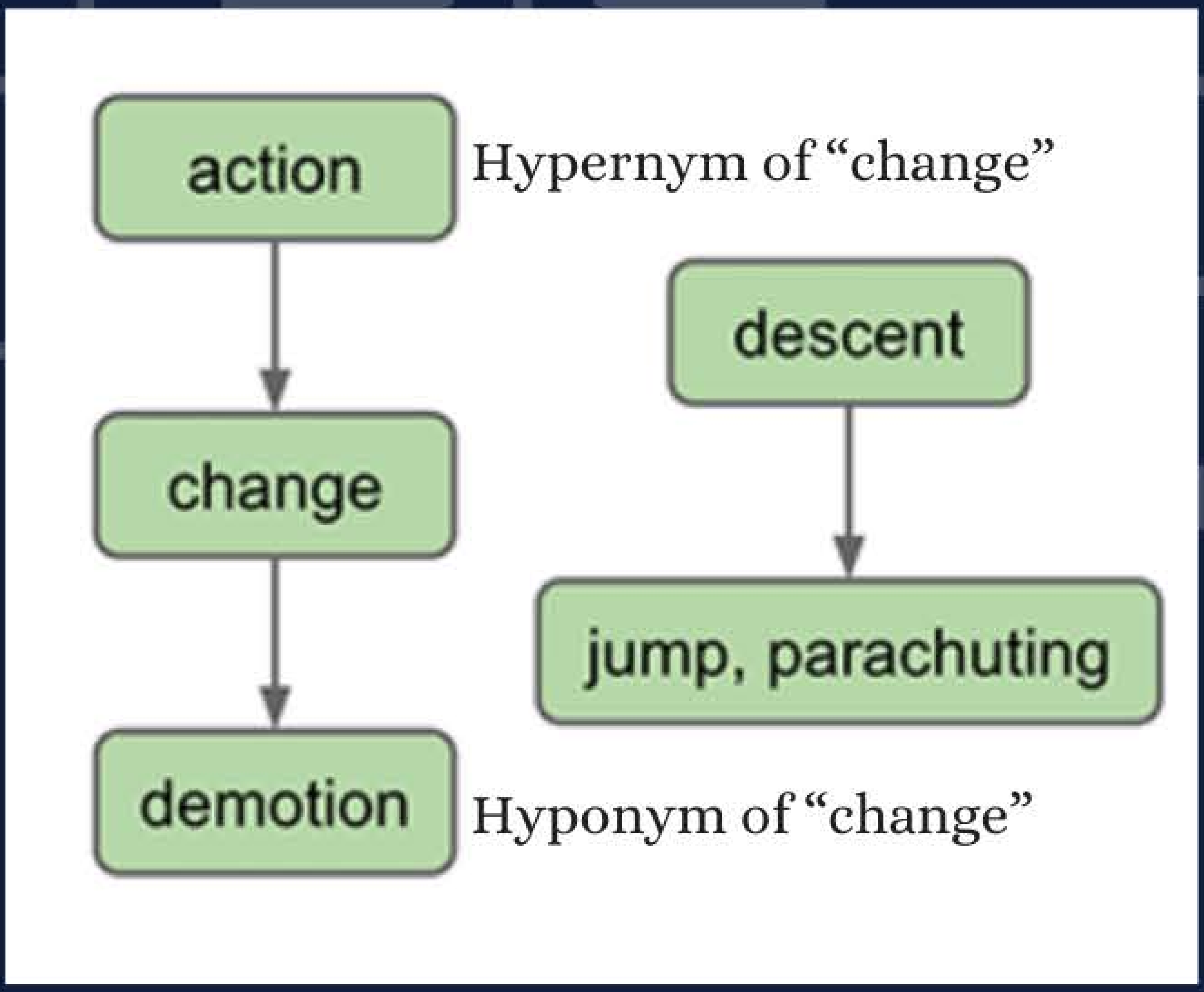


Hypo-Hypernym Handler

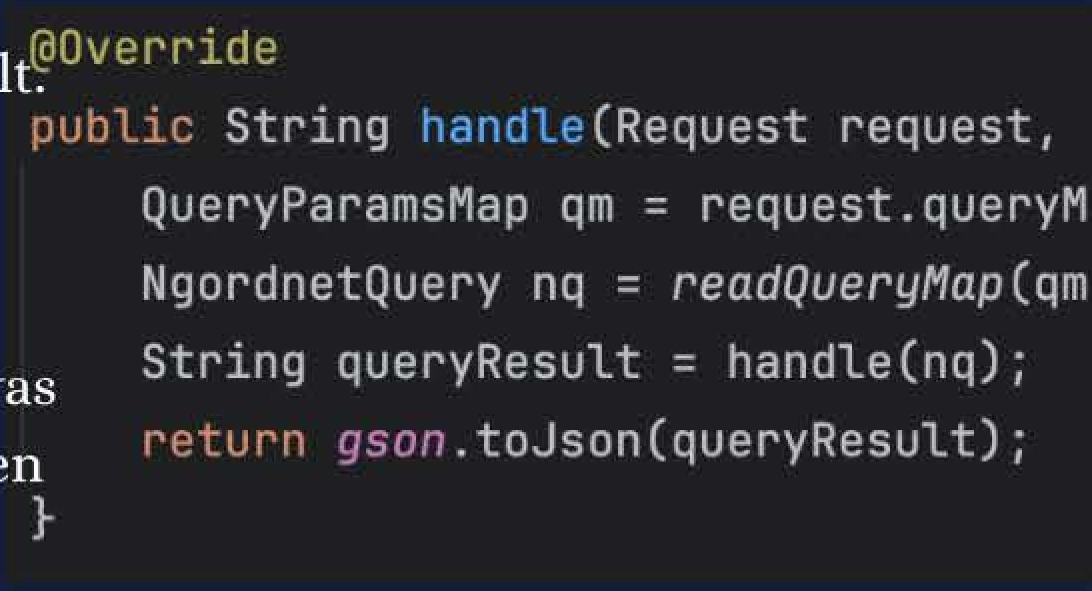
Java

Goal:

Build a website using Google's NGram Database that identifies hypernyms and hyponyms for a word, finds common hyponyms between multiple words, and tracks the frequency of a word's usage in published texts from 1700 to 2020.

Steps and Challenges:

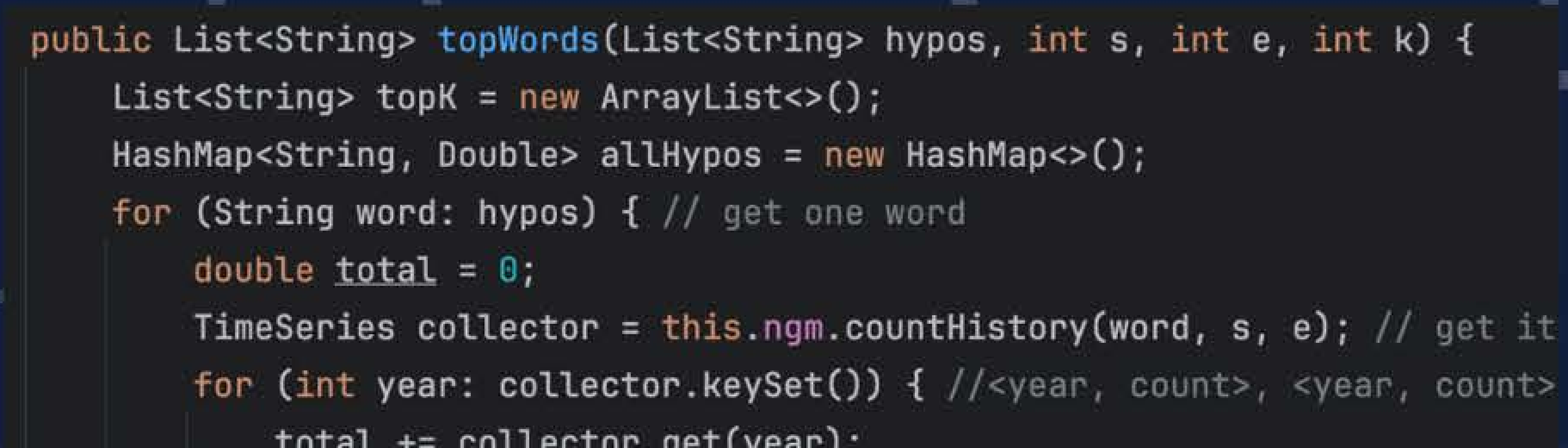
Working with Google's NGram dataset was a learning experience that required me to develop new skills in data parsing. The information wasn't in the most user-friendly format, so I had to figure out how to efficiently extract and organize the data. Once I had parsed the information, the next challenge was grouping words in a way that allowed me to easily find their associated hyponyms and hypernyms. Combining different datasets and using them in various ways proved to be particularly difficult. However, the project became much more manageable once I figured out how to properly contain words along with their related hyponyms and hypernyms. There were countless methods to explore, including ArrayLists, LinkedLists, maps, and trees, each offering a unique approach to structuring the data. The key challenge was to parse through different datasets, collect only the necessary information, and then combine it effectively, all while maintaining efficiency in terms of both space and time.



```
@Override
public String handle(Request request,
    QueryParamsMap qm = request.queryM
    NgordnetQuery nq = readQueryMap(qm
    String queryResult = handle(nq);
    return gson.toJson(queryResult);
}
```

Takeaways:

- The importance of run-time.
  - Working with large pieces of information at a time, being able to think critically about how to clean and store data so as to be able to run through it all, was a game changer. Thinking in this manner from this point on has guided my course of action in all of my projects since then.
- Working with multiple datasets.
  - Finding connectors between datasets was challenging at first, and I didn't know how much or how little to use the information given to me. Being able to think outside the box and work with so much information proved to be a saving grace.
- Complexity is not always better.
  - With so many different ways to code the same idea, using a variety of different data structures, I quickly became lost in the world of Java. I found it most helpful to break down each structure into what I knew about it and what built-in features would be of use. Sticking to a few structures and delving into their capabilities was key in this project.
- Patience amidst distress.
  - I found myself overwhelmed by the grandiose scale of this project as it was hard to focus on just one aspect without being overtaken by the other moving pieces. This project strengthened my ability to work through my ideas in a clean, healthy manner, making me a stronger coder and student.



```
public List<String> topWords(List<String> hypos, int s, int e, int k) {
    List<String> topK = new ArrayList<>();
    HashMap<String, Double> allHypos = new HashMap<>();
    for (String word: hypos) { // get one word
        double total = 0;
        TimeSeries collector = this.ngm.countHistory(word, s, e); // get it
        for (int year: collector.keySet()) { //<year, count>, <year, count>
            total += collector.get(year);
```

Given this is a class project, I cannot put my entire code online. However, I would be more than ready to discuss the code by request.