NOKIA

Praktyczne Aspekty Rozwoju Oprogramowania: TDD na przykładzie C++

Hanna Senhadri (hanna.senhadri@nokia.com)
Kamil Witecki (kamil.witecki@nokia.com)

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 – **część teoretyczna + praktyczna zajęć**
- Pytania w trakcie zajęć – mile widziane
- Czy powinniśmy robić przerwę w trakcie zajęć?

Kwestie organizacyjne… a zdalna forma zajęć

- Sprawdzenie listy obecności – **na podstawie Teams**
- Na zajęciach przysługują dodatkowe punkty za aktywność (+, ++):
  - Wykonanie przykładu publicznie
  - Prezentacja zadania publicznie
  - Pytania i odpowiedzi
- Okno czatu – pomagamy i odpowiadamy na pytania!
- Osoby wypowiadające się: prosimy o oznaczenie się na czacie
- Propozycje dobrych praktyk z poprzednich zajęć?
  - Możemy współdzielić/pokazywać kod przez dedykowany link: https://rextester.com/

    https://godbolt.org/
  - …

NOKIA

czym jest tdd?

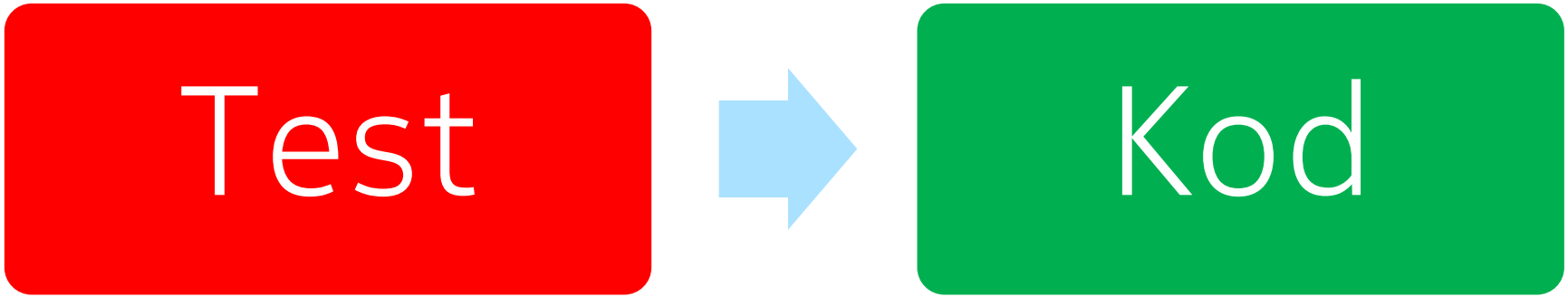
NOKIA

test driven development

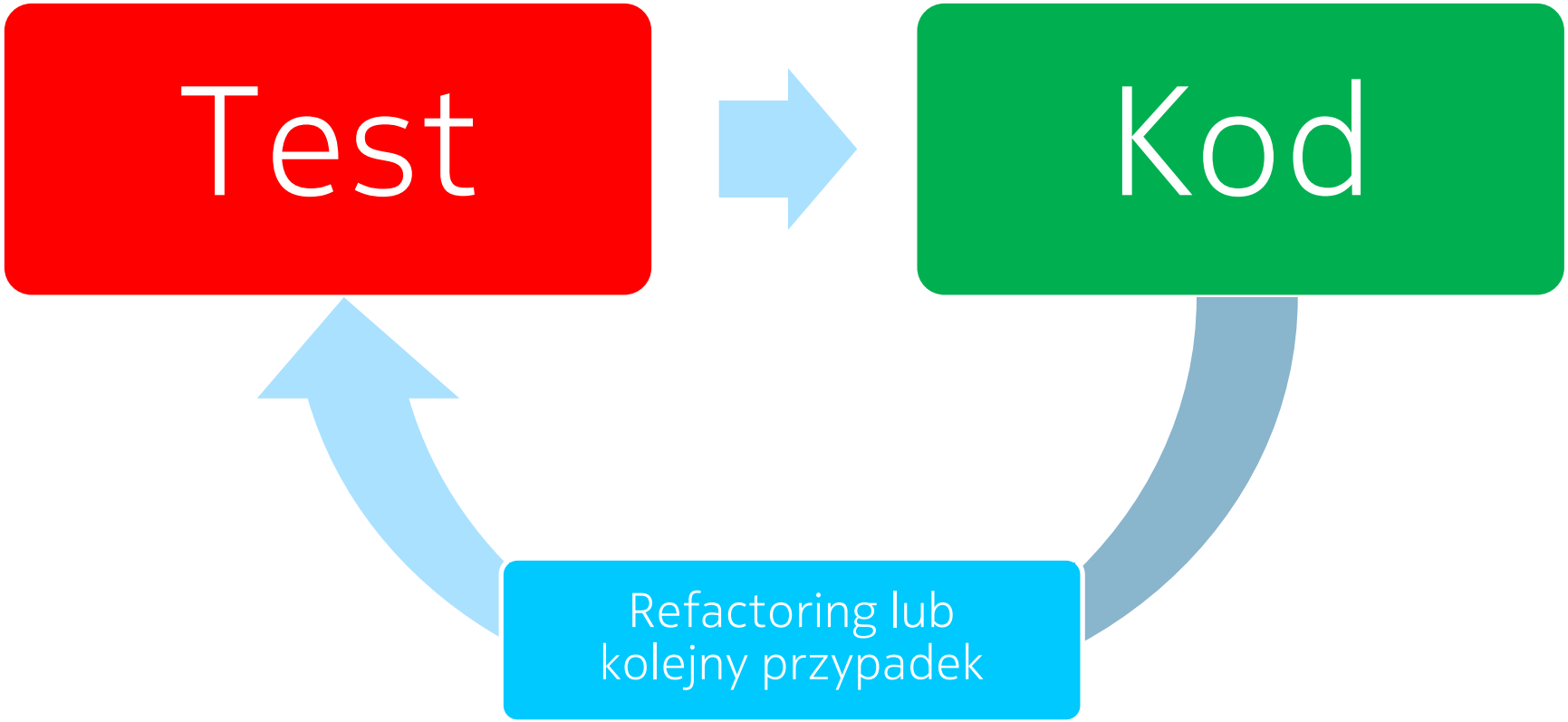
Zasady pracy w TDD

Test

Zasady pracy w TDD



Zasady pracy w TDD



Zadanie na dziś

Liczenie punktów podczas gry w kręgle

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

Wymagania:

Wynik gracza
+ **void** roll(**int** pins) + **int** getScore()

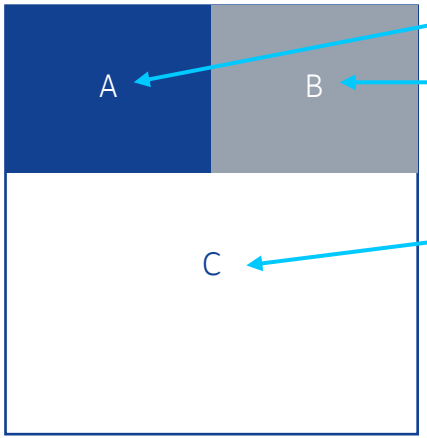
Zadanie na dziś

Liczenie punktów podczas gry w kręgle

Wynik za rundę: ilość przewróconych pinów + **bonusy za spare i strike:**

- **Spare**: przewrócenie 10 pinów podczas dwóch rzutów w jednej rundzie
  - Bonus: Ilość przewróconych pinów podczas następnego rzutu (kolejna runda)
- **Strike**: przewrócenie 10 pinów podczas pierwszego rzutu w rundzie
  - Bonus: Punkty zebrane podczas następnych dwóch rzutów (kolejna runda lub dwie kolejne rundy)

Notacja wyników



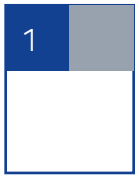
A – wynik za pierwszy rzut

B – wynik za drugi rzut (opcjonalny)

C – wynik gry w danej rundzie

Prosta Gra

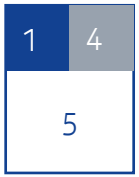
Gracz strąca 1 kręgiel w pierwszym rzucie.



Prosta Gra

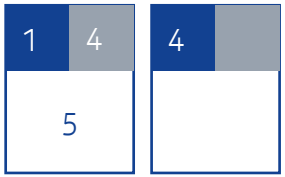
Gracz strąca 4 kręgle w następnym rzucie.

Suma za rundę wynosi 5.



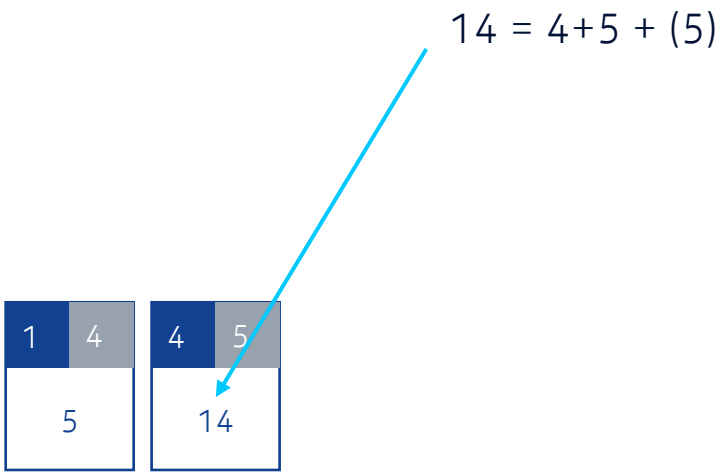
Prosta Gra

Gracz strąca 4 kręgle w następnym rzucie.



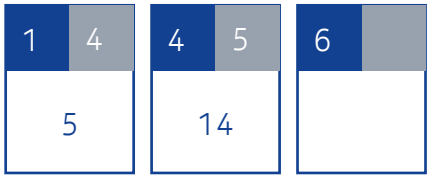
Prosta Gra

Gracz strąca 5 kręgli w następnym rzucie.



Prosta Gra

Gracz strąca 6 kręgli w następnym rzucie.

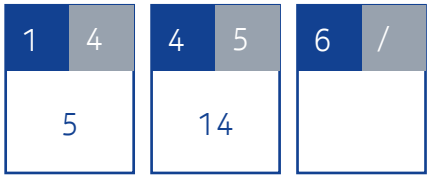


Prosta Gra

Gracz strąca 4 kręgle w następnym rzucie.

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

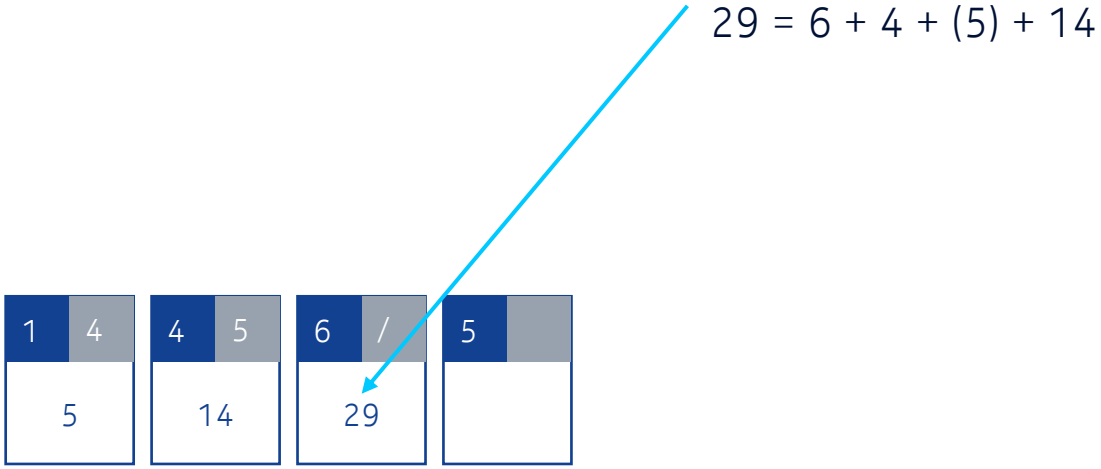
- BONUS: Do wyniku rundy doliczany jest wynik z JEDNEGO następnego rzutu.



Prosta Gra

Gracz strąca 5 kręgli.

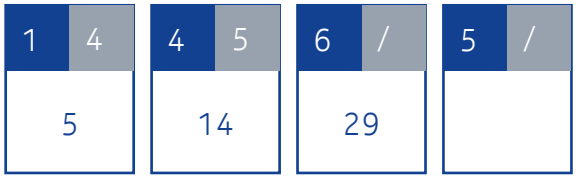
Doliczanie bonusu za spare



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.

49 = 5 + 5 + (10) + 29

Prosta Gra

Gracz strąca 10 kręgli w jednym rzucie

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

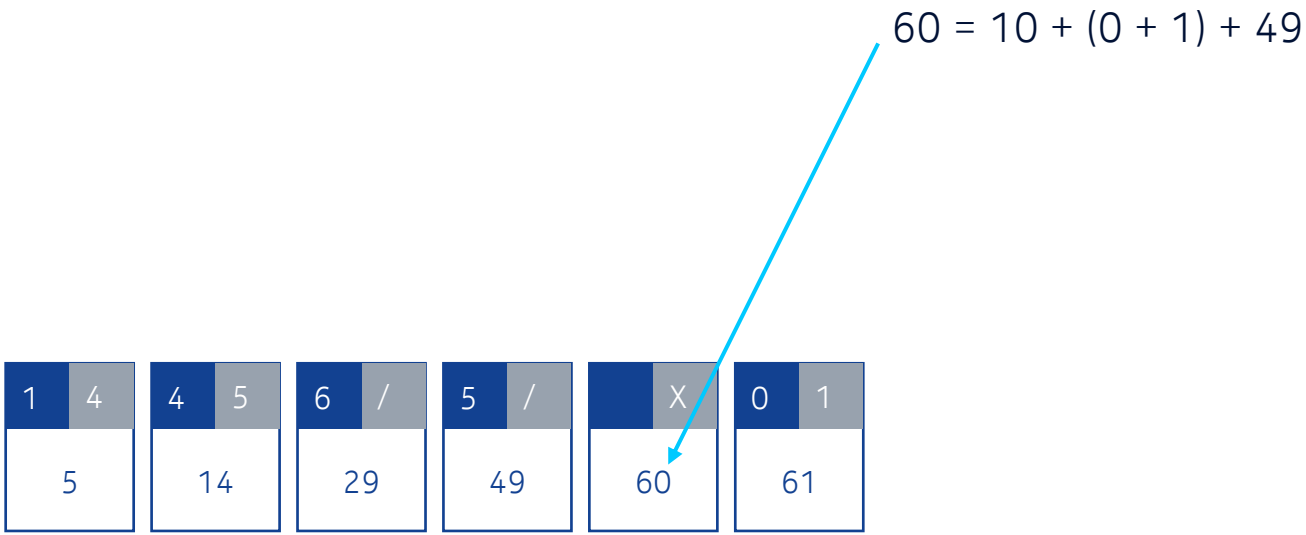
- BONUS: Do wyniku rundy doliczany jest wynik z DWÓCH następnych rzutów.

Prosta Gra

Gracz nie strąca ani jednego kręgla

Prosta Gra

Gracz strąca 1 kręgiel.



Prosta Gra

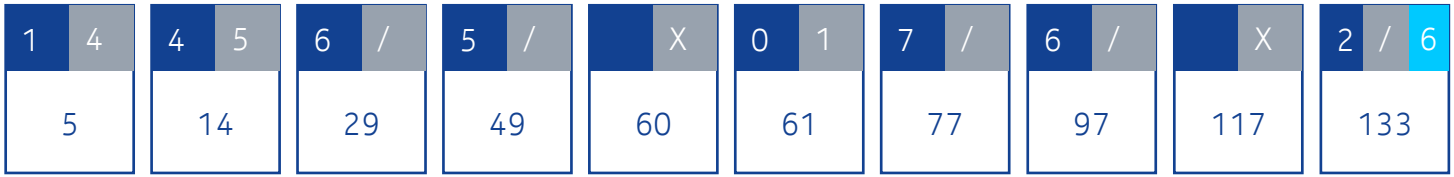
Dalszy ciąg gry.

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 1 (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 1 (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/heireann/paro2021`

- Do weryfikacji wyników można użyć kalkulatora online, przykładowo:
  https://www.bowlinggenius.com/

Implementacja testu

Dodawanie scenariusza testowego

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

Implementacja testu

Dodawanie scenariusza testowego

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

Implementacja testu

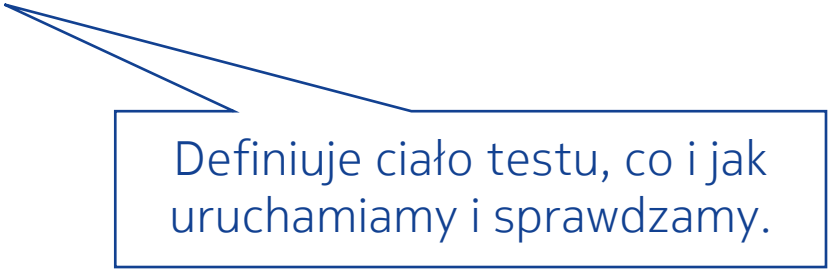
Dodawanie scenariusza testowego

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

Implementacja testu

Dodawanie scenariusza testowego

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



Implementacja testu

Dodawanie scenariusza testowego

```
TEST_CASE("Test_case_name")
{
}
TEST_CASE("Test_case_name1")
{}
…
TEST_CASE("Test_case_nameN")
{}
```

Liczba testów jest teoretycznie 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);
}
```