

STAT540: Computing in Statistics

Basic Python Objects and NumPy Arrays

Yen-Yi Ho

Department of Statistics, University of South Carolina

Section 1

Lists and Indexing

Question 1

```
x = [5, 2, 3]  
x[0]
```

What does this return?

A. 2

B. 5

C. 3

D. An error

Question 1

```
x = [5, 2, 3]  
x[0]
```

What does this return?

A. 2

B. 5

C. 3

D. An error

Answer: B.

Python indexes from **0**, not 1 as in R — so `x[0]` is the *first* element. The habit to unlearn: in R `x[1]` is the first element; in Python it is the second. A negative index counts from the **end** (`x[-1]` is 3); it does not *drop* an entry as it does in R.

Question 2

```
x = [5, 2, 3]  
x * 2
```

What does this return?

- A. [10, 4, 6]
- B. [5, 2, 3, 5, 2, 3]
- C. [[5, 2, 3], [5, 2, 3]]
- D. An error — you cannot multiply a list

Question 2

```
x = [5, 2, 3]
x * 2
```

What does this return?

- A. [10, 4, 6]
- B. [5, 2, 3, 5, 2, 3]
- C. [[5, 2, 3], [5, 2, 3]]
- D. An error — you cannot multiply a list

Answer: B.

On a **list**, `*` *repeats*; it is not entrywise arithmetic. This is one of several signs that lists are not built for numerical computing — that is what **NumPy arrays** are for, where `np.array([5,2,3]) * 2` does give `[10 4 6]`.

Section 2

Tuples and Dictionaries

Question 3

Which object cannot be edited after it is created?

- A. `x = [5, 2, 3]`
- B. `salutations = ("hey", "hi", "ahoy", "sup")`
- C. `stat_540 = {'nstudents': 29}`
- D. All three can be edited

Question 3

Which object cannot be edited after it is created?

- A. `x = [5, 2, 3]`
- B. `salutations = ("hey", "hi", "ahoy", "sup")`
- C. `stat_540 = {'nstudents': 29}`
- D. All three can be edited

Answer: B — the tuple.

Parentheses build a **tuple**, which is **immutable**: assigning into one raises a `TypeError`. Lists (brackets) and dictionaries (braces) are both mutable. Immutability buys memory efficiency, and we will mostly meet tuples as *arguments* — e.g. the shape in `np.ones((3, 5))`.

Question 4

```
stat_540 = {'nstudents' : 29,  
           'time'       : '2:20 - 3:35 pm',  
           'days'      : ['Tue', 'Thu'],  
           'instructor' : 'Dr. Ho'}
```

Which line returns 'Dr. Ho'?

- A. `stat_540[3]`
- B. `stat_540['Dr. Ho']`
- C. `stat_540['instructor']`
- D. `stat_540.instructor[0]`

Question 4

```
stat_540 = {'nstudents' : 29,  
           'time'       : '2:20 - 3:35 pm',  
           'days'      : ['Tue', 'Thu'],  
           'instructor' : 'Dr. Ho'}
```

Which line returns 'Dr. Ho'?

- A. `stat_540[3]`
- B. `stat_540['Dr. Ho']`
- C. `stat_540['instructor']`
- D. `stat_540.instructor[0]`

Answer: C.

A dictionary is accessed **by key**, not by position: **A** raises a `KeyError` (there is no key 3), **B** looks up a *value* as if it were a key, and **D** uses attribute syntax dictionaries do not have. Assignment by key edits in place: `stat_540['instructor'] = 'Dr. Ho'`.

Section 3

NumPy Arrays

Question 5

```
D = rng.random((20, 3))  
D.sum(axis=0)
```

What does this return?

- A. One number — the sum of all 60 entries
- B. 3 numbers — one per **column**
- C. 20 numbers — one per **row**
- D. A 20×3 array of running sums

Question 5

```
D = rng.random((20, 3))  
D.sum(axis=0)
```

What does this return?

- A. One number — the sum of all 60 entries
- B. 3 numbers — one per **column**
- C. 20 numbers — one per **row**
- D. A 20×3 array of running sums

Answer: B.

`axis=0` names the axis that gets **collapsed**, so it operates *down* the columns: shape $(20, 3)$ with axis 0 removed is $(3,)$. Symmetrically `axis=1` runs *across* rows and returns 20 numbers; omitting `axis` collapses everything to one number (**A**).

Q	Idea	Answer
1	Indexing starts at 0	B
2	* repeats a list, it does not multiply it	B
3	Tuples are immutable	B
4	Dictionaries are keyed, not positional	C
5	axis=0 collapses <i>down</i> the columns	B

Lists hold things; NumPy arrays compute with them.