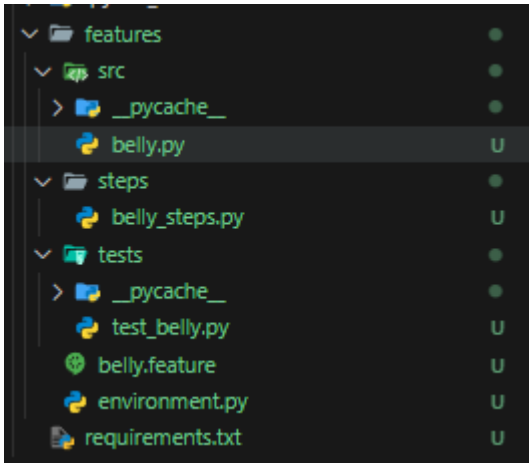
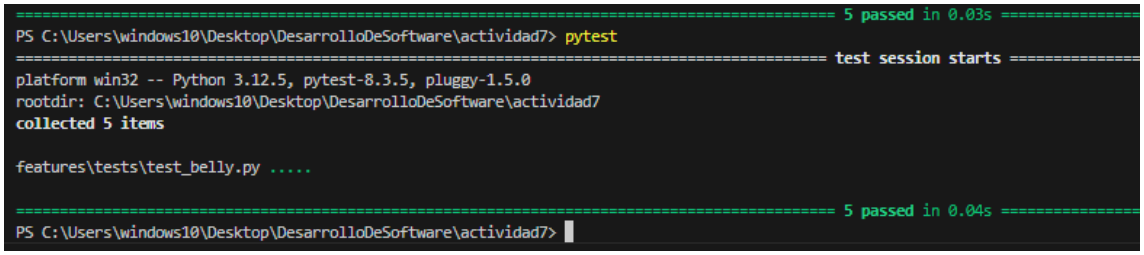
ACTIVIDAD 7;

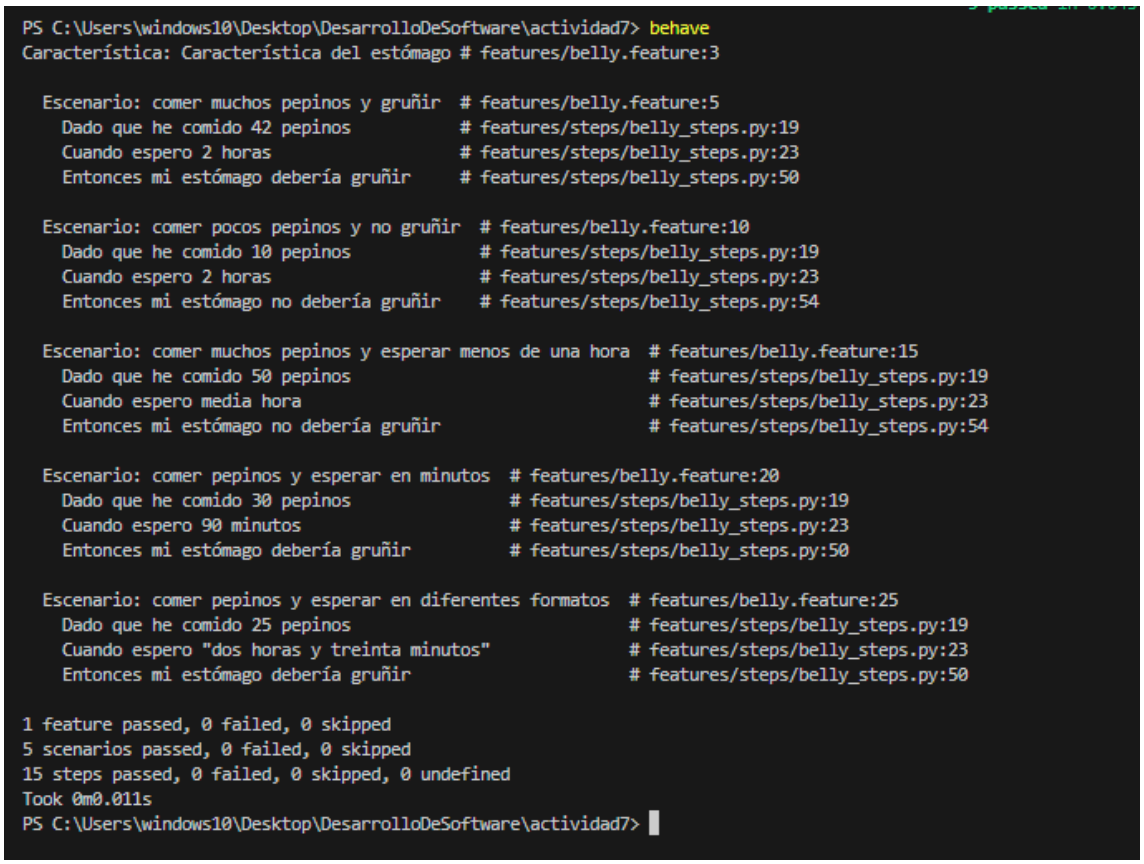
Estructura del proyecto



Hacemos las pruebas de de testeo luego de instalar las librerias



Ejecutamos behave y notamos que esta correctamente



Agregamos los número letras en ingles para convertirlo a numero

```python
# Función para convertir palabras numéricas a números
def convertir_palabra_a_numero(palabra):
    try:
        return float(palabra)
    except ValueError:
        numeros = {
            # Español
            "media": 0.5, "cero": 0, "uno": 1, "una": 1, "dos": 2, "tres": 3, "cuatro": 4,
            "cinco": 5, "seis": 6, "siete": 7, "ocho": 8, "nueve": 9, "diez": 10, "once": 11,
            "doce": 12, "trece": 13, "catorce": 14, "quince": 15, "veinte": 20, "treinta": 30,
            # Inglés
            "half": 0.5, "zero": 0, "one": 1, "two": 2, "three": 3, "four": 4, "five": 5,
            "six": 6, "seven": 7, "eight": 8, "nine": 9, "ten": 10, "eleven": 11,
            "twelve": 12, "thirteen": 13, "fourteen": 14, "fifteen": 15, "twenty": 20,
            "thirty": 30
        }
        return numeros.get(palabra.lower(), 0)

```

Ademas tambien agregamos los features

conflicted-file.md | belly_steps.py U | belly.feature U

actividad7 > features > belly.feature

```
3   Característica: Característica del estómago
31    Escenario: Comer una cantidad fraccionaria de pepinos
35
36    @english
37    Escenario: Esperar usando horas en inglés
38      Dado que he comido 50 pepinos
39      Cuando espero "two hours and thirty minutes"
40      Entonces mi estómago debería gruñir
41
42    @english
43    Escenario: Esperar solo 30 minutos en inglés
44      Dado que he comido 25 pepinos
45      Cuando espero "thirty minutes"
46      Entonces mi estómago no debería gruñir
47
```

Compilamos el codigo agregado en ingles

```
  @spanish
  Escenario: Comer una cantidad fraccionaria de pepinos  # features/belly.feature:31
    Dado que he comido 0.5 pepinos                       # None
    Cuando espero 2 horas                                # None
    Entonces mi estómago no debería gruñir               # None

  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:37
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:22
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:26
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:44

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:43
    Dado que he comido 25 pepinos               # features/steps/belly_steps.py:22
    Cuando espero "thirty minutes"              # features/steps/belly_steps.py:26
    Entonces mi estómago no debería gruñir      # features/steps/belly_steps.py:48

1 feature passed, 0 failed, 0 skipped
2 scenarios passed, 0 failed, 6 skipped
6 steps passed, 0 failed, 18 skipped, 0 undefined
Took 0m0.002s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```

Importamos la librería random

```
from behave import given, when, then
import re
import random
```

Colocamos un random entre los 2 valores

```
    )
    if match_random:
        low = float(match_random.group(1))
        high = float(match_random.group(2))
        total_time_in_hours = random.uniform(low, high)
        print(f"🕐 Tiempo aleatorio generado: {total_time_in_hours:.2f} horas")
        context.belly.esperar(total_time_in_hours)
        return
```

Y agregamos el feature

```
  Escenario: Comer pepinos y esperar un tiempo aleatorio
    Dado que he comido 25 pepinos
    Cuando espero un tiempo aleatorio entre 1 y 3 horas
    Entonces mi estómago debería gruñir
```

No tenemos ningún error

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

    Cuando espero media hora                                  # features/steps/belly_steps.py:42
    Entonces mi estómago no debería gruñir                    # features/steps/belly_steps.py:88

  Escenario: comer pepinos y esperar en minutos  # features/belly.feature:20
    Dado que he comido 30 pepinos                # features/steps/belly_steps.py:37
    Cuando espero 90 minutos                     # features/steps/belly_steps.py:42
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:84

  Escenario: comer pepinos y esperar en diferentes formatos  # features/belly.feature:25
    Dado que he comido 25 pepinos                            # features/steps/belly_steps.py:37
    Cuando espero "dos horas y treinta minutos"              # features/steps/belly_steps.py:42
    Entonces mi estómago debería gruñir                      # features/steps/belly_steps.py:84

  Escenario: Comer pepinos y esperar un tiempo aleatorio  # features/belly.feature:30
    Dado que he comido 25 pepinos                         # features/steps/belly_steps.py:37
    Cuando espero un tiempo aleatorio entre 1 y 3 horas   # features/steps/belly_steps.py:42
    Entonces mi estómago debería gruñir                   # features/steps/belly_steps.py:84

  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:36
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:37
  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:36
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:37
  Escenario: Esperar usando horas en inglés      # features/belly.feature:36
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:37
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:37
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:42
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:84

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:42
    Dado que he comido 25 pepinos               # features/steps/belly_steps.py:37
    Cuando espero "thirty minutes"              # features/steps/belly_steps.py:42
    Entonces mi estómago no debería gruñir      # features/steps/belly_steps.py:88

1 feature passed, 0 failed, 0 skipped
8 scenarios passed, 0 failed, 0 skipped
24 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.010s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```

Agregamos los ejemplos

```
  Escenario: Manejar tiempos complejos
    Dado que he comido 50 pepinos
    Cuando espero "1 hora, 30 minutos y 45 segundos"
    Entonces mi estómago debería gruñir

  Escenario: Manejar tiempos complejos
    Dado que he comido 20 pepinos
    Cuando espero "1 hora, 10 minutos y 10 segundos"
    Entonces mi estómago no debería gruñir
```

Modificamos el codigo para los casos parecido

```python
    if match_random:
        low = float(match_random.group(1))
        high = float(match_random.group(2))
        total_time_in_hours = random.uniform(low, high)
        print(f"🕐 Tiempo aleatorio generado: {total_time_in_hours:.2f} horas")
        context.belly.esperar(total_time_in_hours)
        return

    # Normalizar "y" y "and"
    time_description = time_description.replace(' y ', ' ').replace(' and ', ' ')

    # Regex que captura horas, minutos y segundos (número o palabra)
    pattern = re.compile(
        r'(?:(\d+(?:\.\d+)?|\w+)\s*(horas?|hours?))?'      # grupo 1 horas
        r'\s*(?:(\d+(?:\.\d+)?|\w+)\s*(minutos?|minutes?))?'# grupo 3 minutos
        r'\s*(?:(\d+(?:\.\d+)?|\w+)\s*(segundos?|seconds?))?'# grupo 5 segundos
    )
    match = pattern.match(time_description)

    if match:
        raw_hours = match.group(1) or "0"
        raw_minutes = match.group(3) or "0"
        raw_seconds = match.group(5) or "0"

        hours = convertir_palabra_a_numero(raw_hours)
        minutes = convertir_palabra_a_numero(raw_minutes)
        seconds = convertir_palabra_a_numero(raw_seconds)

        total_time_in_hours = hours + (minutes / 60) + (seconds / 3600)
        print(f"⏳ Tiempo calculado: {total_time_in_hours:.4f} horas")
        context.belly.esperar(total_time_in_hours)
    else:
        raise ValueError(f"No se pudo interpretar la descripción del tiempo: {time_description}")
```

Y ejecutamos los resultados

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:47
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:36
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:86

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:53
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:86

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:53

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:53
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:53
    Dado que he comido 25 pepinos                # features/steps/belly_steps.py:36
    Cuando espero "thirty minutes"               # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir       # features/steps/belly_steps.py:90

1 feature passed, 0 failed, 0 skipped
10 scenarios passed, 0 failed, 0 skipped
30 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.017s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```

Agregamos la función para testear

```python
def test_gruñir_si_comido_muchos_pepinos():
    belly = Belly()
    belly.comer(15)
    belly.esperar(2)
    assert belly.esta_gruñendo() == True
```

Y ejecutamos la función

```
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7> py.test.exe
============================= test session starts =============================
platform win32 -- Python 3.12.5, pytest-8.3.5, pluggy-1.5.0
rootdir: C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7
collected 6 items

features\tests\test_belly.py ......

============================== 6 passed in 0.01s ==============================
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```

Agregamos el feature y ejecutamos

```
39
40    Escenario: Comer muchos pepinos y esperar el tiempo suficiente
41      Dado que he comido 15 pepinos
42      Cuando espero 2 horas
43      Entonces mi estómago debería gruñir
44
45    Escenario: Manejar tiempos complejos
46      Dado que he comido 20 pepinos
47      Cuando espero "1 hora, 10 minutos y 10 segundos"
48      Entonces mi estómago no debería gruñir
49
50
51    @english
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
  Escenario: Comer muchos pepinos y esperar el tiempo suficiente  # features/belly.feature:40
    Dado que he comido 15 pepinos                                 # features/steps/belly_steps.py:36
    Cuando espero 2 horas                                         # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir                           # features/steps/belly_steps.py:86

  Escenario: Manejar tiempos complejos                 # features/belly.feature:45
    Dado que he comido 20 pepinos                      # features/steps/belly_steps.py:36
    Cuando espero "1 hora, 10 minutos y 10 segundos" # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir             # features/steps/belly_steps.py:90

  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:52
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:36
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:86

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:58
    Dado que he comido 25 pepinos               # features/steps/belly_steps.py:36
    Cuando espero "thirty minutes"              # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir      # features/steps/belly_steps.py:90

1 feature passed, 0 failed, 0 skipped
11 scenarios passed, 0 failed, 0 skipped
33 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.014s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```

Agregamos los 2 escenarios

```
  Escenario: Comer suficientes pepinos y esperar el tiempo adecuado
    Dado que he comido 20 pepinos
    Cuando espero 2 horas
    Entonces mi estómago debería gruñir

  Escenario: Comer pocos pepinos y no esperar suficiente tiempo
    Dado que he comido 5 pepinos
    Cuando espero 1 hora
    Entonces mi estómago no debería gruñir
```

Y ejecutamos

```
  Escenario: Comer muchos pepinos y esperar el tiempo suficiente  # features/belly.feature:50
    Dado que he comido 15 pepinos                                 # features/steps/belly_steps.py:36
    Cuando espero 2 horas                                         # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir                           # features/steps/belly_steps.py:86

  Escenario: Manejar tiempos complejos                # features/belly.feature:55
    Dado que he comido 20 pepinos                     # features/steps/belly_steps.py:36
    Cuando espero "1 hora, 10 minutos y 10 segundos" # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir            # features/steps/belly_steps.py:90

  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:62
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:36
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:86

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:68
    Dado que he comido 25 pepinos               # features/steps/belly_steps.py:36
    Cuando espero "thirty minutes"              # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir      # features/steps/belly_steps.py:90

1 feature passed, 0 failed, 0 skipped
13 scenarios passed, 0 failed, 0 skipped
39 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.017s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```

Agregamos la función y ejecutamos

```
42
43  def test_pepinos_restantes():
44      belly = Belly()
45      belly.comer(15)
46      assert belly.pepinos_comidos == 15
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:62
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:36
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:86

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:68
    Dado que he comido 25 pepinos               # features/steps/belly_steps.py:36
    Cuando espero "thirty minutes"              # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir      # features/steps/belly_steps.py:90

1 feature passed, 0 failed, 0 skipped
13 scenarios passed, 0 failed, 0 skipped
39 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.017s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7> py.test.exe
============================================================================== test session start
platform win32 -- Python 3.12.5, pytest-8.3.5, pluggy-1.5.0
rootdir: C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7
collected 7 items

features\tests\test_belly.py .......

============================================================================== 7 passed in 0.06s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7> ^C
```

Agregamos un then más

```
        raise ValueError(f"No se pudo interpretar la descripción del tiempo: {time_description}")

@then('mi estómago debería gruñir')
def step_then_belly_should_growl(context):
    assert context.belly.esta_gruñendo(), "Se esperaba que el estómago gruñera, pero no lo hizo."
```

Ejecutamos

```
@then('mi estómago debería gruñir')
def step_then_belly_should_growl(context):
    assert context.belly.esta_gruñendo(), "Se esperaba que el estómago gruñera, pero no lo hizo."

@then('mi estómago no debería gruñir')
def step_then_belly_should_not_growl(context):
    assert not context.belly.esta_gruñendo(), "Se esperaba que el estómago no gruñera, pero lo hizo."

# Nuevo step para verificar cuántos pepinos se han comido
@then('debería haber comido {expected:d} pepinos')
def step_then_should_have_eaten_cukes(context, expected):
    eaten = getattr(context.belly, 'pepinos_comidos', None)
    assert eaten == expected, f"Se esperaban {expected} pepinos comidos, pero se encontraron {eaten}."
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:66
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:36
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:86

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:72
    Dado que he comido 25 pepinos               # features/steps/belly_steps.py:36
    Cuando espero "thirty minutes"              # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir      # features/steps/belly_steps.py:90

1 feature passed, 0 failed, 0 skipped
14 scenarios passed, 0 failed, 0 skipped
41 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.020s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```

Agregamos el nuevo tes y ejecutamos

```
def test_estomago_gruñendo():
    belly = Belly()
    belly.comer(20)
    belly.esperar(2)
    assert belly.esta_gruñendo() == True
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

    Dado que he comido 25 pepinos               # features/steps/belly_steps.py:36
    Cuando espero "thirty minutes"              # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir      # features/steps/belly_steps.py:90

1 feature passed, 0 failed, 0 skipped
14 scenarios passed, 0 failed, 0 skipped
41 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.020s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7> py.test.exe
============================================================================== t
platform win32 -- Python 3.12.5, pytest-8.3.5, pluggy-1.5.0
rootdir: C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7
collected 8 items

features\tests\test_belly.py ........

==============================================================================
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```

Agregamos un nuevo escenario y ejecutamos

```
  Escenario: Verificar que el estómago gruñe tras comer suficientes pepinos y esperar
    Dado que he comido 20 pepinos
    Cuando espero 2 horas
    Entonces mi estómago debería gruñir

  Escenario: Manejar tiempos complejos
    Dado que he comido 20 pepinos
    Cuando espero "1 hora, 10 minutos y 10 segundos"
    Entonces mi estómago no debería gruñir

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:70
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:36
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:86

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:76
    Dado que he comido 25 pepinos               # features/steps/belly_steps.py:36
    Cuando espero "thirty minutes"              # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir      # features/steps/belly_steps.py:90

1 feature passed, 0 failed, 0 skipped
15 scenarios passed, 0 failed, 0 skipped
44 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.015s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```

Agregamos el ultimo escenario y ejecutamos

```
  Escenario: Ver cuántos pepinos puedo comer antes de que el estómago gruña
    Dado que he comido 8 pepinos
    Cuando pregunto cuántos pepinos más puedo comer
    Entonces debería decirme que puedo comer 2 pepinos más

  Escenario: Verificar que el estómago gruñe tras comer suficientes pepinos y esperar

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

  @english
  Escenario: Esperar usando horas en inglés      # features/belly.feature:80
    Dado que he comido 50 pepinos                # features/steps/belly_steps.py:36
    Cuando espero "two hours and thirty minutes" # features/steps/belly_steps.py:40
    Entonces mi estómago debería gruñir          # features/steps/belly_steps.py:97

  @english
  Escenario: Esperar solo 30 minutos en inglés  # features/belly.feature:86
    Dado que he comido 25 pepinos               # features/steps/belly_steps.py:36
    Cuando espero "thirty minutes"              # features/steps/belly_steps.py:40
    Entonces mi estómago no debería gruñir      # features/steps/belly_steps.py:101

1 feature passed, 0 failed, 0 skipped
17 scenarios passed, 0 failed, 0 skipped
50 steps passed, 0 failed, 0 skipped, 0 undefined
Took 0m0.022s
PS C:\Users\windows10\Desktop\DesarrolloDeSoftware\actividad7>
```