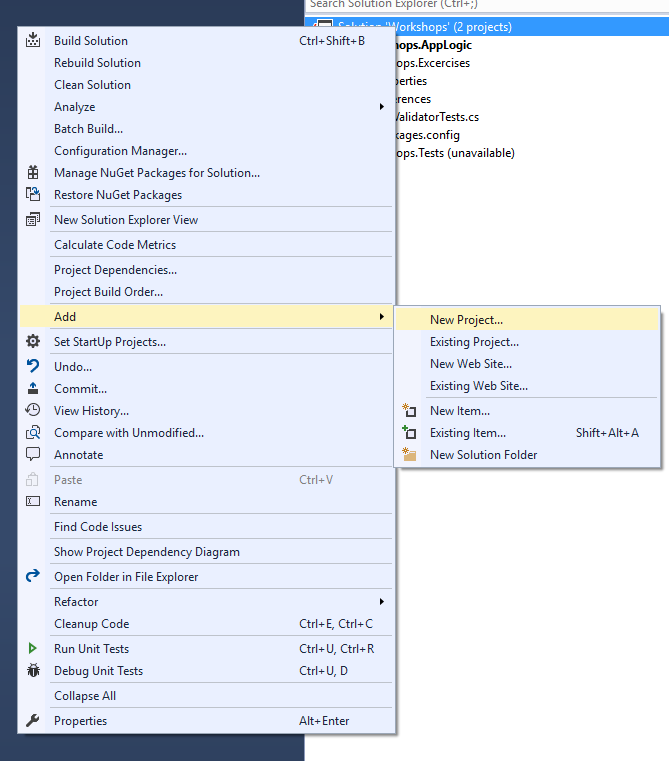
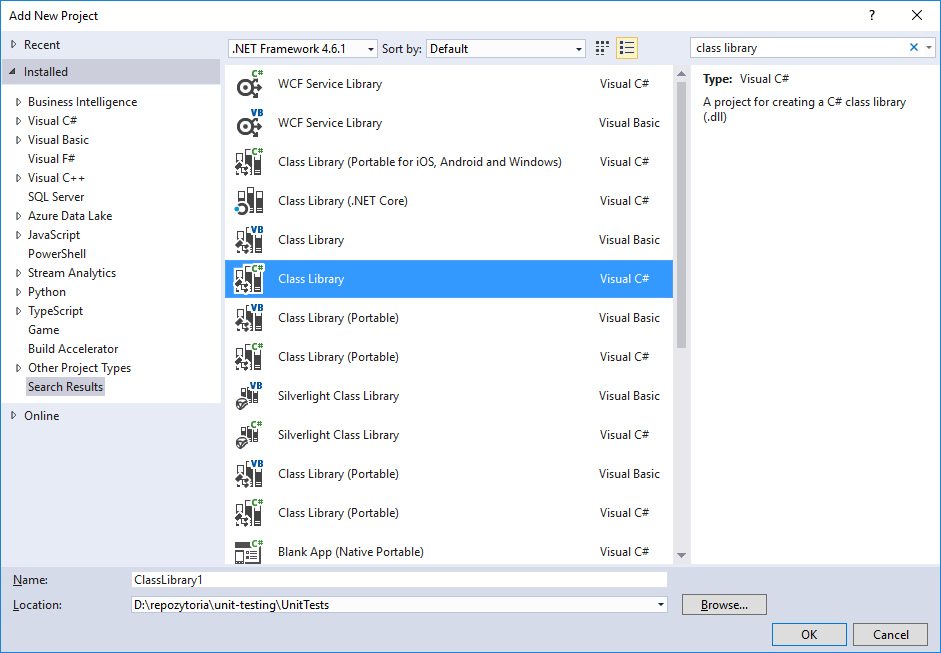
**Przygotowanie projektu z testami**

1.Klikamy prawym przyciskiem myszy na Solution ‘nazwa’ i wybieramy **Add**, następnie **New Project**

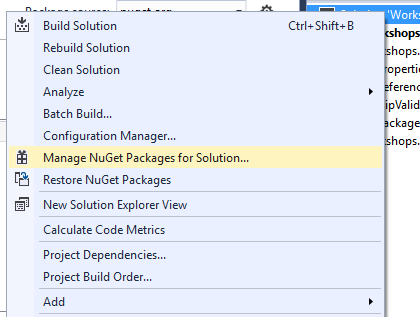


2. W nowo otwartym oknie w polu Search… wpisujemy class library i wybieramy pozycję Class Library Visual C#

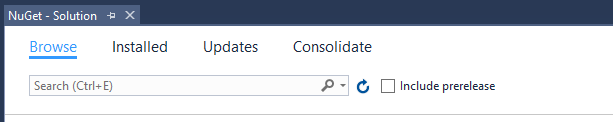


3. Uzupełniamy pole **Name**, następnie zatwiedzamy przyciskiem OK.

4. Z menu z punktu 1. wybieramy pozycję Manage NuGet Packages for Solution



5. W pole **Search** wpisujemy **NUnit.** W panelu po prawej zaznaczamy projekt z testami i klikamy install.



**Przydatne atrybuty:**

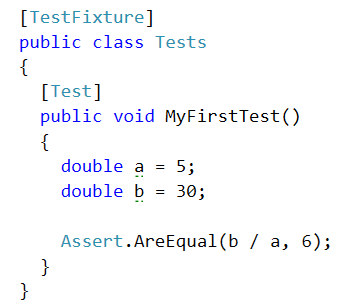
- [TestFixture]

- [Test]

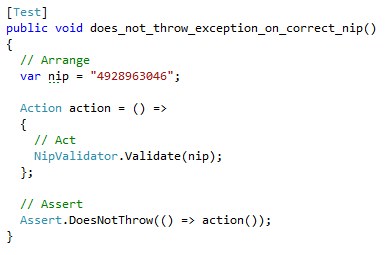
- [TestCase]

- [TestCaseSource]

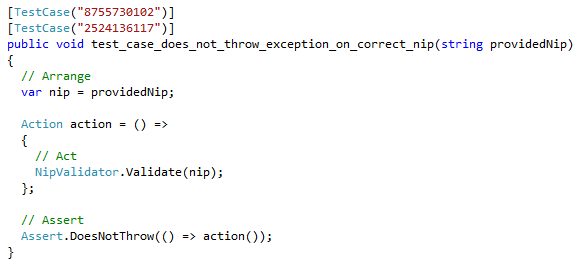
**Przykładowy test jednostkowy:**



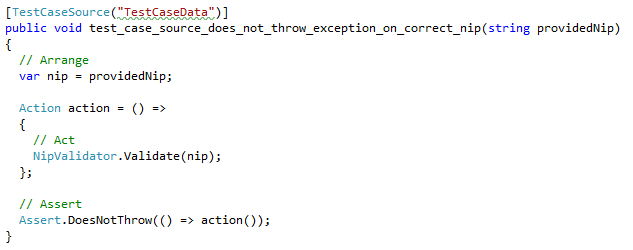
**Wykorzystanie atrybutu [Test]**

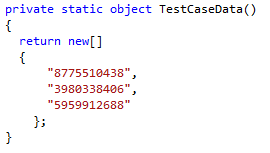


**Wykorzystanie atrybutu [TestCase]**

****

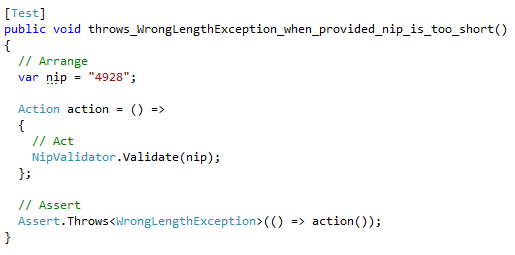
**Wykorzystanie atrybutu [TestCaseSource]**





**Testowanie zgłoszenia wyjątku**

**Assert.Throws(()=>{/\* akcja, która powinna zgłosić wyjątek \*/ });**

****

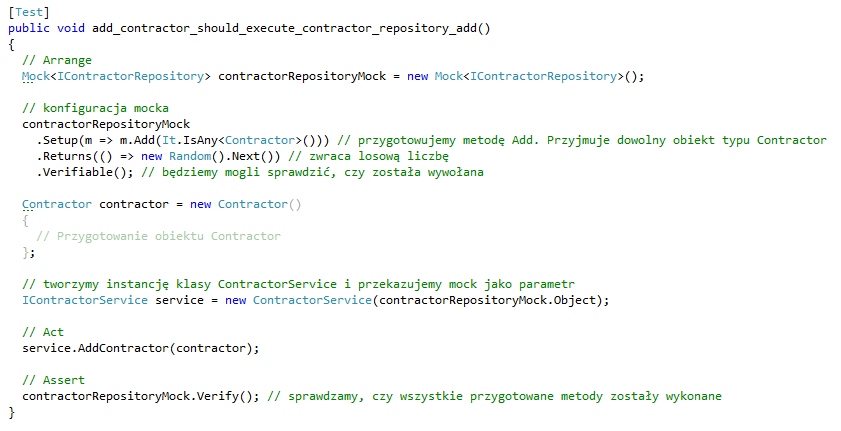
**Sposób walidacji NIP**

* Długość NIPu – 10 znaków
* Wagi: 6, 5, 7, 2, 3, 4, 5, 6, 7, 0
* Mnożymy poszczególne cyfry przez wagę i sumujemy
* Wynik dzielimy modulo przez 11
* Ostatnia cyfra NIPu powinna być równa wynikowi dzielenia

**Sposób walidacji REGON**

* Dopuszczalne 2 długości numeru REGON: 9 oraz 14 znaków
* Wagi:
  + **9 znaków**: 8, 9, 2, 3, 4, 5, 6, 7, 0
  + **14 znaków**: 2, 4, 8, 5, 0, 9, 7, 3, 6, 1, 2, 4, 8, 0
* Mnożymy poszczególne cyfry przez wagę i sumujemy
* Wynik dzielimy modulo przez 11
* Ostatnia cyfra REGONu powinna być równa wynikowi dzielenia

**Wykorzystanie Mocka**

****

Ćwiczenie 1. Test walidacji numeru REGON

Metoda ***Validate*** klasy statycznej ***RegonValidator*** (RegonValidator.Validate(string regon)) sprawdza poprawność numeru REGON podanego jako parametr.

W przypadku gdy podana wartość ma błędną długość powinien być zgłaszany wyjątek ***WrongLengthException***. Jeśli zawiera znaki inne niż cyfry powinien być zgłaszany ***IncorrectCharsException.*** Gdy wynik dzielenia opisanego w instrukcji walidacji REGON jest różny od ostatniej cyfry REGONu powinno to powodować ***NotValidException.*** Prawidłowa wartość nie powinna powodować rzucania wyjątków.

W ćwiczeniu tym należy przy pomocy atrybutów [TestFixture], [Test], [TestCase], [TestCaseSource] przetestować, czy metoda RegonValidator.Validate(string regon)) zachowuje się zgodnie z opisem.

Ćwiczenie 2.

Klasa ***ContractorService*** implementująca interfejs ***IContractorService*** korzysta z implementacji interfejsu ***IContractorRepository***. Wykorzystując mechanizm mocków sprawdzimy, czy