

Autor: Andrés Castellero Moriana

Examen: jUnit (15/Marzo/2021)

Nota aclaratoria:

Estás realizando un examen, y como tal no puedes manejar ningún material de referencia adicional, ni comunicarte de ninguna forma con nadie. Estás sólo tú con estas instrucciones y las herramientas imprescindibles para llevarlas a cabo (Eclipse y LibreOffice/Word). Si se detecta cualquier amago de comunicación, automáticamente suspenderás.

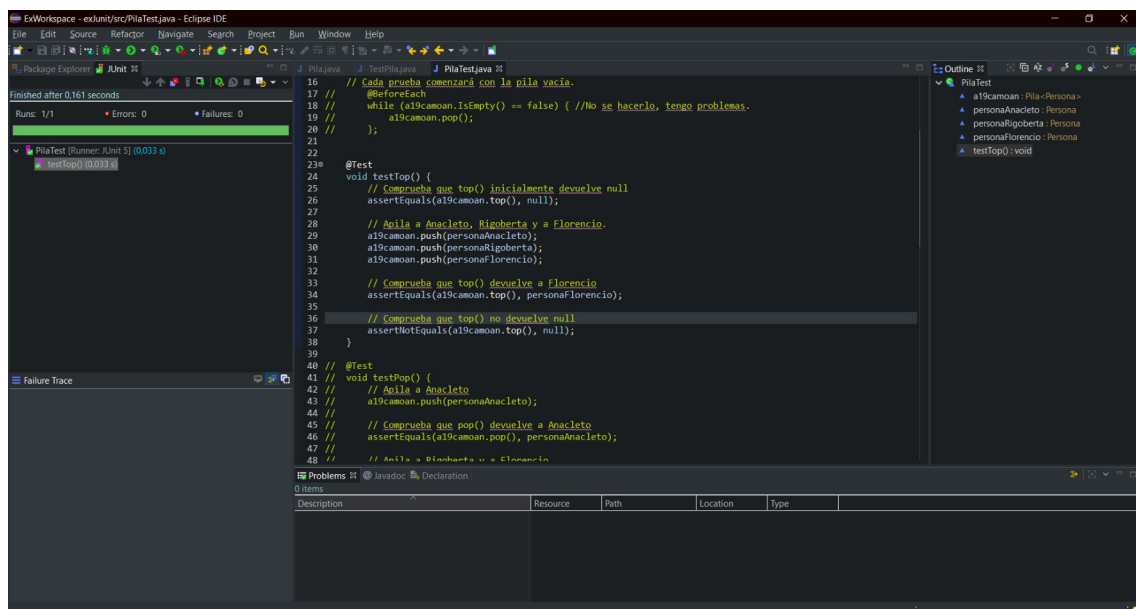
Utilizando el código adjunto que implementa una Pila, sigue las siguientes instrucciones. Al final deberás entregar un fichero comprimido

"NombreApellido1Apellido2ExJUnit.rar" con dos ficheros:

1. NombreApellido1Apellido2ExJUnit.pdf (Exportado a pdf desde OpenWriter)
2. NombreApellido1Apellido2ExJUnitProyecto. (Eclipse)

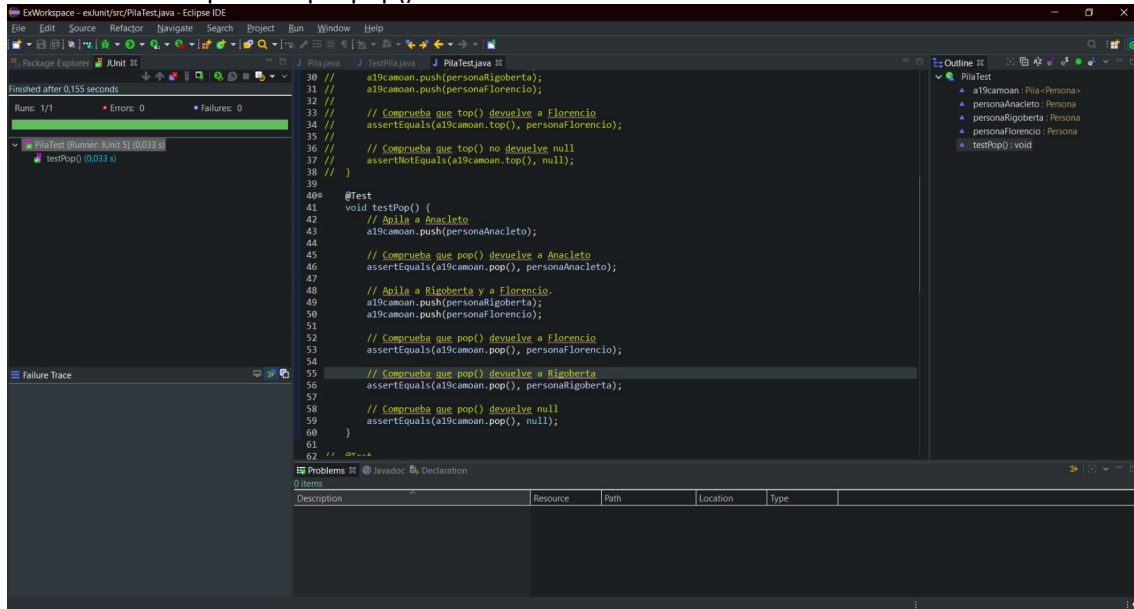
Vamos a diseñar un caso de pruebas para la clase Pila. Para ello trabajaremos sobre una pila de Personas (también se da en el código). Realiza los siguientes pasos:

- Abre Eclipse y crea un proyecto "NombreApellido1Apellido2ExJUnit".
- Añade el paquete pila que contiene Pila.java y TestPila.java.
- En el mismo paquete crea un caso de prueba jUnitPila sobre la clase Pila.
- Para todas las pruebas se van a necesitar tres objetos de la clase Persona. Serán los que apilemos/desapilemos de la Pila. Crea los tres objetos Persona con las siguientes características. Los referenciaremos en el orden indicado.
 - Anacleto Montes, Rigoberta Bosques, Florencio Arenas
- Además de los tres objetos necesitaremos un objeto Pila de Personas para cada una de las pruebas. Llámala como tu usuario "a20mamao".
- Cada prueba comenzará con la pila vacía.
- Debes probar los siguientes métodos:
 - top: Devuelve la cima de la pila.
 1. Comprueba que top() inicialmente devuelve null
 2. Apila a Anacleto, Rigoberta y a Florencio.
 3. Comprueba que top() devuelve a Florencio
 4. Comprueba que top() no devuelve null

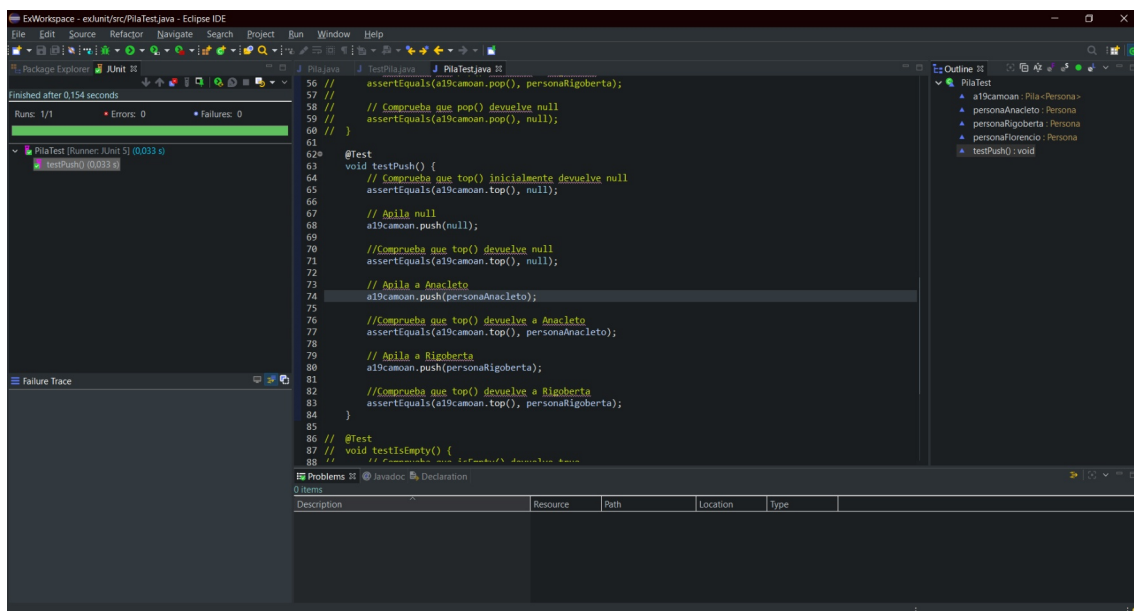


```
16 // Cada prueba comenzará con la pila vacía.
17 //
18 // @BeforeEach
19 // while (a19camoan.isEmpty() == false) { //No se hacerlo, tengo problemas.
20 //     a19camoan.pop();
21 // }
22
23 @Test
24 void testTop() {
25     // Comprueba que top() inicialmente devuelve null
26     assertEquals(a19camoan.top(), null);
27
28     // Apila a Anacleto, Rigoberta y a Florencio.
29     a19camoan.push(personaAnacleto);
30     a19camoan.push(personaRigoberta);
31     a19camoan.push(personaFlorencio);
32
33     // Comprueba que top() devuelve a Florencio
34     assertEquals(a19camoan.top(), personaFlorencio);
35
36     // Comprueba que top() no devuelve null
37     assertNotNull(a19camoan.top());
38 }
39
40 @Test
41 void testPop() {
42     // Apila a Anacleto
43     a19camoan.push(personaAnacleto);
44
45     // Comprueba que pop() devuelve a Anacleto
46     assertEquals(a19camoan.pop(), personaAnacleto);
47
48     // Apila a Rigoberta y a Florencio
```

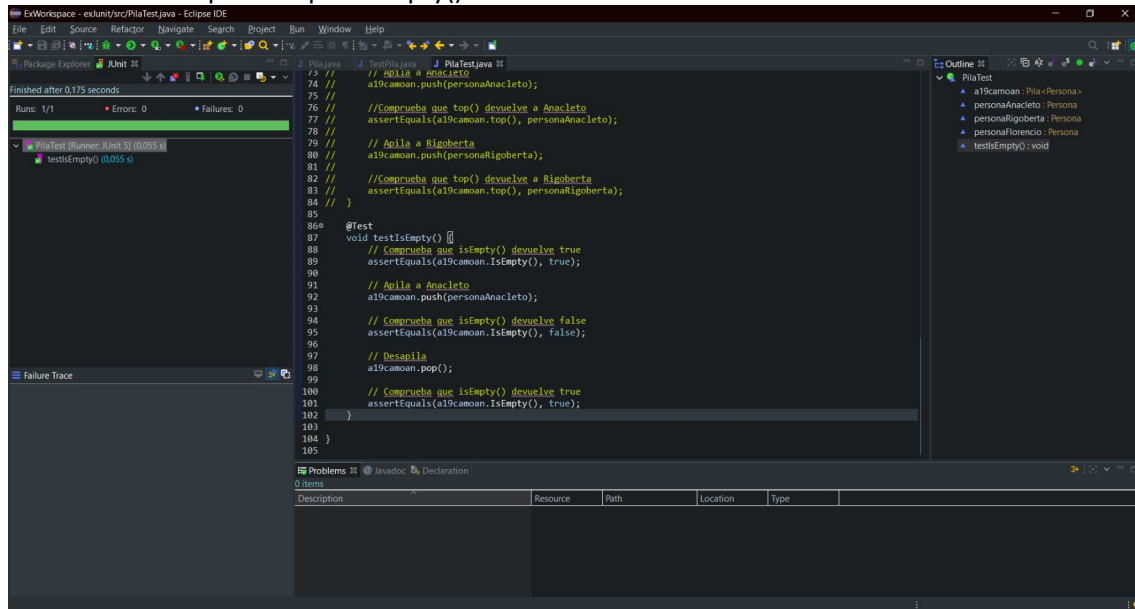
- pop: Desapila
- 1. Apila a Anacleto
- 2. Comprueba que pop() devuelve a Anacleto
- 3. Apila a Rigoberta y a Florencio.
- 4. Comprueba que pop() devuelve a Florencio
- 5. Comprueba que pop() devuelve a Rigoberta
- 6. Comprueba que pop() devuelve null



- push: apila
- 1. Comprueba que top() inicialmente devuelve null
- 2. Apila null
- 3. Comprueba que top() devuelve null
- 4. Apila a Anacleto
- 5. Comprueba que top() devuelve a Anacleto
- 6. Apila a Rigoberta
- 7. Comprueba que top() devuelve a Rigoberta



- isEmpty: comprueba si está vacía
- 1. Comprueba que isEmpty() devuelve true
- 2. Apila a Anacleto
- 3. Comprueba que isEmpty() devuelve false
- 4. Desapila
- 5. Comprueba que isEmpty() devuelve true



Entrega el proyecto, así como los siguientes pantallazos:

1. Cada uno de los métodos implementados
2. La vista con los tiempos en cada uno de los test. Todos han de estar chequeados correctamente, incluida la clase.

