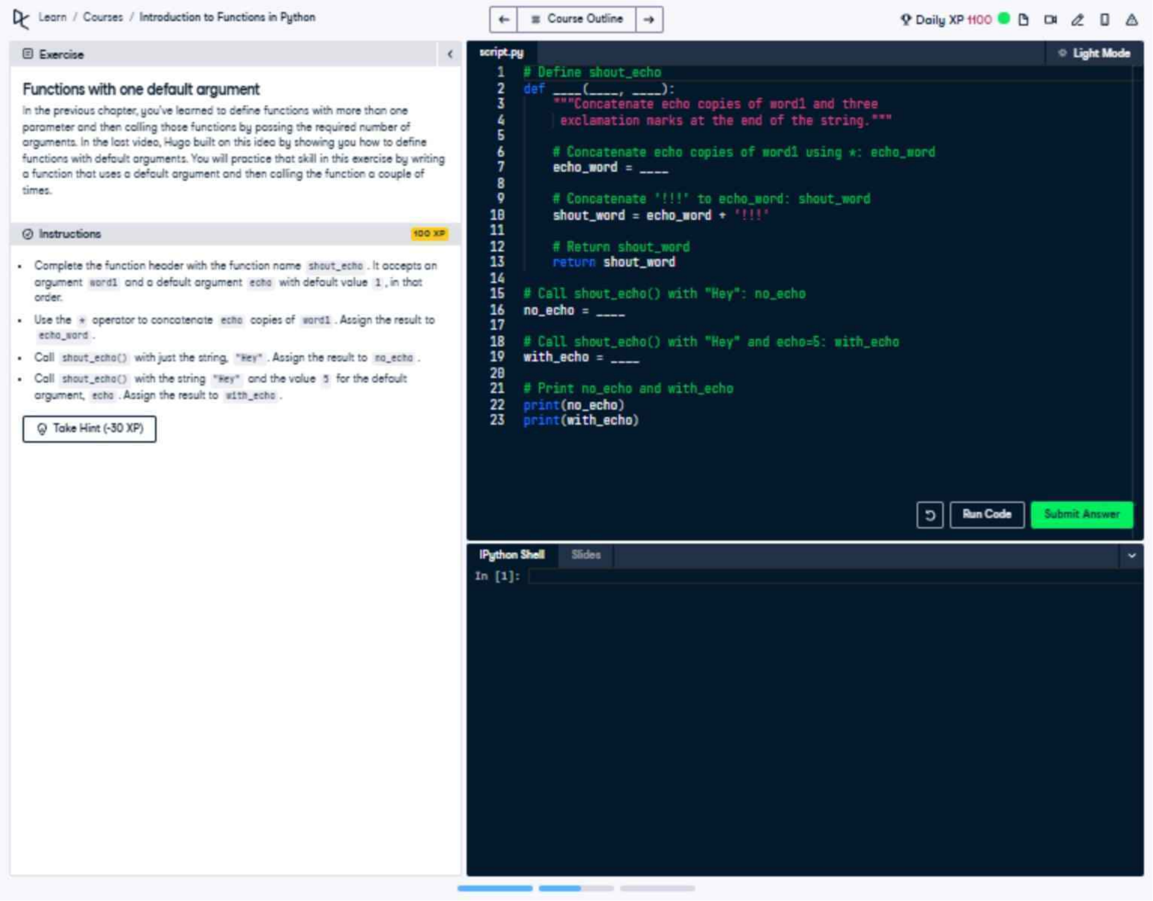
Functions with One Default Argument



Functions with one default argument

In the previous chapter, you've learned to define functions with more than one parameter and then calling those functions by passing the required number of arguments. In the last video, Hugo built on this idea by showing you how to define functions with default arguments. You will practice that skill in this exercise by writing a function that uses a default argument and then calling the function a couple of times.

- Complete the function header with the function name `shout_echo`. It accepts an argument `word1` and a default argument `echo` with default value `1`, in that order.
- Use the `*` operator to concatenate `echo` copies of `word1`. Assign the result to `echo_word`.
- Call `shout_echo()` with just the string, `"Hey"`. Assign the result to `no_echo`.
- Call `shout_echo()` with the string `"Hey"` and the value `5` for the default argument, `echo`. Assign the result to `with_echo`.

```python
# Define shout_echo
def ____(____, ____):
    """Concatenate echo copies of word1 and three
     exclamation marks at the end of the string."""

    # Concatenate echo copies of word1 using *: echo_word
    echo_word = ____

    # Concatenate '!!!' to echo_word: shout_word
    shout_word = echo_word + '!!!'

    # Return shout_word
    return shout_word

# Call shout_echo() with "Hey": no_echo
no_echo = ____

# Call shout_echo() with "Hey" and echo=5: with_echo
with_echo = ____

# Print no_echo and with_echo
print(no_echo)
print(with_echo)
```

Question:

In the previous chapter, you've learned to define functions with more than one parameter and then calling those functions by passing the required number of arguments.
In the last video, Hugo built on this idea by showing you how to define functions with default arguments.
You will practice that skill in this exercise by writing a function that uses a default argument and then calling the function a couple of times.

Instructions:
1. Complete the function header with the function name `shout_echo`. It accepts an argument `word1` and a default argument `echo` with default value `1`, in that order.
2. Use the `*` operator to concatenate `echo` copies of `word1`. Assign the result to `echo_word`.
3. Concatenate `'!!!'` to `echo_word` and assign the result to `shout_word`.

4. Return `shout_word`.
5. Call `shout_echo()` with just the string `"Hey"`. Assign the result to `no_echo`.
6. Call `shout_echo()` with the string `"Hey"` and the value `5` for the default argument `echo`. Assign the result to `with_echo`.
7. Print `no_echo` and `with_echo`.

Answer:

```
# Define shout_echo
def shout_echo(word1, echo=1):
    """Concatenate echo copies of word1 and three exclamation marks at the end of the string."""

    # Concatenate echo copies of word1 using *: echo_word
    echo_word = word1 * echo

    # Concatenate '!!!' to echo_word: shout_word
    shout_word = echo_word + '!!!'

    # Return shout_word
    return shout_word

# Call shout_echo() with "Hey": no_echo
no_echo = shout_echo("Hey")

# Call shout_echo() with "Hey" and echo=5: with_echo
with_echo = shout_echo("Hey", echo=5)

# Print no_echo and with_echo
print(no_echo)
print(with_echo)
```

Explanation:

1. def shout_echo(word1, echo=1): - This defines a function `shout_echo` with a required parameter `word1` and an optional parameter `echo` with a default value of 1.
2. echo_word = word1 * echo - Creates a string by repeating `word1` exactly `echo` times.
3. shout_word = echo_word + '!!!' - Appends `'!!!'` to the string

`echo_word`.

4. return shout_word - Returns the final string `shout_word`.
5. no_echo = shout_echo("Hey") - Calls `shout_echo` with only the required parameter. Since `echo` is not provided, it uses the default value of 1.
6. with_echo = shout_echo("Hey", echo=5) - Calls `shout_echo` with both `word1` and `echo`. This creates a string by repeating "Hey" 5 times, followed by `'!!!'`.
7. print(no_echo) - Prints the result of `shout_echo("Hey")`, which is `"Hey!!!"`.
8. print(with_echo) - Prints the result of `shout_echo("Hey", echo=5)`, which is `"HeyHeyHeyHeyHey!!!"`.