Question 1: Do the test cases above successfully run ? Why or why not?
(Provide two screenshots of these tests and justify the result for each of them)

Both have a NullPointerException
- hasPositiveBalance fails, because the variable is never initialized within the Account function and it calls the account variable instead of the class

- depositIncreasesBalance fails, because of the exception thrown in the previous test
    - int initialBalance = account.getBalance();

Question 2: Do the test cases successfully run ? Why or why not?
(Provide two screenshots of these tests and justify the result for each of them)

- The test cases both run successfully, because the account variable has now been initialized to a string

Question 3: Does the test above run or not, please explain and provide a screenshot?

- The test runs successfully, because it is depositing 50 dollars, and since that is the only amount that has been deposited/withdrawn it is equal to 50. Thus, making it succeed.

Question 4: Write a new test case to show some potential error in the code, and submit the test (code and expected result) and a screenshot of the result. Your answer should make the test descriptive (showing a potential weakness and how you address it).

-   I saw that within the getName() function there seemed to be a problem with getting an account name according to the letters of the account. Overall, I saw that instead it takes the account name exactly and proceeds to fail even if the letters are in the account name. For example, expected "a string starting with "abc"" and the actual was "CompanyName" to fix this just fix the getName() function and don't just return the name.

```
 ⌄ ⊗ HandsOnExperi 11 ms    /usr/lib/jvm/java-11-openjdk/bin/java -ea -Didea.test.cyclic.buffer.size=1048576 -javaagent:/usr/
    ⊗ fail          11 ms
                            ⊞ java.lang.AssertionError <3 internal lines>
                            ⊞    at lab4.HandsOnExperience.fail(HandsOnExperience.java:91) <28 internal lines>


                            Process finished with exit code 255
```

Question 5: What happens if the @Test annotation without specifying the exception?

-   Without clarifying the exception then the exception is thrown making the test fail. Due to you originally specifying the exception is clarifies that the exception InsufficientFundsException was thrown and it expects it to be thrown.

```
 ⌄ ❶ HandsOnExperi 34 ms    /usr/lib/jvm/java-11-openjdk/bin/java -ea -Didea.test.cyclic.buffer.size=1048576 -javaagent:/usr/
    ❶ throwsWher 34 ms
                            lab4.HandsOnExperience$InsufficientFundsException: balance only 0

                                at lab4.HandsOnExperience$Account.withdraw(HandsOnExperience.java:34)
                            ⊞   at lab4.HandsOnExperience.throwsWhenWithdrawingTooMuch(HandsOnExperience.java:102) <28 intern


                            Process finished with exit code 255
```

```
 ⌄ ✔ HandsOnExperi 21 ms    /usr/lib/jvm/java-11-openjdk/bin/java -ea -Didea.test.cyclic.buffer.size=1048576 -javaagent:/usr/
    ✔ throwsWher 21 ms
                            Process finished with exit code 0
```