Name: Akash Singh (Student ID: 801022198)

Program design:

This project implements Lempel-Ziv-Welch algorithm. The project consist of two modules

1) **Encoding**: Encoding class is used to implement the encoding/compression step of the algorithm

- In this module we are encoding the data which we read from the input file which is given through command line argument. The command line argument takes text filename and number of bits as inputs.
- Read the data from file and stored it in a String called Symbols.
- Implemented the Encoding algorithm as per pseudo code and stored the output integers in List called outputs.
- Wrote the output in the encoded file with lzw extension using UTF-16BE encoding

2) **Decoding**: Decoding class is used to implement the encoding/compression step of the algorithm

- In this module we are decoding the data which we have created in the above encoding module

- Decoding class accepts this file along with encoding bits size through command line argument.
- Implemented the decoding algorithm as per decompressing pseudo code and stored the output integers in List called outputs
- Converted this output to string and wrote it as filename_decoded.txt file

Data structure design

Encoding module:

- Linked Hash Map: This is used as a dictionary which have key, value pair as String, Integer respectively
- Array List: This is used to store output codes in Integer form

Decoding module:

- Linked Hash Map: This is used as a dictionary which have key, value pair as Integer, String respectively
- Array List: This is used to store decoded values in String form

The breakdown of the files

Two files encoding and decoding are used to implement compression and decompression steps of the algorithm.

Encoding/Compression:

The Encoding.java files is used to implement compression/encoding

- It takes the input file to be compressed and bits size from command line argument
- It outputs the compressed file using UTF-16BE encoding in java
- It contains a main method which is the acts as driver method from where the program execution for the encoding module starts.

Decoding/Decompression:

The Decoding.java files is used to implement decompression/encoding

- It takes the input file which was generated from encoding module from command line argument
- It reads the compressed encoding file using UTF-16 encoding and outputs the decompressed file as filename_decoded.txt
- It contains a main method which is the acts as driver method from where the program execution for the decoding module starts.

Summary:

I observed that when the bit size given through command line is 12 we get the compressed file i.e. the output generated from encoding module have file size less than the input file given to encoding module. However when the bits are reduced, for instance bits = 4 In this case the file generated by encoding module was found to be greater than the file size of the input file. Thus we can conclude that the compression does not work and the algorithms does not properly work to compress the file.

I also observed that if the input text file size is very less for instance if file consist of just few character then the file generated after encoding is not compressed i.e. I observed that the file size is not reduced further.

Programming language and compiler version

I have used Java 1.8 as a programing language to implement Encoding and decoding modes

Compiler Version 1.8

Running the program:

Place the two java files (Encoding.java and Decoding.java) in current directory

Compile the two java files using cmd as below

Encoding:

```
javac Encoding.java
```

Decoding:

```
javac Decoding.java
```

Run the generated .class file step as below

Running the encoding class file

```
java Encoding inputfilename.txt 12
```

Running the decoding class file

```
java Decoding inputfilename.lzw 12
```

We get inputfilename.lzw on running the Encofing.java file and inputfilename_decoded.txt on running the Decoding.java file. We can verify the inputfilename_decoded.txt with inputfilename.txt file to verify if the two files are same.
