# EECS 4313 Assignment 2
# Black-box and White-box Testing with JUnit

Student Name — Student Number — EECS Account

**Edward Vaisman — 212849857 — eddyv**

**Robin Bandzar — 212200531 — cse23028**

**Kirusanth Thiruchelvam — 212918298 — kirusant**

**Sadman Sakib Hasan — 212497509 — cse23152**

March 5, 2018

# Contents

# 1 Black Box Testing

## 1.1 Equivalence Class Testing

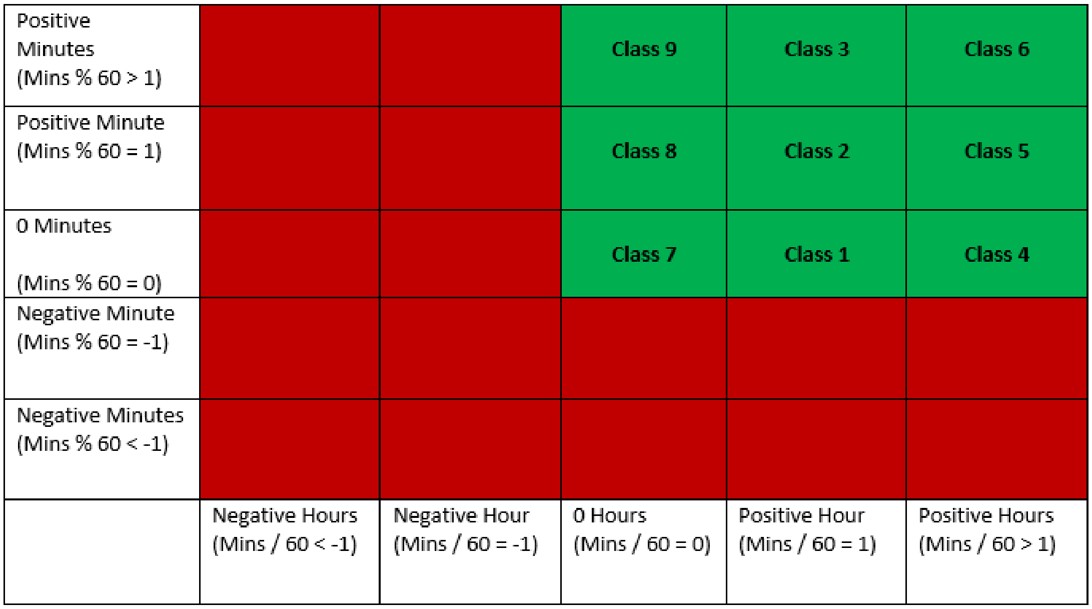
- **Technique**: *Equivalence Class Testing*
- **Class**: *net.sf.borg.common.DateUtil.java*
- **Method**: *minuteString(int mins))*
- **Method Description**: This method generate a human reable string for a particular numbe of minutes.It returns the string interms of hours or minutes or both hours and mintues.
- **mins** - The argument is an integer
- **Justification**: Equivalence class testing is suitable for this method since argument of this method is an integer which is an independent variable and the entire range of input can be partitioned while assuring disjointness and non-redundancy between each partition set. We have chosen these partition integer range based on when we use minute, minutes, hour, and hours.In order to partition the integer argument into hours and minutes,we divide the Minutes by 60 to get the range of hours and the remainder ( minutes % 60) to get the range of the minutes.The paritions for this method are:
  - **Mins / 60 = 1 and Mins % 60 = 0 :** To test 1 hour.
    - Range of hours: [1]
    - Range of minutes: [0]
  - **Mins / 60 = 1 and Mins % 60 = 1 :** To test the 1 hour with 1 minute.
    - Range of hours: (1,+∞)
    - Range of minutes: [1]
  - **Mins / 60 = 1 and Mins % 60 > 1 :** To test 1 hour with minutes more than 1 minute.
    - Range of hours: [1]
    - Range of Minutes : (1, 59]
  - **Mins / 60 > 1 and Mins % 60 = 0 :** To test the hours more than 1 hour.
    - Range of hours: (1,+∞)

        - Range of minutes: [0]
    - **Mins / 60 > 1 and Mins % 60 = 1 :** To test hours more than 1 hour with 1 minute.
        - Range of hours: (1,+∞)
        - Range of Minutes :[1]
    - **Mins / 60 > 1 and Mins % 60 > 1 :** To test the hours more than 1 hour with minutes more than 1 minute.
        - Range of hours: (1,+∞)
        - Range of Minutes : (1, 59]
    - **Mins / 60 = 0 and Mins % 60 = 0 :** To test 0 minutes.
        - Range of hours: [0]
        - Range of minutes: [0]
    - **Mins / 60 = 0 and Mins % 60 = 1 :** To test 1 minute.
        - Range of hours: [0]
        - Range of minutes: [1]
    - **Mins / 60 = 0 and Mins % 60 > 1 :** To test minutes more than 1 minute and less than 60 minutes or 1 hour.
        - Range of hours: [0]
        - Range of minutes: (1,59]

  The method did not specifity how negative minutes should be treated, so we omit the negative integers as an argument for this method.For example, -75 can be converted as -1 hour and 15 minutes or 45 minutes or any other way. Therefore, this case is tested in the whitebox testing after analyzing structure of the method.

- **Evaluation :** The tests are shown below suitable for strong normal equivalence class testing technique since it covers the all the range of outputs for valid inputs and invalid inputs (negative integers) are not tested due to lack of specfication information regarding these values.
    - **Class 1:** Mins / 60 = 1 and Mins % 60 = 0
    - **Class 2:** Mins / 60 = 1 and Mins % 60 = 1
    - **Class 3:** Mins / 60 = 1 and Mins % 60 > 1
    - **Class 4:** Mins / 60 > 1 and Mins % 60 = 0

- **Class 5:** Mins / 60 > 1 and Mins % 60 = 1
- **Class 6:** Mins / 60 > 1 and Mins % 60 > 1
- **Class 7:** Mins / 60 = 0 and Mins % 60 = 0
- **Class 8:** Mins / 60 = 0 and Mins % 60 = 1
- **Class 9:** Mins / 60 = 0 and Mins % 60 > 1



| | | | | | |
|---|---|---|---|---|---|
| Positive Minutes (Mins % 60 > 1) | 🟥 | 🟥 | Class 9 | Class 3 | Class 6 |
| Positive Minute (Mins % 60 = 1) | 🟥 | 🟥 | Class 8 | Class 2 | Class 5 |
| 0 Minutes<br>(Mins % 60 = 0) | 🟥 | 🟥 | Class 7 | Class 1 | Class 4 |
| Negative Minute (Mins % 60 = -1) | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 |
| Negative Minutes (Mins % 60 < -1) | 🟥 | 🟥 | 🟥 | 🟥 | 🟥 |
| | Negative Hours (Mins / 60 < -1) | Negative Hour (Mins / 60 = -1) | 0 Hours (Mins / 60 = 0) | Positive Hour (Mins / 60 = 1) | Positive Hours (Mins / 60 > 1) |

Figure 1: Proving the test cases produce Strong Normal ECT

In the Figure 1, The boxes colored in green represent all the valid output results and red boxes represent the invalid results which cannot be tested since the method did not specify how the negative integers should be treated. The Class 1,Class 2....Class 9 represents the partitions that we derived from how to convert the integers into readable string.

# 2 White Box Testing

- The statement coverage measurements for your Assignment 2 test suite.
- A description of the test cases that you added in this assignment to improve statement coverage. The marker will not read your code in order to see what you tested. You have to describe it.
- The statement coverage measurements for your final submission. Include the screenshots of the test running results and the screenshots of the coverage measurement. If your coverage is not 100%, include a discussion on why that is.
- The Control Flow Graph you created. Indicate the segments clearly (you will probably need to include the code for this).
- The path coverage discussion described in section 2 above.
- Attaching bug reports if bugs are discovered using your testing methods. You should use the same bug report format as in Assignment 1. Do not file these bug reports to the projects bug report system.
- An appendix with the specification of the methods you are testing (if there are new ones).