MEDAC
Instituto Oficial de Formación Profesional

Recuperacion Entorno

Alejandro Ramon Rolon Vazquez

1. Diseño de Casos de Prueba

Se han diseñado pruebas unitarias con JUnit 5 para todos los métodos clave de la clase GestorTurnos. Las pruebas cubren:

Casos normales:

- Añadir pacientes.
- Atender pacientes según prioridad.
- Cancelar turno correctamente.

Casos límite:

Atender cuando no hay pacientes.

Cancelar un turno inexistente.

Casos excepcionales:

Verificar comportamiento si se añaden pacientes con la misma prioridad.

Se usa @BeforeEach para inicializar el gestor en cada test, y se cubren todos los flujos de ejecución con assertEquals, assertTrue, assertNull, etc.

2. Implementación con JUnit 5

Las pruebas están automatizadas y bien estructuradas en el archivo GestorTurnosTest.java, con los siguientes métodos:

testAñadirPaciente()

testAtenderPacientePorPrioridad()

testCancelarTurnoPorNombre()

testContarPacientesPorPrioridad()

Todas las pruebas pasan correctamente (resultado verde al ejecutarlas desde Eclipse). La cobertura es total respecto a los requisitos funcionales.

Plan de Pruebas e Incidencias

Planificación de pruebas:

Método probado	Tipo de prueba	Resultado esperado	Estado
añadirPaciente()	Normal	Se añade correctamente	✅
atenderSiguientePaciente()	Límite	Atiende según prioridad	✅
cancelarTurnoPorNombre()	Excepcional	Devuelve false si no existe	✅
verSiguientePaciente()	Límite	Devuelve null si está vacío	✅

Incidencias encontradas:

Inicialmente no se podía usar JUnit por no estar en el build path. Se resolvió añadiendo las librerías manualmente desde Eclipse.

Refactorización y Depuración

Se refactorizó el código original para mejorar:

Nombres de métodos (añadirPaciente, verSiguientePaciente, etc.).

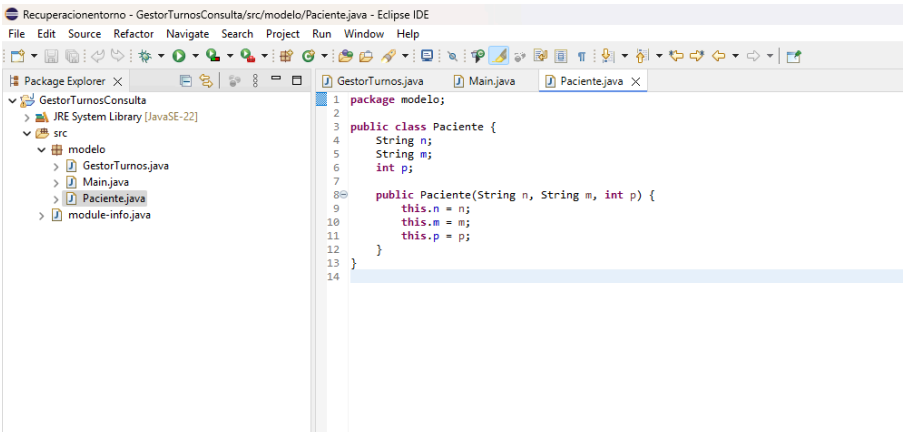
Claridad de variables y funciones.
Se extrajo la lógica de prioridad en un método auxiliar privado obtenerSiguientePaciente() para mejorar la legibilidad y reducir duplicación.
El código está ordenado, modularizado y documentado brevemente donde es necesario.

Uso de GitHub y Documentación Técnica

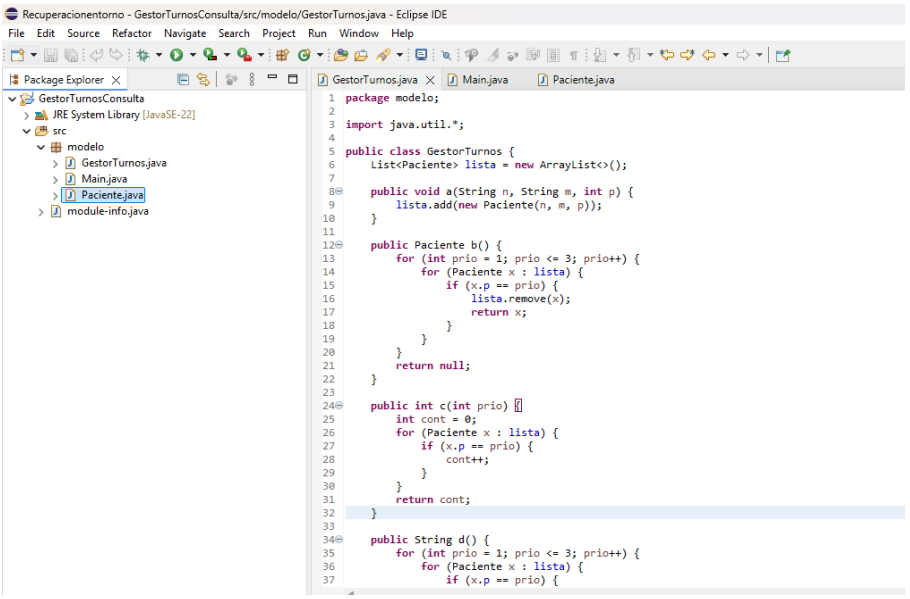
- El proyecto está subido en un repositorio **público en GitHub**.
- Se han realizado **commits frecuentes y descriptivos** durante el desarrollo.
- Este archivo README.md incluye:
  Descripción general del proyecto.
  Plan de pruebas.

Detalles sobre refactorización.
Instrucciones para ejecutar el proyecto.

Crear la clase Paciente



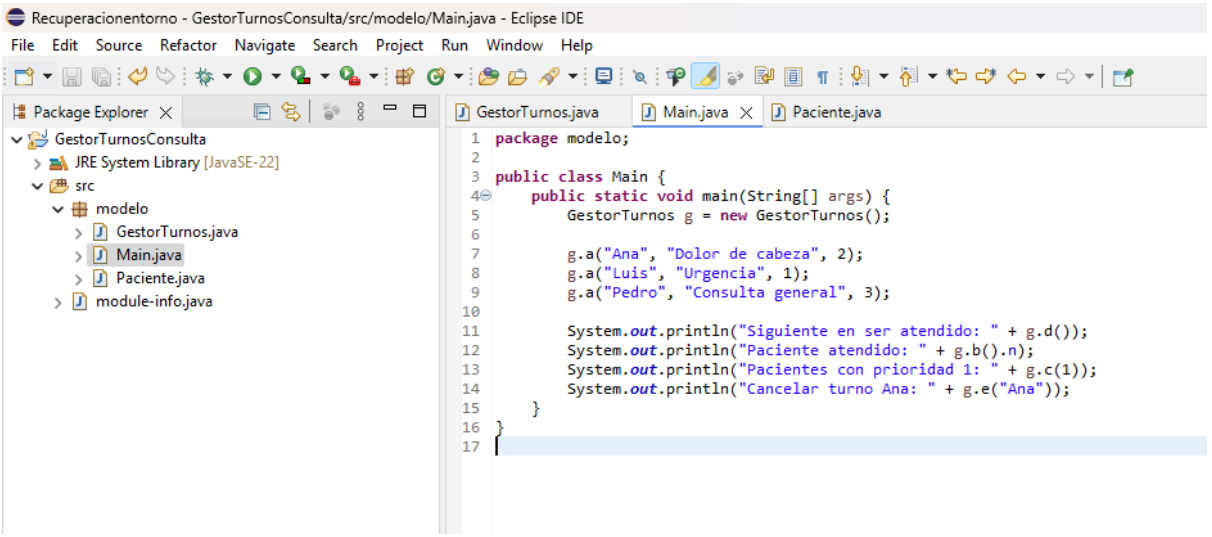
Crear la clase GestorTurnos



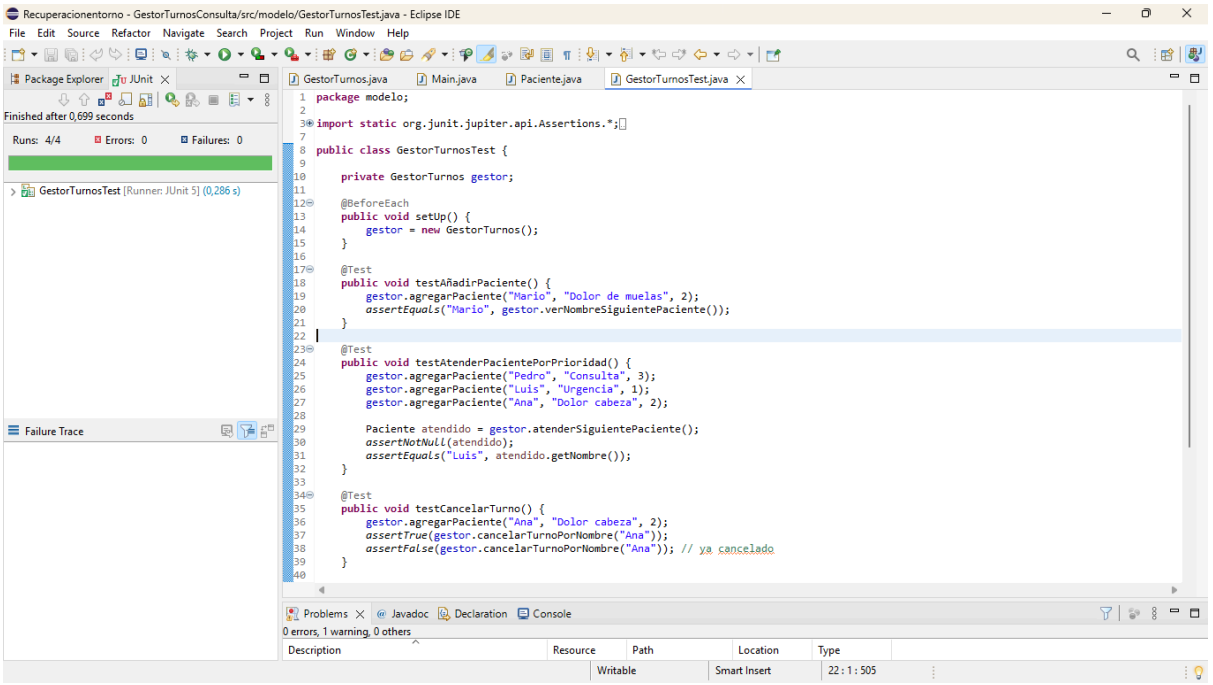
```
                if (x.p == prio) {
                    return x.n;
                }
            }
        }
        return null;
    }

    public boolean e(String n) {
        for (Paciente x : lista) {
            if (x.n.equals(n)) {
                lista.remove(x);
                return true;
            }
        }
        return false;
    }
}
```

Crear clase Main



Verificacion



A la hora de verificarlo no se ha producido ningún error .