**Javadoc Link**

(tba)

**Compilation**

The program consists of 4 different Java files.

To compile all the files run the following command:

javac Encode.java Access.java State.java Date.java

**Execution**

There are two different executable programs; Encode and Access.

**Encode**

Encode expects a single argument consisting of the properly formatted “state” data file.

java Encode [state data file]

Example)

java Encode States90.txt

**Access**

Access expects two arguments consisting of the index file and then the encoded data file.

java Access [index file] [encoded file]

Example)

java Access states.idx states.enc

**Encoding the Data**

Encode.java completes the file encoding in two steps.

First, it parses the provided states data file. It reads each state and its data fields using a Scanner object. The text data is then converted into a State object and put into an ArrayList of State objects. Each State in the data file is present in the ArrayList.

Encode then goes through each State object in the ArrayList and retrieves the values of each field. It writes these fields in a specific order into the encoded file. Each state object is separated by a ‘~’ (tilde) character in the encoded file. Between each of the State’s fields, it separates them using a ‘#’ character. This was necessary because some of the fields are multi-word and you need to know where one field stops and the next starts. As each State is written to the encoded file, there is a counter that keeps track of the total number of characters that have been inserted. Encode.java also creates an index file while this process is occurring. The index file has a single line for each State that is inserted into the encoded file. As the encoded file is being written to, each State name is written to the index file along with its starting position in the encoded file.

Encode generates two files when finished successfully; “states.idx” and “states.enc”.

Original:

OHIO

10847115 7 264.9 41330 35 3 1 1803 17 COLUMBUS

The above state would be encoded like this in the encoded file:

OHIO#10847115#7#264.9#41330#35#3#1#1803#17#COLUMBUS~

And the corresponding entry in the index file would look like this:

OHIO 0

**Accessing Data From the Encoding**

Access.java needs the index file and the encoded data file provided to it when executing the program.

When Access is run, it reads from the index file line by line. For each line, it splits the string into two on the last space character found. We now have the state name and it’s location in the encoded data file. The state is stored in a Java HashMap object as the key and the location is stored as an integer value.

When the user selects that they want to search for a specific state, they enter a specific state name and it is searched for in the HashMap. The HashMap returns the location, and then we are ready to retrieve the data from the encoded file.