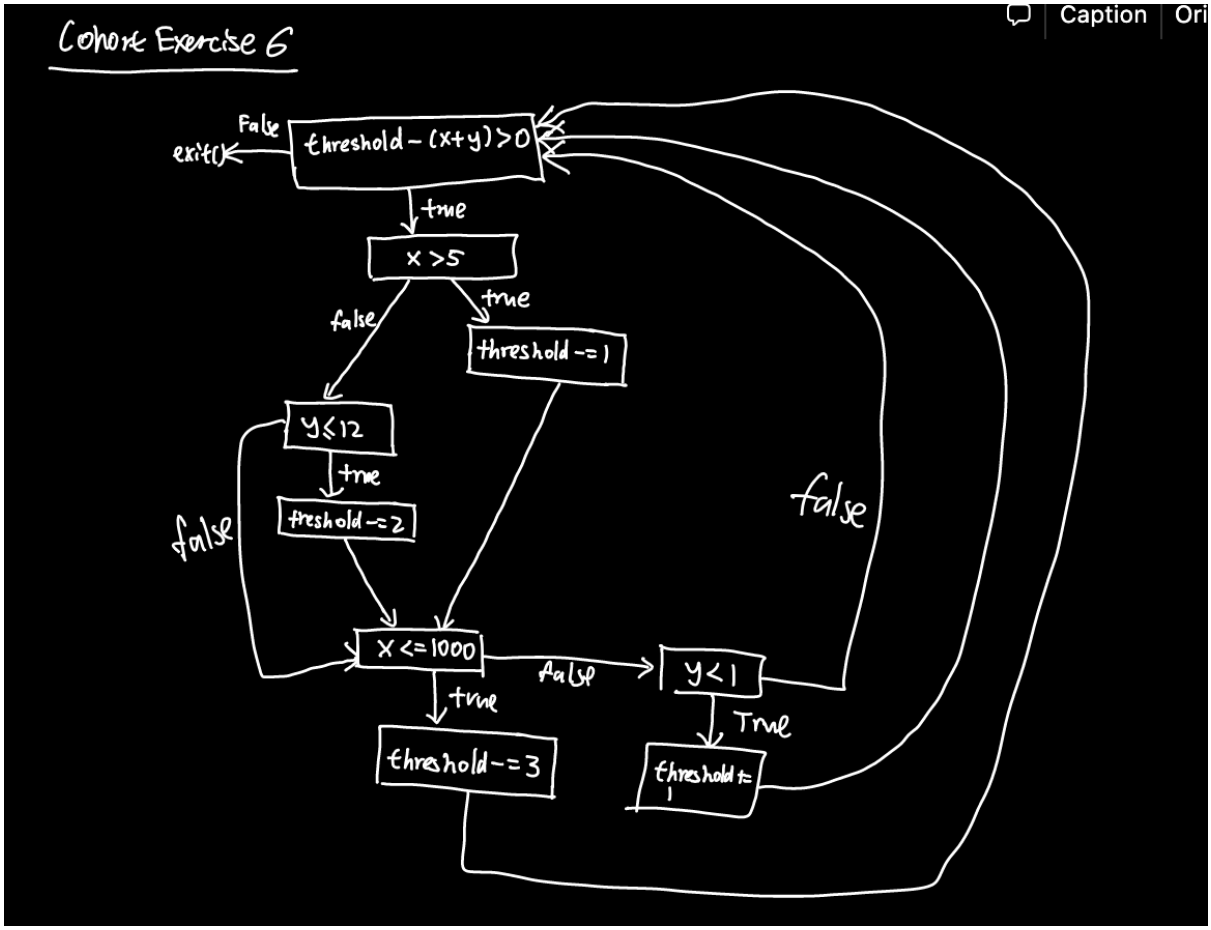
PSET 2

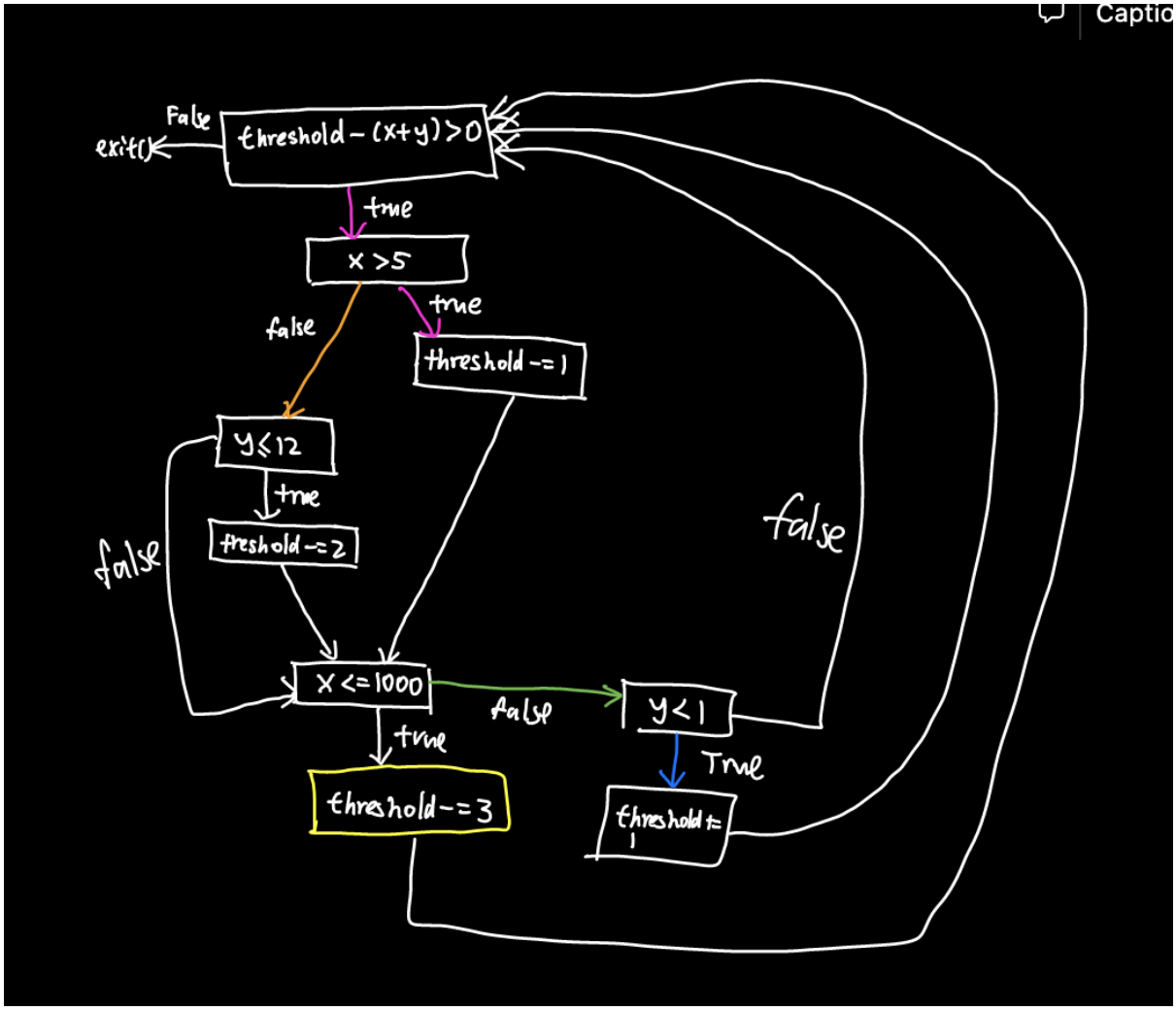
Question 6

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/2d35a608-014e-4ff6-b7b4-30375dfd43f9/PSET_2.pdf



Question 7.

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/8c2072e9-3263-4483-9355-393e57d5a5fc/PSET_3.pdf



Refer to DiskStatementCoverage.java for the following code blocks in the file.

```
public void test1()
```

The function above covers the statements (x≥5) and (x≤ 1000)

```
public void test3()
```

The function above covers the statements (x≤5) and (y≤12)

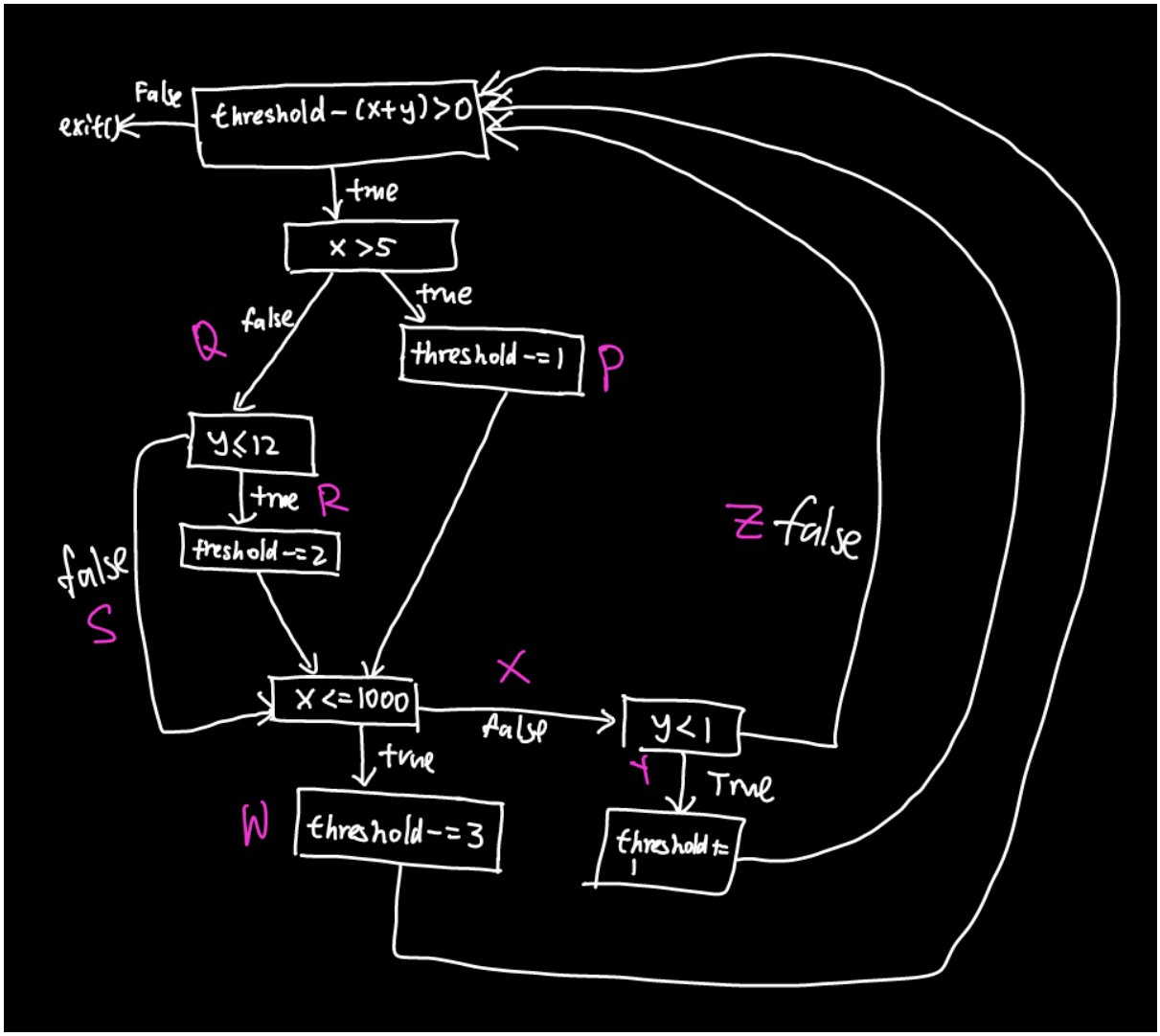
```
public void test2()
```

The function above covers the infinite loop (x>1000) and (y<1) and (x+y < 1000)

The tests written in the file seem to be the minimum number of tests that provide the full statement coverage as it is impossible to traverse the pink and orange paths together as they branch out from the same statement and the orange and green paths together(for the orange path to be executed, the value of x must be less than or equal to 5, hence the yellow statement will always execute. Hence, pairing the green, blue and pink would mean that it is the only case when the green path executes. Hence, 3 tests is the minimum.

Question 8

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/169afbaf-ca2b-4e99-b476-fc3d72b7597b/PSET_4.pdf



4. To achieve branch coverage, we need to add tests that can cover paths S and Z. This cannot be done through the inclusion of including only a single test (x>1000 and y >12) because the condition (threshold - (x+y) > 0) does not hold and the program does not enter the loop. Path Z can never be transversed because both y ≥1 and x > 1000 has to be fulfilled. The path that will traverse S is the path that passes through the statements x < 5 and y > 12 at the same time. A program will never execute branch z, but excluding this and since branch z doesn't contribute statements, branch coverage is achieved.

Question 9

9 paths.

First path, x+y > threshold. Doesn't enter the while loop yet.

paths 2-5: enters each if/else statement if loop only once.

paths 6-9: enter fitst if/else if, and second if/else if.

Question 10

The test cases do not provide condition coverage, condition coverage for this particular program is impossible as y < 1 == false is never executed.

Question 11

The bug is in the infinite loop as implemented in both files of the DIskStatementCoverage.java and the DiskBranchCoverage.java. This is because the threshold, that was modified as a common variable si not decreasing. Hence, an inifinite loop is seen.

Question 12

We first consider on the test cases in Russian.java function that contains the multiplication errors. So for the case of the blackbox, we want to consider the primary test of functionalities. One of which is to test whether the function can take in zero values and then multiply by 0 values. In black box testing, we want to consider all possible use cases to cover. This includes includes every possible input value by the user. from 0, to negative and of course positive. By breaking the code up, we find that negative cases cannot be taken. In white box testing, we aim for branch coverage as well as statement coverage. In our file RussianTest.java, we cover the atomic conditions of n > 0 and n % 2 == 1. Hence, we are able to trace all possible paths. This means 100 % branch coverage. We use RussianFault1.java and RussianFault2.java to prove the two atomic conditions n > 0 and n % 2 == 1 to prove that when n is a multiple of 2, the product is not not added(RussianFault1). For RussianFault2, reveals that when n == 1, no further computations are made. Hence, the white box testing is prepared. For the fault based testing, the fault will be exposed when setting n as a negative integer. Normally, the multiplication would take 2 positive, 1 positive, 1 negative and 2 negative numbers. In this case, only 2 positive, 1

positive and a negative is accounted for. This is a fault as the function is unable to multiply when n < 0.