NOKIA

Nokia Networks

Praktyczne Aspekty Rozwoju Oprogramowania

TDD

- Paweł Wlezień
- Hanna Senhadri

Kwestie organizacyjne

- Kurs Praktyczne Aspekty Rozwoju Oprogramowania został przygotowany przez pracowników firmy Nokia Solutions and Networks
- Uczestnicy kursu otrzymają dawkę wiedzy z zakresu rozwoju oprogramowania wraz z przykładami zastosowań praktycznych
- Sprawdzenie listy obecności
- Na zajęciach przysługują dodatkowe punkty za aktywność
- Czy powinniśmy robić przerwę w trakcie zajęć?

Kwestie organizacyjne… a zdalna forma laboratorium

- Uczestnicy kursu otrzymają dawkę wiedzy z zakresu rozwoju oprogramowania wraz z przykładami zastosowań praktycznych – **część teoretyczna + praktyczna zajęć**
- Sprawdzenie listy obecności – **na podstawie Webexa**
- Na zajęciach przysługują dodatkowe punkty za aktywność:
  - Wykonanie przykładu publicznie
  - Prezentacja zadania publicznie
  - Pytania i odpowiedzi
- Okno czatu – pomagamy i odpowiadamy na pytania!
- Propozycje dobrych praktyk z poprzednich zajęć?

NOKIA

Nokia Networks

czym jest tdd?

NOKIA

Nokia Networks

test driven development

Zasady pracy w TDD

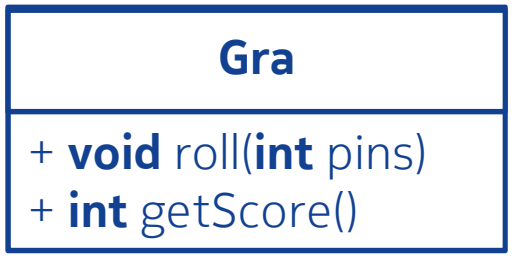
1. Piszemy test, który nie przechodzi (**RED**)
2. Piszemy minimalną implementację, która sprawi, że test przejdzie (**GREEN**)
3. Opcjonalnie dokonujemy refaktoryzacji kodu (**REFACTOR**)

Zadanie na dziś

Liczenie punktów podczas gry w kręgle

- 10 rund, po 2 rzuty
- 10 pinów
- Wynik za rundę: ilość przewróconych pinów

Wymagania:



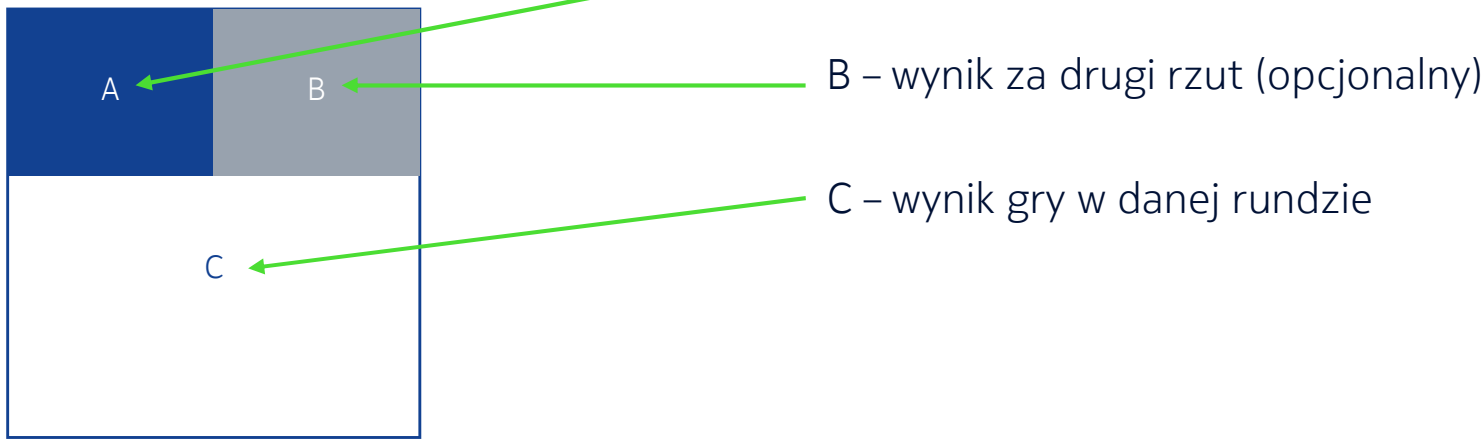
Źródło: Uncle Bob

Zadanie na dziś

Liczenie punktów podczas gry w kręgle

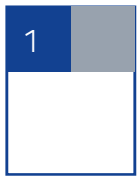
- 10 rund, po 2 rzuty
- 10 pinów
- Wynik za rundę: ilość przewróconych pinów + bonusy za spare i strike
- Spare: przewrócenie 10 pinów podczas dwóch rzutów
  - Bonus: Ilość przewróconych pinów podczas następnego rzutu
- Strike: przerwócenie 10 pinów podczas pierwszego rzutu
  - Bonus: Punkty zebrane podczas następnych dwóch rzutów

Notacja wyników



Prosta Gra

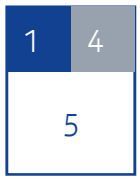
Gracz strąca 1 kręgiel w pierwszym rzucie.



Prosta Gra

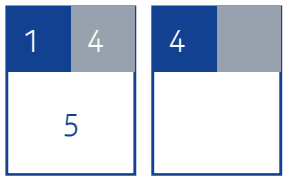
Gracz strąca 4 kręgle w następnym rzucie.

Suma za rundę wynosi 5.



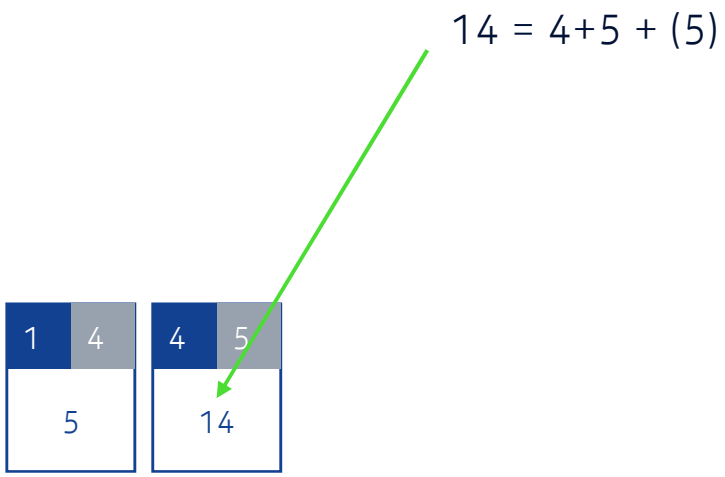
Prosta Gra

Gracz strąca 4 kręgle w następnym rzucie.



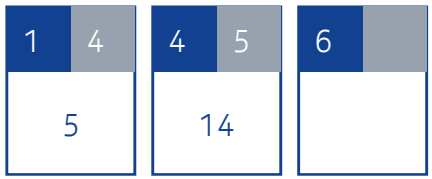
Prosta Gra

Gracz strąca 5 kręgli w następnym rzucie.



Prosta Gra

Gracz strąca 6 kręgli w następnym rzucie.

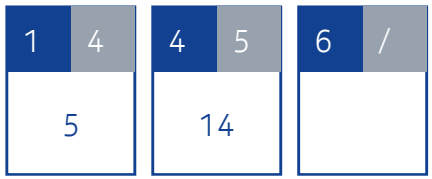


Prosta Gra

Gracz strąca 4 kręgle w następnym rzucie.

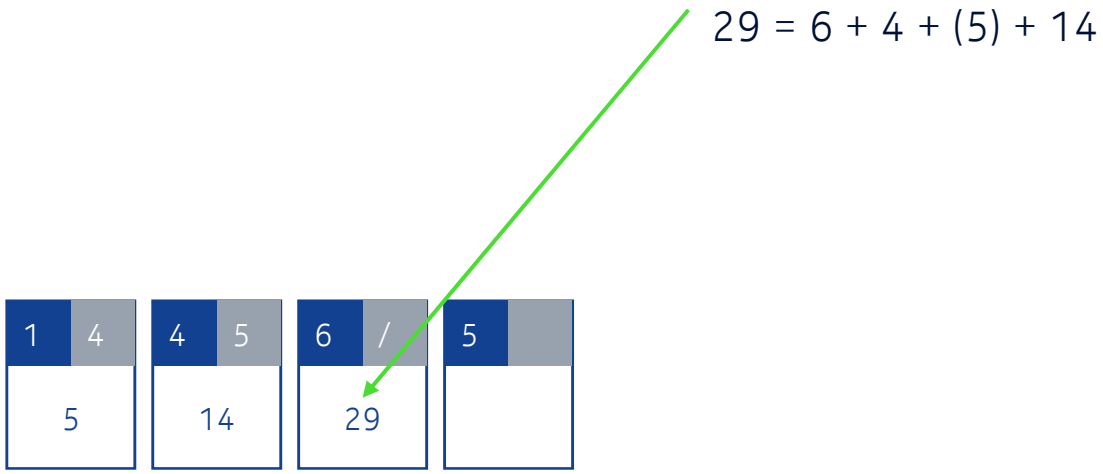
Jest to SPARE, strącenie 10 kręgli w dwóch rzutach podczas jednej rundy.

- BONUS: Do wyniku rundy doliczany jest wynik z JEDNEGO następnego rzutu.



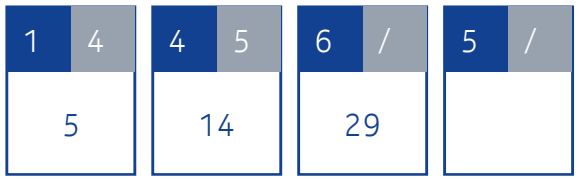
Prosta Gra

Gracz strąca 5 kręgli.
Doliczanie bonusu za spare



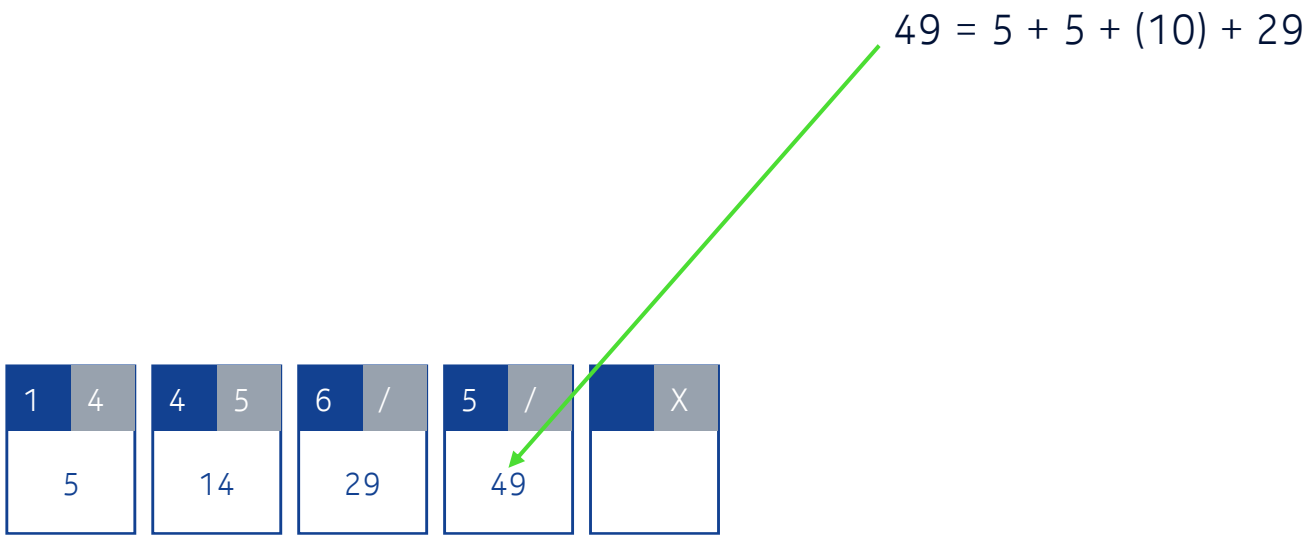
Prosta Gra

Gracz strąca 5 kręgli.
Następny SPARE



Prosta Gra

Gracz strąca 10 kręgli w jednym rzucie
Jest to STIKE, strącenie 10 kręgli w pierwszym rzucie rundy.



Prosta Gra

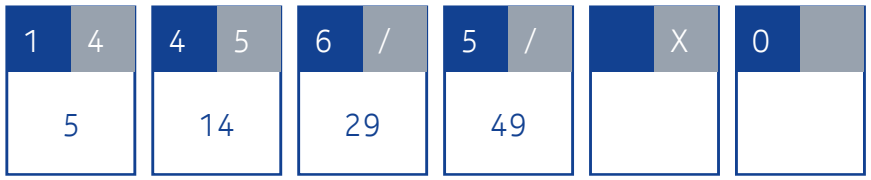
Gracz strąca 10 kręgli w jednym rzucie

Jest to STIKE, strącenie 10 kręgli w pierwszym rzucie rundy.

- BONUS: Do wyniku rundy doliczany jest wynik z DWÓCH następnych rzutów.

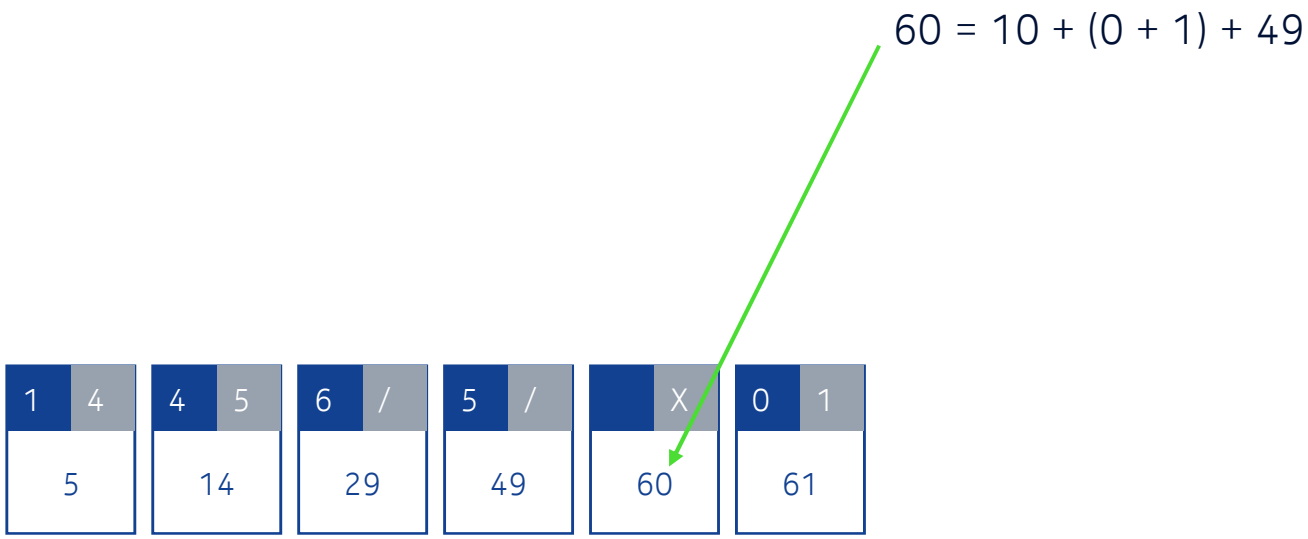
Prosta Gra

Gracz nie strąca ani jednej kręgli



Prosta Gra

Gracz strąca 1 kręgiel.



Prosta Gra

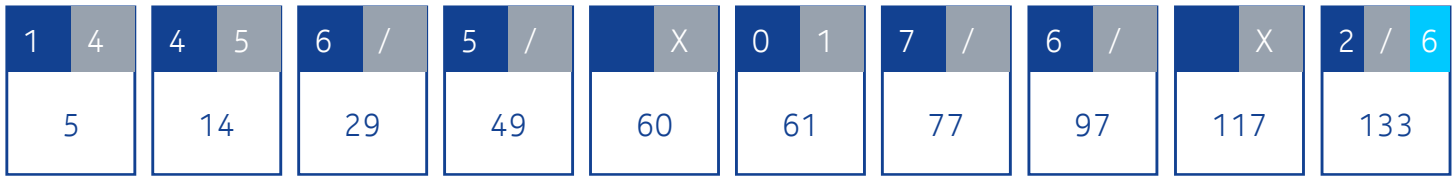
Dalszy ciąg gry.

Prosta Gra

Ostatnia, finałowa runda.

Runda różni się od typowej rundy, gdyż w przypadku gdy gracz rzuci w ostatniej rundzie:

- SPARE (w dwóch rzutach), dostaje jeden dodatkowy rzut
- STRIKE (w jednym rzucie), dostaje dwa dodatkowe rzuty.



Wyniki kilku gier

Rzucając cały czas 9 i spare, wynik gry to 190.

Rzucając cały czas 1, wynik gry to 20.

Rzucając cały czas 10 (Strike), wynik to....

Wyniki kilku gier

Rzucając cały czas 9 i spare, wynik gry to 190.

Rzucając cały czas 1, wynik gry to 20.

Rzucając cały czas 10 (Strike), wynik to.... 300, tzw. **Perfect Game**

Ściągawka

- Biblioteka do testowania Catch – źródła i wiki
  https://github.com/catchorg/Catch2

- Repozytorium z programem startowym
  git clone https://github.com/sharki13/bowling

Implementacja testu

Dodawanie scenariusza testowego

```
TEST_CASE("Test_case_name")
{
}
```

Słowo kluczowe "TEST_CASE" - definuje funckcje która zostanie wykonana jako jeden z testów. Jako argument należy podać nazwę testu która będzie używana przez "test runnera" do komunikacji ewentualnych błedów. Liczba przypadków (TEST_CASE) jest "nieograniczona".

Implementacja testu
Sprawdzanie wyniku

```
REQUIRE(/* warunek_testu */)
```

Warunkiem testu powinno być logiczne wyrażenie zwracające wartości bool, "true" lub "false".

Kiedy wyrażenie zwraca wartość "true", test jest zdany, jeśli zwraca wartość "false" test jest uznany za oblany.

Implementacja testu
Trywialny przykład

Implementacja:

```
class SimpleCalculator {
public:
    int add(int a, int b) {
        return a + b;
    }
};

// prosta klasa z metodą "add"
// zwracająca sumę dwóch liczb całkowitych
```

Test:

```
TEST_CASE("SimpleCalculator_add_test") {
    SimpleCalculator testObj; // tworzenie
instancji testowanego obiektu

    REQUIRE( testObj.add(2, 2) == 4);

    REQUIRE( testObj.add(2, 4) == 6); //
}
```