Los *tests* generados automáticamente por Pynguin presentaron diferentes *Code smells*, entre los cuales se encuentran *Test code duplication*, debido a que en distintos tests se usaban exactamente la misma línea de código ya sea para crear una instancia de *NumberDisplay* o de *ClockDisplay*, lo cual se puede mejorar encapsulando estos *tests* en una clase general que tenga un *setUp* en donde se creen estas instancias para reutilizarlas.
También se encontró el *code smell Eager test*, ya que hay distintos *tests* que simplemente esperan que algo falle, sin especificar qué tipo de error se espera, y por lo tanto no se sabe explícitamente que se está testeando en cada uno.
Por último, se encontró el *code smell Lazy test*, ya que hay muchos *test* que prueban esencialmente la misma funcionalidad solamente cambiando el input en cada uno.

```
 - Total locations identified: 15
 - Location sample coverage: 66.67 %


Running time details
--------------------
 - Clean trial 1 run time: 0:00:00.159836
 - Clean trial 2 run time: 0:00:00.158090
 - Mutation trials total run time: 0:00:03.950300

2022-09-23 23:46:57,901: Trial Summary Report:

Overall mutation trial summary
==============================
 - DETECTED: 15
 - SURVIVED: 8
 - TOTAL RUNS: 23
 - RUN DATETIME: 2022-09-23 23:46:57.901024

2022-09-23 23:46:57,901: Detected mutations:

DETECTED
--------
 - src/display_number.py: (l: 9, c: 22) - mutation from <class 'ast.Mod'> to <class 'ast.Div'>
 - src/display_number.py: (l: 9, c: 23) - mutation from <class 'ast.Add'> to <class 'ast.Mod'>
 - src/display_number.py: (l: 9, c: 23) - mutation from <class 'ast.Add'> to <class 'ast.Pow'>
 - src/display_number.py: (l: 9, c: 23) - mutation from <class 'ast.Add'> to <class 'ast.Div'>
 - src/display_number.py: (l: 9, c: 23) - mutation from <class 'ast.Add'> to <class 'ast.Mult'>
 - src/display_number.py: (l: 9, c: 23) - mutation from <class 'ast.Add'> to <class 'ast.Sub'>
 - src/display_number.py: (l: 9, c: 23) - mutation from <class 'ast.Add'> to <class 'ast.FloorDiv'>
 - src/display_number.py: (l: 17, c: 11) - mutation from <class 'ast.Lt'> to <class 'ast.GtE'>
 - src/display_number.py: (l: 17, c: 11) - mutation from <class 'ast.Lt'> to <class 'ast.Eq'>
 - src/display_number.py: (l: 18, c: 21) - mutation from <class 'ast.Add'> to <class 'ast.Sub'>
 - src/display_number.py: (l: 18, c: 21) - mutation from <class 'ast.Add'> to <class 'ast.Mult'>
 - src/display_number.py: (l: 18, c: 21) - mutation from <class 'ast.Add'> to <class 'ast.FloorDiv'>
 - src/display_number.py: (l: 18, c: 21) - mutation from <class 'ast.Add'> to <class 'ast.Div'>
 - src/display_number.py: (l: 18, c: 21) - mutation from <class 'ast.Add'> to <class 'ast.Mod'>
 - src/display_number.py: (l: 18, c: 21) - mutation from <class 'ast.Add'> to <class 'ast.Pow'>

2022-09-23 23:46:57,901: Timedout mutations:

2022-09-23 23:46:57,901: Surviving mutations:

SURVIVED
--------
 - src/clock_display.py: (l: 4, c: 38) - mutation from None to False
 - src/clock_display.py: (l: 9, c: 14) - mutation from <class 'ast.GtE'> to <class 'ast.NotEq'>
 - src/clock_display.py: (l: 9, c: 14) - mutation from <class 'ast.And'> to <class 'ast.Or'>
 - src/display_number.py: (l: 3, c: 41) - mutation from None to False
 - src/display_number.py: (l: 9, c: 22) - mutation from <class 'ast.Mod'> to <class 'ast.FloorDiv'>
 - src/display_number.py: (l: 10, c: 15) - mutation from <class 'ast.Eq'> to <class 'ast.Lt'>
 - src/display_number.py: (l: 17, c: 11) - mutation from <class 'ast.Lt'> to <class 'ast.LtE'>
 - src/display_number.py: (l: 22, c: 15) - mutation from <class 'ast.Lt'> to <class 'ast.Gt'>
```

La mayoria de las mutaciones que sobrevivieron no tienen relevancia para el contexto actual, ya que esos cambios no afectan el comportamiento del código.
Los mutantes sobrevivientes son equivalentes al código.