Cosc 201
Algorithms and Data Structures
Lecture 15 (20/4/2025)
Hashing Applications

Brendan McCane
brendan.mccane@otago.ac.nz
And
Michael Albert

UNIVERSITY OF OTAGO
Te Whare Wānanga o Otāgo
NEW ZEALAND

The plan

To illustrate the utility and convenience of using hash tables and in particular `HashMap`, I'm going to present code for:

- A basic lookup function for words in te reo Māori and their English equivalents based on a list of most common Māori words provided by the Ministry of Education for educational reference.
- A nonsense word generator that generates words that look like sensible English words (but usually aren't).

Frequent Māori words

- Imagine a sort of a flashcard app for learning some te reo Māori vocabulary.
- You're shown a word, and have to remember the meaning.
- We'd like to work from an existing bank of vocabulary, but also be flexible enough to permit the addition of new words, and/or updating or extending the English equivalents.
- Initial source data is from a list of most common Māori words.
- Remember to check that your keyboard settings allow the use of macrons if you're working with the code.

Plan

- We're building a simple lookup table.
- A `HashMap` will store each Māori word in our list (key) and a `String` representing the English translations (value) (we could use an `ArrayList<String>` as well).
- There's some work in getting the initial data into a form that can easily be read into our program – this is quite typical!
- Mainly for testing, our command-line application will take a Māori word as input and print its translations (if known).

LLMs kindof work like this ...

- The problem is to generate plausible-looking nonsense words in English.
- What does that mean?
- You know it when you see it. ‘Jhgz’ is not plausible, but ‘folgat’ is.
- The plan (initially) is to do the following:
  - Work with a dictionary of words.
  - Choose the first letter according to the correct relative frequencies of first letters of words.
  - Having chosen a letter, choose the next letter according to the frequencies of letters (or end-of-word) that follow it.
  - Stop when you choose end-of-word.
- An extension is to use prefixes of more than one character.
- Note that this is an improvement on the nonsense word generator we looked at it Lecture 8. That version used the letter frequency only. This version will also use the context.