MISE Summer Programming Camp

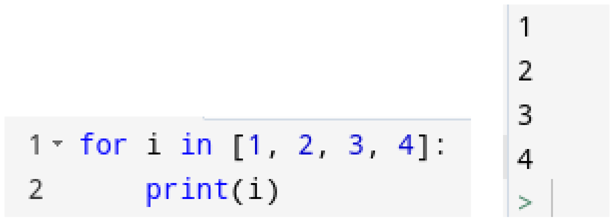
Class 5 Worksheet - Lists of Lists

This worksheet has 3 sections, which should take you around 1 hour to complete. Section 1 will summarize all we know about lists. Section 2 is optional and it asks you to write a simple app to simulate hospital patient processing. Finally on Section 3 you'll have to write some programs and submit them to CodeForces.

1. Review Questions

We learned how to create and manipulate a list in different ways over the first 5 classes. Let's review some of the trickiest concepts.

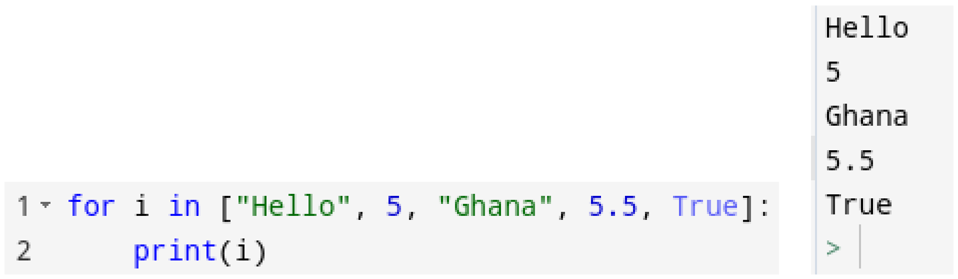
First of all, we've hinted at this a couple of times, but the **range** function is nothing but a function that returns a list. In fact, one of the questions from last week asked you to implement something like a range. In particular, the **for** loop as we learned it actually loops through a list and doesn't need to be through a range. Look at the following example:



```python
for i in [1, 2, 3, 4]:
    print(i)
```

```
1
2
3
4
>
```

The body of the **for** loop repeats 4 times and the variable *i* takes the value of 1 through 4, the elements of the list. We can use any list we want in a **for** loop and Python will loop through its elements:



```python
for i in ["Hello", 5, "Ghana", 5.5, True]:
    print(i)
```

```
Hello
5
Ghana
5.5
True
>
```

Similarly, we can use the for notation to define new lists using something called *list comprehension*. The syntax for this is the following:

```
[ <expression> for item in list if <conditional> ]
```

Here is an example of the above:

```python
l = [x for x in range(1,10) if x % 2 == 0]
print(l)
```

```
[2, 4, 6, 8]
>
```

The way to think about the above is the following: *l* is the list formed of all the elements in the range from 1 to 10 for which the condition `x % 2 == 0` is satisfied, which is the same as all the even numbers between 1 and 9. Here is another example:

```python
l = [(i + 4) / 2 for i in range(8)]
print(l)
```

```
[2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5]
>
```

Here we modify the value of each of the elements in the range by summing 4 and dividing by 2. We can use list comprehension with all data types, for example, we can use it to define a list of lists (or a 2d list) like the following:

```python
l = [[j for j in range(3)] for i in range(4)]
print(l)
```

```
[[0, 1, 2], [0, 1, 2], [0, 1, 2], [0, 1, 2]]
>
```

This is a very useful way to write lists in Python, without having to use lots of **for** loops.

Question 1. For each of the following, write a code that creates a list (potentially using list comprehension) that satisfies the prompt.

a. A list containing the number one.
   Solution: *lst = [1]*
b. A list with one hundred 1s.
c. A list with all even numbers between 1 and 29.
d. A list with all numbers between 1 and 100 except numbers that start with the digit 5.
e. Given a parameter (input) *n*, a list with one 1, followed by two 2s, followed by three 3s… up to *n* (for example, if *n = 3* the solution would be *lst = [1, 2, 2, 3, 3, 3]*) .

2. Writing a Simple App (Optional)

Question 1. In this question we are going to pretend to work at a hospital and we are going to implement a toy patient data processor. We will guide you through the process of building this app. In short, your program should first read a table of patient data, then it should read an option from a menu and according to the option it should display some info about the patients.

a. Let's start by reading patient data. There are **n** patients in this hospital (up to 100). For each one we know 4 things: name (a string with no whitespace); age (an integer between 1 and 100); weight (an integer between 1 and 200); height (an integer between 1 and 250). Your program should first read **n**. Then it should read **n** lines, each of which represents one patient. The format of each line is: <name> <age> <weight> <height>. Here is an example of a possible patient data:
   ```
   3
   Koko 29 55 170
   Panyin 60 86 190
   Ata 5 25 120
   ```
   You should store each patient as a tuple with 4 entries (name, age, weight and height) and store each tuple in a list, in the order of the input. All names will be distinct, so there are no duplicate names. Hint: you will need to use nested loops and lists of tuples. You'll also need to use the **split** function that was mentioned in the last worksheet.
b. After the patient data, your program should display the following text, which represents a menu of options:
   ```
   Ghana Hospital App - Choose one option
   1 - Calculate average age
   2 - Display patient i
   3 - Find patient height
   ```
   After you print this text, your program should read one integer from the input. According to the result it should perform one of the three mentioned operations. If it is any other integer your program should do nothing. We will implement these below.
   i. *1 - Calculate average age*. For this option your program should look at the ages of each patient, calculate the average of all of them and then print the result.
   ii. *2 - Display patient i*. For this option your program should output the information about the patient of index *i*. To do so, it should first print `Select a patient index` and then read an integer *i* from the input, which will be between 1 and the number of patients. It should then print the information of the *i*th patient in the following format: `Patient <name> is <age> years old and weighs <weight>kg and is <height>m tall.`
   iii. *3 - Find patient height.* For this option your program should read the name of a patient and find the height of the patient with that name. It should first print `Select a patient name` and then read a string from the input. It should then find the patient with that name and print its height.
c. To finish your whole program you should put all of the above steps together (if you haven't already). Test your program with different options and patient data and make sure it works like intended. You will also get to submit this to CodeForces (the link to this class' problem set is below).

3. Coding Questions

You should see a new problem set called “Class 5 Problems” (https://codeforces.com/group/K1Fxw6skwV/contest/445808). Only the first problem is mandatory (“Shrubs”). The next two problems (“Card Game” and “Hospital App”) are extra. The last problem (“X-Sum”) is a challenge for those that want to try something harder.