```
Wypisz wszystkie liczby od 1 do 100 - odpowiedzi:
- podzielna przez 3 -> Fizz / podzielna przez 5 -> Buzz
- podzielna przez 3 i 5 -> FizzBuzz / nie podzielna -> ta sama liczbę

1. Test shouldReturn1WhenNumberIs1
2. Metoda check zwraca 1 (w klasie test)
3. Refactor - statyczny imoport equals
4. Test shouldReturn2WhenNumberIs2
5. Metoda check zwraca value of number
6. Refactor - mozemy jeden z tych testow usunac
7. Test shouldReturnFizzWhenNumberIs3
8. Metoda check zwraca Fizz if number==3
9. Refactor - Zmienna statyczna Fizz
10. Metoda check zwraca Fizz if number%3
11. Test shouldReturnBuzzWhenNumberIs5
12. Metoda check zwraca Buzz if number%5
13. Test shouldReturnFizzBuzzWhenNumberIs15
14. Metoda check zwraca FizzBuzz if number%5 i number%3
15. Refaxctor = StringBuilder na FizzBuzz

private String check(Long number) {
    StringBuilder response = new StringBuilder();
    if (number % 3 == 0) {
        response.append(FIZZ);
    }
    if (number % 5 == 0) {
        response.append(BUZZ);
    }
    if (response.length() == 0) {
        response.append(number);
    }
    return response.toString();
}

private class FizzBuzz{
    final Long fizz;
    final Long buzz;

    public FizzBuzz(Long fizz, Long buzz) {
        this.fizz = fizz;
        this.buzz = buzz;
    }

    public FizzBuzz() {
        this(3L,5L);
    }

    private String check(Long number) {
        StringBuilder response = new StringBuilder();
        fizz(number, response);
        buzz(number, response);
        number(number, response);
        return response.toString();
    }

    private void number(Long number, StringBuilder response) {
        if (response.length() == 0) {
            response.append(number);
        }
    }

    private void buzz(Long number, StringBuilder response) {
        if (number % buzz == 0) {
            response.append(BUZZ);
        }
    }

    private void fizz(Long number, StringBuilder response) {
        if (number % fizz == 0) {
            response.append(FIZZ);
        }
    }
}
```