Тестирование реализации банковского счета

Требования к реализации

1. Реализация банковского счета должна предоставлять следующие методы:
    a. `int getBalance()`
    возвращает текущую сумму на счете, представленную как число типа `int`;
    b. `int getMaxCredit()`
    возвращает текущее значение кредитного максимума по данному счету;
    c. `boolean isBlocked()`
    возвращает статус счета — заблокирован он или нет;
    d. `void block()`
    блокирует счет;
    e. `boolean unblock()`
    разблокирует счет, если удачно, возвращает `true`;
    f. `boolean deposit(int sum)`
    в обычной ситуации увеличивает сумму на счете на величину параметра, если удачно, возвращает `true`;
    g. `boolean withdraw(int sum)`
    в обычной ситуации уменьшает сумму на счете на величину параметра, если удачно, возвращает `true`;
    h. `boolean setMaxCredit(int c)`
    устанавливает кредитный максимум равным величине параметра, если удачно, возвращает `true`.
2. В незаблокированном состоянии значение текущей суммы на счете не должно быть меньше отрицательного кредитного максимума.
    При попытке нарушить это требование вызов `withdraw(int sum)` с достаточно большим значением параметра должен возвращать результат `false`, текущая сумма на счете при этом остается неизменной.
3. В незаблокированном состоянии запрещено изменять значение кредитного максимума, `setMaxCredit(int c)` возвращает результат `false`, значение кредитного максимума счета остается неизменным.
4. При превышении параметром `setMaxCredit(int c)` по абсолютной величине границы значения кредитного максимума, равной `BOUND = 1000000`, возвращается результат `false`, значение кредитного максимума счета остается неизменным.
5. В заблокированном состоянии и при значении параметра, не превосходящем BOUND по абсолютной величине, `setMaxCredit(int c)` устанавливает значение кредитного максимума равным значению параметра, возвращая результат `true`.
6. Разблокировка счета с помощью `unblock()` должна быть удачна, если выполнено условие 2, иначе разблокировка неудачна и возвращает результат `false`, не изменяя статус счета.

7. Результат операций `deposit(int sum)` и `withdraw(int sum)` с отрицательным аргументом или аргументом, превосходящим BOUND, должен быть `false`, текущая сумма на счете при этом остается неизменной.
8. Результат операций `deposit(int sum)` и `withdraw(int sum)` влекущий за собой изменение баланса, превосходящее BOUND, должен быть `false`, текущая сумма на счете при этом остается неизменной.
9. Результат операций `deposit(int sum)` и `withdraw(int sum)` для заблокированного счета должен быть `false`, текущая сумма на счете при этом остается неизменной.

Задание

Разработать набор тестов для всех методов реализации банковского счета в классе `root.account.Account`.
Набор тестов должен покрывать все требования.
Кроме того, набор тестов должен обеспечивать покрытие всех ветвлений в коде методов и всех отдельных дизъюнктов в условиях ветвлений.

Принимаются

1. Набор тестов
2. Список ошибок с исправлениями
3. Отчет о покрытии тестами кода

Замечание

Тестирование должно проводиться по принципу «черного ящика»

Максимальный балл по этой версии задания - 7