

TECHNOLOGIE KOMPONENTOWE GRUPA 2 - 27.11.2018

1. Sprawdzenie środowiska pracy:

- a. git --version
- b. dotnet --version
- c. Visual Studio Code

2. Fork projektu:

<https://gitlab.com/lukasikpiotr/company1>

Po ukończeniu zadania proszę o wykonanie Merge Request.

3. Stworzenie projektu Classlib/konsolowego w .Net Core, stworzenie projektu testów w NUnit/XUnit oraz dodanie pliku solucji z referencjami do projektów.

Proszę o zachowanie następującej struktury folderów:

Name
 Company
 CompanyTests
 .gitignore
 .gitlab-ci.yml
 Company.sln

(projekt testów powinien być umieszczony na tym samym poziomie co projekt aplikacji, a nie wewnątrz aplikacji).

4. Stworzenie klas reprezentujących strukturę firmy (3 pkt):

a. klasa pracownik powinna zawierać

- i. imię - string
- ii. nazwisko - string
- iii. stanowisko - typ Stanowisko
- iv. konstruktor inicjujący (imię, nazwisko, stanowisko)
- v. metodę pozwalającą zmienić stanowisko
- vi. metodę ToString() zwracającą imię, nazwisko, stanowisko z pensją

b. klasa stanowisko powinna zawierać

- i. typ stanowiska - enum z wartościami: junior, mid, senior
- ii. stawkę - zależną od typu stanowiska - wartości proszę wybrać indywidualnie

5. Zaprezentowanie działania klas w projekcie z testami. (XUnit lub NUnit) (2 pkt)

6. Stworzenie extension method (metody rozszerzającej) dla: (2 pkt)

- a. String - metoda CountWord - zlicza liczbę słów podanym tekście
- b. String - metoda CountVowel - zlicza liczbę samogłosek

- c. Int - metoda MakeBinary - zapisuje liczbę w postaci binarnej (zapis do string, max wartość int = 1000)
 - d. Int - metoda MakeHex - zapisuje liczbę w postaci heksadecymalnej (zapis do string, max wartość int = 1000)
- 7. Do każdej extension method proszę o utworzenie jednego testu pozwalającego zweryfikować działanie metody (umieszczenie testu dla max wartości). (XUnit lub NUnit) (2 pkt)
- 8. Stworzenie własnej kolekcji FancyList w oparciu o interfejs IList (3 pkt)
 - a. Zaimplementowanie wymaganych metod
 - b. Zaimplementowanie metody rozszerzającej działanie kolekcji o:
 - i. sprawdzenie czy ilość elementów jest parzysta - IsEven
 - ii. sprawdzenie ilość unikalnych elementów w kolekcji
- 9. Stworzenie testów weryfikujących poprawność FancyList: (1 pkt)
 - a. sprawdzenie zachowań listy
 - b. sprawdzenie zachowań metod rozszerzających
- 10. Stworzenie delegaty realizującej zachowanie (zadanie może być zrealizowane w projekcie w testami, jeśli główny projekt to classlib) (2 pkt)
 - a. jako parametr przyjmuje int zwraca string
 - b. jako parametr przyjmuje long zwraca bool
 - c. zaprezentowanie możliwości **multicasting** w delegatach
 - d. dodanie komentarzy opisujących zachowanie stworzonej delegaty