### Prueba de la clase EncuestaAnónima

Vamos a escribir una prueba que verifique un aspecto del comportamiento de AnonymousSurvey. Escribiremos una prueba para verificar que una única respuesta a la pregunta de la encuesta se almacena correctamente:

**test\_survey.py**

from survey import AnonymousSurvey  
  
❶ def test\_store\_single\_response():  
 """Test that a single response is stored properly."""  
 question = "What language did you first learn to speak?"  
❷ language\_survey = AnonymousSurvey(question)  
 language\_survey.store\_response('English')  
❸ assert 'English' in language\_survey.responses

Empezaremos importando la clase que queremos probar, AnonymousSurvey. La primera función de prueba verificará que cuando almacenemos una respuesta a la pregunta de la encuesta, la respuesta acabará en la lista de respuestas de la encuesta. Un buen nombre descriptivo para esta función es test\_store\_single\_response() ❶. Si esta prueba falla, sabremos por el nombre de la función en el resumen de la prueba que hubo un problema al almacenar una única respuesta a la encuesta.

Para probar el comportamiento de una clase, necesitamos crear una instancia de la clase. Creamos una instancia llamada language\_survey ❷ con la pregunta "What language did you first learn to speak?" Almacenamos una única respuesta, English, utilizando el método store\_response(). A continuación, verificamos que la respuesta se ha almacenado correctamente afirmando que English está en la lista language\_survey.responses ❸.

Por defecto, la ejecución del comando pytest sin argumentos ejecutará todas las pruebas que pytest descubra en el directorio actual. Para centrarte en las pruebas de un archivo, pasa el nombre del archivo de pruebas que quieras ejecutar. Aquí ejecutaremos sólo la prueba que escribimos para AnonymousSurvey:

$ pytest test\_survey.py  
========================= test session starts =========================  
--snip--  
test\_survey.py . [100%]  
========================== 1 passed in 0.01s ==========================

Es un buen comienzo, pero una encuesta sólo es útil si genera más de una respuesta. Verifiquemos que se pueden almacenar tres respuestas correctamente. Para ello, añadimos otro método a TestAnonymousSurvey:

from survey import AnonymousSurvey  
  
def test\_store\_single\_response():  
 --snip--  
  
def test\_store\_three\_responses():  
 """Test that three individual responses are stored properly."""  
 question = "What language did you first learn to speak?"  
 language\_survey = AnonymousSurvey(question)  
❶ responses = ['English', 'Spanish', 'Mandarin']  
 for response in responses:  
 language\_survey.store\_response(response)  
  
❷ for response in responses:  
 assert response in language\_survey.responses

Llamamos a la nueva función test\_store\_three\_responses(). Creamos un objeto encuesta igual que hicimos en test\_store\_single\_response(). Definimos una lista que contenga tres respuestas ❶ diferentes , y luego llamamos a store\_response() para cada una de estas respuestas. Una vez almacenadas las respuestas, escribimos otro bucle y afirmamos que cada respuesta está ahora en language\_survey.responses ❷.

Cuando volvemos a ejecutar el archivo de prueba, ambas pruebas (para una sola respuesta y para tres respuestas) pasan:

$ pytest test\_survey.py  
========================= test session starts =========================  
--snip--  
test\_survey.py .. [100%]  
========================== 2 passed in 0.01s ==========================

Esto funciona perfectamente. Sin embargo, estas pruebas son un poco repetitivas, así que utilizaremos otra función de pytest para hacerlas más eficientes.

[anterior](c11_18.html)[Subtema 19 de 22: (Ver todo)](c11.html)[siguiente](c11_20.html)