Описание на тестовете

News:

News\_Exception\_InitialisationsNameNull() – създаваме новина със следните параметри:

* Празно име, Информация – хвърля <ArgumentException> - данните не могат да бъдат празни
* Null, Информация - хвърля <ArgumentException> - данните не могат да бъдат празни
* Име, Празна информация - хвърля <ArgumentException> - данните не могат да бъдат празни
* Име, Null - хвърля <ArgumentException> - данните не могат да бъдат празни

News\_Complete() – създаваме новина с име и информация – новината се създава успешно

News\_DrawNews() – четем новините от базата и ги записваме като List<TextBox> -обектът не е празен

News\_ReadNews() – четем новините от базата като List<News> - обектът не е празен

News\_CheckForNonExistingNew() – подаваме новина, която не е записана в базата – резултат е че новината не е намерена (False)

News\_CheckForExistingNew() - подаваме новина, която е записана в базата – резултат е че новината е намерена (true)

News\_CheckForExistingNewName() - подаваме новина, чието име е записано в базата, но информацията й е различна – резултат е че новината е намерена (true)

News\_CheckForExistingNewInformation() - подаваме новина, чиято информация е записана в базата, но името й е различно – резултат е че новината е намерена (true)

News\_ReverseNews() – Подаваме List<News> със записани 3 различни новини - новините са подредени в обратен ред

Questions:

Questions\_SetAnswerA() – Създаваме обект от тип Questions като му подаваме за верен отговор “a” – верният отговор на въпроса съвпада с първия отговор

Questions\_SetAnswerB() - Създаваме обект от тип Questions като му подаваме за верен отговор “б” – верният отговор на въпроса съвпада с втория отговор

Questions\_SetAnswerC() - Създаваме обект от тип Questions като му подаваме за верен отговор “в” – верният отговор на въпроса съвпада с третия отговор

Questions\_SetAnswerD() - Създаваме обект от тип Questions като му подаваме за верен отговор “г” – верният отговор на въпроса съвпада с четвъртия отговор

Questions\_CheckValidQuestion() – Създаваме обект от тип Questions с 4 отговор и верен отговор 1 и проверяваме валидността на въпроса – въпросът е валиден (true)

Questions\_CheckValidQuestion\_False() - Създаваме обект от тип Questions с 4 отговор и верен отговор, който не съвпада с никой от останалите, и проверяваме валидността на въпроса – въпросът е невалиден (false)

Questions\_SetAnswer\_CheckValidQuestion() - Създаваме обект от тип Questions като му подаваме за верен отговор “г” и проверяваме валидността на въпроса – въпросът е валиден (true)

Questions\_Exist\_False() – Създаваме обект от тип Questions, като на въпроса подаваме стойност, която я няма в базата данни, и правим проверка за съществуването на въпроса – въпросът не е открит в базата (функцията връща true)

Questions\_Exit\_True() - Създаваме обект от тип Questions, като на въпроса подаваме стойност, която съществува в базата, и правим проверка за съществуването на въпроса – въпросът е открит в базата (функцията връща false)

Questions\_ReadAllQuestions\_Empty() – правим проверка за съществуването на въпросите в базата по предмет, като подаваме празна стойност – не са открити въпроси в базата

Questions\_ReadAllQuestions() - правим проверка за съществуването на въпросите в базата по предмет, като подаваме ”Математика”– стойността не е празна

Student:

Student\_Exception\_Initialisations() – подаваме невалидни стойности за полетата на Student обекта – получаваме <ArgumentException> за невалидна стойност

Student\_Evaluation\_When\_Default() – оценяваме Student обект, когато стойността на текущата оценка е 2 – получаваме подадената оценка

Student\_Evaluation() - оценяваме Student обект, когато стойността на текущата оценка е 3 и подаваме 6 – получаваме 5 като оценка

Student\_ReadFromData() – четем съществуващ студент от базата – полетата на обекта са със стойностите от базата

Student\_ReadStudents() – чете всички студенти от базата – броя на обектите е по-голям от 0

Student\_Update\_Grade() – променяме оценката на студент в базата след това го прочитаме и очакваме оценката да бъде променена – оценката е променена

Student\_CheckForExistingStudent() – правим проверка за съществуването на email в базата – имеилът е намерен (true)

Student\_CheckForNonExistingStudent() - правим проверка за съществуването на email в базата, като подаваме незаписан email – имеилът не е намерен (false)

Student\_CheckForNonExistingStudentPhoneNumber() - правим проверка за съществуването на телефонен номер в базата, като подаваме незаписан номер – номерът не е намерен (false)

Student\_CheckForExistingStudentPhoneNumber() - правим проверка за съществуването на телефонен номер в базата – номерът е намерен (true)

Student\_CheckForNonExistingStudentFacultyNumber() - правим проверка за съществуването на факултетен номер в базата, като подаваме незаписан номер – номерът не е намерен (false)

Student\_CheckForExistingStudentFacultyNumber() - правим проверка за съществуването на факултетен номер в базата – номерът е намерен (true)

Teacher:

Teacher\_Exception\_Initialisations() - подаваме невалидни стойности за полетата на Teacher обекта – получаваме <ArgumentException> за невалидна стойност

Teacher\_ReadFromData() - четем съществуващ преподавател от базата – полетата на обекта са със стойностите от базата

Teacher\_ReadTeachers() - чете всички преподаватели от базата – броя на обектите е по-голям от 0

Teacher\_CheckForExistingTeacher() - правим проверка за съществуването на email в базата – имеилът е намерен (true)

Teacher\_CheckForNonExistingTeacher() - правим проверка за съществуването на email в базата, като подаваме незаписан email – имеилът не е намерен (false)

Teacher\_CheckForNonExistingTeacherAdminNumber() - правим проверка за съществуването на административен номер в базата, като подаваме незаписан номер – номерът не е намерен

Teacher\_CheckForExistingTeacherAdminNumber() - правим проверка за съществуването на административен номер в базата – номерът е намерен (true)

Teacher\_CheckForNonExistingTeacherPhoneNumber() - правим проверка за съществуването на телефонен номер в базата, като подаваме незаписан номер – номерът не е намерен (false)

Teacher\_CheckForExistingTeacherPhoneNumber() - - правим проверка за съществуването на телефонен номер в базата – номерът е намерен (true)

ReviewTest

Read\_From\_Data() – четем Review тест от базата, като подаваме email и предмет на съществуващ запис – върнатите обекти са повече от 0