

MODULE 02 - 042: Python - Order of Operations (PEMDAS/PEDMAS)

Understanding Python's Order of Operations

When performing calculations in Python, understanding the **order of operations** is crucial. Without this knowledge, you may end up with incorrect results.

What is the Order of Operations?

The **order of operations** determines the sequence in which calculations are performed. Python follows standard mathematical rules, which dictate that some operations take precedence over others.

Consider the following example:

```
calculation = 8 + 2 * 5 - (9 + 2) ** 2
print(calculation) # Output: -103
```

If you calculate this incorrectly by following a left-to-right approach, you may get the wrong result. Instead, Python follows a **specific sequence**.

1 Using Mnemonics to Remember Order of Operations

A common way to remember the order of operations is the **PEMDAS** or **PEDMAS** rule:

P - Parentheses ()
E - Exponents **
M - Multiplication *
D - Division /
A - Addition +
S - Subtraction -

A useful mnemonic to recall this order is:

Please Excuse My Dear Aunt Sally

Although some variations exist (e.g., switching M and D), **multiplication and division** are processed from **left to right**, so the sequence does not change the final result.

Best Practice: Always use parentheses to clarify operation order and avoid ambiguity.

2 Step-by-Step Breakdown of the Calculation

Let's analyze the example step by step:

Initial Calculation:

```
8 + 2 * 5 - (9 + 2) ** 2
```

Step 1: Parentheses First (P)

```
8 + 2 * 5 - (11) ** 2
```

Step 2: Exponents (E)

```
8 + 2 * 5 - 121
```

Step 3: Multiplication (M)

```
8 + 10 - 121
```

Step 4: Addition & Subtraction (A and S)

```
18 - 121
```

Final Result:

-103

Best Practice: Break down complex calculations step by step to verify accuracy.

3 Practical Applications of PEMDAS in Python

Understanding PEMDAS helps in:

- Writing **accurate** mathematical expressions.
- Debugging **unexpected** calculation results.
- Using **parentheses** effectively to control operation priority.

Example: Controlling Precedence with Parentheses

```
result = (8 + 2) * 5 - 9 + 2 ** 2
print(result) # Output: 43
```

Here, $(8 + 2)$ ensures that **addition happens first** before multiplication.

Best Practice: Use parentheses when needed to **explicitly** define the calculation order.

Summary: Key Takeaways

Operator	Description	Order
()	Parentheses	1st
**	Exponents	2nd
*, /	Multiplication & Division	3rd (Left to Right)
+, -	Addition & Subtraction	4th (Left to Right)

Python Documentation Reference

Python Arithmetic Operators

Python follows standard mathematical precedence rules.

Video Lesson Speech

In this guide, we're going to talk about the order of operations in Python calculations.

Now this is an incredibly important concept because if you do not understand this properly you're going to end up with some very odd calculations and you're going to expect one type of output and end up getting something else.

What the order of operations is?

It is **the order that the python system goes and performs the calculations.**

So, right here I have a full expression that I'm storing inside of a calculation variable so Python is going to go through this entire calculation and then it's going to run it and store it inside of this variable and then we're simply going to print it out.

Now, before I go through and I print it, I want you to think about what this is going to do.

So I want you to try to work this out in your head or on paper and be able to say "*OK the output of this calculation should be....*" some number and see how close you can get to understanding the order of operations because this is something that is not specific to python.

This is actually something that is pretty fundamental in standard math and arithmetic.

And so I want you to be able to see how much you remember this May date you a little bit. Take you back to seventh or eighth grade.

However, it is something that is very important to understand and then come back and we're going to walk through it.

Advice: Use acronyms.

Welcome back if you did perform those calculations I want to first run it and then we're going to go through the process and I'm also going to help give you if you had any issues in understanding how it worked in the order.

I'm going to give you some ways.

Essentially a mental framework for understanding and remembering.

So I'm going to run it and you can see that the value return is negative 103.

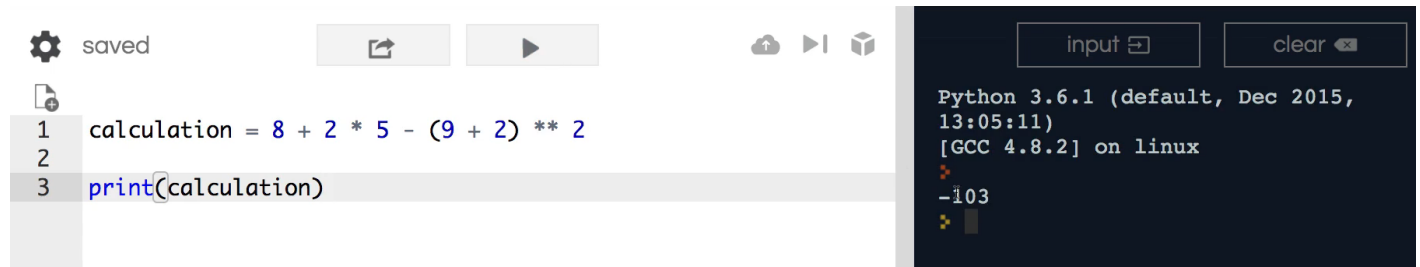
A screenshot of a Python IDE interface. On the left, a code editor shows three lines of Python code: `1 calculation = 8 + 2 * 5 - (9 + 2) ** 2`, `2`, and `3 print(calculation)`. The code is numbered 1, 2, and 3 respectively. Above the code editor are icons for saving, running, and other IDE functions. On the right, a terminal window shows the output of the code: `Python 3.6.1 (default, Dec 2015, 13:05:11)`, `[GCC 4.8.2] on linux`, and `-103`. The terminal also has input and clear buttons.

Figure 1: large

Now, if you were to run this and simply start by saying OK the way you run it is eight plus two which is 10.

And then we multiply ten by five.

So we have 50 and then we're going to subtract and you go like that you're going to end up with the wrong answer.

And so let's see how the order of operations actually goes.

And we're going to use a mnemonic for this.

So I'm going to give us a multiline String and this is a way that I was taught it and I used a process growing up called **PEMDAS** and so I can say:

P
E
M
D
A
S

and there are many different variations to this.

You may hear it called PEMDAS and you may hear these two get switched and it would actually be PEDMAS.

And so there are a bunch of different variations and this isn't one I go with but at the end of the day all of them are going to say the same thing they're all going to mean the same thing and it is all about order of operations.

Now, the way I was taught this was with this little mnemonic sentence that was:

Please Excuse My Dear Aunt Sally

Please
Excuse
My
Dear
Aunt
Sally

That's the way I personally remembered it.

You may remember something back in your own days that will help you do it.

But this is what helped me to remember.

PEMDAS meaning

And now let's talk about what this actually represents.

- Please represents parentheses so what that means is this is all in order. So the P or the parentheses are going to be the first set of items that get processed.

So if we look at this calculation the first thing that's going to be processed is 9 plus 2 because it's inside of parentheses.

- The next one, excuse, is going to be our exponent and just say it and I'm not going to parens but will just say exponents so that means that we're going to take this calculation that gets done.

Then we're going to look at the exponent and we're going to square it because it is an exponent.

- M or my here stands for multiplication. And so we have our multiplication right here. Now if you remember back I said that some people use PEMDAS like we have here and other people use PEDMAS where you switch the D and the M.

Now I have had multiple conversations and I've received more comments on this.

One guide especially when I've taught it in other languages such as JavaScript in Ruby and everyone always seems to want to fight about this one and so I have asked math professors at universities exactly what the right answer is and that truth is there is not one way of doing it and the issue is because once you get to this stage the M and the D which is multiplication and division you go from right to left in it all.

It just doesn't matter.

So you could do the division first, you could do the multiplication first. It doesn't matter you're going to end up with the same exact result and if you don't believe me go through it yourself research it yourself.

This is simply the way I personally do it so that I can remember it but I have seen people switch it up do the division before and multiplication.

It's still going to come out to the same answer.

- Next, we have Aunt which stands for addition. So with the addition this is where we're going to have our plus sign. So right here we have it right here at the beginning.

Now notice we have a plus sign inside of the parens and that is the first process we're going to run because it's inside of these parentheses.

- Lastly we have subtraction which is our dash and I'll put these here:

```
Please -> Parens ()
Excuse -> Exponents **
My -> Multiplication *
Dear -> Division /
Aunt -> Addition +
Sally -> Subtraction -
```

And, in the show notes just so you have access to them and that's divisions that slash multiplication is our Asterix and I'm just going to format all of these just so they're all the same.

And then here parens.

Okay, so that is the order.

But now let's actually take a look back and let's actually build this out. So instead of just trusting that that is the right answer. Let's see how they were able to. When I say they I mean the Python programming language lets see how it was able to process that. I'm going to come inside of our comments and we're going to do this just like back in school. So we're going to go through and take it step by step. So right here we were following the PEDMAS rules or the PEMDAS rules. We are going to start with the parens. So I'm going to copy this down to the next line and we're simply going to add nine plus two. So we're going to get 11 right here and that is all we're going to do on that line. Coming down our next order. Remember it's exponents. So it's going to be 11 squared. Thankfully this is part of the reason why I used a two here because I actually know what 11 square it is. If I had done something bigger that we would have had to break out the calculator. So you know that 11 squared is 121. So now we have eight plus two times five minus 121. Let's copy this down and looking back up and referencing what we have left.

The next one on the list is multiplication. So here we have two times five. So we know that this one is 10 and now we have addition and subtraction and from here we can really just do this from right to left and so we don't even have to do the rest of it you're going to end up with 8 plus 10 which is 18, 18 minus 121 is going to be the right answer 103. Now if you don't believe me I don't blame you because arithmetic is definitely not my strong suit. Let's just say 8 plus 10 18 minus 121. And that is equal to negative 103. So that is the full process from here ends all the way through subtraction.

The screenshot shows a Python IDE with a file named 'saved'. The code editor on the left contains the following Python code:

```

1  """
2  Please -> Parans ()
3  Excuse -> Exponents **
4  My -> Multiplication *
5  Dear -> Division /
6  Aunt -> Addition +
7  Sally -> Subtraction -
8
9  8 + 2 * 5 - (9 + 2) ** 2
10 8 + 2 * 5 - 11 ** 2
11 8 + 2 * 5 - 121
12 8 + 10 - 121
13 -103
14 """
15
16 calculation = 8 + 2 * 5 - (9 + 2) ** 2
17
18 print(calculation)

```

The output console on the right shows the execution of the code:

```

Python 3.6.1 (default, Dec 2015,
13:05:11)
[GCC 4.8.2] on linux
-103

```

Figure 2: large

You don't have to actually perform this task you have to understand the way it works because say that you were to call variables and you were to do it with the wrong order of operations. You're going to end up with the wrong output. So it's important to understand that this is the entire process that Python uses when it's running your calculations.

Code

```

"""
Please -> Parans ()
Excuse -> Exponents **
My -> Multiplication *
Dear -> Division /
Aunt -> Addition +
Sally -> Subtraction -
8 + 2 * 5 - (9 + 2) ** 2
8 + 2 * 5 - 11 ** 2
8 + 2 * 5 - 121
8 + 10 - 121
-103
"""

calculation = 8 + 2 * 5 - (9 + 2) ** 2

print(calculation)

```