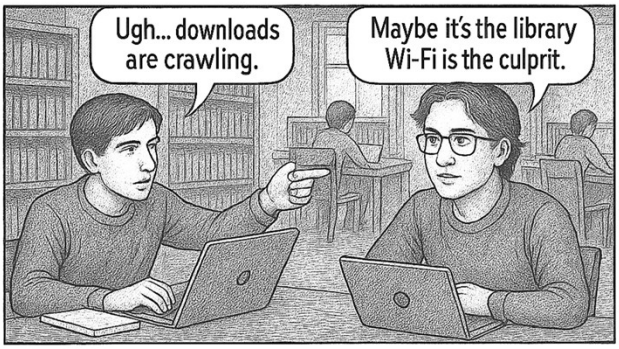
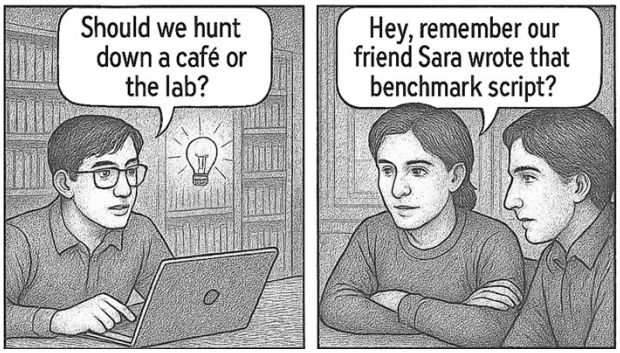
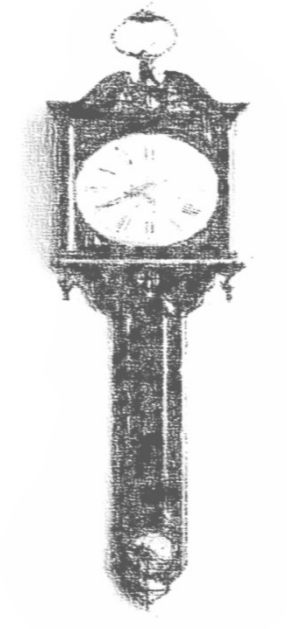
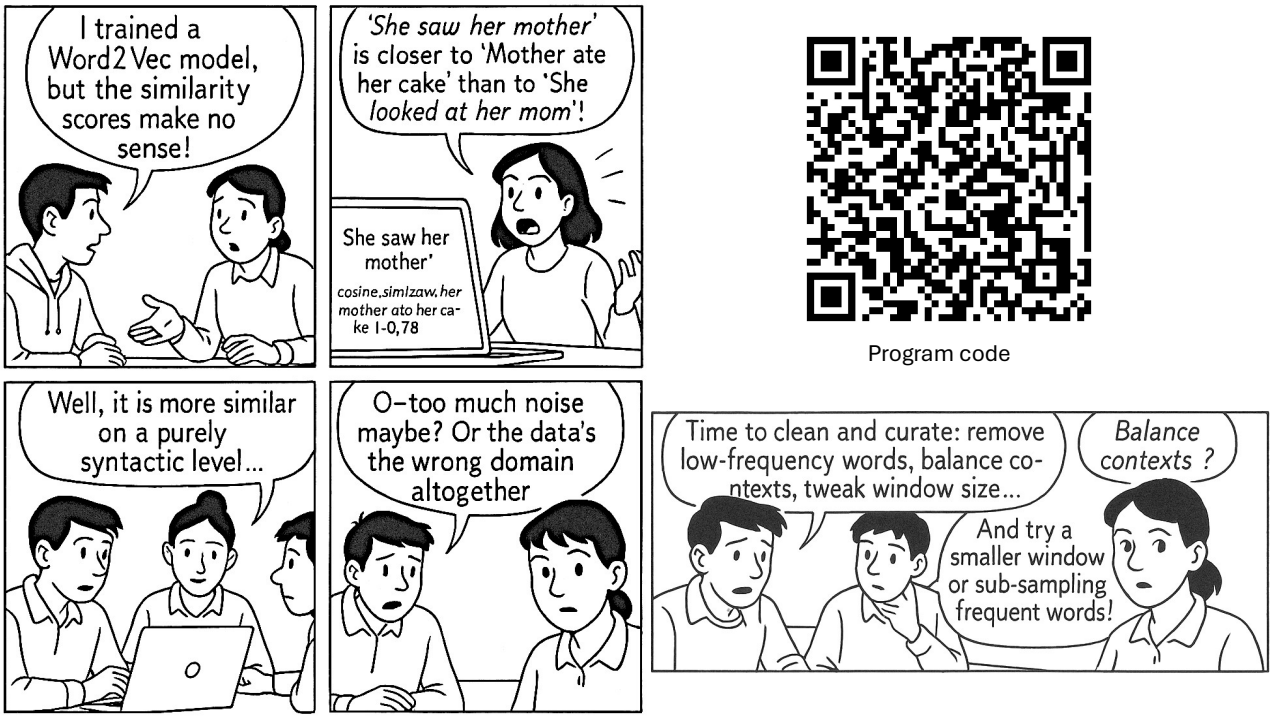


Program code

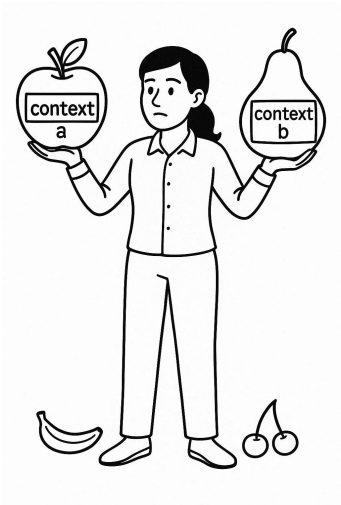


- **Script Purpose:** Summarise what Sara's benchmark script evaluates in one sentence.
- **Wall-Clock Time:** Define "wall-clock time" in this context and explain its significance as a metric.
- **Downloaded Files:**
  a) Identify the type of file the script downloads.
  b) What URL template (BASE_URL) is utilized to construct those file URLs?
- **Download Order:** In what order are files downloaded by each approach?
- **URL Manipulation:** Alter the folder segment (before page number) in BASE_URL from /4/ to /1/, /2/, and /3/. What occurs when you execute the script each time? Why?
- **Choosing an Approach:** Among the three download methods (Hybrid Async-Multiprocess, Sequential, Multithreading), which would you choose for the quickest, most dependable bulk download?
- **How It Works:** Describe in your own words the functioning of your selected download method (for instance, "Multiprocessing distributes the URL list across processes…" or "Asyncio sends multiple non-blocking requests in a single loop…").

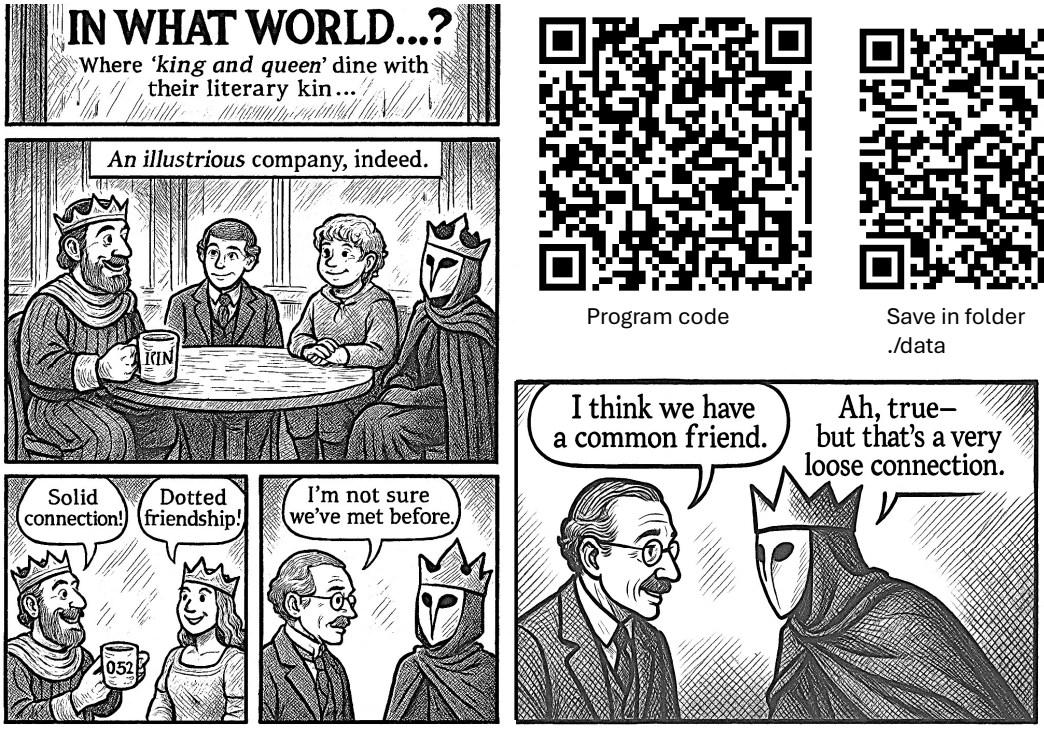




Program code



1. **Run the script and inspect the code.**
   - What does "Rank" refer to in this context?
   - What does the "Score" measure?
2. **Refactor Long Methods**
   - Structure particularly long methods in the program by adding subheadings and, where possible, extract these code sections into separate methods.
3. **Define the optimal outcome.**
   - What would you expect for an ideal result?
   - Why isn't this ideal achieved given the available training data?
4. **Assess the impact of corpus size.**
   - In your view, does a larger training corpus always lead to better performance?
   - Given that our example datasets are very small, would adding more data likely improve the results?
5. **What is meant by "strategies"?**
   - Exactly what does each function—apply_boosting_strong, apply_reduction_strong, apply_doubling (and the double-doubling variant), and apply_phrase_swapping—do?
6. **Compare results across training sets based on what is in the datasets**
   - Which strategy would you choose in a project, where you want to deal with bias in the data? Consider practical arguments.



Program code

Save in folder ./data

1. **Interpret the Visualization**
   - How do you "read" the resulting Graphviz diagram?
   - What do the **edge colors**, **line styles** (solid vs. dotted), and **pen widths** encode?
   - How do node shapes or fill-colors distinguish "king" vs. "queen" nodes?
2. **Refactor Long Methods**
   - Structure particularly long methods in the program by adding subheadings and, where possible, extract these code sections into separate methods.
3. **Compare Individual Models**
   - Examine the subgraphs for the austen, tolkien, and got models. Are there neighbours for each target word ("king" and "queen"), that appear in all three models? Note differences across models. What might these differences reveal about the source texts?
4. **Evaluate the Combined Model**
   - The combined model is trained on the union of all sentences.
   - Do its nearest neighbors for "king" and "queen" seem more semantically coherent or varied than in individual models?
   - Which model (individual vs. combined) would you choose to embed the concepts of royalty most robustly, and why?

Save in folder

Save in folder ./data

Program code

1. **Run & Compare**
   - Compare the two generated graphs: what structural similarities can you spot?
2. **Spot the Differences**
   - Identify and explain at least three differences between the German and English page structures (e.g. number of subsections, ordering of topics, depth of heading levels).
3. **Refactor Long Methods**
   - Structure particularly long methods in the program by adding subheadings and, where possible, extract these code sections into separate methods.
4. **Image Mapping**
   - Look at how figure captions are pulled into the tree.
   - Try to map individual images/captions in the German version to those in the English version. What patterns or anomalies do you notice (e.g. missing captions, different placement)?
5. **BeautifulSoup4 in a Nutshell**
   Research BeautifulSoup . What is it, why is it useful, and what kinds of HTML tasks it helps you perform.
6. **HTML Tags & Semantics**
   - List every HTML tag that the script's parse_structure function extracts.
   - For each tag, explain its usual semantic role in an HTML document.
7. **Swap in "Rubens"**
   - Modify the call in main() (or elsewhere) so the script fetches and analyzes the page for **"Peter_Paul_Rubens"** instead of Rembrandt.
   - Run it and describe in English the high-level structure and any interesting image captions you find.