Objectives

- Practice writing Python code in a text editor
- Practice running Python code from a command line
- Practice basic I/O using `input()`, `print()`, and formatting
- Practice with python variables and data types

Getting Started

First, do the attendance quiz as soon as you arrive. Make sure to submit the whole quiz after getting the question "correct". You will have to check in with the TA at the end of the tutorial before you leave as well in order to get your tutorial mark.

Note: It is always a good idea to save any code you may work on. You may find that something you need to do in the future is somewhat similar to the work you will do here.

Problem 1 (Age Calculation)

When it is evaluated, the `input` function displays a prompt and reads text from the console (i.e., typed in using the keyboard). When executed, the following program code will prompt a user to enter their name, assign the input string to the variable called `name`, and then print it out to the screen.

```python
name = input("Enter your name : ")
print("Hello, " + name)
```

Write a simple program that prompts a user for their name and birth year. The program will then display (output) a greeting to the user (with their name) and say how old the person is. For example, when the program is executed, the output may look like the following (what the user types in with the keyboard is highlighted):

```python
name = input("Enter your name : ")
print("Hello, " + name)
```
Enter your name: Bob
What year were you born, Bob? 1998
Hello, Bob. You are now 20 years old.

Note: The input() functions returns the input as a string. In order to convert the string into an integer use the int() function. For example, if \( x = "12" \), then \( \text{int}(x) \) is the integer 12. Also recall that the str() function will convert a number to a string, which is useful when trying to print a mix of numbers and text.

**Problem 2 (Multiplication Table)**

Write a program that asks the user for a number, and then displays a well-formatted multiplication table for that number.

Please enter a number: 8

<table>
<thead>
<tr>
<th>Factor</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>6</td>
<td>48</td>
</tr>
<tr>
<td>7</td>
<td>56</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
</tr>
<tr>
<td>9</td>
<td>72</td>
</tr>
</tbody>
</table>

**Problem 3 (Duration Conversion)**

Write a program that asks the user for an arbitrary number of days, and then determines three equivalent lengths of time – as measured in years, months, and weeks. You may ignore the leap-year correction and assume that 1 year = 365 days, and for the purposes of this exercise, assume that 1 month = 30 days.

How many days? 600

600 days is equivalent to 85.71428571428571 weeks.
600 days is equivalent to 20.0 months.
600 days is equivalent to 1.64385616438356 years.
Problem 4 (Duration Conversion – Part 2)

Modify the program you created for Problem 3 so that it asks the user for an arbitrary number of days and then converts that length of time into a combination of years, months, weeks, and days. To clarify, once the user has provided the number of days, determine the number of full years (i.e., 365 days) that would have elapsed in that time, followed by the number of full months (i.e., 30 days), the number of full weeks (i.e., 7 days), and the number of days remaining. Display the result of these calculations to the user on a single line.

How many days? 600
600 days is 1 years, 7 months, 3 weeks, and 4 days.

[Bonus] Problem 5 (Making Change)

Write a program that asks the user for two floating point numbers, representing the price of a bill, and the amount paid for that bill. Your program should then calculate and print out the required number of each of the following denominations in order to make change: Dollars, Quarters, Dimes, Nickels, and Pennies.

Enter the bill amount: 3.13
Enter the amount paid: 8.60

Your change is $5.47.
That's 5 dollars, 1 quarters, 2 dimes, 0 nickels, and 1 pennies.

Note: when working with floating point numbers you are likely to have some rounding errors. This may mean that your solution works for some numbers, but is off by a penny for others. Consider a second bonus problem: How can you solve this problem without rounding errors?