



## Exercise 34: Accessing Elements of Lists

Lists are pretty useful, but unless you can get at the things in them they aren't all that good. You can already go through the elements of a list in order, but what if you want say, the fifth element? You need to know how to access the elements of a list. Here's how you would access the *first* element of a list:

```
animals = ['bear', 'tiger', 'penguin', 'zebra']
bear = animals[0]
```

You take a list of animals, and then you get the first (1st) one using `0`?! How does that work? Because of the way math works, Python starts its lists at 0 rather than 1. It seems weird, but there are many advantages to this, even though it is mostly arbitrary.

The best way to explain why is by showing you the difference between how you use numbers and how programmers use numbers.

Imagine you are watching the four animals in our list (`['bear', 'tiger', 'penguin', 'zebra']`) run in a race. They cross the finish line in the *order* we have them in this list. The race was really exciting because the animals didn't eat each other and somehow managed to run a race. Your friend shows up late and wants to know who won. Does your friend say, "Hey, who came in *zeroth*?" No, he says, "Hey, who came in *first*?"

This is because the *order* of the animals is important. You can't have the second animal without the first (1st) animal, and you can't have the third without the second. It's also impossible to have a "zeroth" animal since zero means nothing. How can you have a nothing win a race? It just doesn't make sense. We call these kinds of numbers "ordinal" numbers, because they indicate an ordering of things.

Programmers, however, can't think this way because they can pick any element out of a list at any point. To programmers, the list of animals is more like a deck of cards. If they want the tiger, they grab it. If they want the zebra, they can take it too. This need to pull elements out of lists at random means that they need a way to indicate elements consistently by an address, or an "index," and the best way to do that is to start the indices at 0. Trust me on this: the math is *way* easier for these kinds of accesses. This kind of number is a "cardinal" number and means you can pick at random, so there needs to be a 0 element.

How does this help you work with lists? Simple, every time you say to yourself, "I want the third animal," you translate this "ordinal" number to a "cardinal" number by subtracting 1. The "third" animal is at index 2 and is the penguin. You have to do this because you have spent your whole life using ordinal numbers, and now you have to think in cardinal. Just subtract 1 and you will be good.

Remember: ordinal == ordered, 1st; cardinal == cards at random, 0.

1  
2  
3  
4  
MAIN



Follow

Let's practice this. Take this list of animals, and follow the exercises where I tell you to write down what animal you get for that ordinal or cardinal number. Remember if I say "first," "second," then I'm using ordinal, so subtract 1. If I give you cardinal (like "The animal at 1"), then use it directly.

```
animals = ['bear', 'python', 'peacock', 'kangaroo', 'whale',  
'platypus']
```

1. The animal at 1.
2. The third (3rd) animal.
3. The first (1st) animal.
4. The animal at 3.
5. The fifth (5th) animal.
6. The animal at 2.
7. The sixth (6th) animal.
8. The animal at 4.

For each of these, write out a full sentence of the form: "The first (1st) animal is at 0 and is a bear." Then say it backwards: "The animal at 0 is the 1st animal and is a bear."

Use your Python to check your answers.

## Study Drills

1. With what you know of the difference between these types of numbers, can you explain why the year 2010 in "January 1, 2010," really is 2010 and not 2009? (Hint: you can't pick years at random.)
2. Write some more lists and work out similar indexes until you can translate them.
3. Use Python to check your answers.

### WARNING

Programmers will tell you to read this guy named "Dijkstra" on this subject. I recommend you avoid his writings on this unless you enjoy being yelled at by someone who stopped programming at the same time programming started.

## Video

### Purchase The Videos For \$29.59

For just \$29.59 you can get access to all the videos for Learn Python The Hard Way, **plus** a PDF of the book and no more popups all in this one location. For \$29.59 you get:

- All 52 videos, 1 per exercise, almost 2G of video.
- A PDF of the book.

### Buying Is Easy

Buying is easy. Just fill out the fc below and we'll get started.

Full Name

Full Name




Email Address


- Email help from the author.
- See a list of everything you get before you buy.

When you buy the videos they will immediately show up **right here** without any hassles.

Already Paid? Reactivate Your Purchase Right Now!

Email Address

☒   

☐  Us

Pay With Credit Card  
(by Stripe™)

your PayPal  
account.

Buy Learn Python The Hard  
Way, 3rd Edition

<b>Zed Shaw</b> PDF + Videos + Updates <b>\$29.95</b>	<b>Amazon</b> Paper + DVD <b>\$22.74</b> <b>InformIT</b> eBook + Paper <b>\$43.19</b>	<b>Amazon</b> Kindle <b>\$17.27</b>	<b>B&amp;N</b> Paper + DVD <b>\$22.96</b>	<b>B&amp;N</b> Nook (No Video) <b>\$17.27</b>
		<b>Interested In Ruby?</b> Ruby is also a great language. <b>Learn Ruby The Hard Way</b>		