Lab 2 Solutions

Summer Session A, 2023, Ethan M.

July 1, 2023

1 Task 1

```
[1]: list1 = [1, "hi", 3.4, "PSTAT 5A"]
```

```
[2]: type(list1)
```

```
[2]: list
```

2 Task 2

```
[3]: ## part (a)
     x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
[4]: ## part (b)
     x[1]
```

```
[4]: 2
```

```
[5]: ## part (c)
     x[0]
```

```
[5]: 1
```

3 Task 3

```
[6]: x = [1, "two", 3.5, "four", "five five"]
```

Part (a): Should be `list`, which is the data *class* of the variable `x`.

Part (b): Should be `str`, which is the data *type* of the quantity `"two"`.

Part (c): Should be `int`, which is the data *type* of the quantity `1`.

```
[7]: type(x)
```

```
[7]: list
```

```
[8]: type(x[1])
```

```
[8]: str
```

```
[9]: type(x[0])
```

```
[9]: int
```

4 Task 4

Answers may vary.

5 Task 5

```
[10]: from datascience import *
```

```
[11]: ## part (a)
      profs = Table().with_columns(
          "Professor", ["Dr. Swenson", "Dr. Wainwright", "Dr. Mouti"],
          "Office", ["South Hall", "Old Gym", "Old Gym"],
          "Course", ["PSTAT 130", "PSTAT 120A", "PSTAT 126"]
      )
```

```
[12]: ## part (b)
      profs
```

```
[12]: Professor      | Office     | Course
      Dr. Swenson    | South Hall | PSTAT 130
      Dr. Wainwright | Old Gym    | PSTAT 120A
      Dr. Mouti      | Old Gym    | PSTAT 126
```

```
[13]: ## part (b)
      profs.column("Course")
```

```
[13]: array(['PSTAT 130', 'PSTAT 120A', 'PSTAT 126'],
            dtype='<U10')
```

```
[14]: profs_new = profs.with_row(
          ["Dr. Ravat", "South Hall", "PSTAT 120B"]
      )
```

```
[15]: profs_new
```

```
[15]: Professor      | Office     | Course
      Dr. Swenson    | South Hall | PSTAT 130
      Dr. Wainwright | Old Gym    | PSTAT 120A
      Dr. Mouti       | Old Gym    | PSTAT 126
      Dr. Ravat       | South Hall | PSTAT 120B
```

6 Task 6

[16]:

```
my_list = [1, 2, 3]
my_array = make_array(1, 2, 3)
```

[17]:

```
my_array
```

[17]:

```
array([1, 2, 3])
```

[18]:

```
sum(my_list)
```

[18]:

```
6
```

[19]:

```
sum(my_array)
```

[19]:

```
6
```

[20]:

```
my_list + 2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[20], line 1
----> 1 my_list + 2

TypeError: can only concatenate list (not "int") to list
```

[21]:

```
my_array + 2
```

[21]:

```
array([3, 4, 5])
```

7 Task 7

[22]:

```
"statistics" > "Statistics"
```

[22]:

```
True
```

We see that Python gives a higher value to lowercase letters than uppercase letters.

8 Task 8

[23]:

```
x = make_array(1, 2, 3)
y = make_array(2, 3, 1)

x < y
```

[23]:

```
array([ True,  True, False], dtype=bool)
```

Indeed, we see that Python has performed element-wise comparisons.

9 Task 9

First, `x` is assigned the value `2`. Then, Python checks whether `x` is less than `2`. Since `2` is not less than `2`, Python then checks whether `x` is less than `3`. Since `2` **is** less than `3`, Python assigns `x` the value `"goodbye"` and stops running any further code in the conditional expression.

[24]:

```
x = 2

if x < 2:
    x = "hello"
elif x < 3:
    x = "goodbye"
else:
    x = "take care"
```

[25]:

```
x
```

[25]:

```
'goodbye'
```

10 Task 10

Answers will vary.

11 Task 11

[26]:

```
def cent_to_far(x):
    """
    returns the corresponding temperature in Fahrenheit of a temperature of x degrees Centigrade
    """

    return (5/9) * x + 32
```

[27]:

```
cent_to_far(68)
```

[27]:

```
69.77777777777777
```

12 Task 12

The key is to note that, if `x` is even, then `x % 2` will be zero. Conversely, if `x` is odd, then `x % 2` will be one.

[28]:

```
def parity(x):
    """
    returns the parity of the number x
    """

    if x % 2 == 0:
        return "even"
    else:
        return "odd"
```

[29]:

```
parity(2)
```

[29]:

```
'even'
```

[30]:

```
parity(3)
```

[30]:

```
'odd'
```