# Упражнения: Компонентно тестване

Можете да проверите решенията си в **Judge системата**: <https://judge.softuni.org/Contests/4180/19-Unit-Testing-Basics>

**Използвайте** предоставения **скелет** за всяко от упражненията. За всяка задача трябва да **добавите** правилната **проектна референция**.

Преди да предадете решението си, се уверете, че **using** декларацията на проекта е в **неактивно състояние**.

Text

Description automatically generated with medium confidence

## Компонентно тестване на класа Collection<T>

Даден е C# класът class Collection<T>, който съдържа индексирана поредица от елементи:

Text

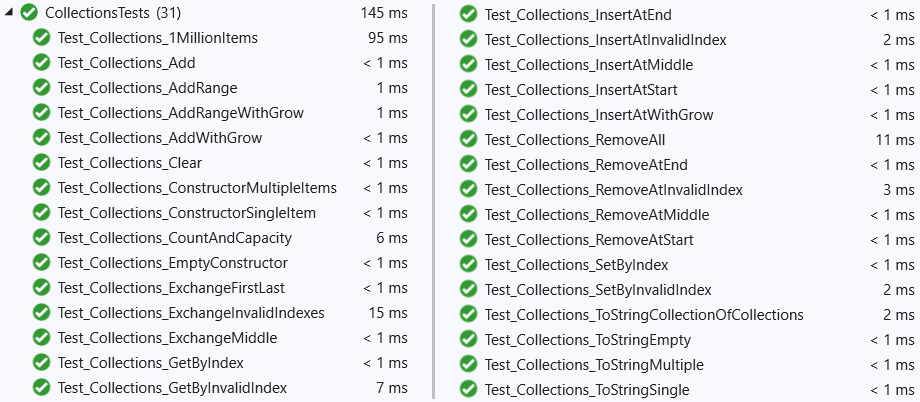
Description automatically generated

Създайте нов **NUnit** **проект** и го кръстете Collections.Tests. Добавете клас CollectionTests.cs и **добавете** **проектна референция**. Напишете **компонентни тестове** за класа Collection<T> в проекта Collections. Уверете се, че **обхватът на тестовете** е висок и че всички интересни случаи са покрити: тествайте **всички публични методи**, тествайте **валидни** и **невалидни** диапазони, опитайте се да покриете и всички останали **специални случаи**.

### Насоки

* Тествайте **всички публични методи**.
* Обмислете **различните сценарии**, които може да се случат, напр. добавяне на елемент в началото, в края, в средата, добавяне с auto-grow, добавяне на невалидна позиция, добавяне в празна колекция и т.н.
* Уверете се, че тествате всеки метод с **валидни** и **невалидни** данни.
* Имплементирайте тест с 1 милион елемента (без **timeout**).

Можете да имплементирате следните компонентни тестове:



## Мениджър на коли

Предоставен е проект, **съдържащ само един клас** - "**Car**". Неговата **основна цел** е да **представя функционалности** на колата. Всяка кола съдържа информация за нейната **марка** (make), **модел** (model), **консумация на гориво** (fuel consumption), **количество гориво** (fuel amount) и **капацитет за гориво** (fuel capacity). Към резервоара на всяка кола може да се **добавя гориво** и всяка кола може да **измине определено разстояние**. За да може шофьорът да **кара колата**, тя трябва да има **достатъчно гориво**.

Всичко, дадено в скелета, работи правилно и **не трябва** да го променяте.

В скелета е предоставен тестов проект с име "**CarManager.Tests**". Трябва да напишете **всички компонентни тестове** там. Този тестов проект трябва да има **само един клас**:

* “**CarTests**” – тук трябва да поставите **целия код**, който тества класа "**Car**" и неговите функционалности.

Вашата задача е да напишете **компонентни тестове**, които да тестват **всяка част** от кода в класа the "**Car**":

* **Всички** **конструктори**
* **Всички** **свойства** (**getter**-и и **setter**-и)
* **Всички методи** и **валидации** в класа

Преди да **предадете** своето решение в **Judge**, **премахнете всички референции и namespaces**, които реферират към другия проект. **Качете единствено** проекта "**CarManager.Tests**", който съдържа **класа с тестовете**. **Премахнете** папките "**bin**" и "**obj**" **преди** да предадете работата си.

### Забележки

* Можете да тествате двата конструктора заедно.
* Не трябва да тествате **auto** свойствата.
* Всички валидации трябва да бъдат тествани.
* Няма ограничение за тестовете, които ще напишете, но се фокусирайте върху основните функционалности.

## Бойна арена

Предоставен е проекта "**FightingArena**", който съдържа **два класа** - "**Warrior**" and "**Arena**". Вашата задача е да **напишете тестове**, които **покриват всички функционалности**. Но преди да започнете да пишете тестове, трябва да се **запознаете** със **структурата** на проекта и неговата **бизнес логика**. Всяка арена (**Arena**)има **колекция** от воини (**Warrior**), които ще водят битки. На арената воините трябва да могат да се **включват в битки** и да **бият друг воин**. Всеки воин има **уникално име (name)**, **нанесена щета** (**damage**) и **HP**. Воините могат да **атакуват** други воини. Разбира се, трябва да има валидации:

* **Името** не можеда бъде **null** или **празен стринг.**
* **Нанесената щета** не можеда бъде **0** или **отрицателна.**
* **HP** не можеда бъде **отрицателно.**
* **Воинът** не може да **атакува**, ако неговото **HP** е **под 30**.
* **Воинът** не може да **атакува опоненти** с **HP под 30**.
* **Воинът** не може да **атакува** **по-силни врагове**.

На **арената** трябва да се изпълнят следните **валидации:**

* Воини, които **вече са се включили**, не трябва да могат да се **включват отново**.
* Не може да **има битка**, ако някой от **воините не е записан** за битките.

В скелета е предоставен тестов проект с име "**FightingArena.Tests**". Трябва да **напишете всички компонентни тестове** там. В този тестов проект трябва да има **два класа**:

* "**WarriorTests**" – тук трябва да сложите **целия код**, който тества класа "**Warrior**" и неговите **функционалности.**
* "**ArenaTests**" – тук трябва да сложите **целия код**, който тества класа "**Arena**" и неговите **функционалности.**

Трябва да тествате **всяка част** от кода в класовете "**Warrior**" и "**Arena**":

* Всички **конструктори**
* Всички **свойства** (**getter**-и и **setter-**и)
* Всички **методи** и **валидации** в класа

Преди да **предадете** своето решение в **Judge**, **премахнете всички референции и namespaces**, които реферират към другия проект. **Качете единствено** проекта " **FightingArena.Tests** ", който съдържа **двата класа с тестове**. **Премахнете** папките "**bin**" и "**obj**" **преди** да предадете работата си.

### Забележки

* Всичко, предоставено в скелета, работи правилно.
* Не трябва да променяте нищо по структурата на проекта.
* Не трябва да тествате **auto** свойствата.
* Всички валидации трябва да бъдат тествани.
* Няма ограничение за тестовете, които ще напишете, но се фокусирайте върху основните функционалности.