28th April 21

Previous Day

- lists()
- split ()
- indexing
- append()
- extend()
- pop()
- remove()
- len()
- nested list
- insert()
- MCQs

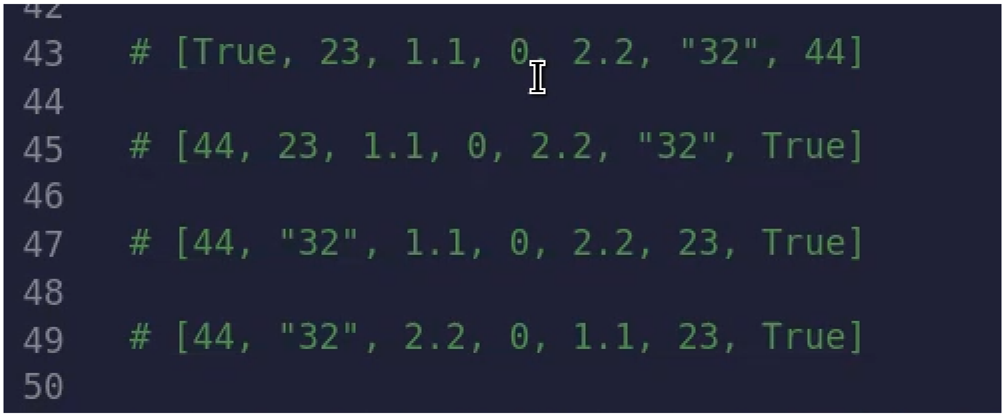
Python Tuple and Dictionary

Lecture Flow

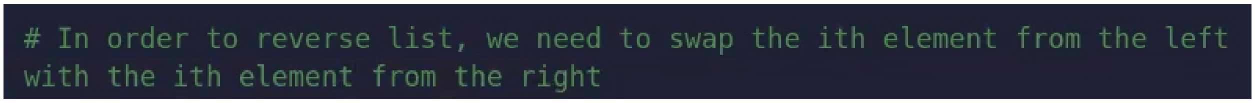
- reverse()
- program
- tuple
- mcqs

Topic and Explanation

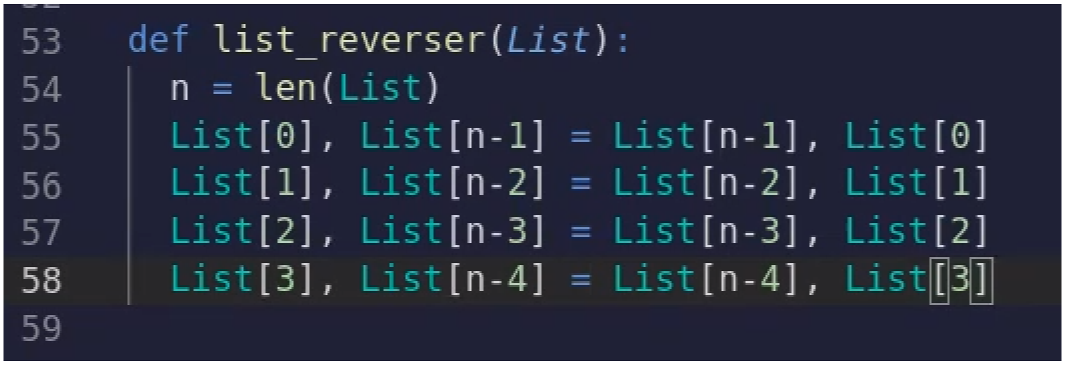
Reverse()



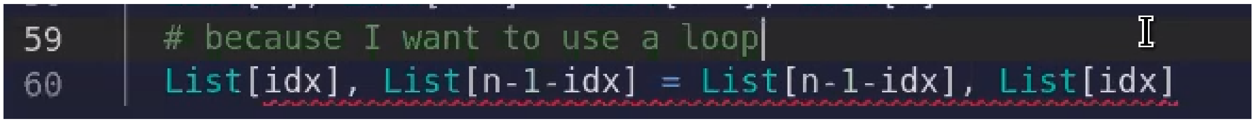
```
# [True, 23, 1.1, 0, 2.2, "32", 44]

# [44, 23, 1.1, 0, 2.2, "32", True]

# [44, "32", 1.1, 0, 2.2, 23, True]

# [44, "32", 2.2, 0, 1.1, 23, True]

```



```
# In order to reverse list, we need to swap the ith element from the left
with the ith element from the right
```

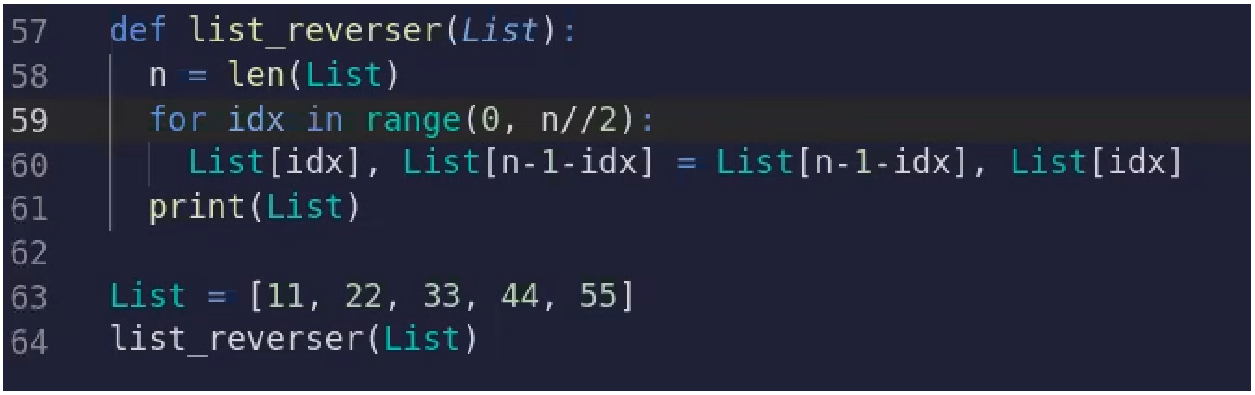


```
def list_reverser(List):
  n = len(List)
  List[0], List[n-1] = List[n-1], List[0]
  List[1], List[n-2] = List[n-2], List[1]
  List[2], List[n-3] = List[n-3], List[2]
  List[3], List[n-4] = List[n-4], List[3]

```



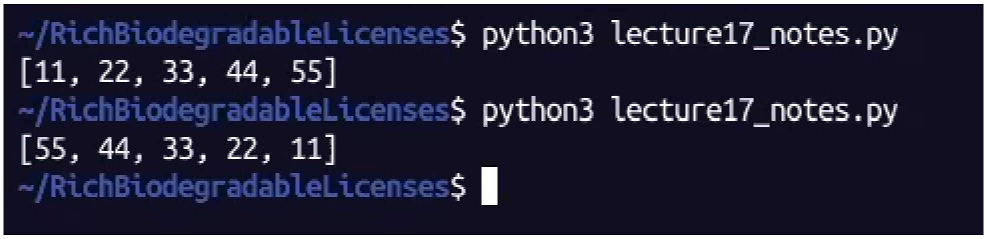
```
  # because I want to use a loop
  List[idx], List[n-1-idx] = List[n-1-idx], List[idx]
```

Program

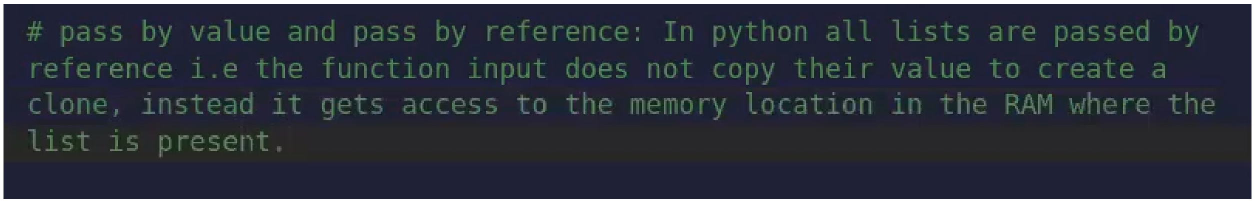


```python
def list_reverser(List):
  n = len(List)
  for idx in range(0, n//2):
    List[idx], List[n-1-idx] = List[n-1-idx], List[idx]
  print(List)

List = [11, 22, 33, 44, 55]
list_reverser(List)
```

Output

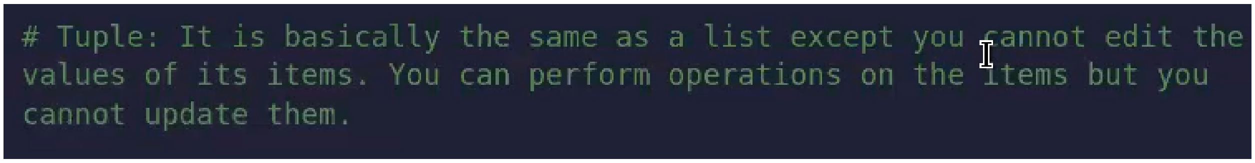


```
~/RichBiodegradableLicenses$ python3 lecture17_notes.py
[11, 22, 33, 44, 55]
~/RichBiodegradableLicenses$ python3 lecture17_notes.py
[55, 44, 33, 22, 11]
~/RichBiodegradableLicenses$
```

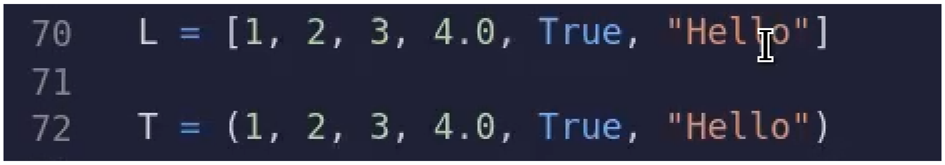


```
# pass by value and pass by reference: In python all lists are passed by
reference i.e the function input does not copy their value to create a
clone, instead it gets access to the memory location in the RAM where the
list is present.
```

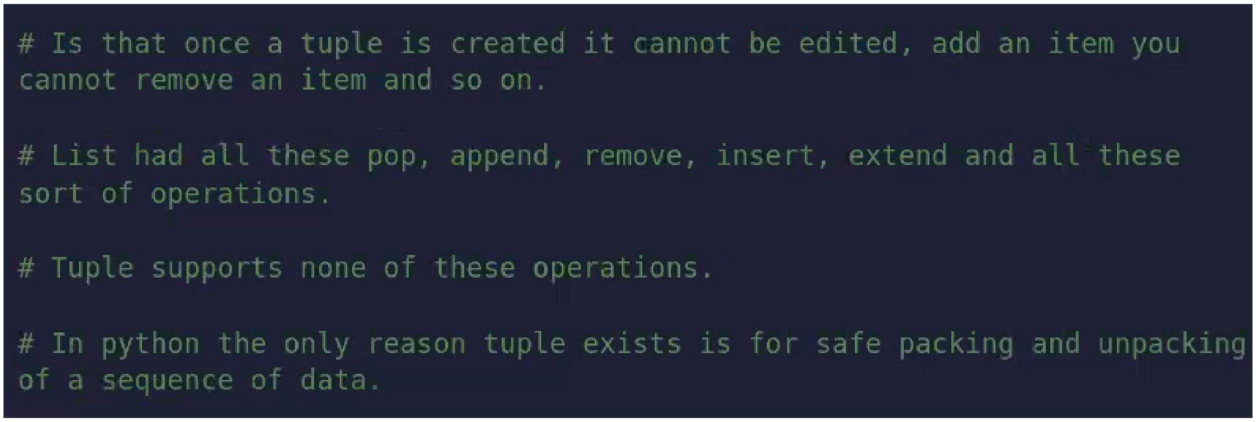
Tuple



```
# Tuple: It is basically the same as a list except you cannot edit the
values of its items. You can perform operations on the items but you
cannot update them.
```

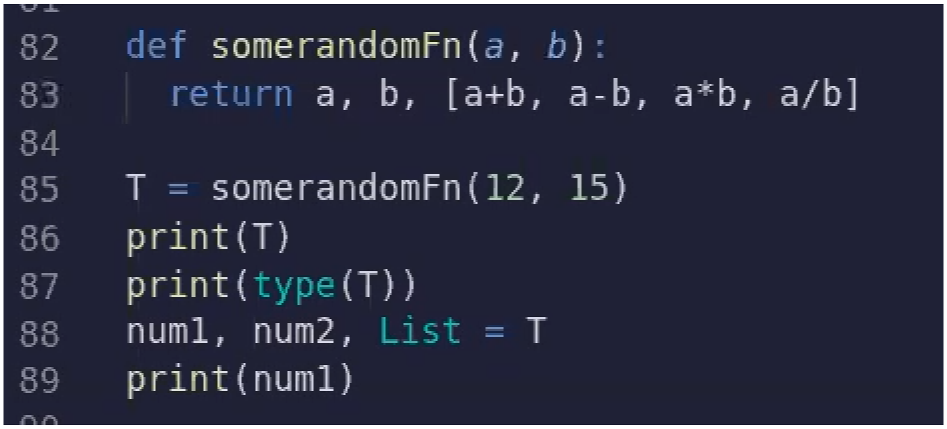


```python
L = [1, 2, 3, 4.0, True, "Hello"]

T = (1, 2, 3, 4.0, True, "Hello")
```

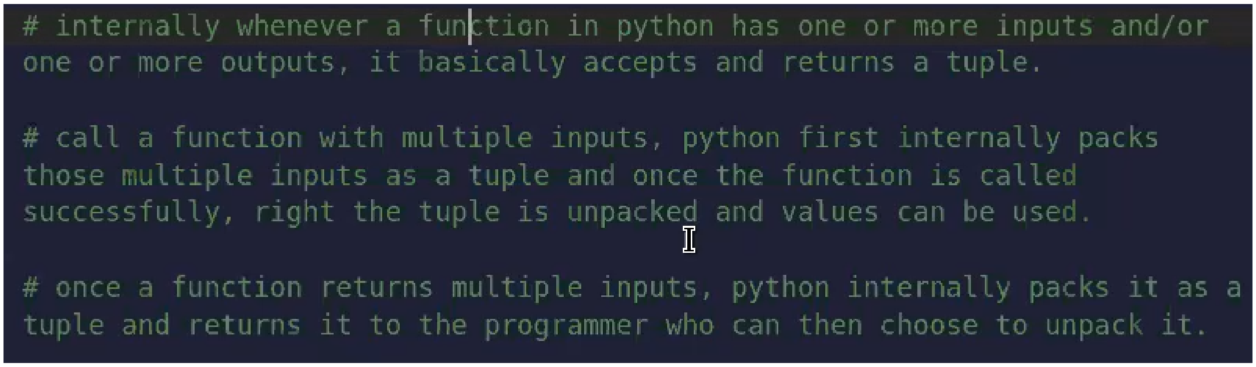


```
# Is that once a tuple is created it cannot be edited, add an item you
cannot remove an item and so on.

# List had all these pop, append, remove, insert, extend and all these
sort of operations.

# Tuple supports none of these operations.

# In python the only reason tuple exists is for safe packing and unpacking
of a sequence of data.
```

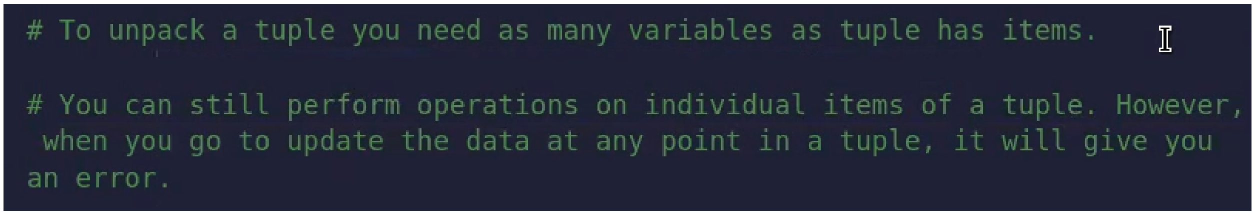
Packing Unpacking Problem



```python
def somerandomFn(a, b):
    return a, b, [a+b, a-b, a*b, a/b]

T = somerandomFn(12, 15)
print(T)
print(type(T))
num1, num2, List = T
print(num1)
```

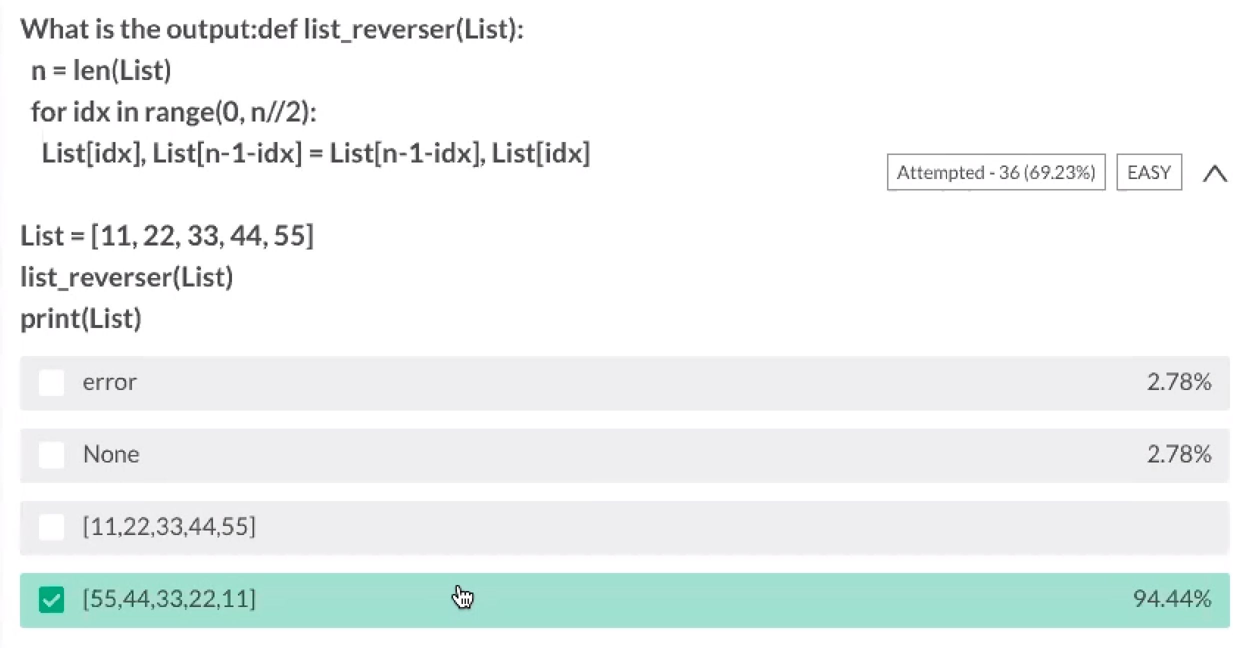


```
# internally whenever a function in python has one or more inputs and/or
one or more outputs, it basically accepts and returns a tuple.

# call a function with multiple inputs, python first internally packs
those multiple inputs as a tuple and once the function is called
successfully, right the tuple is unpacked and values can be used.

# once a function returns multiple inputs, python internally packs it as a
tuple and returns it to the programmer who can then choose to unpack it.
```



```
# To unpack a tuple you need as many variables as tuple has items.

# You can still perform operations on individual items of a tuple. However,
 when you go to update the data at any point in a tuple, it will give you
an error.
```

MCQs



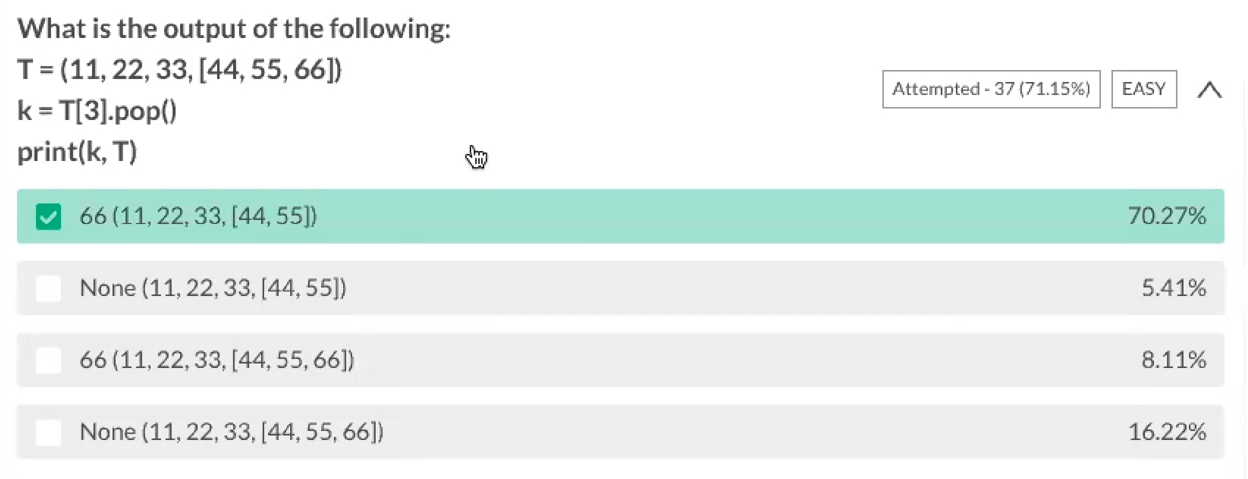
```
What is the output:def list_reverser(List):
 n = len(List)
 for idx in range(0, n//2):
  List[idx], List[n-1-idx] = List[n-1-idx], List[idx]

List = [11, 22, 33, 44, 55]
list_reverser(List)
print(List)
```

☐ error 2.78%

☐ None 2.78%

☐ [11,22,33,44,55]

☑ [55,44,33,22,11] 94.44%



What is the output of the following:

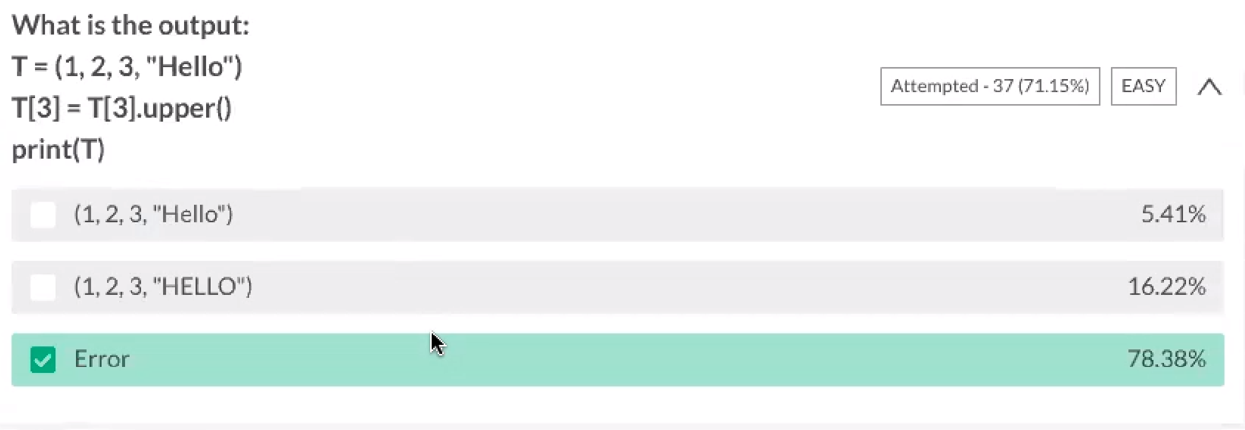
```
T = (11, 22, 33, [44, 55, 66])
k = T[3].pop()
print(k, T)
```

☑ 66 (11, 22, 33, [44, 55]) 70.27%

☐ None (11, 22, 33, [44, 55]) 5.41%

☐ 66 (11, 22, 33, [44, 55, 66]) 8.11%

☐ None (11, 22, 33, [44, 55, 66]) 16.22%



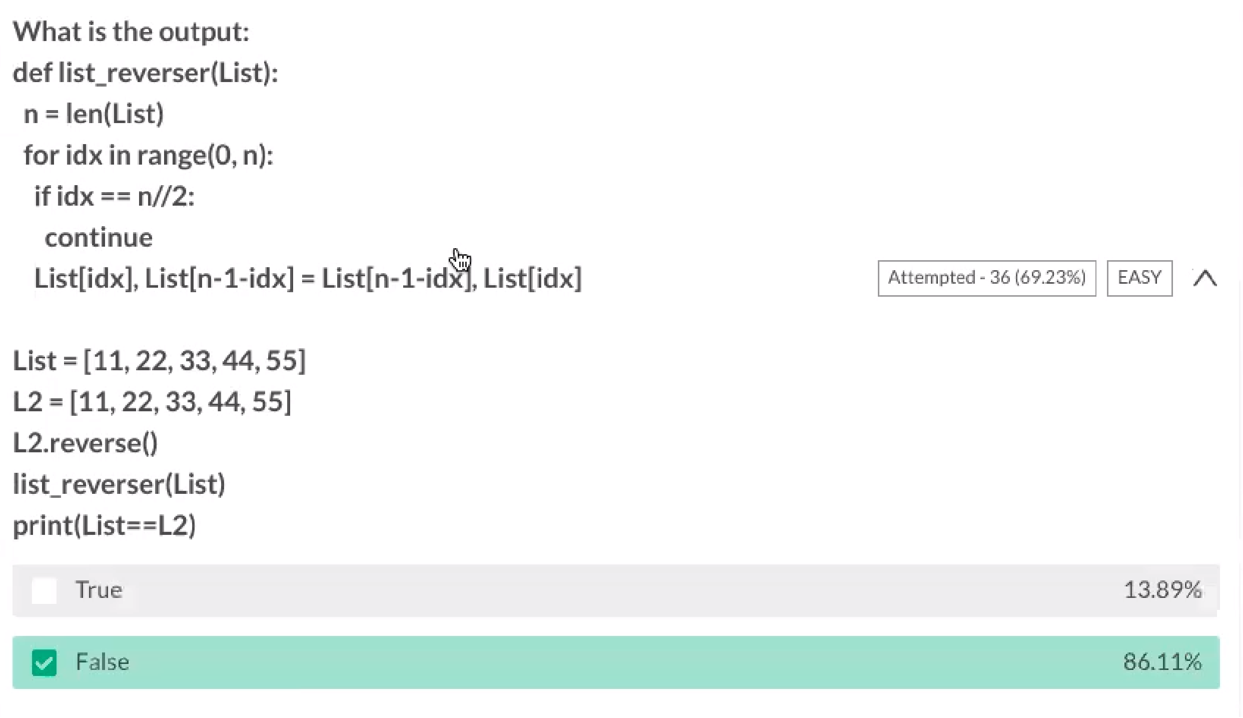
What is the output:

```
T = (1, 2, 3, "Hello")
T[3] = T[3].upper()
print(T)
```

☐ (1, 2, 3, "Hello") 5.41%

☐ (1, 2, 3, "HELLO") 16.22%

☑ Error 78.38%



What is the output:

```
def list_reverser(List):
 n = len(List)
 for idx in range(0, n):
  if idx == n//2:
   continue
  List[idx], List[n-1-idx] = List[n-1-idx], List[idx]

List = [11, 22, 33, 44, 55]
L2 = [11, 22, 33, 44, 55]
L2.reverse()
list_reverser(List)
print(List==L2)
```

☐ True 13.89%

☑ False 86.11%