'm done!" so the experimenter can check your work. Then, continue to the next task.

Task Sequence A

Task 1: Make a global change to code

Change the value of the `shark` variable from "`Sammy`" to "`Jerry`". Make sure this change is reflected in every snippet, output, and text reference in the tutorial.

Task 2: Make a local change to code

Add a new snippet that shows that the console output changes when you update the `print` statement in the "hello" method. Add this snippet after the first snippet containing this code:

```
print('Hello, World!')
```

In the new snippet you add, change the message from `'Hello, World!'` to `'Hello, Earth!'`. Add an output that shows that shows that the print output has changed to "Hello, Earth!". This change should be *local*, i.e. it should have no effect on the outputs of previous snippets.

In the interest of time, you do *not* need to write any new text.

Task 3: Revert a local change to code

Add a new snippet, after the snippet you created in Task 2, that sets the print message back to 'Hello, World!'. Add an output that shows the output change back to 'Hello, World!'.

In the interest of time, you do *not* need to write any new text.

Task Sequence B

Task 1: Make a global change to code

For the object variable `pizza_01`, change the "`name`" parameter to be "`garden`", instead of "`artichoke`". Make sure this change is reflected in every snippet, output, and text reference in the tutorial.

Task 2: Make a local change to code

Add a new snippet that shows that no log messages are printed when the logging level is set to `logging.INFO`. Add this snippet after the first snippet containing this code.

```
logging.basicConfig(level=logging.DEBUG)
```

In the new snippet you add, change the logging level to `logging.INFO`. Add an output that shows that output messages no longer appear in the console output. This change should be *local*, i.e. it should have no effect on the outputs of previous snippets.

In the interest of time, you do *not* need to write any new text.

Task 3: Revert a local change to code

Add a new snippet, after the snippet you created in Task 2, that sets the logging level back to `logging.DEBUG`. Add an output that shows the log messages appear again.

In the interest of time, you do *not* need to write any new text.

Open-Ended Tutorial Creation Task

Write a tutorial introducing someone to some basics of object-oriented programming in Python.

You have been provided with the source code file `objects.py`. Assume this code is your, and now you are writing a tutorial about how to write that code.

You have 25 minutes. Since this is time-constrained, we don't expect the tutorial to be fully-polished and complete. Think about writing bullet points and ellipses as notes to yourself on what you would write, rather than perfecting the descriptions in the text.

At the end of the tutorial, you should have touched on:

- How do you declare a class
- How do you access the values of member variables
- How do you declare a class that inherits another one

For each of these steps, you should include some output that readers can use to verify their work.

The main purpose of this task is to get a sense of which features of Torii work well for you for creating tutorials, and what we should improve before making the tool more widely available.