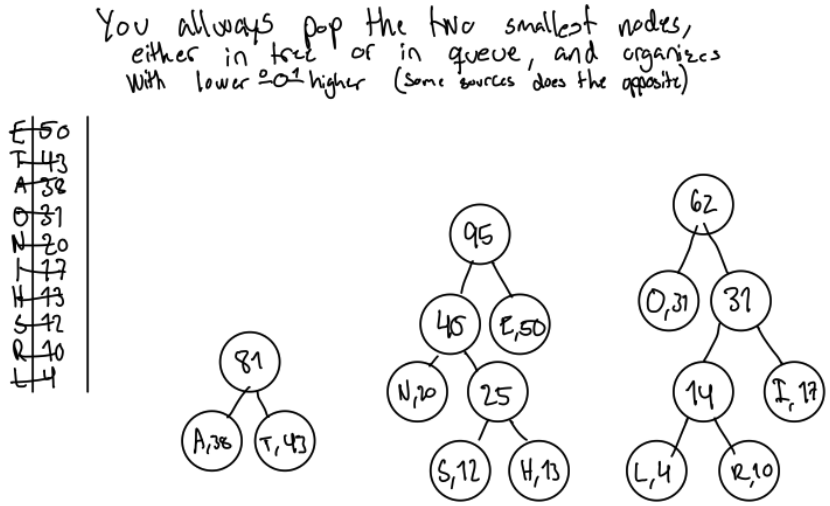
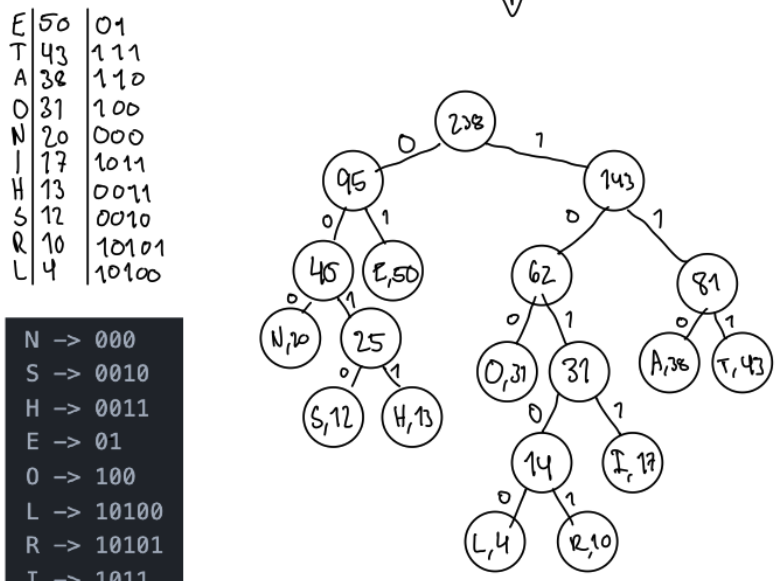
Assignment 4

Task 1

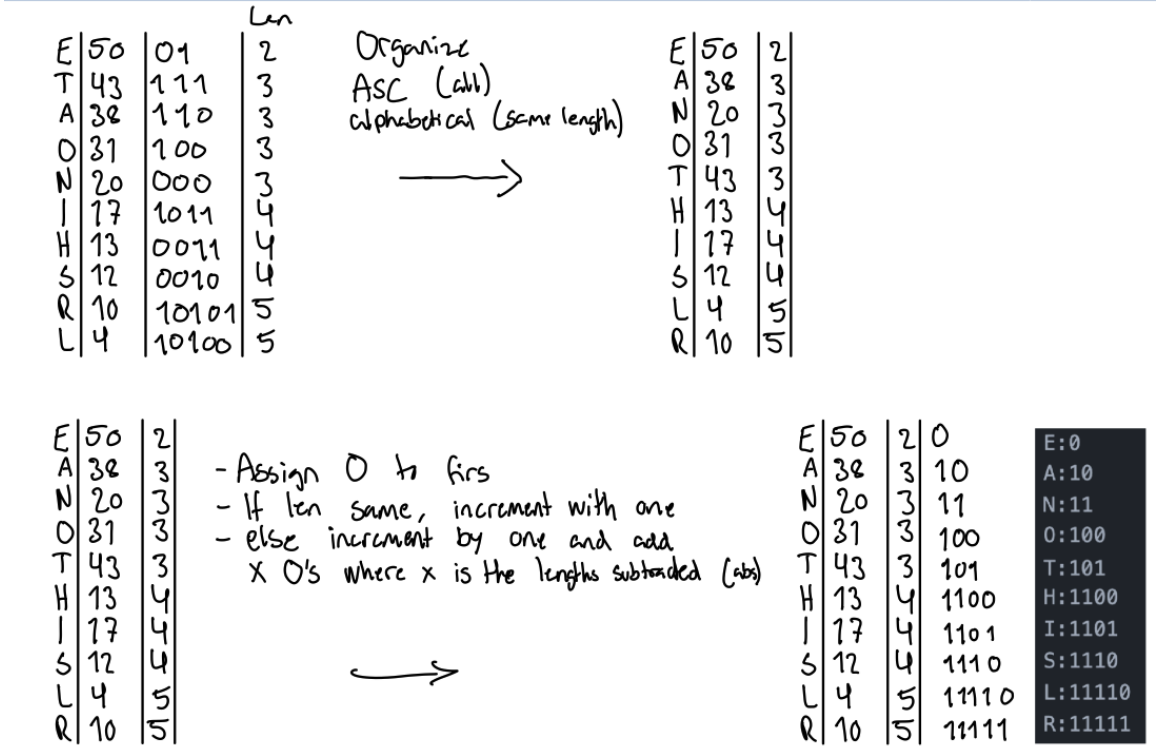
Task 1.1



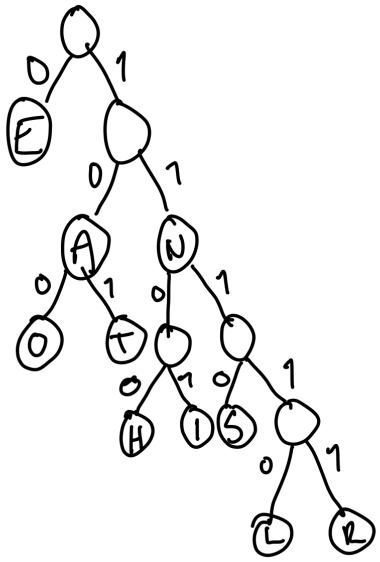



```
N -> 000
S -> 0010
H -> 0011
E -> 01
O -> 100
L -> 10100
R -> 10101
I -> 1011
A -> 110
T -> 111
```

Normal Hoffmann - have to make it canonical:



Can draw a canonical tree matching this?



The average length of the code is:

$$AvgLen = \frac{\sum_{l \in Letters} CodeLen(l) * Freq(l)}{\sum_{l \in Letters} Freq(l)} = \frac{646}{238} = 2.714$$

The average length of the code if the frequency of the letters was equal:

$$AvgLen = \frac{Total\ bits\ needed}{Number\ of\ characters} = \frac{(3 * 6) + (4 * 4)}{10} = \frac{34}{10} = 3.4$$

Task 1.2

For å dekode starter man bare fra start og finner den første av sekvensene i tabellen, og fortsetter videre. Jeg har oppdelt bitsekvensen etter hver verdi i tabellen for å vise det tydeligere.

Letter	Code
A	10
T	11
H	000
U	010
N	011
L	0010
O	0011

000 10 0010 0010 0011

- 000 = H
- 10 = A
- 0010 = L
- 0010 = L
- 0011 = O

0011 10 11 010

0011 = O

10 = A

11 = T

010 = U

Task 2

Task 2.1

Apache Lucene is a Java library providing a high-performance text search engine for indexing and search. In this demonstration we use the Lucene-demo library for indexing

a collection of documents, and search for relevant documents using keywords and these indexes.

It should be mentioned that Lucene has the following properties/features:

- many powerful query types: phrase queries, wildcard queries, proximity queries, range queries and more
- pluggable ranking models, including the Vector Space Model and Okapi BM25
- fielded searching (e.g. title, author, contents)
- index size roughly 20-30% the size of text indexed

And much more!

Task 2.2

The first 30 lines of the main method includes the process of setting indexPath and docsPath, and break down the arguments sent in to the method. When we first reach the try-catch, we get to the indexing. First, it opens an instance of the class `FSDirectory` for writing to this directory. Then, we create an object of the `StandardAnalyzer` class, which is further passed on to the `IndexWriterConfig`. Furthermore, we set the `writerConfig` to either append or create mode based on the use of "-update" argument or not. Lastly, they create an instance of the `IndexWriter`, and use the method `indexDocs()` for indexing using this writer. We close the writer and print the timing, and the main method ends with a catch of possible `IOExceptions`.

During indexing we get the following output to the console:

```
Indexing to directory 'lucene/src/main/java/index/task3'...
adding lucene/src/main/java/data/docs/Text6.txt
adding lucene/src/main/java/data/docs/Text5.txt
adding lucene/src/main/java/data/docs/Text4.txt
adding lucene/src/main/java/data/docs/Text1.txt
adding lucene/src/main/java/data/docs/Text3.txt
adding lucene/src/main/java/data/docs/Text2.txt
221 total milliseconds
```

Task 2.3

```
Searching for: god
2 total matching documents
1. lucene/src/main/java/data/docs/Text1.txt
2. lucene/src/main/java/data/docs/Text2.txt
Execution time (in ms) for the query 'God': 26

Searching for: circumstances
1 total matching documents
1. lucene/src/main/java/data/docs/Text6.txt
Execution time (in ms) for the query 'circumstances': 2

Searching for: claims of duty
6 total matching documents
1. lucene/src/main/java/data/docs/Text6.txt
2. lucene/src/main/java/data/docs/Text1.txt
3. lucene/src/main/java/data/docs/Text4.txt
4. lucene/src/main/java/data/docs/Text5.txt
5. lucene/src/main/java/data/docs/Text2.txt
6. lucene/src/main/java/data/docs/Text3.txt
Execution time (in ms) for the query 'claims of duty': 10
```

For answering on the question regarding the use of "OR" or "AND" in the query model, I ran a quick test:

```
Searching for: claims
1 total matching documents
1. lucene/src/main/java/data/docs/Text6.txt
Execution time (in ms) for the query 'claims': 129

Searching for: duty
1 total matching documents
1. lucene/src/main/java/data/docs/Text6.txt
Execution time (in ms) for the query 'duty': 2

Searching for: of
6 total matching documents
1. lucene/src/main/java/data/docs/Text1.txt
2. lucene/src/main/java/data/docs/Text6.txt
3. lucene/src/main/java/data/docs/Text4.txt
4. lucene/src/main/java/data/docs/Text5.txt
5. lucene/src/main/java/data/docs/Text2.txt
6. lucene/src/main/java/data/docs/Text3.txt
Execution time (in ms) for the query 'of': 1

Searching for: claims of duty
6 total matching documents
1. lucene/src/main/java/data/docs/Text6.txt
2. lucene/src/main/java/data/docs/Text1.txt
3. lucene/src/main/java/data/docs/Text4.txt
4. lucene/src/main/java/data/docs/Text5.txt
5. lucene/src/main/java/data/docs/Text2.txt
6. lucene/src/main/java/data/docs/Text3.txt
Execution time (in ms) for the query 'claims of duty': 11
```

As you may see in the console output above, the terms "claims" and "duty" is only found in text 6, but the term "of" is found in all 6. As the resulting list of the term "claims of duty" includes 6 documents, the model have to use the "OR" operator.

Task 2.4

The *Enron Email Dataset:*

https://www.cs.cmu.edu/~enron/enron_mail_20150507.tar.gz

Norwegian University Science Technology

```
Searching for: norwegian university science technology
3970 total matching documents
1. lucene/src/main/java/data/maildir/lay-k/inbox/1126.
2. lucene/src/main/java/data/maildir/lay-k/inbox/958.
3. lucene/src/main/java/data/maildir/kaminski-v/deleted_items/781.
4. lucene/src/main/java/data/maildir/kaminski-v/inbox/136.
5. lucene/src/main/java/data/maildir/skilling-j/inbox/156.
6. lucene/src/main/java/data/maildir/kaminski-v/deleted_items/776.
7. lucene/src/main/java/data/maildir/kaminski-v/inbox/140.
8. lucene/src/main/java/data/maildir/skilling-j/notes_inbox/383.
9. lucene/src/main/java/data/maildir/skilling-j/discussion_threads/1144.
10. lucene/src/main/java/data/maildir/skilling-j/all_documents/1454.
Execution time (in ms) for the query 'Norwegian University Science Technology': 156
```

Norwegian University of Science and Technology

```
Searching for: norwegian university of science and technology
5055 total matching documents
1. lucene/src/main/java/data/maildir/lay-k/inbox/1126.
2. lucene/src/main/java/data/maildir/kaminski-v/deleted_items/781.
3. lucene/src/main/java/data/maildir/kaminski-v/inbox/136.
4. lucene/src/main/java/data/maildir/lay-k/inbox/958.
5. lucene/src/main/java/data/maildir/skilling-j/inbox/156.
6. lucene/src/main/java/data/maildir/kaminski-v/deleted_items/776.
7. lucene/src/main/java/data/maildir/kaminski-v/inbox/140.
8. lucene/src/main/java/data/maildir/skilling-j/notes_inbox/383.
9. lucene/src/main/java/data/maildir/skilling-j/discussion_threads/1144.
10. lucene/src/main/java/data/maildir/skilling-j/all_documents/1454.
Execution time (in ms) for the query 'Norwegian University of Science and Technology': 222
```

I thought it was interesting to see the differences when including "of" and "and" of the university's full name or not. Two differences is to be spotted, one being the slight increase in time, and the other being the similar results except for one inbox climbing some spots. If searching for **"Norwegian University of Science and Technology"** as a phrase, we get the following results:

```
Searching for: "norwegian university of science and technology"
1 total matching documents
1. lucene/src/main/java/data/maildir/lay-k/inbox/1126.
Execution time (in ms) for the query '"Norwegian University of Science and technology"': 193
```

For curiosity I also searched for **Apple:**

```
Searching for: apple
672 total matching documents
1. lucene/src/main/java/data/maildir/corman-s/sent_items/3.
2. lucene/src/main/java/data/maildir/lewis-a/deleted_items/701.
3. lucene/src/main/java/data/maildir/farmer-d/inbox/97.
4. lucene/src/main/java/data/maildir/corman-s/deleted_items/6.
5. lucene/src/main/java/data/maildir/hernandez-j/discussion_threads/202.
6. lucene/src/main/java/data/maildir/hernandez-j/sent/108.
7. lucene/src/main/java/data/maildir/hernandez-j/_sent_mail/108.
8. lucene/src/main/java/data/maildir/hernandez-j/all_documents/586.
9. lucene/src/main/java/data/maildir/corman-s/inbox/23.
10. lucene/src/main/java/data/maildir/corman-s/deleted_items/55.
Execution time (in ms) for the query 'Apple': 178
```

The Lucene search is quick, being able to search through 500.000 emails and rank regarding a query in 0.178s.