# Yahtzee Version 1 Testing: Upper Section

The main focus of this week is **testing**, so now you will implement unit tests for some of your functions.

We should always test our code with various inputs. It might work correctly with some of them but not work with others. Now, you will test your functions by calling them and checking the output. You will also use the `pytest` module to run your tests.

The functions that you will test are:

- `sum_of_given_number()`
- `fill_upper_section()`

## What to do

Create a new file called `test_yahtzee1.py` in the same folder as `yahtzee1.py`.

1. Import the functions from `yahtzee1.py` that you want to test in the file `test_yahtzee1.py`.

2. Write three unit tests for the `sum_of_given_number` function, covering the following cases:
a) All dice have different face values
b) All dice have the same face value
c) Some dice have the same face value, other dice have different face value. Each test must have a hardcoded roll (that satisfies the test case) and six assertions, one for each possible `number` (1 to 6).

3. Write one unit test for the `fill_upper_section` function. This test must have a hardcoded roll and one assertion.

Use the following template. All unit tests defined in the template **must be present and implemented** in your code (you may **not** omit anything).

```
from yahtzee1 import sum_of_given_number, fill_upper_section

def test_sum_of_given_number_all_different():
    """
    Tests sum_of_given_number() with one roll
```

```
    where all dice have different face values.
    """
    roll = # TODO (hardcode a roll that satisfies this test case)
    assert sum_of_given_number(roll, 1) == # TODO
    assert sum_of_given_number(roll, 2) == # TODO
    assert sum_of_given_number(roll, 3) == # TODO
    assert sum_of_given_number(roll, 4) == # TODO
    assert sum_of_given_number(roll, 5) == # TODO
    assert sum_of_given_number(roll, 6) == # TODO
    # (in total, you have six assertions in this test)

def test_sum_of_given_number_all_same():
    """
    Test sum_of_given_number() with one roll
    where all dice have the same face value.
    """
    roll = # TODO (hardcode a roll that satisfies this test case)
    # TODO: Add assertions for all six values of `number`
    # (in total, you will have six assertions in this test)

def test_sum_of_given_number_some_different():
    """
    Test sum_of_given_number() with one roll
    where some dice have the same face value.
    """
    roll = # TODO (hardcode a roll that satisfies this test case)
    # TODO: Add assertions for all six values of `number`
    # (in total, you will have six assertions in this test)

def test_fill_upper_section():
    """
    Test fill_upper_section() with any one roll of your choice.
    """
    roll = # TODO (hardcode a roll that satisfies this test case)
    assert fill_upper_section(roll) == # TODO
```

Run your tests by calling `python -m pytest test_yahtzee1.py` in the terminal.

All tests should pass.

## How to debug

Your unit tests can be failing for two reasons:

1. Your code has errors

2. Your unit tests have errors

Make sure that you understand what is going wrong and ensure that the tests are correct before changing the code.

## Program name

Save your unit tests as `test_yahtzee1.py`.

## Demo

https://asciinema.org/a/4cjtjTqlrvSgOV6eVqQaHrPK3

## Submitting

Submit `test_yahtzee1.py` via eClass.