

# Lab 5

## Slicing, Modules and Tuples

Z. Hutchinson  
zachary.s.hutchinson@maine.edu

October 11, 2022

### Instructions

Work is due by the end of your lab session and must be submitted to Brightspace in the proper place. Unless otherwise instructed, submissions must be python files (e.g. files that end with `.py`). Any other format, even if it is plain text, will **not** be graded. Messy or otherwise unreadable code will lose points. Lab submissions can be all in the same file, but please label with comments to which task code belongs. **IMPORTANT:** Any code that is commented out will not be graded. **RUN YOUR CODE TO MAKE SURE IT WORKS!!!**

### Task 1 - Slicing

This task involves slicing strings. For this task you will use the following code. Each subtask is independent.

```
quote = "Most of the good programmers do programming "+ \
        "not because they expect to get "+ \
        "paid or get adulation by the public, "+ \
        "but because it is fun to program."
```

#### A

From the string above slice and print the phrase, *good programmers*.

#### B

Slice the quote in half and print the first half. HINT: If you do it right, the last letter printed is the 'g' in 'get'.

#### C

Print the quote above with the last two words sliced off. However, keep the period. So the new quote should end with the word 'fun' followed by a period.

#### D

Split the string using the `split()` function. Split breaks the string wherever there is white space. It returns a list of sub-strings. Then slice and print every second word in the split quote.

#### E - EXTRA

Using slicing and loops, find and print all words that start with a 'p'.

## Task 2 - Modules

### A - Random randint

The random module contains a ton of functions to generate pseudo-random numbers or choices in code. Import the random module and use the `randint()` function to generate a random number between 1-10. [Documentation for randint\(\)](#)

### B - Random choice

Another good function in the random module is `choice()`. Choice takes a sequence (e.g., list or tuple) as an actual parameter and returns a randomly chosen element from it. Split the quote from Task 1 into a list of strings and then use `choice()` to randomly select one of the words from the quote. [Documentation for choice\(\)](#)

### C - Random shuffle

Another good one from random is `shuffle()`. As you might guess, this function randomly shuffles a sequence. Shuffle takes a sequence as an actual parameter and shuffles the sequence **in place**. In place means that it changes the list itself (it uses the same memory locations) and does not return a new list. [Documentation for shuffle\(\)](#)

### D - Others

There's nothing to code for this subtask. Spend some time looking at the module list: [List of Modules](#). It's a good idea to check here if you have to code something that seems like a common thing. Here are the first modules I started using (other than random):

- **math**: Sine, cosine, tangent, log and all that good stuff.
- **datetime**: When you need to keep track of dates/times in your program.
- **json**: Parsing/creating json files.
- **pickle**: Serialization. Serialization is the saving of the state of your program.
- **tkinter**: Make graphical user interfaces.

## Task 3 - Tuples

There's nothing to code for this section. Task 3 exists to tell you to remember the one big difference between a *list* and a *tuple*. Lists are mutable; they can be changed after they creation. Tuples are immutable; they cannot be changed.

Example: Put this list into your code file.

```
L = [6,7,8,9,10]
```

Now try to change the first element:

```
L[0]=27  
print(L)
```

It works because a list is mutable. It can be changed.

Now switch the square brackets to parenthesis and run it again.

```
L = (6,7,8,9,10)  
L[0]=27  
print(L)
```

It crashes. This is because tuples are immutable. They cannot be changed. If you want to change a tuple you must make a new tuple.

```
L = (6,7,8,9,10)
L = (27,) + L[1:]
print(L)
```

In the above we concatenate a slice of L with a new single element tuple containing 27. Note that you can slice a tuple. Slices work because they create new tuples (just like they create new lists). Also note, that to create a single element tuple, you must use parenthesis and a comma. If you just write, (27), it isn't a tuple, but a single integer.