CS 312

Lab 6 – Single Error Correcting (SEC) Code

Description

Develop a Single Error Correcting (SEC) Code in the **Java** programming language, which recives two (binary) files and computes if a transmission error has occurred, and if detected, the bit location of the error.

The input of the program is the following:

- ***Transmit file*** – a binary file containing the transmitted bits (data and check bits)
- ***Receive file*** – a binary file containing the received bits (data and check bits). Four different received files are provided with different data and scenarios.

The program should do the following:

- Retrieve the binary data from the **transmitted** file and calculate its size (in bytes).
- Compute the data (M) and check (K) bits in the file
- Calculate and display the bit location of the K check bits.
- Extract and display the K check bits.

- Retrieve the binary data from the **received** file and calculate its size (in bytes).
- Compare the file sizes, and if they are different, display error message and exit program.
- If file sizes are the same then:
    - Compute the data (M) and check (K) bits in the file
    - Calculate and display the bit location of the K check bits.
    - Extract and display the K check bits.

- Compute the **syndrome word**, by doing the XOR between the two check bits of the transmitted and received files.
    - If the syndrome contains all 0's, no error has been detected.
    - If the syndrome contains one and only one bit set to 1, then an error has occurred in one of the check bits. No correction is needed.
    - If the syndrome contains more than one bit set to 1, then the numerical value of the syndrome indicates the position of the data bit in error.

Example

Assume that the transmitted file contains the following data using the SEC correction code:
111011001011011001100

Transmitted file read

The program should read in the transmitted file data and display the following in the console as given below.

```
Transmitted file content: 111011001011011001100
Total number of bytes read: 21 bytes
```

Calculate the data (M) and check (K) bits

The program should compute the M and K bits from the data using the following equation:

$$2^K - 1 \geq M + K$$

This is then displayed as the following:

```
M data bits is: 16
K check bits is: 5
```

Display the check bits for transmitted file

The bit location and those bits from the transmitted file should then be displayed as the following: Note: Locations indexed from right to left starting from 0; check bit values listed from left to right.

```
Location of the k check bits are: 0 1 3 7 15
The k check bit values are: 1 1 1 0 0
```

Received file read

The program should then read in the received file data and display the following in the console as given below. Assume that it is **111011101011001000111.**

```
Received file content: 111011101011001000111
Total number of bytes read: 21 bytes
```

File size checking

The program should then check if both the files are the same size. If they are not (assume that the file data is **1110110010110110011001)**, then it should display an error message and terminate as given below.

```
Received file content: 1110110010110110011001
Total number of bytes read: 22 bytes
Files are not the same size!
```

Display the check bits for received file

The bit location and those bits from the received file should then be displayed as the following:

```
Location of the k check bits are: 0 1 3 7 15
The k check bit values are: 1 0 0 1 1
```

Note: Locations indexed from right to left starting from 0; check bit values listed from left to right.

Calculate the syndrome word

If the two files are equal and the syndrome word does not contain any 1's then the following should be displayed:

```
The syndrome word is:  0 0 0 0 0
No error detected in the received file.
```

If the syndrome word contains one and only one bit set to 1, then an error has occurred in one of the check bits. No correction is needed. This should be displayed as the following:

```
The syndrome word is:  0 0 0 0 1
Only one syndrome bit is set to 1. The error bit is one of the check bits. No correction needed.
```

If the syndrome contains more than one bit set to 1, then the numerical value of the syndrome indicates the position of the data bit in error. The syndrome word is calculated by doing the XOR operation on the two check bit fields.

$$\begin{array}{cccccc} & 1 & 1 & 1 & 0 & 0 \\ \oplus & 1 & 0 & 0 & 1 & 1 \\ \hline & 0 & 1 & 1 & 1 & 1 \end{array}$$

The equivalent decimal bit location is calculated from the syndrome work (i.e. 15 in this example), which should be displayed as the following:

```
The syndrome word is:  0 1 1 1 1
The location of the error bit in the received data is: 15
```

Note: Locations indexed from right to left starting from 0; check bit values listed from left to right.

Hence, the location 15 is in fact at the location with index number 14. Right?P

The following is the total program convocation:

```
Transmitted file content: 11101100101101100100
Total number of bytes read: 21 bytes

M data bits is: 16
K check bits is: 5

Location of the k check bits are: 0 1 3 7 15
The k check bit values are: 1 1 1 0 0

Received file content: 11101110101100100011 1
Total number of bytes read: 21 bytes

Location of the k check bits are: 0 1 3 7 15
The k check bit values are: 1 0 0 1 1

The syndrome word is:  0 1 1 1 1
The location of the error bit in the received data is: 15
```

Experiment

One transmitted file and four different received files are provided and the code needs to be test on all these different files. Please ensure that the code is properly documented.

Submission

The **student's name**, **ID** and **honor code** is required all submitted files.

Submit the **Java** code file to Canvas

The grading rubric is given as the following:

Description	**Points**
Binary file reading	20
Data and check bit calculations	20
Correct syndrome word calculation	20
Input and output display	20
Correct error checking	10
Code documentation	10