## Документация за извършените промени в кода на проект „Bulls and Cows“

## Отбор „Osmium“, Май 2013

1. Преименувахме проекта от KPK\_DOMASHNO на BullAndCows
2. Форматирахме кода, чрез премахване на излишните празни редове, добавяне на нужни празни редове, скоби, и преименуване на променливи и меоди
3. Отделихме ScoreBoard методите и полетата в друг отделен клас , който кръстихме ScoreBoard.cs

* в ScoreBoard.cs добавихме константа PlayersToShow, за запазване броя на играчите, които да се показват в класирането.
* премахнахме tryAddToScoreboard() метода, тъй като вече запазваме всички резултати, но показваме толкова колкото пише в константата PlayersToShow
* добавихме нов метод AddPlayer() за добавяне на играч в класирането
* добавихме нов метод Show(), койтозаменя DisplayTop() метода, за показване на класирането
* заменихме полето klasirane, което беше от тип SortedList<string, int> с ranking, като вече типа му е List<Player>
* добавихме нов метод SortRanking(), за да може да се сортира списъка с играчите, метода се извиква всеки път, когато е добавен нов играч.

1. Добавихме енумерация за командите в играта (Command.cs)
2. Създадохме нов клас Player, който да държи информацията за играчите в играта

* Player имплементира интерфейса IComparable<>
* добавихме две полета/свойства: Name и Attemps
* добавихме конструктор с две параметри: name и attemps
* пренаписахме ToString() метода
* имплементирахме CompareTo() метода, за да можем да сравним 2 играча. Сравнението става по Attemps

1. Създадохме нов клас ConsolePrinter.cs, който принтира всички съобщения на конзолата
2. Създадохме интерфейс IPrinter.cs, който се имплементира от ConsolePrinter
3. Създадохме нов клас GameEngine, който управлява играта и имплементира логиката и:

* използва ConsolePrinter, когато трябва да печата на конзолата.
* използва Generator за генериране на тайното число.
* съдържа обект от ScoreBoard, в който добавя всички резутати.
* чете команди от конзолата, за които извършва съответните операции.

1. Създадохме нов клас Generator, който се генерира тези елементи от играта, които трябва да са произволни. В него има методи:

- SecretNumber() - генерира произволно n-цифрено число (в играта: 4-цифрено).

- HelpIndex() - Избира произволен индекс от числото.

- HelpNumber() - Връща стринг с разкрита една цифра от числото.

10. Добавихме XML документация на всички публични класове, енумерации, методи, свойства и интерфейси в проекта.

11. Добавихме unit test-ове за всички класове.

11.1. ConsolePrinterTests:

- Unit test за всяко съобщение (използваме Console.SetOut())

11.2. GeneratorTests

- SecterNumberTest() - тест дали има еднакви числа в масив от 10 числа, генерирани random.

- HelpIndexTest() - тест дали за N на брой извиквания, където N е дължината на числото, методът ще върно поне една двойка различни индекси.

- HelpNumberTest() - тест дали за определено число и индекс се разкрива правилната цифра.

- HelpNumberOutOfRangeTest() - тест дали методът хвърля IndexOutOfRange exception при подаване на грешен индекс.

11.3. PlayerTests

- Тестове за невалидно име (null, String.Empty, Whitespace name).

- Тестове за attempts <= 0.

- Тестове на конструктора на класа.

- Тест за PlayerName setter-a.

- Тест на ToString().

- Тестове на CompareTo() метода.

11.4. ScoreBoardTests

- Тестове на конструктора на класа (празен ScoreBoard, един играч, много играчи, играч -> null)

- Тест за коректност на сортирането.

- Тест за коректност на ToString().

- Teст за коректност на ToString() при празен ScoreBoard.

11.5. GameEngineTests

- Тест за констуктора на класа.

- Тестове за правилното изпълнение на всяка команда (Exit, Restart, Top, Help).

За следващите тестове използваме Console.SetOut()/Console.SetIn()

- Тестове за правилно отпечатване на резутата при проба за познаване на числото.

- Тест за изход при грешно подадена команда (грешен текс, null).

- Тест на exitFromGame полето, при подадена команда Exit.

- Тест за ползване на командата Help повече от позволения брой пъти.

- Тест за коректност на добавянето на Player в ScoreBoard.