Согласно тестовому заданию необходимо было написать программу расчёта страховки автомобиля на ЯП Java. На входе функция получает стаж водителя, количество аварий за последний год, мощность двигателя. На выходе пользователь получает стоимость страховки.

В ходе написания программы было принято решение включить в код обработку исключений, чтобы не допустить некорректного ввода данных:

1. Было подсчитано, что стаж вождения не может превысить (примерно) 81 года (при условии, что водитель начал ездить в 18 лет). В консоль выводится сообщение “Impossible” при попытке ввода опыта больше 80.
2. Также обрабатывается исключение при попытке ввода опыта меньше 0.
3. Также обрабатывается исключение при попытке ввода количества аварий меньше 0.
4. Также были наложены ограничения на ввод данных о мощности двигателя: мощность не может быть меньше 0 и больше 800.

В процессе написания класса тестирования были выбраны две техники тест-дизайна: тестирование классами эквивалентности и попарное тестирование.

|  |  |  |
| --- | --- | --- |
| **Классы эквивалентности** | | |
| *Опыт* | *Аварии* | *Мощность двигателя* |
| 0,1-1 |  | 1-50 |
| 1,1-4,9 |  | 51-100 |
| 5-81 |  | 101-150 |
|  |  | 151-800 |

Затем с помощью сервиса <https://pairwise.teremokgames.com> была составлена таблица пар тестирования.

Затем в каждом классе эквивалентности были отобраны средние значения:

|  |  |  |
| --- | --- | --- |
| *Опыт* | *Аварии* | *Двигатель* |
| 0,5 | 0 | 25 |
| 2 | 1 | 75 |
| 35 | 2 | 125 |
|  | 1000\* | 350 |

\*условно аварий может быть + ∞

Проведя вышеуказанные манипуляции, были получены входные данные для тестового метода. Проверялись только положительные тестовые случаи.

PS. Предполагаю, что тест-кейсы с повторяющимся результатом (210 и 1200) можно не выполнять.