The Python Book

By Kalu Kalu

Lists

Lists are a collection of objects. For instance of you can have a list of integers, a list of floats, a list of strings, or a list of any combination of the above. You can even have a list of lists (what's known as a two dimensional list (or in Mathematics a matrix). Let's create some lists.

To create a list we simply start with opening and closing brackets, [ ] . Inside the opening and closing brackets we insert any python objects that we want, separated by a comma. Enough talking, let's create some lists!

ex17.py

```
exam_grades = [80, 93, 85, 94]
students = ['eke', 'jordan', 'uche', 'nnenna']
gpas = [3.8, 3.6, 3.33, 4.0]

print(exam_grades)
print(students)
print(gpas)
```

Congrats on creating a list. Now let's do some simple list selection to see how to select elements from the list.

List Item Selection

Now that we have created let list let's see how we can select items from a list. Selecting items from a list is similar to substring selection that we discussed earlier. To select items from a list you use bracket-notation.

ex18.py

```
exam_grades = [80, 93, 85, 94]
students = ['Eke', 'Jordan', 'Uche', 'Nnenna']
gpas = [3.8, 3.6, 3.33, 4.0]

uche_grade = exam_grades[2]
uche_name = students[2]
uche_gpa = gpas[2]

first_two_grades = exam_grades[0:2]
last_two_grades = exam_grades[2:4]
```

```
print(uche_grade)
print(uche_name)
print(uche_gpa)
print(first_two_grades)
print(last_two_grades)
```

Our output should look like this:

```
85
Uche
3.33
[80, 93]
[85, 94]
```

Let's discuss a bit of what happened. Just like with string substring we can select items with bracket-notation. Bracket refers to [ ] . To select an element from the list you just add [ ] to the list name and write the start and stop points you want.

Selecting a Single Item

If you have a list, students = ['eke', 'jordan', 'uche', 'nnenna'] , and we want to select the third name in the list, we would do student[2]. Do not forget that in programming we often start counting with 0. Since students[0] would give us the first position, students[2] is actually the third position number.

Selecting Multiple Items

How do you select multiple items from a list? We select items from a list with "bracket-notation" and adding two position numbers separated by a colon . For instance, students[0:2] would give me the first two items at positions 0 through 1, ['eke', 'jordan']. As we will discussed with strings selection, and will discuss again below, when selecting using bracket-notation the second number after the colon is always **EXCLUSIVE.** Python interprets the statement students[0:2] as meaning get me items from positions 0 **up until but not including** position 2.

When Doing String Selection With Bracket Notation The Second Argument Is Exclusive

From Position X Up Until Position Y

Earlier we had a piece of code that said students[0:2] Surprising this gave us ['eke', 'jordan'] and not ['eke', 'jordan', 'uche']. That is because the second number is EXCLUSIVE. That means we do not include the element at that position, we include everything UP UNTIL. On the other hand, the first number is always include. So you should think of [0:2] as saying give me all the objects **from** position 0 **up until** position 2.

Let’s use our lists to draft a little story about each student.

Position Numbers Start With 0

As you may have just noticed, when discussing position numbers in programming we usually start with zero. For instance, the first element in the students list, students = ['Eke', 'Jordan', 'Uche', ‘Nnenna’], is “eke”. The number we used use to describe this position is position 0, not position 1. The second element, “jordan” , would position 1, and so on.

When Doing String Selection With Bracket Notation The Second Argument Is Exclusive

From Position X Up Until Position Y

Earlier we had a piece of code that said students[0:2] Surprising this gave us ['eke', ‘jordan’] and not ['eke', 'jordan', ‘uche']. That is because the second number is EXCLUSIVE. That means we do not include the element at that position, we include everything UP UNTIL. On the other hand, the first number is always include. So you should think of [0:2] as saying give me all the objects **from** position 0 **up until** position 2.

Let’s use our lists to draft a little story about each student.

ex19.py

```
exam_grades = [80, 93, 85, 94]
students = ['Eke', 'Jordan', 'Uche', 'Nnenna']
gpas = [3.8, 3.6, 3.33, 4.0]

text1 = "\n\nHi "
text2 = ". This is Professor Obama. Congratulations on finishing your Constitutional Law exam. "
text3 = "Your final grade was a "
text4 = ". Now your gpa is a "

eke_story = text1 + students[0] + text2 + text3
eke_story += str(exam_grades[0]) + text4 + str(gpas[0])

```

```
jordan_story = text1 + students[1] + text2 + text3
jordan_story += str(exam_grades[1]) + text4 + str(gpas[1])

uche_story = text1 + students[2] + text2 + text3
uche_story += str(exam_grades[2]) + text4 + str(gpas[2])

nnenna_story = text1 + students[3] + text2 + text3
nnenna_story += str(exam_grades[3]) + text4 + str(gpas[3])


print(eke_story)
print(jordan_story)
print(uche_story)
print(nnenna_story)
```

Our output will look like this:

```
Hi Eke. This is Professor Obama. Congratulations on finishing your
Constitutional Law exam. Your final grade was a 80. Now your gpa is a
3.8


Hi Jordan. This is Professor Obama. Congratulations on finishing your
Constitutional Law exam. Your final grade was a 93. Now your gpa is a
3.6


Hi Uche. This is Professor Obama. Congratulations on finishing your
Constitutional Law exam. Your final grade was a 85. Now your gpa is a
3.33


Hi Nnenna. This is Professor Obama. Congratulations on finishing your
Constitutional Law exam. Your final grade was a 94. Now your gpa is a
4.0
```

Do not be confused with str(exam_grades[2]) . Let's break down what this means. This means we grab the element in position two of exam_grades. This is element the integer, 85 (remember for position numbers we start counting at 0). In order to add a integer to a string we must first convert it to a string. str(exam_grades[2]) will convert the integer 85 into a string, '85', that way we can do string concatenation in order to make our story. First the expression exam_grade[2] is evaluated. This is evaluated into the element 85. After str(85) is evaluated, resulting in "85". Then we add

The For-Loop

How A For-Loop Works

If we want to do something to every single item in a list, we can use a for-loop. Please type the following code.

ex20.py

```
1.  visited = ['China', 'United States', 'Nigeria', 'Japan']
2.
3.  for country in visited:  1
4.      print("Here is the country” )
5.      print(country)  2
6.
7.  print("\nThat is all the countries I have been too. The End.” )  3
```

If we run this file on our terminal we should get the following output.

```
China
United States
Nigeria
Japan
```

In line 1 we write the for-loop definition. Let’s look at the syntax for a for-loop definition. The first thing we need is the keyword “for”. After that we can create a variable name we will use to refer each item in our list. We can choose any name, and we choose to use the variable name “country” because it is a good description of each item in our list. Lastly we use the key word “in” followed by our list, “visited”. Finally, to end our definition we need a colon, “:”.

Next, 2 comes the for-loop body. Here we can use the variable name we created. What ever code we write in the for-loop body will be ran one-by-one for each item in our list. We can refer to the item using “country”, the variable name we created in the for-loop definition.

How Do We End Our For-Loop?

How do we end our for-loop? Python knows to end our for-loop by looking for the first line that does not start with indentation (space at the beginning). In line number 7, 3 we have our first line that does not start with indentation. Therefore we know that the for-loop body has ended. After the for-loop has ended, python continues one-by-one as usually running the lines in your programming.

Step-By-Step Order Of How Python Runs Our Program

The order in which python runs our program is very important, especially with a for-loop. The way this program is ran is first line 1, then line 2, then line 3.

In line 3 we are told to enter into a loop of code. We will run the code n lines 4 and 5, for every item in the list. So we enter into the loop and run the code on line 4, and the the code on line 5. The first time we run the code, the variable name “country” refers to “China”, the first item in the list. After we have finished running line 5 the first time, we restart our loop, this time we country meaning “United States.”. So again python runs line number 4 and line number 5 a second time. After that our loop restarts for the third time. Now “country” refers to “Nigeria”. We run line number 4 and 5 a third time, and then our loop restarts for the fourth and final time. For our last time in the loop “country” refers to “Japan”. Since there are no more items in our visited list, our for-loop is finished. We can finally run the next line of code, line 7, and then our program finishes.

Let’s do something more interesting. In ex21.py we will print the string “I have been to “ plus our country.

ex21.py

```
1.  visited = ['China', 'United States', 'Nigeria', 'Japan']
2.  count = 1
3.  for country in visited:
4.      print("\nCountry number " + str(count))
5.      print("I have been to " + country)
6.      count +=1
7.
8.  print("\n\nThat is all the countries I have been too.")
```

For-Loop Practice

Let’s practice some for-loops. We want to create a program than can add all the numbers together in a list. Please type the following code.

A note: Below you will see, total += num . Do not be confused. The code total+= num means the same thing as total = total + num.

ex22.py

```
1.  numbers = [1, 0, 3, 4]   1
2.  total = 0   2
3.
```

```
4.  for num in numbers:  3
5.      total += num  4
6.
7.  message = "The total is " + str(total)
8.  print(message)
```

In 1 we create our list. Our list is simply a list of integers that we want to add. Next, in 2 we create our counter variable. We will call it total. We can set the total to the value 0. Each time we loop through a number in our numbers list, we want to add that number to the total.

We accomplish this result in 3 and 4. As usual, in 3 we write our function definition. We choose the variable name “num”. Next in the body of our for-loop 4 we simply add the “num” number to our total. Remember, num represents each number in our loop.

Conditional Logic: For-Loops and If Statements

We can logic such as if/else statements in our code. Please type the following code. This program will tell you if a number is even or odd.

ex23.py

```
numbers = [2, 3, 5, 8, 9]

for num in numbers:
    if num % 2 == 0:
        print("The number " + str(num) + " is even")
    else:
        print("The number " + str(num) + " is odd” )
```

That was pretty good. But we can do better. We can use the input() function to take user data. Type your code the following way instead in ex24.py.

ex24.py

```
numbers = [2, 3, 5, 8, 9]

for num in numbers:
    if num % 2 == 0:
        print("The number " + str(num) + " is even")
    else:
        print("The number " + str(num) + " is odd” )
```