

In this tutorial, we will learn even more about definitions.

Syntax: Lispy JS PY Scala 3

What is the result of running this program?

Syntax: Lispy JS PY Scala 3

```
def f(x):  
    return 3  
print(f(12 / 0))
```

Run 

error

Syntax: Lispy JS PY Scala 3

You got it right! 🎉🎉🎉

Syntax: Lispy JS PY Scala 3

Function calls bind their formal parameters (in this case, there is one formal parameter, `x`) to the values of actual parameters (in this case, there is one actual parameter, `12 / 0`). The program errors when it tries to evaluate `12 / 0`.

Click [here](#) to run this program in the Stacker.

What is the result of running this program?

Syntax: Lispy JS PY Scala 3

```
x = y + 1  
y = 2  
print(x)  
print(y)
```

Run 

error

Syntax: Lispy JS PY Scala 3

You got it right! 🎉🎉🎉

Syntax: Lispy JS PY Scala 3

The first definition tries to bind `x` to the value of `y + 1`. To evaluate `y + 1`, we need the value of `y`. But `y` is not bound to a value at that moment.

Click [here](#) to run this program in the Stacker.

What is the result of running this program?

Syntax: Lispy JS PY Scala 3

```
x = 12 / 0  
print(3)
```

Run 

error

Syntax: Lispy JS PY Scala 3

Error

You got it right! 🎉🎉🎉

Syntax: Lispy JS PY Scala 3

When you define a variable (in this case, `x`), you have to bind it to a value, no matter whether or not you need the value of that variable later in the program. The program errors when it tries to evaluate `12 / 0`.

Click [here](#) to run this program in the Stacker.

In what order are definitions and expressions evaluated?

Syntax: Lispy JS PY Scala 3

definitions are evaluated when encountered, and expressions are evaluated in the order they appear in the script.

Syntax: Lispy JS PY Scala 3

- Variables are bound to values. Specifically, every variable definition evaluates the expression immediately and binds the variable to the value, even if the variable is not used later in the program; every function call evaluates the actual parameters immediately and binds the values to formal parameters, even if the formal parameter is not used in the function.
- Every block evaluates its definitions and expressions in reading order (i.e., top-to-bottom and left-to-right).

Any feedback regarding these statements? Feel free to skip this question.

Syntax: Lispy JS PY Scala 3

(You skipped the question.)

Syntax: Lispy JS PY Scala 3

Please scroll back and select 1-3 programs that together make these points.

Syntax: Lispy JS PY Scala 3

- Variables are bound to values. Specifically, every variable definition evaluates the expression immediately and binds the variable to the value, even if the variable is not used later in the program; every function call evaluates the actual parameters immediately and binds the values to formal parameters, even if the formal parameter is not used in the function.

You don't need to select all such programs.

(You selected 3 programs)

Syntax: Lispy JS PY Scala 3

Click the left arrow to go back to the previous question.

Syntax: Lispy JS PY Scala 3

Okay. How do these programs ([1,4,7](#)) support the point?

They are all trying to bind values to variables

Syntax: [Lispy](#) [JS](#) [PY](#) [Scala](#) [3](#)

Please scroll back and select 1-3 programs that together make these points.

Syntax: [Lispy](#) [JS](#) [PY](#) [Scala](#) [3](#)

- Every block evaluates in top-to-bottom, left-to-right order.

You don't need to select all such programs.

(You selected 1 programs)

Syntax: [Lispy](#) [JS](#) [PY](#) [Scala](#) [3](#)

Okay. How does this program ([4](#)) support the point?

Syntax: [Lispy](#) [JS](#) [PY](#) [Scala](#) [3](#)

y is not defined before x

Syntax: [Lispy](#) [JS](#) [PY](#) [Scala](#) [3](#)



Here is a program that confused many students

Syntax: [Lispy](#) [JS](#) [PY](#) [Scala](#) [3](#)

```
def addy(x):
    return x + y
s = addy(1)
y = 2
print(s)
```

Run 

Please

1. Run this program in the stacker by clicking the green run button above;
2. The stacker would show how this program produces its result(s);
3. Keep clicking  Next until you reach a configuration that you find particularly helpful;
4. Click  Share This Configuration to get a link to your configuration;
5. Submit your link below;

<https://smol-tutor.xyz/stacker/?syntax=Python&randomSeed=smol-tutor&nNext=2&program=%28defun+%28addy+x%29%0A++%28%2B+x+y%29%29%0A%28defvar+s+%28addy+1%29%29%0A%28defvar+y+2%29%0A%0As%0A&readOnlyMode=>

Syntax: [Lispy](#) [JS](#) [PY](#) [Scala](#) [3](#)

Please write a couple of sentences to explain how your configuration explains the result(s) of the program.

Syntax: [Lispy](#) [JS](#) [PY](#) [Scala](#) [3](#)

An error occurs due to y not being defined before the definition of addy()

Syntax: Lispy JS PY Scala 3

Let's review what we have learned in this tutorial.

Syntax: Lispy JS PY Scala 3

- Variables are bound to values. Specifically, every variable definition evaluates the expression immediately and binds the variable to the value, even if the variable is not used later in the program; every function call evaluates the actual parameters immediately and binds the values to formal parameters, even if the formal parameter is not used in the function.
- Every block evaluates its definitions and expressions in reading order (i.e., top-to-bottom and left-to-right).

You have finished this tutorial 🎉🎉🎉

Please the finished tutorial to a PDF file so you can review the content in the future. **Your instructor (if any) might require you to submit the PDF.**

Start time: 1711096860135