This homework practices some of the fundamental Python programming concepts that will be used throughout the course. Download and run the file `HW2.py`; see the handouts in Lab2 if you need help with this. Be sure to cite any **collaborators** you worked with. Write the estimated **amount of time** you spent at the bottom of the file.

1 Manipulating Numbers

There are nine lines of code in `HW2.py`, labeled `Line A` through `Line I`. Modify the existing strings with the following information.

1. Determine which lines are expressions and which lines are assignments. Replace `<STRING>` with the string `Assignment` or `Expression`.
2. Determine the values of `x`, `y`, and `z` after each line is executed. I have provided the first two lines and the last line. Replace `<VALS>` with the strings denoting the values; be careful about integers (e.g. `10`) vs. floats (e.g. `10.0`).

2 Lists

1. The variable `numList` is a list of four integers. Write expressions within the `print()` functions **using only operators and indices into numList** (`numList[0]`, `numList[1]`, `numList[2]`, or `numList[3]`)) to print the values in the variable name.
2. The variable `mySchedule` is a list of strings. Write the classes you are currently taking as strings in the list.
   (a) Print the length of `mySchedule`.
   (b) Using indexes, print the first and last items of `mySchedule`.
   (c) Try the line

   ```
   print(mySchedule[len(mySchedule)])
   ```

   What happens? Why? Write your answer (including any error you may get) in the comments. You can then add a pound sign (#) at the beginning of the line to "comment it out."
3. The `range()` function takes an integer and returns (something like) a list of integers, starting from 0 and counting up to **but not including** that integer.
   (a) Evaluate `range(5)`, `range(10)`, and `range(1)`. In Python3, you need to tell Python 3 to evaluate `range()` like a list:

   ```
   print(list(range(5)))
   print(list(range(10)))
   print(list(range(1)))
   ```

   (b) Use the `range()` function to print **all the indices** of the list `mySchedule`. This output should change if the number of elements in `mySchedule` changes.

3 FOR Loops

1. Recall that a `FOR` loop has the following syntax (to print the elements in `numList`):

   ```
   for element in numList:
       print('Element:',element)
   ```

2. Using a `FOR` loop, print the elements in `mySchedule`.
3. Suppose we want to print the index, the value *and* the length of that element.
   (a) Use a `FOR` loop and the `range()` function to print **all the indices** of the list `mySchedule`.
   (b) Modify your code to also print the **value** and the **length** of the class in addition to the index. An example printed line will look like:
   `The class at index 0 is bio131 with length 6.`
   Hint: You can combine different types within a `print()` function with a comma. Look at other print functions in this HW.
4. You have already computed the number of elements in `mySchedule` using the `len()` function. Now, create a variable called `numClasses` and set it to `0`. Use a `FOR` loop to count the number of elements in `mySchedule` by adding `1` to `numClasses` for each element. Print the value stored in `numClasses` to the screen.

4 Write a Change Counter

Suppose you have two lists that each contain four values. The list `coinDenominations` contains the values of a penny, a nickel, a dime, and a quarter (e.g., `[1,5,10,25]`). You reach into your pocket and pull out a bunch of change. `numberOfCoins` contains the number of pennies, nickels, dimes, and quarters in your hand (e.g., `[3,1,0,2]`). Use a `FOR` loop to count the amount of change (in dollars) in your pocket (e.g., `$0.58`). Print the final amount to the screen. You can assume that both lists contain the same *order* of coins (penny, nickel, dime, quarter).
Hint: First use a `FOR` loop to print the denomination & number of each coin on a single line (one line for each coin).

5 String Slices

We have learned how to get a single character in a string. To extract a *substring* of the string (a "chunk" of the string), we can use Python slices. Given a string `myString` and two integers `i` and `j` where `i<j`, writing `myString[i:j]` returns a string starting at the `i`th index in `myString` up to **but not including** the `j`th index. Evaluate each of the following lines.

```
stringOfNumbers = '0123456789'
print(stringOfNumbers)
print(stringOfNumbers[5])
print(stringOfNumbers[3:5])
print(stringOfNumbers[3:6])
print(stringOfNumbers[:5])
print(stringOfNumbers[5:])
print(stringOfNumbers[:5] + '*' + stringOfNumbers[5:])
```

6 Hypothetical Gene

In the variable `hypotheticalgene`, the character `'e'` stands for a nucleotide in an exon and the character `'i'` stands for a nucleotide in an intron. The variables `exon1start` and `exon2start` contain the start indexes of the exons. The variable `intron1start` contains the start index of the intron. The variable `genelen` contains the length of the hypothetical gene.

1. Use the indices to print the length of the two exons.
2. Use Python slices to store strings that correspond to the exons and the intron in variables called `exon1`, `exon2`, and `intron`.
3. Print (a) the length of `hypotheticalgene` and (b) the sum of the lengths of `exon1`, `exon2`, and `intron`. The two values should be the same.

7 Extra Exercises (Optional)

Exon starts and intron starts as lists. Suppose that the exon and intron positions are stored in two lists. Write this in your file:

```
exonstarts = [exon1start,exon2start]
intronstarts = [intron1start, genelen]
```

Use a `FOR` loop to print the strings for exon1 and exon2.

SRC Gene. The SRC gene is located on human Chromosome 20. The name, SRC, is pronounced "sarc" and it is short for "sarcoma," a certain class of cancers. SRC is an oncogene, which means it has the potential to cause cancer if it is mutated or the c-Src protein it encodes is expressed at high levels.

Let's consider the exon and intron boundaries for the SRC gene. The exon and intron start positions are located in the `SRCexonstarts` and `SRCintronstarts` lists. The last number in `SRCintronstarts` denotes one position beyond the last exon.

1. Print the number of exons listed for SRC.
2. Print the average exon length.
3. Print the average intron length.
4. Print the variance in exon lengths.
5. Print the variance in intron lengths.
6. From these numbers, what do you notice about exons vs. introns? Write your answer in the comments.