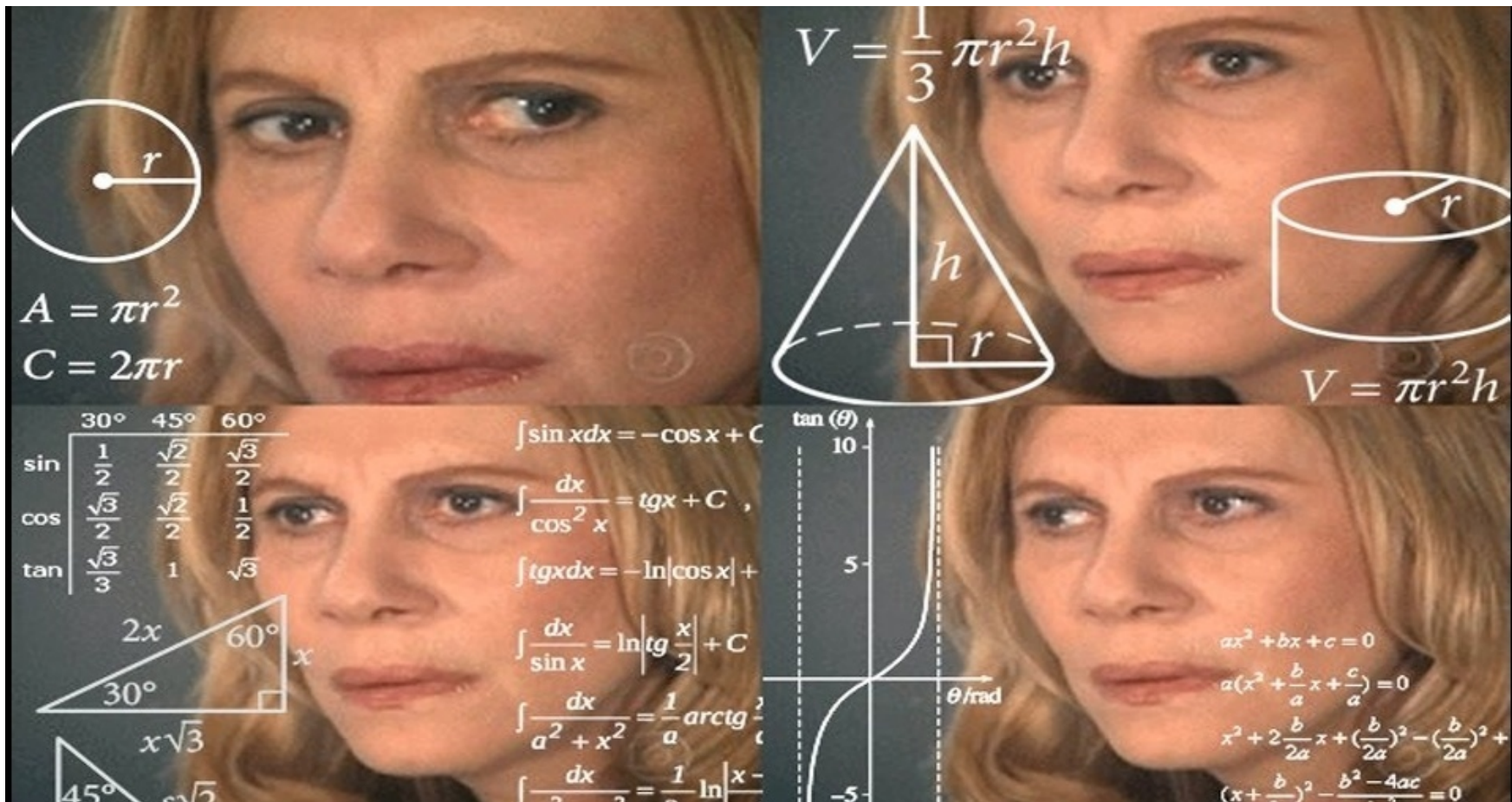


EXERCÍCIOS



`print()` outputs to the screen. Whatever you put between the parentheses will be output!
E.g. `print(5)` will output `5`!

What is the output of this code?

```
print(42)
```

SELECT THE OUTPUT

42

2

4

Variables can be used in calculations. The value of the variable will be used for the computation of the result.

Select the code to return the output

80

SELECT THE CODE

```
income = 100  
costs = 20  
print(costs - income)
```

```
income = 100  
costs = 20  
print(income - costs)
```

```
income = 100  
costs = 20  
print(profit)
```

You can use the `type()` function to get the data type of a variable, e.g. `type(10)` gives you `<type 'int'>`.

What is the output of this code?

```
x = 5  
print(type(x))
```

SELECT THE OUTPUT

`<type 'int'>`

`<type 'float'>`

What is the output of this code?

```
a = '1'  
b = '2'  
print(a + b)
```

SELECT THE OUTPUT

12

3

'3'

'12'

You can also access values using negative indexes, which will go from right-to-left. So, the last element would be at index .

Complete the code to return the output

```
l = [1, 2, 3]  
print(l[])
```

2

FILL IN THE BLANKS

-1

-2

-3

You access values from lists of lists using indexes with `[]`; e.g. to get the **second** element of the **first** sublist: `l[0][1]`.

Complete the code to return the output

```
l = [[7, 6], [8, 9], [10, 11]]  
print(l[?][?])
```

8

FILL IN THE BLANKS

1

2

0

-1

You can replace elements in a list by using index notation with assignment; e.g. `l[0] = 10` will set the first element of `l` to `10`.

What is the output of this code?

```
l = [5, 6, 4, 2]
l[2] = 100
print(l)
```

SELECT THE OUTPUT

`[5, 100, 4, 2]`

`[5, 6, 4, 100]`

`[5, 6, 100, 2]`

With integer variables `a` and `b`, `b = a` **copies** the value of `a` into `b`. So, if `a` changes `b` will not change.

What is the output of this code?

```
a = 1
b = a
a = 6
print(a)
print(b)
```

SELECT THE OUTPUT

1
6

6
1

6
6

With lists `a` and `b`, `b = a` does not copy the values of `a` into `b`, but instead they share those values. So, if `a` changes, `b` changes too!

What is the output of this code?

```
a = [1, 9]
b = a
a[0] = 9
print(a)
print(b)
```

SELECT THE OUTPUT

```
[9, 9]
[1, 9]
```

```
[1, 9]
[1, 9]
```

```
[9, 9]
[9, 9]
```



DataCamp



DataCamp - Learn R, Python & SQL

DataCamp



UNINSTALL

OPEN



Downloads



982



Education



Similar