**INTRODUCTION TO FUNCTIONS**

**Introduction to Functions**

In programming, as we start to write bigger and more complex programs, one thing we will start to notice is we will often have to repeat the same set of steps in many different places in our program.

Let’s imagine we were building an application to help people plan trips! When using a trip planning application we can say a simple procedure could look like this:

1. Establish your origin and destination

2. Calculate the distance/route

3. Return the best route to the user

We will perform these three steps every time users have to travel between two points using our trip application. In our programs, we could rewrite the same procedures over and over (and over) for each time we want to travel, but there’s a better way! Python gives us a useful concept called [*functions*](https://www.codecademy.com/resources/docs/python/functions?page_ref=catalog).

Functions are a convenient way to group our code into reusable blocks. A function contains a sequence of steps that can be performed repeatedly throughout a program without having to repeat the process of writing the same code again.

In this lesson, we are going to explore the idea of a function by slowly building out a Python program for our trip planning steps!

At the end of this lesson, you’ll know how to:

* Write a function and return values from it.
* Allow functions to take custom input.
* Experiment with how functions access our other python code.

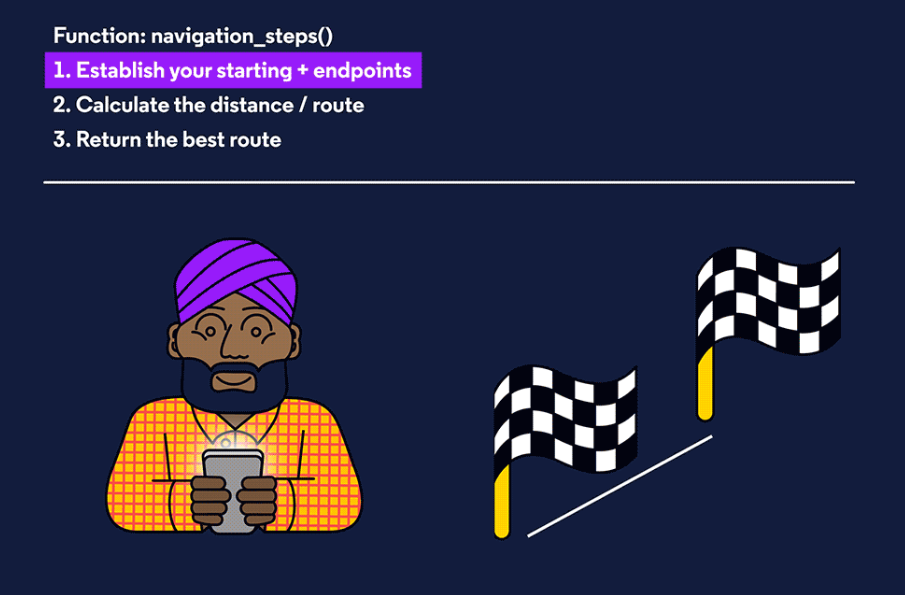
And much more!

**Instructions**

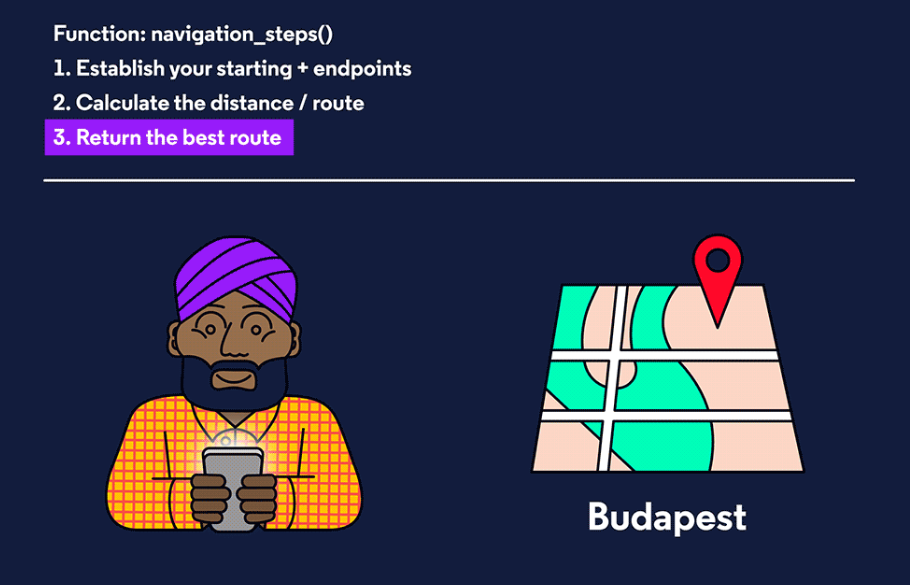
Review the visual for the function navigation\_steps().

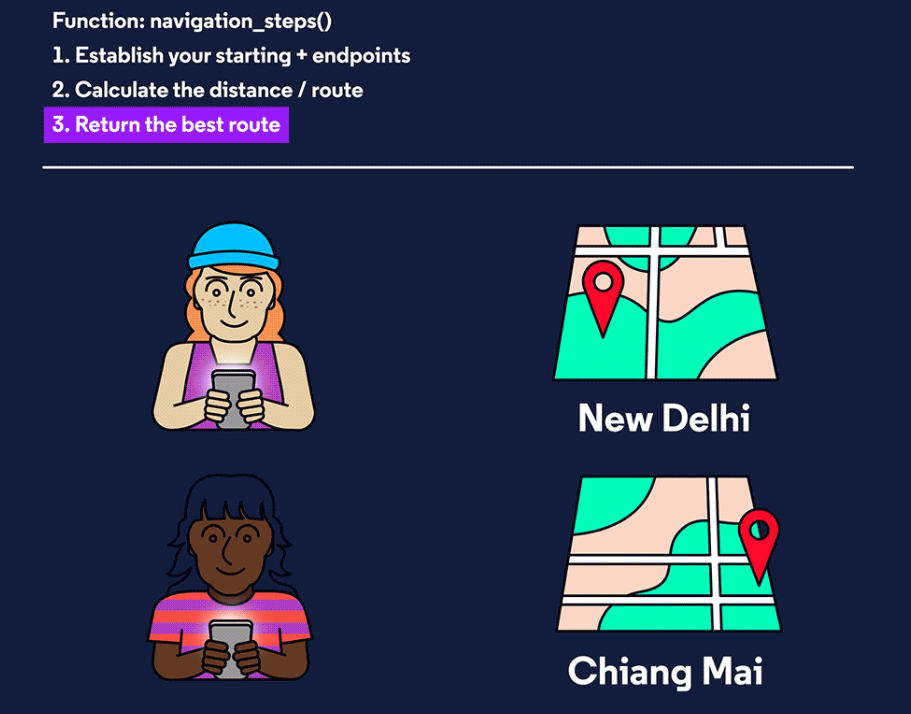
Notice how the function navigation\_steps() serves as a container for the three steps in the procedure and can be reused across multiple users as they plan their trips to different locations.

Click **Next** when you are ready to learn more about functions.









**Why Functions?**

Let’s come back to the trip planning application we just discussed in the previous exercise. The steps we talked about for our program were:

1. Establish an origin and destination

2. Calculate the distance/route

3. Return the best route

If we were to convert our steps into Python code, a very simple version that plans a trip between two popular New York tourist destinations might look like this:

print("Setting the Empire State Building as the starting point and Times Square as our destination.")  
   
print("Calculating the total distance between our points.")   
   
print("The best route is by train and will take approximately 10 minutes.")

Anytime we want to go between these two points we would need to run these three print statements (for now we can assume the best route and time will stay the same).

If our program now had 100 new people trying to find the best directions between the Empire State Building and Times Square, we would need to run each of our three print statements 100 times!

Now, if you’re thinking about using a loop here, your intuition would be totally right! Unfortunately, we won’t be always traveling between the same two locations which means a loop won’t be as effective when we want to customize a trip. We will address this in the upcoming sections!

For now, let’s gain an appreciation for functions.

**Instructions**

**1.**

***Run*** the pre-written print() statements to see what they output.

Checkpoint 2 Passed

**2.**

Write the same set of print statements three more times. Run the code again and see the output.

Checkpoint 3 Passed

Hint

Make sure that the three print statements are all duplicated three more times.

**3.**

Hopefully now you have some perspective about your life without functions!

In the next section, we will learn how we can [refactor](https://en.wikipedia.org/wiki/Code_refactoring) our code to utilize functions to reuse code.

Click ***Run*** your code again and then click ***Next*** to continue.

**travel.py**

# First user wants to travel between these two points!

print("Setting the Empire State Building as the starting point and Times Square as our destination.")

print("Calculating the total distance between our points.")

print("The best route is by train and will take approximately 10 minutes.")

# Second user wants to travel between these two points!

print("Setting the Empire State Building as the starting point and Times Square as our destination.")

print("Calculating the total distance between our points.")

print("The best route is by train and will take approximately 10 minutes.")

# Third user wants to travel between these two points!

print("Setting the Empire State Building as the starting point and Times Square as our destination.")

print("Calculating the total distance between our points.")

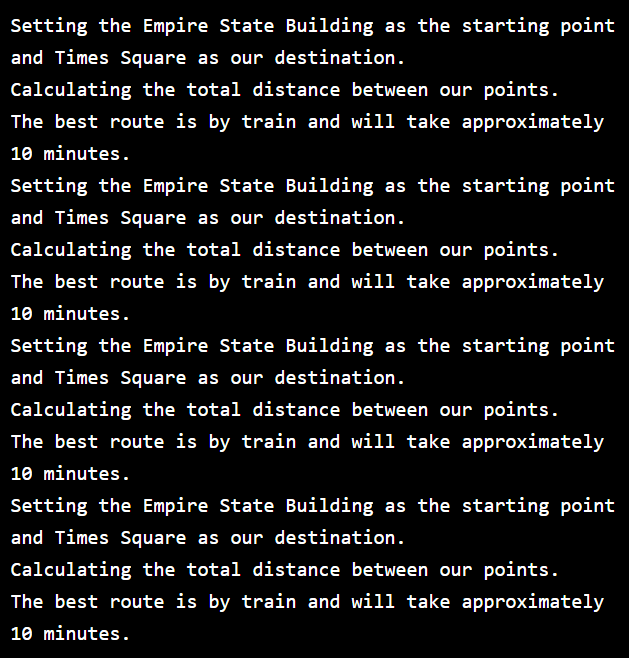
print("The best route is by train and will take approximately 10 minutes.")

# Fourth user wants to travel between these two points!

print("Setting the Empire State Building as the starting point and Times Square as our destination.")

print("Calculating the total distance between our points.")

print("The best route is by train and will take approximately 10 minutes.")



**Defining a Function**

A *function* consists of many parts, so let’s first get familiar with its core - a function definition.

Here’s an example of a function definition:

def function\_name():  
  # functions tasks go here

There are some key components we want to note here:

* The def keyword indicates the beginning of a function (also known as a function header). The function header is followed by a name in snake\_case format that describes the task the function performs. It’s best practice to give your functions a descriptive yet concise name.
* Following the function name is a pair of parenthesis ( ) that can hold input values known as parameters (more on parameters later in the lesson!). In this example function, we have no parameters.
* A colon : to mark the end of the function header.
* Lastly, we have one or more valid python statements that make up the function body (where we have our python comment).

Notice we’ve indented our # function tasks go here comment. Like loops and conditionals, code inside a function must be indented to show that they are part of the function.

Here is an example of a function that greets a user for our trip planning application:

def trip\_welcome():  
  print("Welcome to Tripcademy!")   
  print("Let's get you to your destination.")

***Note:*** Pasting this code into the editor and clicking ***Run*** will result in an empty output terminal. The print() statements within the function will not execute since our function hasn’t been used. We will explore this further in the next exercise; for now, let’s practice defining a function.

**Instructions**

**1.**

Two of the most common NYC attractions include the Empire State Building and Times Square.

In *travel.py*, we’ll write a function that prints the directions via subway from the Empire State Building to Times Square.

First, define a function, directions\_to\_timesSq(). Leave the body of the function empty for now.

***Note:*** When we run the code, we will see an error: SyntaxError: unexpected EOF while parsing. This will occur when we don’t populate a function with any statements. We will populate it with code in the next step.

EOF stands for “End of File” — Python is telling you that it was expecting some code in the body of the function, but it hit the end of the file first.

Checkpoint 2 Passed

Hint

Remember the core of a function - a definition. Check to make sure you have all the components for a function definition.

def my\_function\_name():

**2.**

Within the body of the function, use three print() statements to output the following directions:

Walk 4 mins to 34th St Herald Square train station  
Take the Northbound N, Q, R, or W train 1 stop  
Get off the Times Square 42nd Street stop

Remember, if you run your code, you shouldn’t see any output in the terminal at this point. Check out the hint if you want to see how to see the output (we will be doing it in the next section as well!)

Checkpoint 3 Passed

Hint

Check your statements for spaces, capitalization, and spelling. If you are still having problems getting the checkpoint to pass, try and copy and paste the text directly from the narrative into your print statement.

If you are interested in seeing the output, call your function like this:

directions\_to\_timesSq()

**travel.py**

# Your code below:

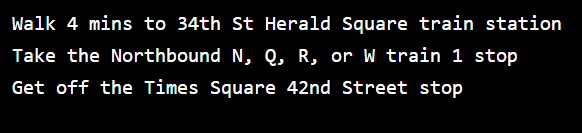
def directions\_to\_timesSq():

  print("Walk 4 mins to 34th St Herald Square train station")

  print("Take the Northbound N, Q, R, or W train 1 stop")

  print("Get off the Times Square 42nd Street stop")

directions\_to\_timesSq()

****