**Задание 1. Напишите код класса с применением принципа инкапсуляции**

* Создайте проект EncapsulationExperiments и создайте в нём класс Main с методом main.

* В рамках этого задания вам нужно создать класс, который будет контролировать работу детского карточного банковского счёта:

|  |
| --- |
| **public class ChildBankAccount {**    **}** |

* Создайте в этом классе две переменные: одну, в которой будет храниться баланс, и вторую, в которой будет храниться максимально возможный баланс:

|  |
| --- |
| **private double balance;**  **private double maxBalance;** |

* Создайте конструктор с параметром maxBalance и устанавливайте значение переменной maxBalance из этого параметра. Это будет максимальный баланс на который можно пополнить счёт.
* Создайте в классе метод пополнения карточного счёта с параметром   value, содержащим количество денег, на которое необходимо пополнить счёт:

.

|  |
| --- |
| **public boolean depositMoney(double value) {**    **}** |

Метод должен защищать balance от ошибочных изменений — уменьшения (в случае, если value меньше нуля) и увеличения сверх лимита (выше значения maxBalance).

В методе напишите проверку того, что сумма счёта после пополнения не станет выше максимально допустимой — выше значения переменной maxBalance. Если это условие выполняется, увеличьте баланс счёта на значение value и верните true. Если условие не выполняется или переменная value меньше нуля, метод должен возвращать false.

* Создайте метод списания денег со счёта:

|  |
| --- |
| **public boolean debitMoney(double value) {**    **}** |

Этот метод должен защищать переменную balance от увеличения (в случае, если будет передано значение value меньше 0) и от уменьшения ниже нуля.

Напишите в этом методе проверку того, что баланс счёта не станет отрицательным, если из него вычесть value, а также проверку того, что value — неотрицательное число. В случае, если всё в порядке, метод должен вернуть true. Если хотя бы одно из условий не выполняется, метод должен вернуть false.

* Создайте также в классе ChildBankAccount метод getBalance, который будет возвращать текущий баланс счёта.

* Напишите в методе main класса Main код, который будет проверять корректность реализации методов класса ChildBankAccount:

|  |
| --- |
| **ChildBankAccount account =**  **new ChildBankAccount(10000);**  **account.depositMoney(1000);**  **account.depositMoney(2000);**  **account.depositMoney(10000);**  **account.depositMoney(-1000);**  **System.*out*.println("Balance: " + account.getBalance());**  **account.debitMoney(500);**  **account.debitMoney(422.75);**  **account.debitMoney(50000);**  **account.debitMoney(-50);**  **System.*out*.println("Balance: " + account.getBalance());** |

* Выполните получившийся код и сверьте результат с эталонным:

|  |
| --- |
| **Balance: 3000.0**  **Balance: 2077.25** |

* Если что-то не получилось, постарайтесь самостоятельно внести исправления в свой код и добиться необходимого результата — вывода в консоль текста, полностью идентичного показанному выше.

* В случае возникновения трудностей вы, как и всегда в заданиях для самостоятельного выполнения, можете воспользоваться рекомендациями под видео, под которым была указана ссылка на это задание.

* Если всё получилось, поздравляем! Вы написали код класса, в котором значение переменной balance защищено от некорректных изменений, а работа с этой переменной осуществляется скрытым в методах depositMoney и debitMoney кодом. Это наглядная демонстрация принципа инкапсуляции, которого вам будет необходимо придерживаться в будущем, создавая свои классы и методы.