Since we want to test both of these images, let's create a variable called image_path that contains the path of the image we want to test.

```
image_path = 'data/test/not_wildfire/non_fire.205.png'
```

Now, we can define a function called predict() that takes in the image path and returns an output 1 or 0, depending on the neural network's classification.

Inside the function, we're first going to load the model that we saved in cnn.py, and then we are going to load the image and convert it to an array. Let's do this now:

```
def  predict(image_path):
    cnn = load_model('model')
    image = load_img(f'{image_path}', target_size=(128, 128))
    image = img_to_array(image)
```

Notice how we use the same image size as we did when training the network. This is necessary as the neural network expects the same size image each time it predicts; if the size is different, the network will throw an error.

There's just one small technicality we have to deal with before we use the classifier to predict what category the image belongs to. When we trained our network, we created a DirectoryIterator with a column containing the batch each image was in.

Since we're only testing on a single image, there are no batches; this will cause the classifier to raise an error because of missing dimension. Thus, we add a column containing the integer 1, indicating an arbitrary batch number.

After doing this, we can use the .predict method of our model and input the image we have just transformed. Since neural networks output the probability of the image being a certain category in a double nested array, we can simply index twice into the returned prediction and transform the number into an integer.

```
def predict(image_path):
```

```
    cnn = load_model('model')
    image = load_img(image_path, target_size=(128, 128))
    image = img_to_array(image)
    image = np.expand_dims(image, axis=0)
    prediction = int(cnn.predict(image)[0][0])
    return prediction
```

Nice! Now we can call the function and assign the return value into a variable called prediction.

```
prediction = predict(image_path)
```

Now, although we don't strictly need to, let's create a simple if/else statement to make the program print either 'Wildfire' or 'Not a wildfire' to make it easier for us to read the output.

```
if prediction == 1:
    print('Wildfire')
elif prediction == 0:
    print('Not a wildfire')
```

If we run the program, we will receive the output 'Not a wildfire', which is exactly what we were expecting. Let's test the network on the other image that we had selected by changing the image path.

```
image_path = 'data/test/wildfire/fire.619.png'
```

This time, we receive the output ‘Wildfire’, which means that our neural network has once again successfully categorized the image!