

Program Report

- Programmer: McKinnley Workman
- Language: Java
- Description:
 - The purpose of this program is to either perform or verify a CRC checksum for a given file to detect any modifications made to the raw data.
 - The functional components can be seen clearly in the javadocs (javadoc *.java) but basically the main working component is the function that calculates the crc for a given binary input. Otherwise there functions that help parse inputs, convert things between hex and binary, xor binary, print, etc but these all just allow the crc to be calculated efficiently.
- Erroneous Behaviour: none. It's simply perfect.
- To Compile:
 - `>>javac *.java`
- To Run:
 - `>>java crcfile <mode> <filename>`
 - Test example1: `>> java crcfile c inputTestFile0.txt`
 - Test example2 (a verify that will pass): `>> java crcfile v inputTestFile1.txt`
 - Test example3 (a verify that will fail): `>> java crcfile v inputTestFile2.txt`
 - Test example4 (a nonHex input file): `>> java crcfile v inputTestFile3.txt`
- Example output
 - `>> java crcfile c inputTestFile0.txt`

The input file (hex):

4ab124ab124ab124ab124ab124ab12

The input file (bin):

1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010

The polynomial that was used (binary bit string):

0001 0000 1001 1000 1101

We will append 16 zero's at the end of the binary input

The binary string answer at each XOR step of CRC calculation:

1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0010 1111 1101 0100 1100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0000 1110 1110 0101 0110 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011

The computed CRC for this file is

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0011 1011 1101 0100(bin) = 3bd40000000(hex)

- THIS VERIFY SHOULD PASS>> java crcfile v inputTestFile1.txt

The input file (hex):

AB124AB124AB124AB124AB124AB124AB124AB123bd4

The input file (bin):

1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010

The polynomial that was used (binary bit string):

0001 0000 1001 1000 1101

We will append 16 zero's at the end of the binary input

The binary string answer at each XOR step of CRC calculation:

1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0010 1111 1101 0100 1100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0000 1110 1110 0101 0110 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0000 0110 1010 1001 0000 0010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0000 0010 1000 1111 0011 0110 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0000 0000 1001 1100 0010 1100 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0000 0000 0001 1000 1110 1010 0011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0000 0000 0000 1000 0111 0010 1110 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0001 1011 1010 1001 0110 0000 0000 0000

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 1011 0011 0001 1011 0000 0000 0000

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0011 0111 1101 1101 1000 0000 0000

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0001 0110 1110 1100 0010 0000 0000

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0110 0111 0100 1111 0000 0000

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0010 0101 0010 1100 0100 0000

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0100 0001 1101 1110 0000

The computed CRC for this file is

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0011 1011 1101 0100(bin) =

3bd4000000(hex)

The checksum calculated matches the one found!

- THIS VERIFY SHOULD FAIL>>>> java crcfile v inputTestFile2.txt

The input file (hex):

AB124AB124AB124AB124AB124AB124AB124AB12AE12

The input file (bin):

1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010

The polynomial that was used (binary bit string):

0001 0000 1001 1000 1101

We will append 16 zero's at the end of the binary input

The binary string answer at each XOR step of CRC calculation:

1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010
1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011 0001 0010 0100 1010 1011
0001 0010 0100 1010 1011 0001 0010 0000 0000 0000 0000

0000 0000 0000 0000 0000 0010 0101 0010 1100 0100 0000

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0100 0001 1101 1110 0000

The computed CRC for this file is

0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000 0000 0000 0000 0011 1011 1101 0100(bin) =
3bd40000000(hex)

The checksum calculated does not match the one found!

○ >>Test example4 (a nonHex input file): >> java crcfile v inputTestFile3.txt
Input file is not hex

- This program is entirely my own work.
- Extras:
 - create java docs with the following command to get a quick read easy view of functionality: >> javadoc *.java
 - cat README provided for quick functional descriptions