*UNBC* Laboratory Assignment
Computer Science 101—Winter 2023

# Numbers as Strings

## Due Date:

This assignment is due by Friday, January 13 *by the beginning of lecture*.

## Purpose:

This laboratory assignment is a chance to review problem decomposition by methods.

This is likely the last laboratory assignment to be focussed solely on `static` methods. Use this assignment to practice good functional decomposition.

## Numbers to Strings

⇒ Implement a `static String numberToString(int n)` that converts a integer $n$ between 1 (inclusive) and one billion (exclusive) into a `String`. The `numberToString` method can cover a larger range if you wish. It should take appropriate action if the argument is out of range.

For instance `numberToString(312000789)` should return `three hundred twelve million seven hundred eighty-nine`.

*Carefully* think about how to decompose this problem. For instance, if you have a method that can handle the range 1–999, how can you use that method to create another method that handles the range 1–999 999?

Aim to create methods that have well-defined purposes, but which have no more than around twenty lines of code each. Exploit small private static arrays of `String`s if you can.

⇒ Implement a test method that provides *automated* testing of the `numberToString` method. Here, "automated" means that the test method does not require human input. Think carefully about what test cases provide good automated testing.

⇒ Determine the total number of `'w'`s in the words "one, two, three, . . . , thiry-four million nine hundred ninety-nine thousand nine hundred ninety-eight, thiry-four million nine hundred ninety-nine thousand nine hundred ninety-nine, thiry-five million."

© 2019–2023 David Casperson. This document may be freely copied, provided that this copyright notice is preserved.

## Coding Practices

- Your program must consist of at least three classes:
  - one containing functions to convert positive integers to strings;
  - one containing test functions to test the class above; and
  - One to count `'w'`s.
- Your code should obey the following rules.
  - All member variables must be either private or final (both is fine).
  - Non-void return type functions should have minimal side-effects. Executing the same non-void function twice in a row with the same arguments should produce the same result.
- Your code must conform to the follwing standards:
  - Class names **must be** capitalized.
  - Method names, member variables, and local variables **must** begin with a lower-case letter.
  - (exceptions) public static final variables may have all-caps names.
  - Methods and member variable declarations must be indented with respect to the surrounding class.
  - More generally, method bodies, loop bodies, and so on must be indented.
- Your test methods should be automated. That is, they should not require user input, and should provide fairly terse output, especially if they succeed.

## English Grammar for Numbers

- The first nineteen positive integers are
  - `one` `two` `three` `four`
  - `five` `six` `seven` `eight`
  - `nine` `ten` `eleven` `twelve`
  - `thirteen` `fourteen` `fifteen` `sixteen`
  - `seventeen` `eighteen` `nineteen`

- The first 9 multiples of ten are: `ten`, `twenty`, `thiry`, `forty`, `fifty`, `sixty`, `seventy`, `eighty`, and `ninety`.
- Note the spellings of `four`, `fourteen`, and `forty`.
- Also note that `eighteen` has only one `'t'`.
- For $20 \leq n < 100$ where $n = a \cdot 10 + b$, and $2 \leq a < 10$, and $0 \leq b < 10$, first comes the word for $10\,a$ , then, if $b > 0$, a hyphen and the word for $b$.
- For $100 \leq n < 1000$ where $n = a \cdot 10^2 + b$, and $1 \leq a < 10$, and $0 \leq b < 10^2$, first comes the word for $a$ then a space then "`hundred`", then, if $b > 0$, a space and the words for $b$.
- For $10^3 \leq n < 10^6$ where $n = a \cdot 10^3 + b$, and $1 \leq a < 1000$, and $0 \leq b < 10^3$, first comes the word(s) for $a$ then a space then "`thousand`", then, if $b > 0$, a space and the words for $b$.
- For $10^6 \leq n < 10^9$ where $n = a \cdot 10^6 + b$, and $1 \leq a < 1000$, and $0 \leq b < 10^6$, first comes the word(s) for $a$ then a space then "`million`", then, if $b > 0$, a space and the word(s) for $b$.
- The word "`and`" is not used for integers. (`one hundred and twenty-three thousandths` means `100.023`; whereas `one hundred twenty-three thousandths` means `0.123`)