Lab 7: white box testing

Exercise 1

For the following function define the control flow graph, and define test cases to obtain the highest possible node coverage, edge coverage, multiple condition coverage, loop coverage, path coverage. For the test cases, write only the input value.

```
1 boolean racer_disqualified(int times[3], int winner_times[3], int n_penalties, int*
penalties) {
2
3      bool disqualified = false;
4      int i;
5      int max_time;
6      int tot_penalties;
7
8      for (i = 0; i < n_penalties; i++) {
9             tot_penalties += penalties[i];
10            if (penalties[i] > 100)
11                   disqualified = true;
12     }
13
14     if (tot_penalties > 100 || n_penalties > 5)
15            disqualified = true;
16
17     for (i=0; i<3; i++) {
18
19     max_time = winner_times[i] * 1.5;
20
21     if (time[i] > max_time[i])
22            disqualified = true;
23 }
24
25 }
```

Coverage type	Number of test cases needed to obtain 100% coverage	Coverage obtained with test cases defined (%)	Test cases defined
Node			
Edge			
Multiple condition line 8			
Loop line 6			
Path			

Exercise 2

A function converts a sequence of chars in an integer number. The sequence can start with a '-' (negative number). If the sequence is shorter than 6 chars, it is filled with blanks (to the left side). The integer number must be in the range minint = -32768 to maxint = 32767.The function signals an error if the sequence of chars is not allowed

```
1 int grade(int answers[5][4], int solutions[5][4]) {
2
3      int grade = 0;
4      int i, j;
5
6      for (i = 0; i < 5; i++) {
7             for (j = 0; j < 4; j++) {
8                    if (answers[i][j] == solutions[i][j] && solutions[i][j] == 1)
9                           grade+=3;
10                   else if (answers[i][j] != solutions[i][j] && solutions[i][j] == 0)
11                          grade-=1;
12            }
13     }
14     if (grade < 0)
15            return 0;
16     else
17            return grade;
18 }
```

Coverage type	Number of test cases needed to obtain 100% coverage	Coverage obtained with test cases defined (%)	Test cases defined
Node			
Edge			
Multiple condition line 8			
Loop line 6			
Path			

Exercise 3

Given the example methods for black box testing used in previous lab:

https://git-softeng.polito.it/se-2021/softeng_blackbox_lab

1) Pull the new version of the project. You will find source code for the methods that were previously provided as .jar libraries
2) Install the EclEmma code coverage tool for Eclipse (https://www.eclemma.org/installation.html). If you are using IntelliJ IDEA, you can use the bundled code coverage runner or JaCoCo (https://www.eclemma.org/jacoco/)
3) Compute statement and decision coverage for the test cases you have already defined for the black box testing lab.
4) Fill the white box test report with:
    a. Identification of the test cases
    b. Code coverage report (a screen capture)
    c. Loop coverage analysis for loops in the code. For each loop define three test cases: one for 0 iterations; one for 1 iteration; one for 2+ iterations.