Начиная с текущего домашнего задания, мы больше не будем создавать класс Application, в котором мы ранее описывали статический метод main. Теперь в области кода (src/java/lessonX/taskY) мы будем описывать код, необходимый для решения задачи, а тесты будем описывать в области для написания тестов test/java/lessonX/taskY/<TestClass>.

**Задача 0**

В текущий проект добавить поддержку фреймворка Maven.

Включить автоматический импорт подключаемых библиотек (Enable Auto-import)

Добавить в зависимости библиотеку Junit Jupiter API. При подключении использовать последнюю стабильную версию на момент выполнения задачи.

**Задача 1**

Создать класс Calculator, который будет иметь 2 статических метода: первый метод будет считать сумму переданных в него параметров, второй метод – произведение этих параметров и возвращать результат операции.

Методы должны принимать переменное количество параметров (int… values).

В области для написания тестов реализовать тестирование полученного решения для входных данных из таблице. Тесты для суммирования и умножения сделать в двух отдельных классах

|  |  |  |
| --- | --- | --- |
| Тестируемый метод | Входные данные | Ожидаемый результат |
| Сложение | 5, 3 | 8 |
| Сложение | 1, 2, 3, 4, 5, 6 | 21 |
| Умножение | 3, 2 | 6 |
| Умножение | 1, 2, 3, 4, 5, 6 | 720 |

**Задача 2**

Создать класс Human, у которого есть следующие поля: firstName (имя), age (возраст). Поля сделать приватными. Конструктор оставить по умолчанию. Реализовать сеттеры и геттеры для полей, которые будут валидировать поля следующим образом:

Возраст – должен быть не менее 0 и не более 120.

Имя – должно содержать от 3 до 12 символов, начинаться с большой буквы, следующие буквы должны быть маленькими. Имя должно быть на русском языке.

В случае, если поля не валидны, должно выбрасываться исключение IllegalArgumentException с сообщением в формате:

Некорректный возраст: <возраст>

Некорректное имя: <имя>

В области для написания тестов необходимо написать тесты для следующих входных данных (имя тестировать в одном классе, возраст в другом):

|  |  |  |
| --- | --- | --- |
| Сеттер поля: | Входные данные | Ожидаемый результат |
| name | Иван | Поле установлено значением <Иван> |
| name | Ив | Исключение с сообщением, поле не установлено |
| name | Ивансдлиннымименем | Исключение с сообщением, поле не установлено |
| name | Иван5аа | Исключение с сообщением, поле не установлено |
| age | 0 | Поле установлено значением 0 |
| age | 60 | Поле установлено значением 60 |
| age | 120 | Поле установлено значением 120 |
| age | -1 | Исключение с сообщением, поле не установлено |
| age | 121 | Исключение с сообщением, поле не установлено |

**Задача 3**

Создать класс Generator, содержащий методы randomInteger(int start, int end), randomString(int startLength, int endLength).

Метод randomInteger должен генерировать случайное число от start до end включительно.

Метод randomString должен генерировать случайную строку из заглавных и прописных латинских букв случайной длиной от startLength до endLength включительно.

Протестировать написанное решение со случайными входными данными границ генерации, путем цикла из 10000 генераций. Для каждой генерации использовать случайные значение граничных значений. Для assertions использовать методы, позволяющие передать дельту получаемых результатов.

**Задача 4**

Подключить к проекту библиотеки org.apache.poi:poi и org.apache.poi:poi-ooxml. Версию выбрать последнюю стабильную на момент выполнения задания.

Ознакомиться с базовым функционалом библиотеки (работа с Excel-файлами, пункт 3 статьи):

<https://www.baeldung.com/java-microsoft-excel>

Создать класс Passport, описать поля: серию и номер.

Создать класс Person, описать поля: фамилия, имя, отчество (String), дата рождения (Date), паспорт (Passport)

Создать класс FileHelper, содержащий статический публичный метод List<Person> getPersonsFromExcel(String path, String sheetName). Метод должен принимать в качестве path – путь к файлу, sheetName – название листа с данными в Excel-документе. Метод должен парсить данные с листа из файла, преобразовывать их в класс Person, а затем возвращать список полученных людей.

В случае, если документ не существует, или в документе нет листа с именем sheetName, метод должен возвращать null. В случае, если при парсе строки возникает ошибка (например, дата не в формате дд.мм.гггг), в таком случае в список персон информация о таком человеке добавляться не должна.

В качестве входного файла использовать homework\_task/lesson11/persons.xlsx

Написать следующие тесты:

Парсинг из документа из листа Persons1, из листа Persons2, из несуществующего листа, из несуществующего документа