Assignment No. 3

Meet Patel

(B00899516)

CSCI 6704: Advance Topics in Networks

Dr. Srini Sampalli

Dalhousie University

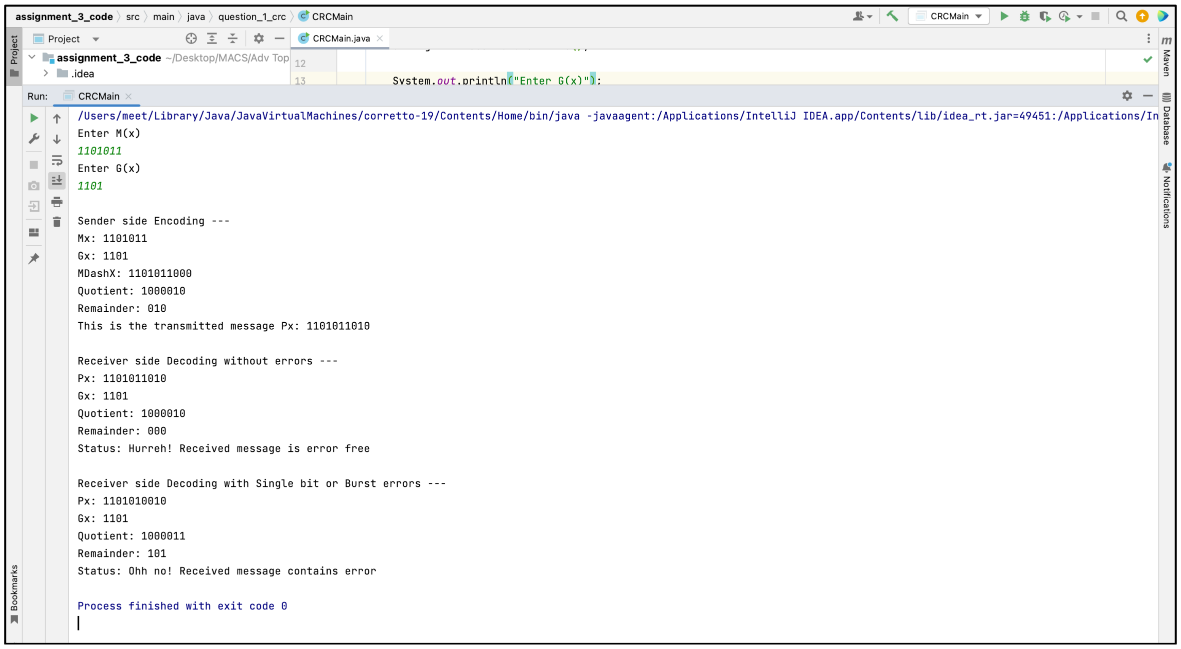
Fall 2022

Programming Exercise 1 <CRC Error Check>

In this exercise, you will implement the sending and receiving CRC protocol by writing functions/methods in a program for each of the following:

a. Method 1 (Sender): Given a message M(x) (a bit string) and the reference polynomial G(x) (another bit string), the method should compute the CRC remainder and determine P(x) - the bit string that will be transmitted.
   As an example, if M(x) = 1101011 and G(x) = 1101, the method should return P(x) = 1101011010.
   You can store and process the numbers as characters (1 and 0) or strings.
   Do not use built-in methods to perform the CRC division.
b. Method 2 (Receiver): Given a bit string with CRC remainder appended, this method should divide the bit string by G(x) and determine if the message is error-free or not.
c. Use the above methods in a test program that accepts from user input the values of G(x) and the input string, introduce random errors in the transmitted bit string and demonstrate how the receiver can detect the error.

Sample Run for CRC Error Check – 1


assignment_3_code src main java question_1_crc CRCMain
Project
assignment_3_code ~/Desktop/MACS/Adv Top
.idea
CRCMain.java
Run: CRCMain

```
System.out.println("Enter G(x)");
```

```
/Users/meet/Library/Java/JavaVirtualMachines/corretto-19/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=49451:/Applications/In
Enter M(x)
1101011
Enter G(x)
1101

Sender side Encoding ---
Mx: 1101011
Gx: 1101
MDashX: 1101011000
Quotient: 1000010
Remainder: 010
This is the transmitted message Px: 1101011010

Receiver side Decoding without errors ---
Px: 1101011010
Gx: 1101
Quotient: 1000010
Remainder: 000
Status: Hurreh! Received message is error free

Receiver side Decoding with Single bit or Burst errors ---
Px: 1101010010
Gx: 1101
Quotient: 1000011
Remainder: 101
Status: Ohh no! Received message contains error

Process finished with exit code 0
```

Figure 1: Output of CRC Error Check for input string M(x) = 1101011 and G(x) = 1101

Sample Run for CRC Error Check – 2

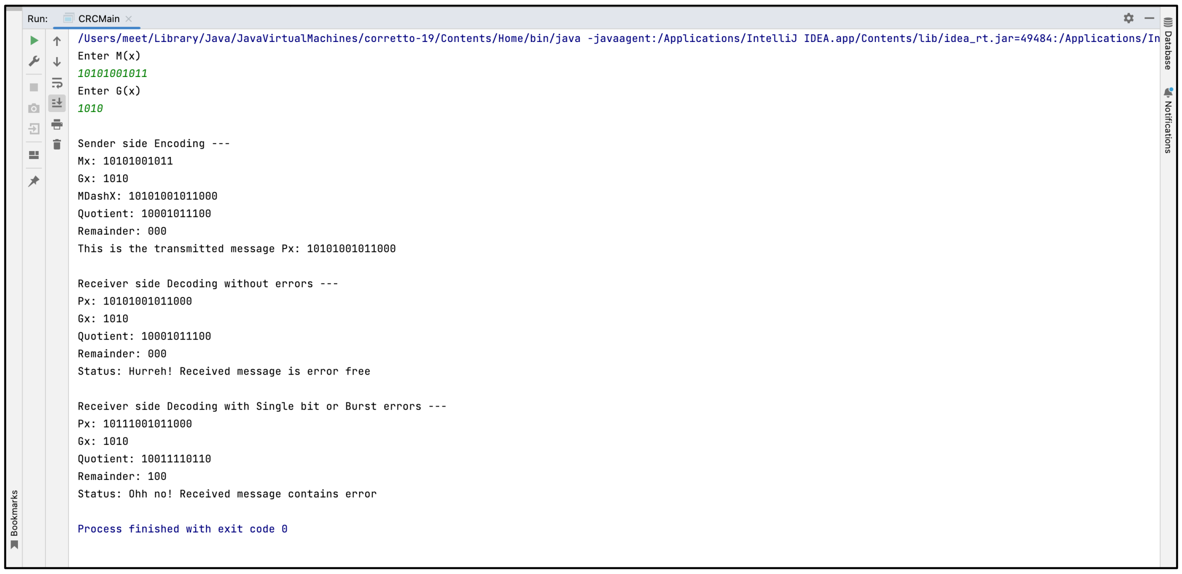
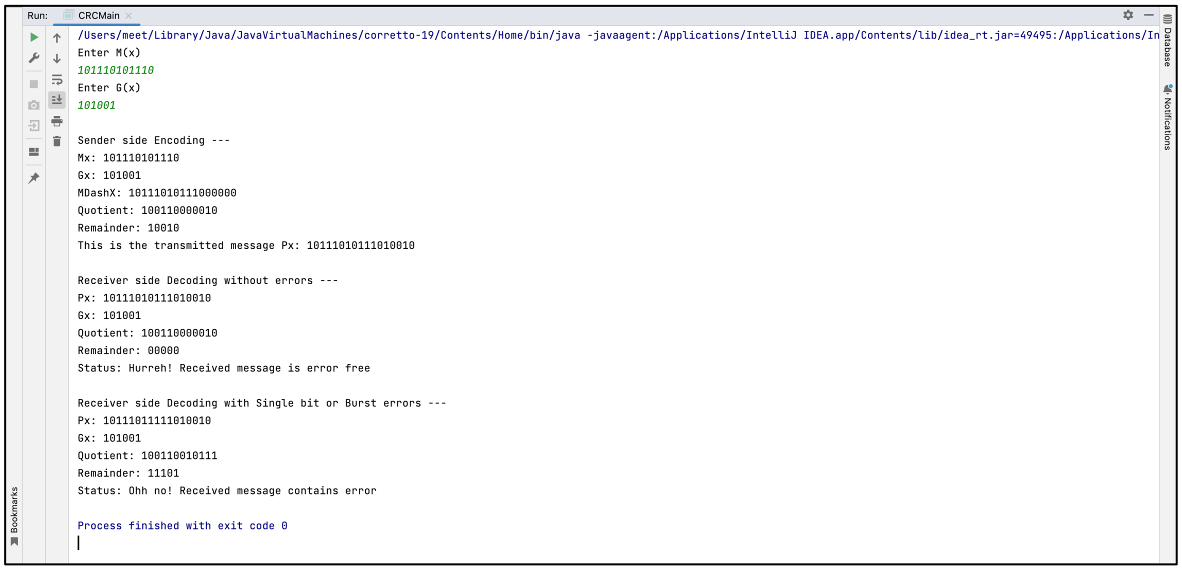

```
/Users/meet/Library/Java/JavaVirtualMachines/corretto-19/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=49484:/Applications/In
Enter M(x)
10101001011
Enter G(x)
1010

Sender side Encoding ---
Mx: 10101001011
Gx: 1010
MDashX: 10101001011000
Quotient: 10001011100
Remainder: 000
This is the transmitted message Px: 10101001011000

Receiver side Decoding without errors ---
Px: 10101001011000
Gx: 1010
Quotient: 10001011100
Remainder: 000
Status: Hurreh! Received message is error free

Receiver side Decoding with Single bit or Burst errors ---
Px: 10111001011000
Gx: 1010
Quotient: 10011110110
Remainder: 100
Status: Ohh no! Received message contains error

Process finished with exit code 0
```

Figure 2: Output of CRC Error Check for input string M(x) = 10101001011 and G(x) = 1010

Sample Run for CRC Error Check – 3


```
/Users/meet/Library/Java/JavaVirtualMachines/corretto-19/Contents/Home/bin/java -javaagent:/Applications/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=49495:/Applications/In
Enter M(x)
101110101110
Enter G(x)
101001

Sender side Encoding ---
Mx: 101110101110
Gx: 101001
MDashX: 10111010111000000
Quotient: 100110000010
Remainder: 10010
This is the transmitted message Px: 10111010111010010

Receiver side Decoding without errors ---
Px: 10111010111010010
Gx: 101001
Quotient: 100110000010
Remainder: 00000
Status: Hurreh! Received message is error free

Receiver side Decoding with Single bit or Burst errors ---
Px: 10111011111010010
Gx: 101001
Quotient: 100110010111
Remainder: 11101
Status: Ohh no! Received message contains error

Process finished with exit code 0
```

Source Code

The source code for **CRC error check** is saved in package “**question_1_crc**”.

It contains three JAVA files which are listed below:

- **CRCComputation** – This java file contains the main logic. This is the java class which perform CRC Error check
- **CRCSender –** This file contains sender’s structure
- **CRCReciver –** This file contains receiver’s structure
- **CRCMain –** This is the file that tests CRCComputation

Programming Exercise 2 <Bridge Processing Simulation>

In this exercise, you will write a simple program to simulate the bridge-processing flowchart for a bridge that we discussed in the lectures. You are given a text file (BridgeFDB.txt) that contains the forwarding database of a bridge with four ports.

Source Code

The source code for **Bridge Processing Simulation** is saved in package "**question_2_bridge_processing**".

It contains three JAVA files which are listed below:

- **BridgeProcessingSimulation** – This java file is responsible for the performing the main logic of reading the RandomFrames.txt file and then updating the BridgeFDB.txt file and finally the generation of BridgeOutput.txt file.

- **BridgeProcessingSimulationMain**– This Java file which is responsible for testing the main logic.

This source code package also contains following files

A) Text file containing the output (BridgeOutput.txt)
B) BridgeFDB.txt file (updated)