# W2. Lists and strings

This week you will gain experience slicing and interacting with Python sequential data.
You will submit results of your experiments in file *w2.py*.

## Step 1. Try strings and lists in IDLE shell

```
>>> alist = [0, 1, 2, 3]
You can label data (here, a list) with a name (here, the name alist)
(no response from Python)

>>> alist
[0, 1, 2, 3]
You can see the data (here, a list) referred to by a name (here, alist)

>>> alist[1:]
[1, 2, 3]
You can slice lists (here, using the name alist)

>>> alist[::-1]
[3, 2, 1, 0]
You can reverse lists (or strings!) using "skip"-slicing with a -1 as the amount to skip.

>>> [0, 1, 2, 3][1:]
[1, 2, 3]
You can slice lists using the raw list instead of the name you gave the list.
(Not that this would be very useful, admittedly!)

>>> 100*alist + [42]*100
(a list with 500 elements that I'm too lazy to type here)

>>> alist = 42
You can reassign the name alist to another value, even of a different type—now, alist names the integer 42, instead of the list it used to represent.
(no response from Python)

>>> alist
42
```

## Step 2. Explore Errors or Exceptions

If you didn't type things in perfectly, Python will reply with an error or an *exception*, as it is often called. See if you can make Python create the following exceptions, but don't spend more than a minute or so in total!

SyntaxError
TypeError (try slicing an integer for example!)
ZeroDivisionError
IndexError (try an out-of-bounds index)
OverflowError

(to obtain this error you'll need to use floating-point values in some large mathematical expression!)

## Step 3. Slicing and indexing… Lists!

This problem will exercise your slicing-and-indexing skills. First, **use the File->New Window menu options to create an editor window. You will use this window to create your *w2lists.py* file. To do:** Copy the following lines into the editor window:

```
'''Your name
w2lists.py slicing and indexing challenge
'''
pi = [3, 1, 4, 1, 5, 9]
e = [2, 7, 1]
```

**Fill in the appropriate information at the top of your file.** Then save the contents under the name *w2lists.py*. Now, when you press **F5**, these two lists will be recognized by the Python shell.

The challenge is to create several lists using **only** the list labeled pi, the list labeled e, and the four list operations here:

- list indexing such as pi[0]
- list slicing such as e[1:]
- list concatenation (+), such as pi[:1] + e[1:]  (do *not* use + to add values numerically)
- the list-making operator ( [ , ]) -  for example:  [e[2], e[0]]

**For each one, place your answer into an appropriate variable in your *w2lists.py* file as you see in the example below.** Include a comment on the line above and a print statement on the line below. That way, each time you use F5 to reload the file, the results will print — it's much easier to check that way!

Though not mandatory, you might try to use **as few operations as possible**.

**Example problem:**

- Use pi and/or e to create the list [2, 5, 9]. Store this list in the variable answer0.

**Answer to the example problem**

```
# Creating the list [2, 5, 9] from pi and e
answer0 = [e[0]] + pi[-2:]    # adds the lists:  [2] + [5, 9]
print(answer0)
```

## The challenges

1. Use pi and/or e to create the list [7, 1]. Store this list in the variable answer1.
2. Use pi and/or e to create the list [9, 1, 1]. Store this list in the variable answer2.
3. Use pi and/or e to create the list [1, 4, 1, 5, 9]. Store this list in the variable answer3.
4. Use pi and/or e to create the list [1, 2, 3, 4, 5]. Store this list in the variable answer4.

## Step 4. Practicing with strings

This problem continues in the style of the last one, but uses a single string rather than lists. So, these challenges ask you to create specified strings that result from using *only* the following saying, which you should copy into your *w2strings.py* file at this point:

```
# starting string
s = 'all day i dream about sports'
```

You may use any combination of these four string operators:

- String indexing, e.g. s[0]
- String slicing, e.g. s[1:3]
- String concatenation (+), e.g., s + s
- Repetition (*), e.g., 42*s

Again, **less is more**! However, **any correct answer is OK**.

## The challenges

**Using *s*:**

5. Create string ‘sos’. Store this string in the variable answer5.
6. Create ‘adidas’. Store this string in the variable answer6.
7. Create ‘store’. Store this string in the variable answer7.
8. Create ‘most’. Store this string in the variable answer8.

Submit the results to the Google classroom.