Python Code Challenges: Functions

Python Code Challenges involving functions.

This article will help you review Python functions by providing some code challenges.

Some of these challenges are difficult! Take some time to think about them before starting to code.

You might not get the solution correct on your first try — look at your output, try to find where you're going wrong, and iterate on your solution.

Finally, if you get stuck, use our solution code! If you "Check Answer" twice with an incorrect solution, you should see an option to get our solution code. However, truly investigate that solution — experiment and play with the solution code until you have a good grasp of how it is working. Good luck!

Function Syntax

As a refresher, function syntax looks like this:

```
def some_function(some_input1, some_input2):
  # … do something with the inputs …
  return output
```

For example, a function that returns the sum of the first and last elements of a given list might look like this:

```
def first_plus_last(lst):
  return lst[0] + lst[-1]
```

And this would produce output like:

```
>>> first_plus_last([1, 2, 3, 4])
5
>>> first_plus_last([8, 2, 5, -8])
0
>>> first_plus_last([-10, 2, 3, -4])
-14
```

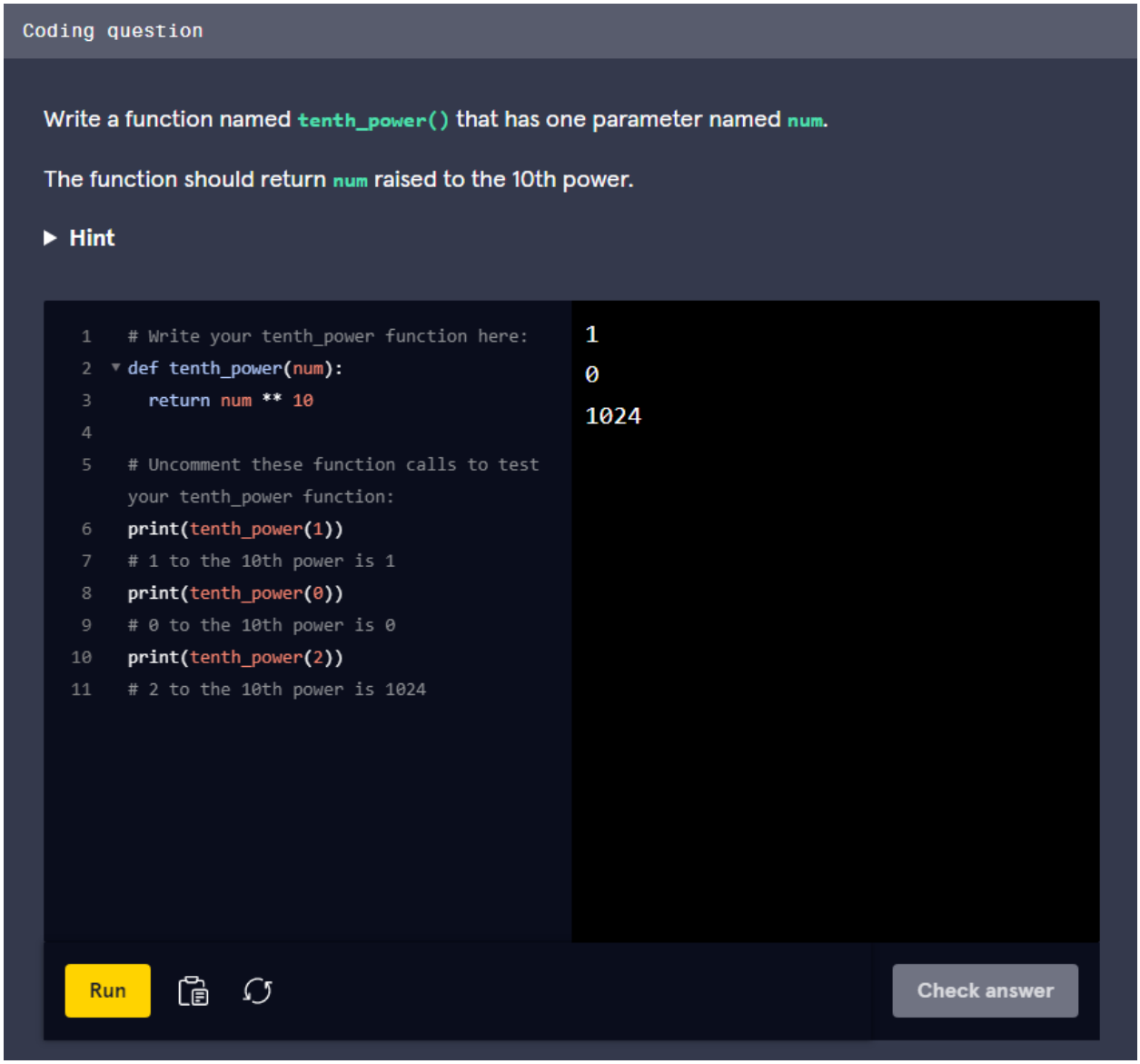
Challenges

We've included 5 challenges below. Try to answer all of them and polish up your problem-solving skills and your function expertise.

1. Tenth Power

Let's create some functions which can help us solve math problems! For this first function, we are going to take the tenth power of a number. In order to do this we need to do three things:

1. Set up the function header for `tenth_power` which accepts one parameter
2. Take the tenth power of the input value
3. Return the result



Let's go over this solution:

```
def tenth_power(num):
  return num ** 10
```

This is one way to solve this problem using two lines of code.

The first line is the function header which uses `def` to define the function followed by the function name. Within the parentheses we include `num` which is our formal parameter. Because of this, `num` is the variable name for the value we pass into this function.

On our next line, we use `return` to show that this function is going to return a value when it is called. Next to the `return` keyword, we define what value is going to be returned. Since we need the tenth power of our input value, we can use the power operator in python which is `**`. Using this knowledge, in order to get the tenth power of our input value, we use `num ** 10`.

2. Square Root

Another useful function for solving math problems is the square root function. We can create this using similar steps from the last problem. The code will look very similar. We need to:

1. Set up the function header for `square_root` which accepts one parameter
2. Take the square root of the input value
3. Return the result

Write a function named `square_root()` that has one parameter named `num`.

Use exponents (`**`) to return the square root of `num`.

▼ Hint

Remember to use `def` when defining the function. To take the square root of a value, you can use the power operator `**`. The square root of a number is the same as taking the ½ power of the number. For example, the square root of 6 would look like: `6 ** 0.5`.

```
# Write your square_root function here:
def square_root(num):
    return int(num ** (1/2))

# Uncomment these function calls to test your square_root function:
print(square_root(16))
# should print 4
print(square_root(100))
# should print 10
```

```
4
10
```

Run

Check answer

Let's go over one possible solution:

```
def square_root(num):
  return num ** 0.5
```

As you can see, this solution is very similar to the last problem. In this case, the only changes we need are the function name and changing the power value to 0.5. We define the function called `square_root` with one parameter. We then take the one half power of the input value which is mathematically the same as taking the square root and return it.

3. Win Percentage

Next, we will create a function which calculates the percentage of games won. In order to do this, we will need to know how many total games there were and divide the

number of wins by the total number of games. For this function, there will be two input parameters, the number of wins and the number of losses. We also need to make sure that we return the result as a percentage (in the range of 0 to 100). In order to create this method we need the following steps:

1. Define the function header with two parameters, `wins` and `losses`
2. Calculate the total number of games using the number of wins and losses
3. Get the ratio of winning using the number of wins out of the total number of games.
4. Convert the ratio to a percentage
5. Return the percentage

Create a function called `win_percentage()` that takes two parameters named `wins` and `losses`.

This function should return out the total percentage of games won by a team based on these two numbers.

▶ Hint

```
# Write your win_percentage function here:
def win_percentage(wins, losses):
    total = wins + losses
    radio_of_winning = wins / total
    return int(radio_of_winning * 100)

# Uncomment these function calls to test your win_percentage function:
print(win_percentage(5, 5))
# should print 50
print(win_percentage(10, 0))
# should print 100
```

```
50
100
```

Run

Check answer

You got it!

Let's go over how we completed this function:

```
def win_percentage(wins, losses):
  total_games = wins + losses
  ratio_won = wins / total_games
  return ratio_won * 100
```

First, we defined our function with two parameters, one for games won and one for games lost. Next, we calculated the total number of games using the number of wins + losses. After that, we use calculate the ratio of wins out of the total number of games by dividing `wins` by our `total_games` variable. Since this gives us a ratio and we want it in percentage form, we multiply the answer by 100 and return it.

4. Average

Let's create a function which takes the average of two numbers. These two numbers will be the input of the function and the output will be the average of the two. In order to do this, we need to do a few steps:

1. Define the function with two input parameters, `num1` and `num2`
2. Calculate the sum of the two numbers
3. Divide the sum by the number of inputs to the function
4. Return the answer

Write a function named `average()` that has two parameters named `num1` and `num2`.

The function should return the average of these two numbers.

▼ Hint

To calculate the average of two numbers we add the first and second number, then divide the result by 2: `(first + second) / 2`

```
1  # Write your average function here:
2  def average(num1, num2):
3      return int((num1 + num2) / 2)
4
5  # Uncomment these function calls to test
   your average function:
6  print(average(1, 100))
7  # The average of 1 and 100 is 50.5
8  print(average(1, -1))
9  # The average of 1 and -1 is 0
```

```
50
0
```

Run

Check answer

Let's look at this solution:

```
def average(num1, num2):
  return (num1 + num2) / 2
```

In this solution, we defined the function with two parameters one line and returned the average on the next line. When returning the value, we used parentheses around the addition to cause the two numbers to be added together first before dividing by two.

5. Remainder

For the final challenge in this group, we are going to take the remainder of two numbers while performing other mathematical operations on them. We are going to multiply the numerator by 2 and divide the denominator by 2. After the two values have been modified, we will divide them and return the remainder. In order to do this we will need to:

1. Define the function to accept two parameters
2. Multiply the first input value by 2
3. Divide the second input value by 2
4. Calculate the remainder of the modified first input value divided by the modified second input value (using modulus)
5. Return the answer

Write a function named `remainder()` that has two parameters named `num1` and `num2`.

The function should return the remainder of twice `num1` divided by half of `num2`.

▼ Hint

In order to calculate the remainder of two numbers, we can use the modulus operator `%`. For example, the remainder of 5 divided by 2 is equal to 1 and we can get this result using `5 % 2`.

```
# Write your remainder function here:
def remainder(num1, num2):
  return int((num1 * 2) % (num2 / 2))

# Uncomment these function calls to test your remainder function:
print(remainder(15, 14))
# should print 2
print(remainder(9, 6))
# should print 0
```

```
2
0
```

Run

Check answer

Let’s go over how we did it:

```
def remainder(num1, num2):
  return (2 * num1) % (num2 / 2)
```

Our solution It shortened to use only two lines of code. The first line defines the function with two input parameters. The second line performs the two mathematical operations on either side of the modulus within parenthesis. This causes the two calculations to be performed before taking the remainder of the left side divided by the right side.