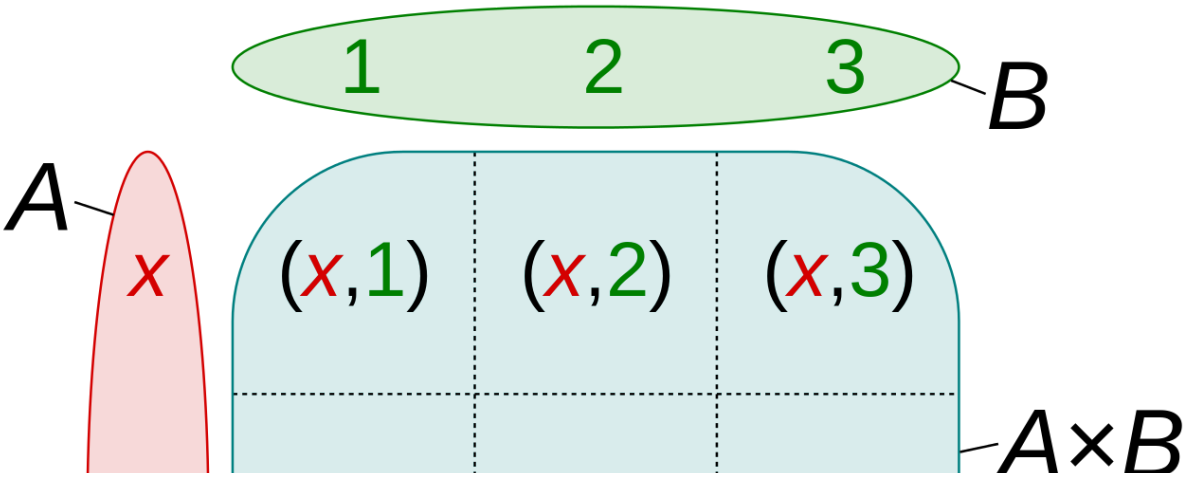
PYTHON SNIPPETS

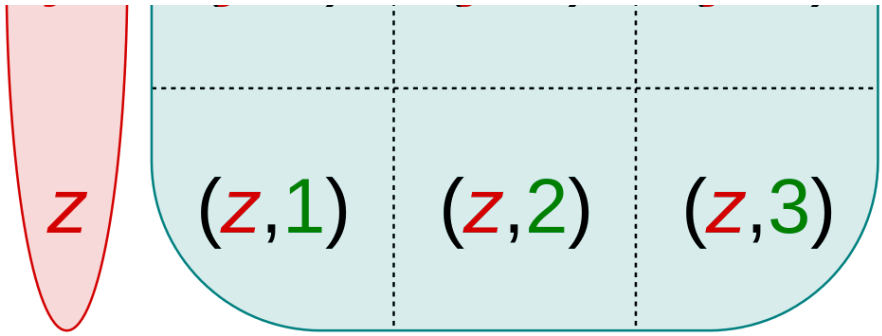
Python Itertools Part 1 – Product

Phil Best
3 min read · Jun 10

The `product` function is one of several handy combinatoric iterators included in the `itertools` module. Combinatoric iterators are related to an area of mathematics called enumerative combinatorics, which is concerned with the number of ways a given pattern can be formed.

`product` gives us a way to quickly calculate the cartesian product of two or more collections. An example of the cartesian product can be found below:





`product` also frequently serves as an alternative approach to having multiple `for` clauses inside list comprehensions.

In a previous post, we provided code for a list comprehension that would calculate all the possible roll combinations for two six-sided dice.

```
roll_combinations = [(d1, d2) for d1 in range(1, 7) for d2 in range(1, 7)]
```

We can do very much the same thing using the `product` function.

```
from itertools import product

dice_combinations = product(range(1, 7), repeat=2)
```

So what's going on here?

The `product` function accepts any number of iterables as positional arguments, and has an optional keyword only parameter called `repeat`.

When we provide two or more iterables as arguments, the `product` function will find all the ways we can match an element from one of these iterables to an item in every other iterable. For example, we might have a pair of lists like so:

```
list_1 = ["a", "b", "c"]
list_2 = [1, 2, 3]
```

When we pass these lists to the product function, we get the following:

```
cartesian_product = product(list_1, list_2)
# ('a', 1) ('a', 2) ('a', 3) ('b', 1) ('b', 2) ('b', 3) ('c', 1) ('c', 2) ('c', 3)
```

If we were to add a third iterable, every one of these tuples would be matched up to an item in this third iterable. For example, if we had a third list containing `"x"`, `"y"`, and `"z"`, we would get output like this:

```
# ('a', 1, 'x') ('a', 1, 'y') ('a', 1, 'z') ('a', 2, 'x') ... etc.
```

The `repeat` parameter is most useful for when we want to use the same iterable multiple times. We can see an example of this in our code for finding roll combinations. We can easily add more and more dice by increasing the value of `repeat`.

If we set a `repeat` value of `2` or more when we have multiple iterables,

cartesian product. The following functions are identical in terms of functionality:

```
c_product_1 = product(["a", "b", "c"], [1, 2, 3], repeat=2)
c_product_2 = product(["a", "b", "c"], [1, 2, 3], ["a", "b", "c"], [1, 2, 3])
```

That just about wraps up our introduction to the `itertools product` function. I hope you learnt something new, and I encourage you to play around with the things we've covered here to really understand how it all works.

We release new snippet posts every Monday, and something a little more substantial on Thursdays, but just in case you forget, you might want to follow us on Twitter to keep up to date with all our content. Next Monday we'll be covering some other cool `itertools` functions, so make sure to check back next week! Alternatively, you can give our Complete Python Course a try, where we go into depth on a number of these topics.

We'd also love to hear about cool tips and tricks you think we should write about, so get in touch on Twitter, or join our Discord server!

Phil Best
I'm a freelance developer, mostly working in web development. I also create course content for Teclado!

Read More

Share this

Python Snippets

Assignment expressions in Python →

Python's namedtuples →

Python's divmod Function →

→ See all 26 posts

CODING INTERVIEW PROBLEMS

Coding Interview Problems: Find the Longest Word in a Paragraph

In this post we tackle another common coding interview problem: finding the longest word in a paragraph. Avoid some common pitfalls and ace your interview.

Phil Best
8 min read · Jun 13

Share this

Like what you see? Enter your e-mail to hear when new posts come out!

E-mail*

Name*

Receive our newsletter and get notified about new posts we publish on the site, as well as occasional special offers.

Sign Up

Privacy Policy