Model 1 Working with Lists

Recall that a variable can hold multiple values in the form of a list. The values are separated by commas and wrapped in square brackets.

Lists have ***methods*** (built-in functions) that can be called using dot notation. For example, to add a new element to the end of a list, we can use the `append` method.

Python code	Shell output
`rolls = [4, 6, 6, 2, 6]`	
`len(rolls)`	
`print(rolls[5])`	
`rolls.append(1)`	
`print(rolls)`	
`print(rolls[5])`	
`lucky.append(1)`	
`lucky = []`	
`print(lucky[0])`	
`lucky.append(5)`	
`print(lucky)`	
`print(lucky[0])`	
`rolls.count(6)`	
`rolls.remove(6)`	
`print(rolls)`	
`help(rolls.remove)`	
`help(rolls)`	

Questions (15 min) Start time:

1. What is the result of calling the `append` method on a list?

2. What must be defined prior to using a method like `append`?

3. Explain why two lines in Model 1 caused an `IndexError`.

4. What is the result of calling the `remove` method on a list?

5. Based on the `help` output, name several list methods not shown in Model 1. Do not include methods that begin and end with two underscores (e.g., __add__).

6. Give one example of a list method that requires an argument and one that does not.

7. Describe the similarities and differences between using a list method like `append` and Python built-in functions like `print`.

8. Complete the function below (two lines are missing). It should prompt the user for numbers and build a list by adding one number at a time to the end of the list. The loop terminates when the user inputs the number 0.

```
def input_numbers():
    x = 1


    while x != 0:
        x = int(input("Enter the next number: "))


    return numbers
```

Model 2 Indexing and Slicing

A string is a sequence of characters in single quotes (') or double quotes ("). Depending on the application, we can treat a string as a single value (e.g., `dna`), or we can access individual characters using square brackets (e.g., `dna[0]`). We can also use ***slice notation*** (e.g., `dna[4:8]`) to refer to a range of characters. In fact, all types of sequences (including `list` and `tuple`) support indexing and slicing.

Python code	Shell output
`dna = 'CTGACGACTT'`	
`dna[5]`	
`dna[10]`	
`len(dna)`	
`dna[:5]`	
`dna[5:]`	
`dna[5:10]`	
`triplet = dna[2:5]`	
`print(triplet)`	
`dna[-5]`	
`dna[-10]`	
`dna[:-5]`	
`dna[-5:]`	
`triplet = dna[-4:-1]`	
`print(triplet)`	

Questions (15 min) Start time:

9. What is the *positive* index of each character in the `dna` string? Check your answers above.

Character:	C	T	G	A	C	G	A	C	T	T
Index:										

10. What is the *negative* index of each character in the `dna` string? Check your answers above.

Character:	C	T	G	A	C	G	A	C	T	T
Index:										

11. Based on the previous questions, what are `dna[2]` and `dna[-2]`? Explain your answers.

12. Explain the IndexError you observed. What is the range of indexes for the `dna` string?

13. Consider the notation of the operator `[m:n]` for slicing the string.

a) Is the value at the start of the resulting string the same as the value at index `m` (i.e., `dna[m]`)? If not, describe what it is.

b) Is the value at the end of the resulting string the same as the value at index `n` (i.e., `dna[n]`)? If not, describe what it is.

c) Explain what it means when only a single number is referenced when creating a slice, such as `[m:]` or `[:n]`.

14. What is the simplest way to get the first three characters of `dna`? What is the simplest way to get the last three characters?

15. Write a Python expression that slices `'GACT'` from `dna` using positive indexes. Then write another expression that slices the same string using negative indexes.

16. Write a Python assignment statement that uses the `len` function to assign the last letter of `dna` to the variable `last`.

17. Write a Python assignment statement that uses a negative index to assign the last letter of `dna` to the variable `last`.

Model 3 Lists of Lists

Connect Four (® Hasbro, Inc.) is a two-player game in which the players take turns dropping colored discs into a six-row by seven-column grid. The objective of the game is to be the first player to form a horizontal, vertical, or diagonal line of four of one's own discs. (paraphrased from https://en.wikipedia.org/wiki/Connect_Four)


```
# current state of the game
grid = [
    [' ', ' ', ' ', ' ', ' ', ' ', ' '],
    [' ', ' ', ' ', ' ', ' ', ' ', ' '],
    ['Y', ' ', ' ', ' ', 'Y', 'Y', ' '],
    ['R', ' ', ' ', 'Y', 'R', 'R', ' '],
    ['R', 'R', 'Y', 'R', 'Y', 'R', ' '],
    ['R', 'Y', 'R', 'Y', 'Y', 'Y', 'R'],
]
```

Enter the `grid` code above into a Python Shell, and run each line of the table below. If the output is longer than one line, summarize it with a few words.

Python code	Shell output
`print(grid)`	
`print(grid[5])`	
`print(grid[5][0])`	
`type(grid)`	
`type(grid[5])`	
`type(grid[5][0])`	
`len(grid)`	
`len(grid[5])`	
`len(grid[5][0])`	
`import pprint`	
`help(pprint)`	
`pprint.pprint(grid)`	
`for item in grid:` `  print(item)`	
`for i in range(len(grid)):` `  print(grid[i])`	

Questions (15 min) Start time:

18. What does `grid` look like when you first `print` it? (How is the output different from the original format shown in Model 3?)

19. What does `grid` look like when you use `pprint` instead? Explain what `pprint` means.

20. When viewed as a rectangle, how many "rows" and "columns" does `grid` have?

21. What type of object is `grid`? What type of objects does it contain?

22. What type of object is `grid[5]`? What type of objects does it contain?

23. In the expression `grid[5][0]`, which index corresponds to the row, and which index corresponds to the column?

24. Is `grid` a list of rows or a list of columns? Justify your answer.

25. Describe how to append one more row to `grid`.

26. What is necessary to append a "column" to `grid`?

Model 4 Nested Dictionaries

Containers can be nested in arbitrary ways. For example, the following data could be described as a "dictionary of dictionaries of integers and lists of strings".

Enter the following code into a Python Shell, and complete the table. If the output is longer than one line, summarize it with a few words.

```
movies = {
    "Casablanca": {
        "year": 1942,
        "genres": ["Drama", "Romance", "War"],
    },
    "Star Wars": {
        "year": 1977,
        "genres": ["Action", "Adventure", "Fantasy"],
    },
    "Groundhog Day": {
        "year": 1993,
        "genres": ["Comedy", "Fantasy", "Romance"],
    },
}
```

Questions (15 min) Start time:

27. Explain the `TypeError` you encountered.

28. In the expression `movies["Casablanca"]["genres"]`, describe the purpose of the strings `"Casablanca"` and `"genres"`.

Python code	Shell output
`movies`	
`movies["Casablanca"]`	
`movies["Casablanca"]["year"]`	
`movies["Casablanca"]["genres"]`	
`type(movies)`	
`type(movies["Casablanca"])`	
`type(movies["Casablanca"]["year"])`	
`type(movies["Casablanca"]["genres"])`	
`len(movies)`	
`len(movies["Casablanca"])`	
`len(movies["Casablanca"]["year"])`	
`len(movies["Casablanca"]["genres"])`	
`for key in movies:` `  print(key)`	
`for key, val in movies.items():` `  print(key, val)`	

29. When iterating a dictionary using a `for` loop (i.e., `for x in movies`), what gets assigned to the variable?

30. What is wrong with the following code that attempts to `print` each movie?

```
for i in range(len(movies)):
    print(movies[i])
```

31. Write nested loops that output every *genre* found under the `movies` dictionary. You should have nine total lines of output.

32. Each movie in Model 4 has a title, a year, and three genres.

a) Is it necessary that all movies have the same format?

b) Name one advantage of storing data in the same format:

c) Show how you would represent The LEGO Movie (2014) with a runtime of 100 min and the plot keywords “construction worker” and “good cop bad cop”.