**Project 1 Test Plan**

**Program goals and Objectives**

The purpose of this program is to convert milliseconds to hours, minutes, and seconds using a convertMillis method.

**Program Functional Requirements**

1. The user needs to be prompted for a long integer of milliseconds
2. The program must use a convertMillis method that has the user input milliseconds as an input parameter.
3. The convertMillis method must return a string as hours:minutes:seconds.
4. The program must output the returned string from the convertMillis method.

**Test Matrix**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case | Input | Expected Result | Actual Result | Outcome (pass/fail) |
| 1 | 555550000 | 154:19:10 | 154:19:10 | pass |
| 2 | 100 | 0:0:0 | 0:0:0 | pass |
| 3 | 555555555555555 | 154320987:39:15 | 154320987:39:15 | pass |
| 4 | 5 trillion | Error message | 0:0:0 | fail |
| 5 | 10^24 | Error message | Exception in thread "main" java.util.InputMismatchException | pass |

**Pseudocode / Flowchart**

Function Main

Declare long millis

Output "Enter time in milliseconds: "

Input millis

Output converMillis(millis)

Function converMillis(millis)

Declare long seconds = millis / 1000

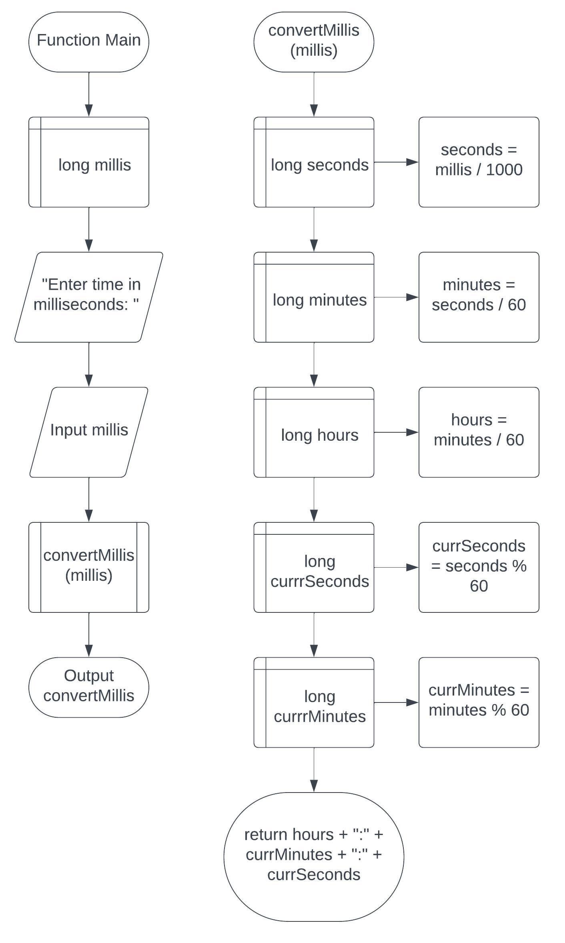
Declare long minutes = seconds / 60

Declare long hours = minutes / 60

Declare long currSeconds = seconds % 60

Declare long currMinutes = minutes % 60

Return hours + ":" + currMinutes + ":" + currSeconds



**Project 2 Test Plan**

**Program goals and Objectives**

The goal of this program is to return a string of a user input number with the user specified width using a “format” method. If the number is longer than the width, the method must return the string representation for the number.

**Program Functional Requirements**

1. The user needs to be prompted for an input number and an input width.
2. The program must execute the “format” method
3. The “format” method has parameters (number, width) and returns a string for the number with one or more prefix 0s. The size of the string is the width.
4. If the number is longer than the width, the method returns the string representation for the number.
5. The program outputs the string return of the “format” method (ex: The formatted number is 00034)

**Test Matrix**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case | Input | Expected Result | Actual Result | Outcome (pass/fail) |
| 1 | 34 5 | The formatted number is 00034 | The formatted number is 00034 | pass |
| 2 | 34 1 | The formatted number is 34 | The formatted number is 34 | pass |
| 3 | Test 1 | Error message | Exception in thread "main" java.util.InputMismatchException | pass |
| 4 | 34 five | Error message | Exception in thread "main" java.util.InputMismatchException | pass |
| 5 | 34 0 | The formatted number is 34 | Exception in thread "main" java.util.DuplicateFormatFlagsException: Flags = '0' | fail |
| 6 | 34 -5 | The formatted number is 34 | Exception in thread "main" java.util.IllegalFormatFlagsException: Flags = '-0' | fail |

**Pseudocode / Flowchart**

Function Main

Declare int number = 0

Declare int width = 0

Output "Enter an Integer: "

Input number

Output "Enter a width: "

Input width

Output "The formatted number is " + format(number, width)

Function format(number, width)

return ("%0" + width + "d", number)

