ITCS323 Computer Data Communication

Programming Assignment: Error code

By

Mr. Waris Damkham 6388014

Mr. Dhammawat Siribunchawan 6388055

Submitted to

Asst. Prof. Boonsit Yimwadsana

Asst. Prof. Thitinan Tanidham

Asst. Prof. Vasaka Visoottiviseth

Faculty of Information and Communication Technology

Mahidol University

2021

Parity Bit

How to run

How to run "Paritybit.java" The program will randomly input data like dataword, word_size, parity_type, and array_size. As you see in the main function, in lines 449–542, they print out the test case for the parity generator and the test case for the parity checker.

Testing Results:

```
workspaceStorage\1cf4c5b932c7065ecd00cfb22100d0e6\redhat.
java\jdt_ws\Parity bits_f69459d3\bin' 'ParityGenerator'
Test case for parity generator
=====================================================
Test case 1
Parity type is one-dimensional-even
The dataword sent 00111110100111O
Actual array of dataword is [0, 0, 1, 1, 1, 1, 1, 0, 1, 0
, 0, 1, 1, 1, 0]
Received [0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1]
=====================================================
=====================================================
Test case 2
Parity type is one-dimensional-even
The dataword sent 00111110100111011010100
Actual array of dataword is [0, 0, 1, 1, 1, 1, 1, 0, 1, 0
, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0]
Received [0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1,
 0, 0, 1, 1, 1, 1, 1, 0, 1]
=====================================================
=====================================================
Test case 3
Parity type is one-dimensional-odd
The dataword sent 00111110100111011010100111101001
Actual array of dataword is [0, 0, 1, 1, 1, 1, 1, 0, 1, 0
, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1
, 1, 0, 1]
Received [0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1,
 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1]
=====================================================
=====================================================
Test case 4
Parity type is one-dimensional-odd
The dataword sent 00111110100111011010100111101001010000
110
Actual array of dataword is [0, 0, 1, 1, 1, 1, 1, 0, 1, 0
, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1
, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0]
Received [0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1,
 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1,
 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0]
=====================================================
=====================================================
Test case 5
Parity type is two-dimensional-even
The dataword sent 00111110100 1110110101 0011110100 10100
00111 010000011
Actual array of dataword is [1, 1, 0, 0, 0, 0, 1, 0, 1, 1
]
Actual array of dataword is [1, 1, 1, 0, 0, 1, 1, 0, 1, 1
]
=====================================================
Test case for Parity checker
=====================================================
Test case 1
Parity type is one-dimensional-even
The dataword sent 111011000000111
PASS
=====================================================
=====================================================
Test case 2
Parity type is one-dimensional-even
The dataword sent 1110110000001110011111
PASS
=====================================================
Test case 6
Parity type is two-dimensional-even
The dataword sent 1110110 0000011 1001111
FAIL
=====================================================
=====================================================
Test case 7
Parity type is two-dimensional-even
The dataword sent 1110110000001 1100111111000 0110011111010 0000100011000
FAIL
=====================================================
=====================================================
Test case 8
Parity type is two-dimensional-even
The dataword sent 111011000 000111001 111110000 110011111 010000010 001100001 111110110 010001000
 010001100 100001110 110000101
FAIL
=====================================================
=====================================================
Test case 9
Parity type is two-dimensional-even
The dataword sent 111011 000000 111001
PASS
=====================================================
=====================================================
Test case 10
Parity type is two-dimensional-even
The dataword sent 11101 10000 00111 00111 11100 00110 01111 10100
FAIL
=====================================================
PS C:\Users\Jigsaw\Desktop\Comdata\Parity bits>
```

CRC

How to run

How to run "CRC.c" The program will receive input data from users, including CRC-type, Dataword, and Generator. To generate a remainder.

Testing Results:

```
Enter CRC-type
4
Enter Word size
6
Enter Dataword
100100
Enter Generator
1011
 Remainder is : 11
```

```
Enter CRC-type
4
Enter Word size
6
Enter Dataword
111001
Enter Generator
1011
 Remainder is : 1
```

```
Enter CRC-type
8
Enter Word size
8
Enter Dataword
1110110
Enter Generator
100100
 Remainder is : 1100000
```

```
Enter CRC-type
16
Enter Word size
8
Enter Dataword
10101010
Enter Generator
1111000000110011
 Remainder is : 111011011100000
```

```
Enter CRC-type
16
Enter Word size
8
Enter Dataword
1011001010
Enter Generator
1010101010101010
 Remainder is : 1111100001100
```

```
Enter CRC-type
4
Enter Word size
6
Enter Dataword
101010
Enter Generator
1111
 Remainder is : 1
```

```
Enter CRC-type
8
Enter Word size
6
Enter Dataword
101010
Enter Generator
10101010
 Remainder is : 1
```

```
Enter CRC-type
4
Enter Word size
6
Enter Dataword
100100
Enter Generator
1010
 Remainder is : 11
```

```
Enter CRC-type
16
Enter Word size
8
Enter Dataword
1011001010
Enter Generator
1001001000011
 Remainder is : 101011100111000
```

```
Enter CRC-type
8
Enter Word size
6
Enter Dataword
100100
Enter Generator
10101010
 Remainder is : 110010
```

Checksum

How to run

How to run "Checksum.java" The program will receive input data from users, including Num_block, Word_size, and Dataword. To generate a codeword based on the checksum and using the same input to test the validity of the codeword, pass or fail.

Testing Results:

```
********START Generator********
Enter Num block
5
Enter Word Size
4
Enter Dataword number 1:
0111
Enter Dataword number 2:
1011
Enter Dataword number 3:
1100
Enter Dataword number 4:
0000
Enter Dataword number 5:
0110
Codeword: 111 1011 1100 0 110 1001
********END Generator********
********START Checker********
Codeword: 0000
Validity of codeword: PASS
********END Checker********
```

```
********START Generator********
Enter Num block
3
Enter Word Size
8
Enter Dataword number 1:
11110000
Enter Dataword number 2:
01001100
Enter Dataword number 3:
10101010
Codeword: 11110000 1001100 10101010 00011000
********END Generator********
********START Checker********
Codeword: 00000000
Validity of codeword: PASS
********END Checker********
```

```
********START Generator********
Enter Num block
2
Enter Word Size
8
Enter Dataword number 1:
10101001
Enter Dataword number 2:
00111001
Codeword: 10101001 111001 00011101
********END Generator********
********START Checker********
Codeword: 00000000
Validity of codeword: PASS
********END Checker********
```

```
********START Generator********
Enter Num block
3
Enter Word Size
8
Enter Dataword number 1:
11011111
Enter Dataword number 2:
01001010
Enter Dataword number 3:
10110101
Codeword: 11011111 1001010 10110101 00100000
********END Generator********
********START Checker********
Codeword: 00000000
Validity of codeword: PASS
********END Checker********
```

```
********START Generator********
Enter Num block
3
Enter Word Size
8
Enter Dataword number 1:
11011011
Enter Dataword number 2:
01001000
Enter Dataword number 3:
01010101
Codeword: 11011011 1001000 1010101 10000110
********END Generator********
********START Checker********
Codeword: 00000000
Validity of codeword: PASS
********END Checker********
```

```
********START Generator********
Enter Num block
4
Enter Word Size
7
Enter Dataword number 1:
1100101
Enter Dataword number 2:
1010101
Enter Dataword number 3:
1100110
Enter Dataword number 4:
1100101
Codeword: 1100101 1010101 1100110 1100101 1110111
********END Generator********
********START Checker********
Codeword: 0000000
Validity of codeword: PASS
********END Checker********
```

```
********START Generator********
Enter Num block
3
Enter Word Size
8
Enter Dataword number 1:
11010111
Enter Dataword number 2:
01001101
Enter Dataword number 3:
01001010
Codeword: 11010111 1001101 1001010 10010000
********END Generator********
********START Checker********
Codeword: 00000000
Validity of codeword: PASS
********END Checker********
```

```
********START Generator********
Enter Num block
3
Enter Word Size
7
Enter Dataword number 1:
1101110
Enter Dataword number 2:
0101011
Enter Dataword number 3:
1101011
Codeword: 1101110 101011 1101011 1111001
********END Generator********
********START Checker********
Codeword: 0000000
Validity of codeword: PASS
********END Checker********
```

```
********START Generator********
Enter Num block
2
Enter Word Size
5
Enter Dataword number 1:
11011
Enter Dataword number 2:
01001
Codeword: 11011 1001 11010
********END Generator********
********START Checker********
Codeword: 00000
Validity of codeword: PASS
********END Checker********
```

```
********START Generator********
Enter Num block
3
Enter Word Size
10
Enter Dataword number 1:
0101100101
Enter Dataword number 2:
0101011111
Enter Dataword number 3:
0010101011
Codeword: 101100101 101011111 10101011 0010010000
********END Generator********
********START Checker********
Codeword: 0000000000
Validity of codeword: PASS
********END Checker********
```

Hamming Code

How to run

How to run " HammingCode.java" The program will randomly input data like dataword, and codeword. As you see in the main function, in lines 369–454, they print out the test case for the hamming generator and the test case for the hamming checker.

Testing Results: