Mahidol University
Faculty of Information and Communication Technology

Error Control Programming Project

By

Miss Suphavadee Cheng
ID 6488120, Section 1

A Report Submitted in Partial Fulfillment of
the Requirements for

ITCS323 Computer Data Communication

Faculty of Information and Communication Technology
Mahidol University
2022

<u>How to run your programs</u>

✓ Programming Language: Java
  o Open Eclipse
  o Create Project: File > New > Java Project
  o Place File: Put four files from folder 'JavaCode-Sec1-6488120' in the 'src' of this project
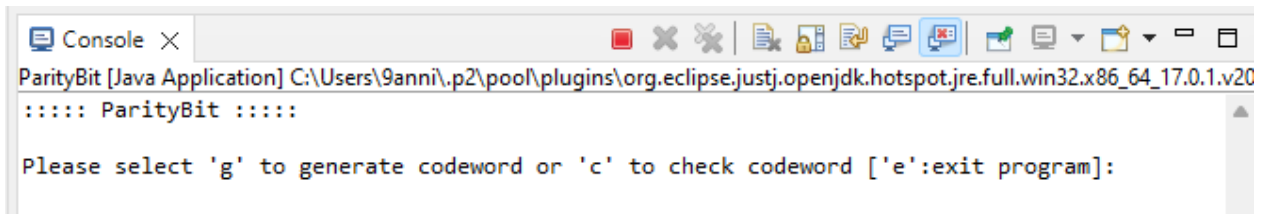
Parity bit class

1. Users have three options to choose from by type the character.

   'g' : generate codeword
   'c' : check codeword
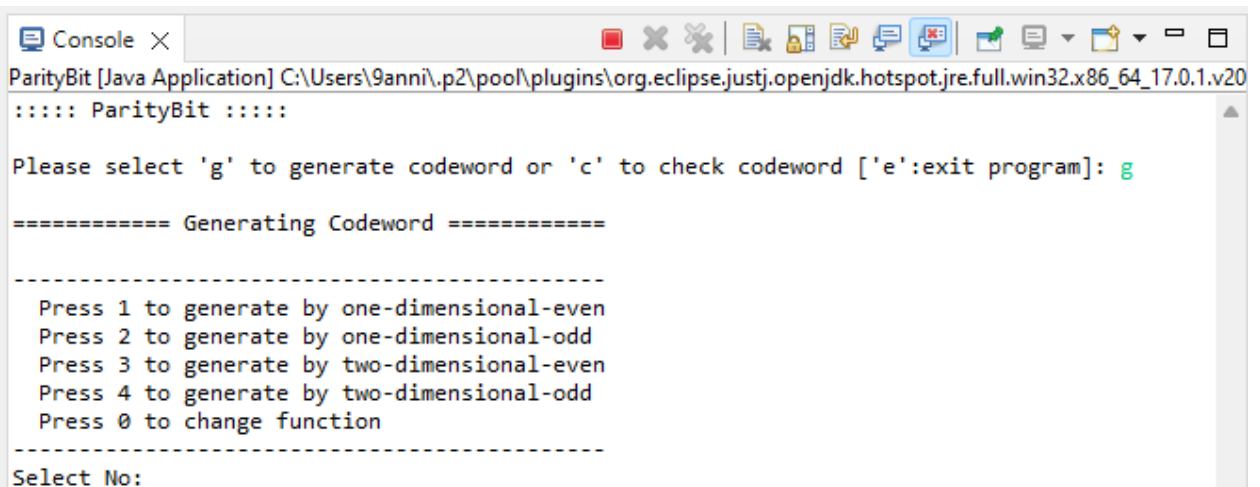   'e' : exit program (This will stop the program)

2. Let's go to each of the characters.

   2.1 Type 'g'

Users can generate the type of parity by typing the number.

Press 1: Generate codeword by using one-dimensional-even
Press 2: Generate codeword by using one-dimensional-odd
Press 3: Generate codeword by using two-dimensional-even
Press 4: Generate codeword by using two-dimensional-odd
Press 0: Change to another function (this will back to select 'g', 'c', or 'e' again)

DATAWORD: 11000 01100 00100 00011 10110

Press 1, one-dimensional-even (Generate)

```
Console ×
ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v202
-------------------------------------------------
Select No: 1

Type: One-Dimensional-Even
Insert Dataword: 11000 01100 00100 00011 10110
Codeword: 110000 011000 001001 000110 101101
```

Press 2, one-dimensional-odd (Generate)

```
Console ×
ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
-------------------------------------------------
Select No: 2

Type: One-Dimensional-Odd
Insert Dataword: 11000 01100 00100 00011 10110
Codeword: 110001 011001 001000 000111 101100
```

Press 3, two-dimensional-even (Generate)

```
Console ×
ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v202
-------------------------------------------------
Select No: 3

Type: Two-Dimensional-Even
Insert Dataword: 11000 01100 00100 00011 10110
Codeword: 110000 011000 001001 000110 101101 001010
```

Press 4, two-dimensional-odd (Generate)

```
Console ×
ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v202
-------------------------------------------------
Select No: 4

Type: Two-Dimensional-Odd
Insert Dataword: 11000 01100 00100 00011 10110
Codeword: 110001 011001 001000 000111 101100 110100
```

Press 0, change function

```
Console ×
ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
-------------------------------------------------
Select No: 0

Change Function
```

------------------ *error* ------------------

Case: Bits in dataword exceed the MAX & Bits in each dataword are not the same

Console ✕

ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20

Select No: 1

Type: One-Dimensional-Even
Insert Dataword: 11 100000 10000000000000000
XXX Not same bits & Too many bits XXX
Codeword: 0

Case: Bits in dataword exceed the MAX

Console ✕

ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20

Select No: 1

Type: One-Dimensional-Even
Insert Dataword: 100011111 100000011
XXX Too many bits XXX
Codeword: 0

Case: Bits in each dataword are not the same

Console ✕

ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
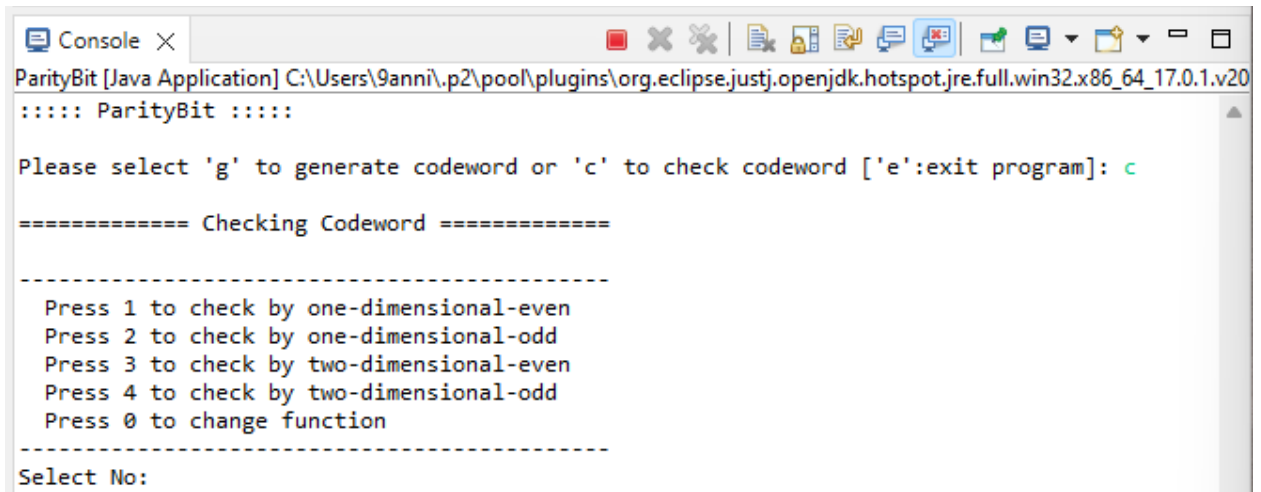
Select No: 1

Type: One-Dimensional-Even
Insert Dataword: 11000 011 100000 11
XXX Not same bits XXX
Codeword: 0

2.2 Type 'c'

Users can check the type of parity by typing the number.

Press 1: Check codeword by using one-dimensional-even
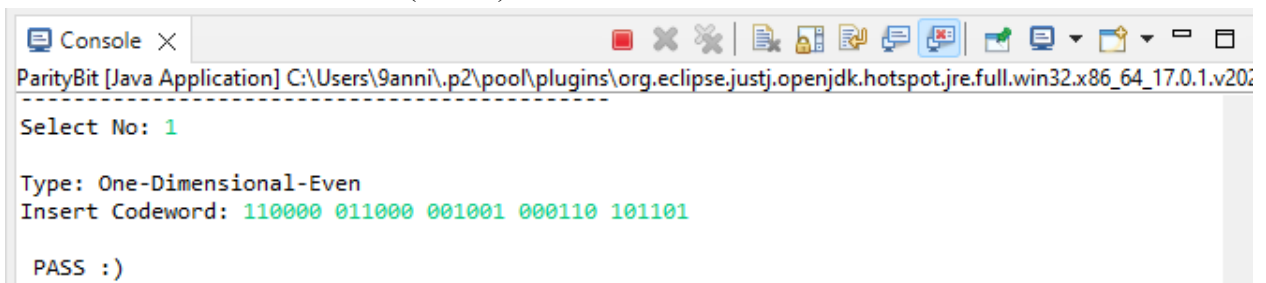Press 2: Check codeword by using one-dimensional-odd
Press 3: Check codeword by using two-dimensional-even
Press 4: Check codeword by using two-dimensional-odd
Press 0: Change to another function (this will back to select 'g', 'c', or 'e' again)

CODEWORD: 110000 011000 001001 000110 101101
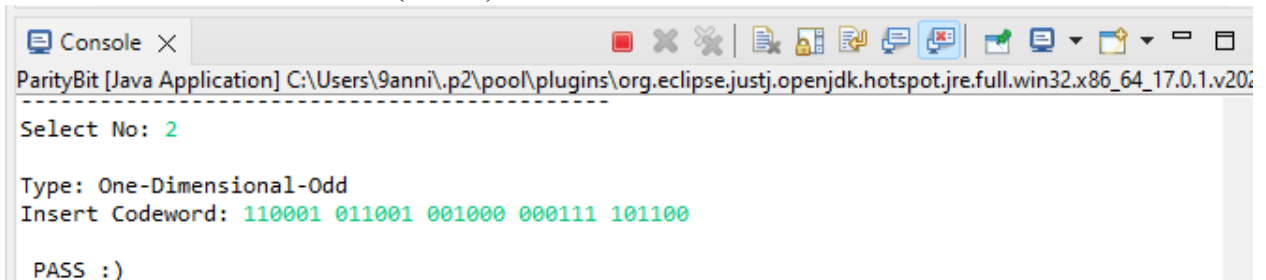Press 1, one-dimensional-even (Check)

CODEWORD: 110001 011001 001000 000111 101100
Press 2, one-dimensional-odd (Check)

CODEWORD: 110000 011000 001001 000110 101101 001010

Press 3, two-dimensional-even (Check)


```
Console X

ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v202
------------------------------------------------
Select No: 3

Type: Two-Dimensional-Even
Insert Codeword: 110000 011000 001001 000110 101101 001010

 PASS :)
```

CODEWORD: 110001 011001 001000 000111 101100 110100

Press 4, two-dimensional-odd (Check)


```
Console X

ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v202
------------------------------------------------
Select No: 4

Type: Two-Dimensional-Odd
Insert Codeword: 110001 011001 001000 000111 101100 110100

 PASS :)
```

Press 0, change function


```
Console X

ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
------------------------------------------------
Select No: 0

Change Function
```

2.3 Type 'e'


```
Console X

<terminated> ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86

Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]: e
```

CRC

1. Users have three options to choose from by type the character.

   'g' : generate codeword
   'c' : check codeword
   'e' : exit program (This will stop the program)

   ```
   Console ×
   CRC [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v202111
   ::::: CRC :::::

   Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]:
   ```

2. Let's go to each of the characters.

   2.1 Type 'g'

   ```
   Console ×
   CRC [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v202111
   ::::: CRC :::::

   Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]: g

   ============ Generating Codeword ============

   ---------------------------------------------
     Press 1 to generate by CRC-8
     Press 2 to generate by CRC-16
     Press 3 to generate by CRC-32
     Press 0 to change function
   ---------------------------------------------
   Select No:
   ```

   Users can generate from this following type of CRC by typing the number.

   Press 1: Generate codeword by using CRC-8
   Press 2: Generate codeword by using CRC-16
   Press 3: Generate codeword by using CRC-32
   Press 0: Change to another function (this will back to select 'g', 'c', or 'e' again)

DATAWORD: 1101

Press1, CRC-8 (Generate)

```
Console ×                                    ■ ✖ ✖ | ▤ ▥ ▣ ▱ ▣ | ▥ ▢ ▾ ▢ ▾ ▭ ▢
CRC [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v2021111
-----------------------------------------------
Select No: 1

Type: CRC-8
Insert Dataword: 1101
Codeword: 110100000010
```

Press2, CRC-16 (Generate)
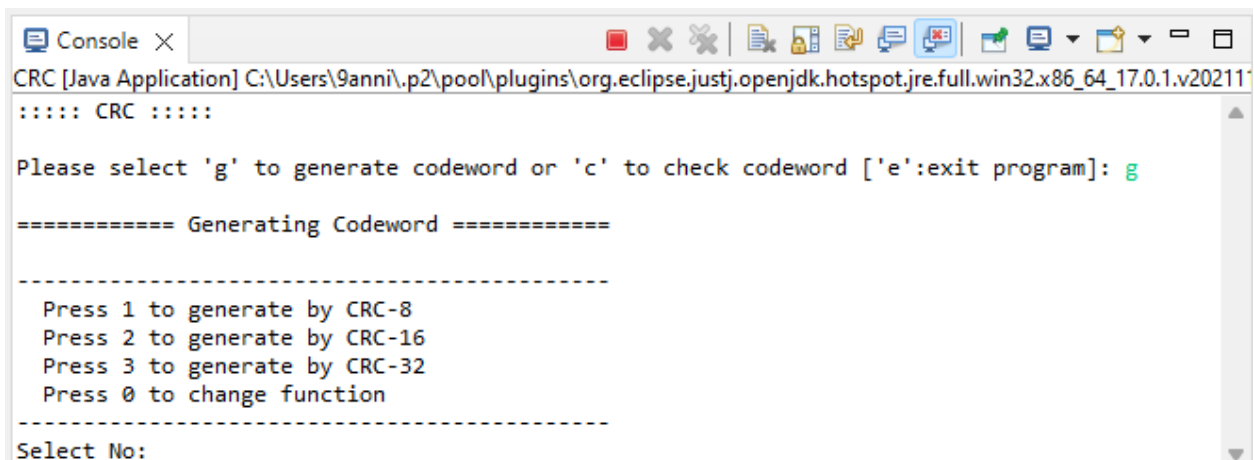
```
Console ×                                    ■ ✖ ✖ | ▤ ▥ ▣ ▱ ▣ | ▥ ▢ ▾ ▢ ▾ ▭ ▢
CRC [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v2021111
-----------------------------------------------
Select No: 2

Type: CRC-16
Insert Dataword: 1101
Codeword: 11011000000000101101
```

Press3, CRC-32 (Generate)

```
Console ×                                    ■ ✖ ✖ | ▤ ▥ ▣ ▱ ▣ | ▥ ▢ ▾ ▢ ▾ ▭ ▢
CRC [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v2021111
-----------------------------------------------
Select No: 3

Type: CRC-32
Insert Dataword: 1101
Codeword: 1101001100011100110110000110110011
```

Press 0, change function

```
Console ×                                    ■ ✖ ✖ | ▤ ▥ ▣ ▱ ▣ | ▥ ▢ ▾ ▢ ▾ ▭ ▢
ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
-----------------------------------------------
Select No: 0

Change Function
```

2.2 Type 'c'

Users can generate from this following type of CRC by typing the number.

Press 1: Check codeword by using CRC-8
Press 2: Check codeword by using CRC-16
Press 3: Check codeword by using CRC-32
Press 0: Change to another function (this will back to select 'g', 'c', or 'e' again)

CODEWORD:110100000010
Press1, CRC-8 (Check)

CODEWORD: 110110000000101101
Press 2, CRC-16 (Check)

CODEWORD: 110001 011001 001000 000111 101100 110100

Press 3, CRC-32 (Check)

```
Console  X
CRC [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v2021111
---------------------------------------------
Select No: 3

Type: CRC-32
Insert Codeword: 11010011000111001101100011011010011
Syndrome: 00000000000000000000000000000000

 PASS :)
```

Press 0, change function

```
Console  X
ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
---------------------------------------------
Select No: 0

Change Function
```

2.3 Type 'e'

```
Console  X
<terminated> ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86

Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]: e
```
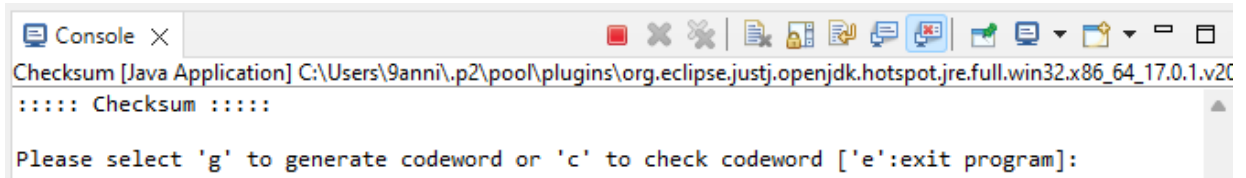
Check Sum

1. Users have three options to choose from by type the character.

   'g' : generate codeword
   'c' : check codeword
   'e' : exit program (This will stop the program)

   ```
   Console ×
   Checksum [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
   ::::: Checksum :::::

   Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]:
   ```

2. Let's go to each of the characters.

   2.1 Type 'g'

   ```
   Console ×
   Checksum [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
   ::::: Checksum :::::

   Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]: g

   ============ Generating Codeword ============

   --------------------------------------------
     Press 1 to generate codeword
     Press 0 to change function
   --------------------------------------------
   Select No:
   ```

   Users can generate the codeword using the checksum method by typing the number.

   Press 1: Generate codeword by using checksum
   Press 0: Change to another function (this will back to select 'g', 'c', or 'e' again)

   DATAWORD: 1100 0011 1001 0011 0101
   Press1, Generate

   ```
   Console ×
   Checksum [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
   --------------------------------------------
   Select No: 1

   Insert Dataword: 1100 0011 1001 0011 0101
   5 Word

   sum: 0010 0000 (32)
   wrapped sum: 0010
   actual checksum: 1101
   Codeword: 1100 0011 1001 0011 0101 1101
   ```

Press 0, change function

```
-------------------------------------------
Select No: 0

Change Function
```

2.2 Type 'c'

```
::::: Checksum :::::

Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]: c

============= Checking Codeword =============

-------------------------------------------
  Press 1 to check codeword
  Press 0 to change function
-------------------------------------------
Select No:
```

Users can check the codeword using the checksum method by typing the number.

Press 1: Check codeword by using check sum
Press 0: Change to another function (this will back to select 'g', 'c', or 'e' again)

CODEWORD: 1100 0011 1001 0011 0101 1101
Press1, Check

```
-------------------------------------------
Select No: 1

Insert Codeword: 1100 0011 1001 0011 0101 1101
6 Word

add: 0010 1101 (45)
wrapped sum: 1111
calcuated checksum: 0000

 PASS :)
```

Press 0, change function

2.3 Type 'e'

Hamming Code

1. Users have three options to choose from by type the character.

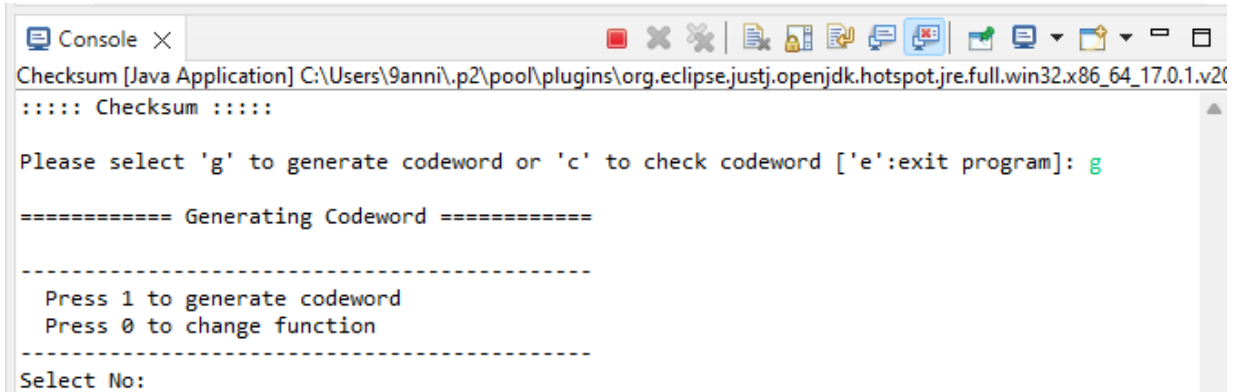   'g' : generate codeword
   'c' : check codeword
   'e' : exit program (This will stop the program)

   ```
   Console ×

   HammingCode [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.
   ::::: Hamming Code :::::

   Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]:
   ```

2. Let's go to each of the characters.

   2.1 Type 'g'

   ```
   Console ×

   HammingCode [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.
   ::::: Hamming Code :::::

   Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]: g

   ============ Generating Codeword ============

   -----------------------------------------------
     Press 1 to generate by Hamming code
     Press 0 to change function
   -----------------------------------------------
   Select No:
   ```
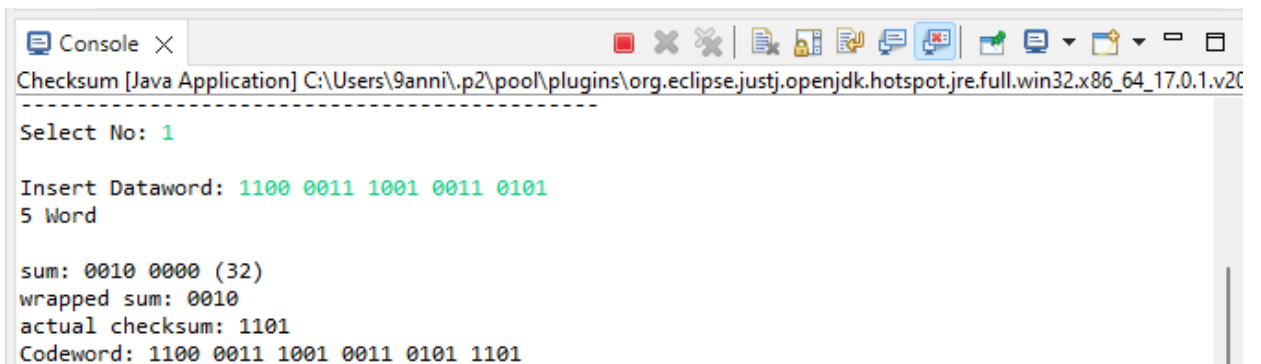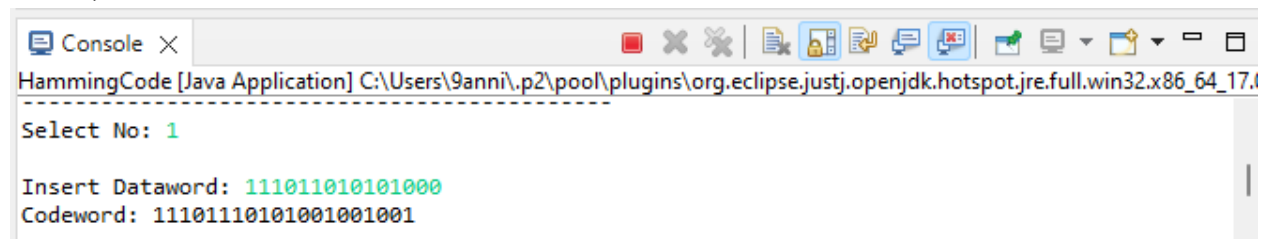
   Users can generate the codeword using the hamming code method by typing the number.

   Press 1: Generate codeword by using hamming code
   Press 0: Change to another function (this will back to select 'g', 'c', or 'e' again)
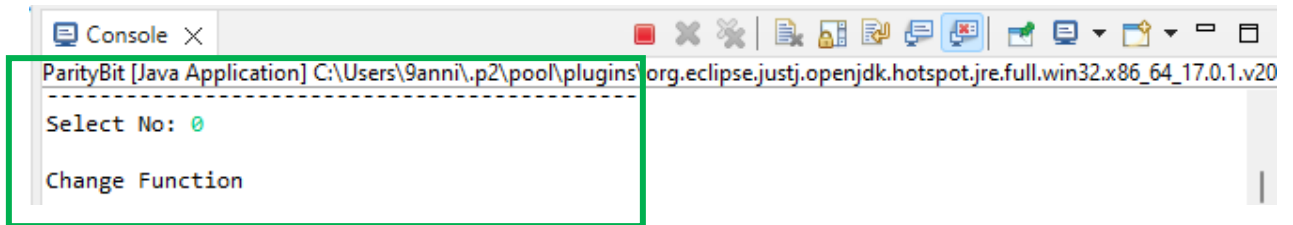
   DATAWORD: 111011010101000
   Press1, Generate
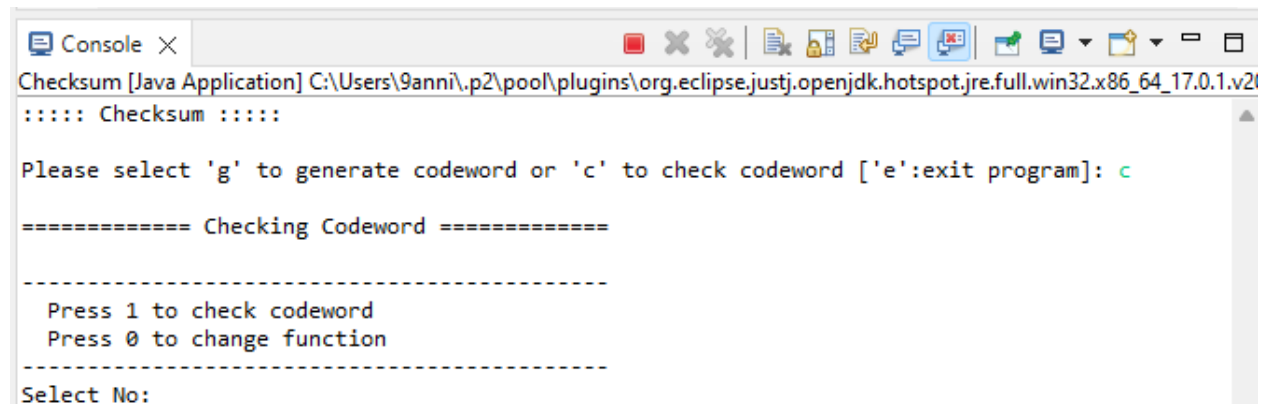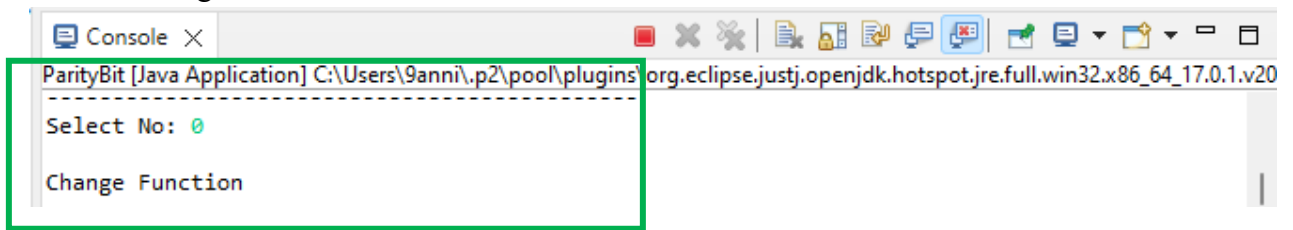
   ```
   Console ×

   HammingCode [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0
   -----------------------------------------------
   Select No: 1

   Insert Dataword: 111011010101000
   Codeword: 11101110101001001001
   ```
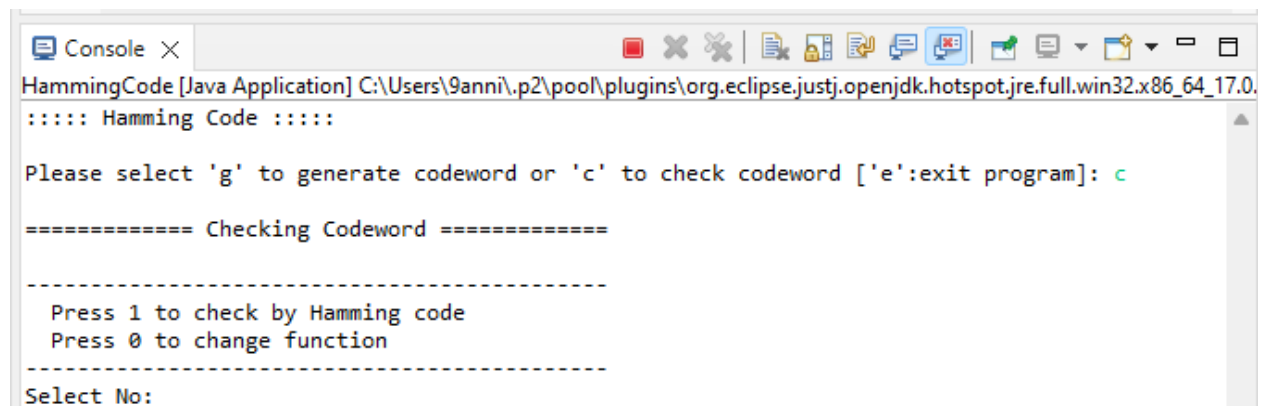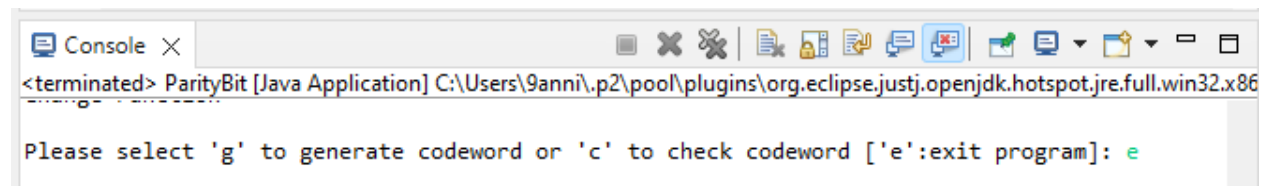
Press 0, change function

```
------------------------------------------
Select No: 0

Change Function
```

2.2 Type 'c'

```
:::::: Hamming Code ::::::

Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]: c

============= Checking Codeword =============

------------------------------------------
  Press 1 to check by Hamming code
  Press 0 to change function
------------------------------------------
Select No:
```
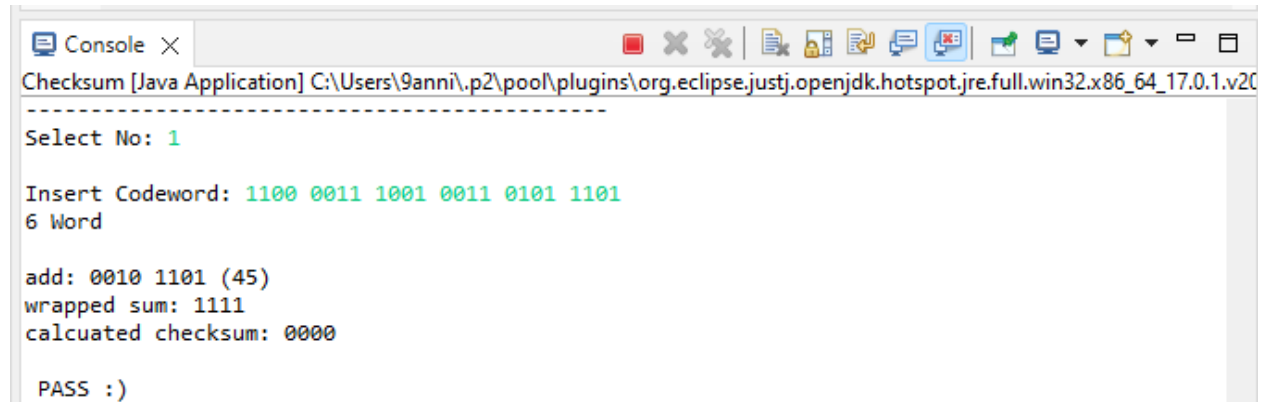
Users can check the codeword using the hamming code method by typing the number.

Press 1: Check codeword by using hamming code
Press 0: Change to another function (this will back to select 'g', 'c', or 'e' again)

CODEWORD: 11101110101001001001
Press1, Check

```
------------------------------------------
Select No: 1

Insert Codeword: 11101110101001001001

 PASS :)
```
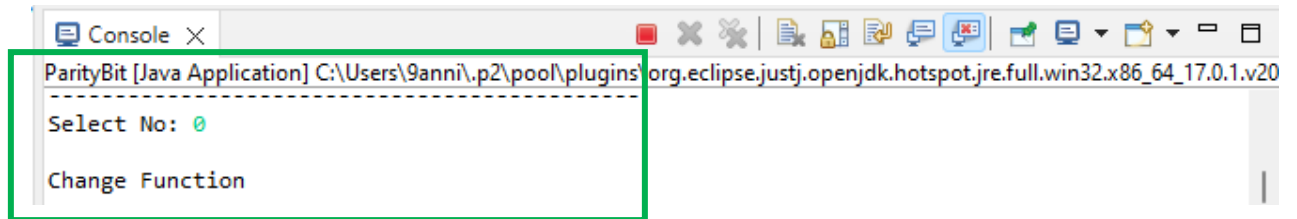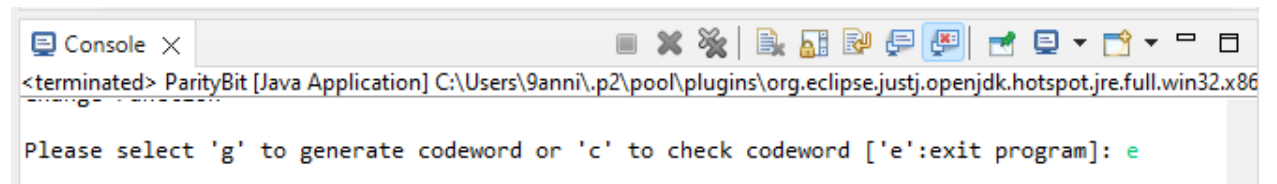
Press 0, change function


```
Console ✕

ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20
------------------------------------------------------------
Select No: 0

Change Function
```

2.3 Type 'e'


```
Console ✕

<terminated> ParityBit [Java Application] C:\Users\9anni\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86

Please select 'g' to generate codeword or 'c' to check codeword ['e':exit program]: e
```

Parity Bit

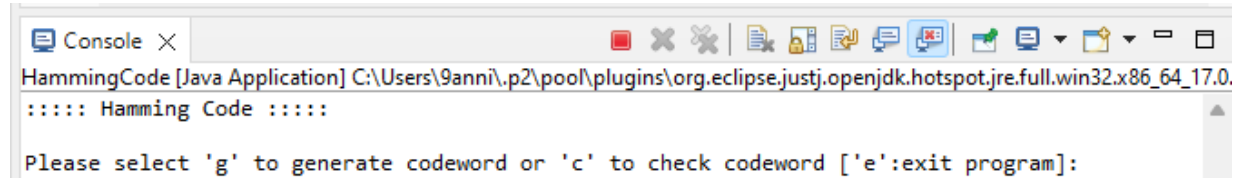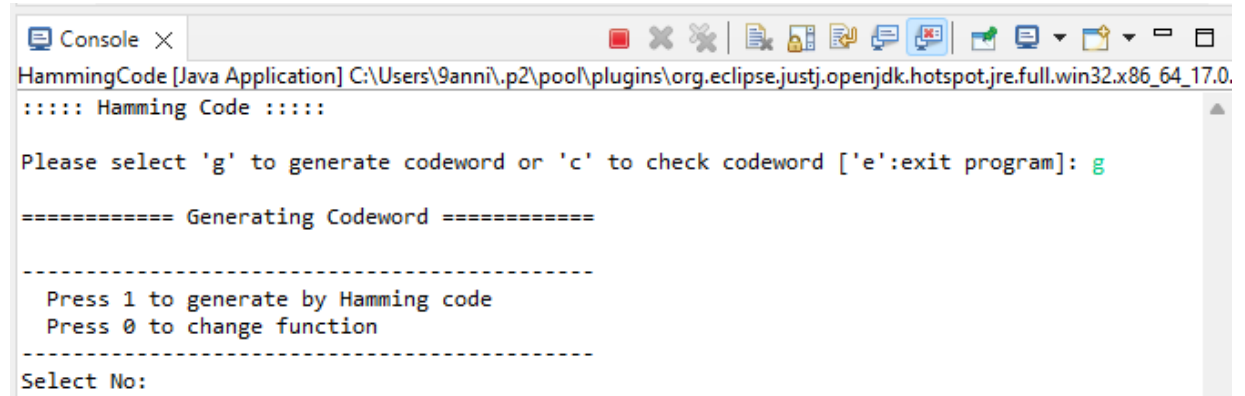|	
	One-Dimensional-Even
1	Generate: Type: One-Dimensional-Even Insert Dataword: 100111 100001 001100 001001 010001 Codeword: 1001110 1000010 0011000 0010010 0100010 Check: Type: One-Dimensional-Even Insert Codeword: 1001110 1000010 0011000 0010010 0100010 PASS :)
2	Generate: Type: One-Dimensional-Even Insert Dataword: 11111 01101 00110 01110 00011 01110 00001 Codeword: 111111 011011 001100 011101 000110 011101 000011 Check: Type: One-Dimensional-Even Insert Codeword: 111111 011011 001100 011101 000110 011101 000011 PASS :)
3	Generate: Type: One-Dimensional-Even Insert Dataword: 011 000 011 100 001 101 Codeword: 0110 0000 0110 1001 0011 1010 Check: Type: One-Dimensional-Even Insert Codeword: 0110 0000 0110 1001 0011 1010 PASS :)
4	Generate: Type: One-Dimensional-Even Insert Dataword: 1101000 0011010 0000010 0001011 0111101 1111111 0001000 0101110 Codeword: 11010001 00110101 00000101 00010111 01111011 11111111 00010001 01011100 Check:

	Type: One-Dimensional-Even Insert Codeword: 11010001 00110101 00000101 00010111 01111011 11111111 00010001 01011100 PASS :)
5	Generate: Type: One-Dimensional-Even Insert Dataword: 00010 01100 00101 11111 10100 01110 01000 01101 00010 01010 00111 Codeword: 000101 011000 001010 111111 101000 011101 010001 011011 000101 010100 001111 Check: Type: One-Dimensional-Even Insert Codeword: 000101 011000 001010 111111 101000 011101 010001 011011 000101 010100 001111 PASS :)
6	Generate: Type: One-Dimensional-Even Insert Dataword: 101011 001111 000001 010010 100010 Codeword: 1010110 0011110 0000011 0100100 1000100 Check: Type: One-Dimensional-Even Insert Codeword: 1010110 0011110 0000011 0100100 1000101 FAIL :(
7	Generate: Type: One-Dimensional-Even Insert Dataword: 00001 10011 00100 01110 01001 10110 11110 Codeword: 000011 100111 001001 011101 010010 101101 111100 Check: Type: One-Dimensional-Even Insert Codeword: 000011 100111 101001 011101 010011 101101 111100 FAIL :(
8	Generate: Type: One-Dimensional-Even Insert Dataword: 010 001 100 000 011 111 110 Codeword: 0101 0011 1001 0000 0110 1111 1100 Check: Type: One-Dimensional-Even Insert Codeword: 1101 0011 1001 0000 0110 1111 1100 FAIL :(
9	Generate:

<table>
<tr><td rowspan="2"></td><td>

```
Type: One-Dimensional-Even
Insert Dataword: 001100 110100 001010 000110 000111 100110 010001
Codeword: 0011000 1101001 0010100 0001100 0001111 1001101 0100010
```

Check:

```
Type: One-Dimensional-Even
Insert Codeword: 0011000 1101001 0010100 0001100 0001101 1001101 0100010

  FAIL :(
```
</td></tr>
</table>

10	Generate:

```
Type: One-Dimensional-Even
Insert Dataword: 11000 00110 01010 11100 00011 11111 00000
Codeword: 110000 001100 010100 111001 000110 111111 000000
```

Check:

```
Type: One-Dimensional-Even
Insert Codeword: 110000 001100 010110 111001 000110 111111 000000

  FAIL :(
```

One-Dimensional-ODD

1	Generate:

```
Type: One-Dimensional-Odd
Insert Dataword: 101100 001110 010000 011100 010000
Codeword: 1011000 0011100 0100000 0111000 0100000
```

Check:

```
Type: One-Dimensional-Odd
Insert Codeword: 1011000 0011100 0100000 0111000 0100000

  PASS :)
```

2	Generate:

```
Type: One-Dimensional-Odd
Insert Dataword: 1100 1010 0001 1011 0000 0100
Codeword: 11001 10101 00010 10110 00001 01000
```

Check:

<table>
<tr><td></td><td>Type: One-Dimensional-Odd
Insert Codeword: 11001 10101 00010 10110 00001 01000

  PASS :)</td></tr>
<tr><td>3</td><td>Generate:

Type: One-Dimensional-Odd
Insert Dataword: 001 100 010 000 011 111
Codeword: 0010 1000 0100 0001 0111 1110

Check:

Type: One-Dimensional-Odd
Insert Codeword: 0010 1000 0100 0001 0111 1110

  PASS :)</td></tr>
<tr><td>4</td><td>Generate:

Type: One-Dimensional-Odd
Insert Dataword: 00001 00010 00011 00100 00111
Codeword: 000010 000100 000111 001000 001110

Check:

Type: One-Dimensional-Odd
Insert Codeword: 000010 000100 000111 001000 001110

  PASS :)</td></tr>
<tr><td>5</td><td>Generate:

Type: One-Dimensional-Odd
Insert Dataword: 000001 111110 001000 011001
Codeword: 0000010 1111100 0010000 0110010

Check:

Type: One-Dimensional-Odd
Insert Codeword: 0000010 1111100 0010000 0110010

  PASS :)</td></tr>
<tr><td>6</td><td>Generate:

Type: One-Dimensional-Odd
Insert Dataword: 000100 010011 101000 001011 101101 111111
Codeword: 0001000 0100110 1010001 0010110 1011011 1111111

Check:

Type: One-Dimensional-Odd
Insert Codeword: 0001000 0100110 1010001 0010110 1011011 1110111

  FAIL :(</td></tr>
<tr><td>7</td><td>Generate:</td></tr>
</table>

	Type: One-Dimensional-Odd Insert Dataword: 1011 1000 0011 1101 0001 Codeword: 10110 10000 00111 11010 00010 Check: Type: One-Dimensional-Odd Insert Codeword: 00110 10000 00111 11010 00010 FAIL :(
8	Generate: Type: One-Dimensional-Odd Insert Dataword: 0101101 1111111 0101010 1100101 1001000 0001101 Codeword: 01011011 11111110 01010100 11001011 10010001 00011010 Check: Type: One-Dimensional-Odd Insert Codeword: 01011011 11111110 01010100 11001011 10010000 00011010 FAIL :(
9	Generate: Type: One-Dimensional-Odd Insert Dataword: 010101 000101 111010 000101 101010 000101 Codeword: 0101010 0001011 1110101 0001011 1010100 0001011 Check: Type: One-Dimensional-Odd Insert Codeword: 0101010 0001011 1110101 0001111 1010100 0001011 FAIL :(
10	Generate: Type: One-Dimensional-Odd Insert Dataword: 0001 1011 1000 0010 0111 1001 0001 0011 1110 0101 Codeword: 00010 10110 10000 00100 01110 10011 00010 00111 11100 01011 Check: Type: One-Dimensional-Odd Insert Codeword: 10010 10110 10000 00100 01110 10011 00010 00111 11100 01011 FAIL :(

Two-Dimensional-Even
1   Generate:

	Type: Two-Dimensional-Even Insert Dataword: 10001 00001 11101 01100 01110 01010 Codeword: 100010 000011 111010 011000 011101 010100 001010 Check: Type: Two-Dimensional-Even Insert Codeword: 100010 000011 111010 011000 011101 010100 001010 PASS :)
2	Generate: Type: Two-Dimensional-Even Insert Dataword: 10111 01111 01101 00011 01011 00001 10001 Codeword: 101110 011110 011011 000110 010111 000011 100010 011011 Check: Type: Two-Dimensional-Even Insert Codeword: 101110 011110 011011 000110 010111 000011 100010 011011 PASS :)
3	Generate: Type: Two-Dimensional-Even Insert Dataword: 001 101 001 111 100 110 011 Codeword: 0011 1010 0011 1111 1001 1100 0110 0110 Check: Type: Two-Dimensional-Even Insert Codeword: 0011 1010 0011 1111 1001 1100 0110 0110 PASS :)
4	Generate: Type: Two-Dimensional-Even Insert Dataword: 0001 1110 1110 0110 1000 0011 1011 Codeword: 00011 11101 11101 01100 10001 00110 10111 01111 Check: Type: Two-Dimensional-Even Insert Codeword: 00011 11101 11101 01100 10001 00110 10111 01111 PASS :)
5	Generate: Type: Two-Dimensional-Even Insert Dataword: 0011101 0001011 1101000 0110100 Codeword: 00111010 00010111 11010001 01101001 10010101 Check:

	Type: Two-Dimensional-Even Insert Codeword: 00111010 00010111 11010001 01101001 10010101 PASS :)
6	Generate: Type: Two-Dimensional-Even Insert Dataword: 010001 010100 001101 110100 000010 001010 011110 111010 101000 010011 Codeword: 0100010 0101000 0011011 1101001 0000101 0010100 0111100 1110100 1010000 0100111 1010110 Check: Type: Two-Dimensional-Even Insert Codeword: 0100010 0101000 0011011 1101001 0000101 0010100 0111100 1110100 1010000 0100111 1010111 FAIL :(
7	Generate: Type: Two-Dimensional-Even Insert Dataword: 01110 11010 00011 10100 01110 10000 01001 01001 Codeword: 011101 110101 000110 101000 011101 100001 010010 010010 111010 Check: Type: Two-Dimensional-Even Insert Codeword: 011101 110101 000110 101000 011101 100001 010010 010011 111010 FAIL :(
8	Generate: Type: Two-Dimensional-Even Insert Dataword: 0011 0100 0111 0110 1000 Codeword: 00110 01001 01111 01100 10001 11101 Check: Type: Two-Dimensional-Even Insert Codeword: 00110 11001 01111 01100 10001 11101 FAIL :(
9	Generate: Type: Two-Dimensional-Even Insert Dataword: 10011 11110 01011 00000 00100 Codeword: 100111 111100 010111 000000 001001 000101 Check: Type: Two-Dimensional-Even Insert Codeword: 100111 111100 010111 000000 001001 000111
10	Generate:

```
Type: Two-Dimensional-Even
Insert Dataword: 010 111 011 100
Codeword: 0101 1111 0110 1001 0101
```

Check:

```
Type: Two-Dimensional-Even
Insert Codeword: 0101 1111 0110 1001 0111

 FAIL :(
```

Two-Dimensional-Odd

1	Generate: ```Type: Two-Dimensional-Odd Insert Dataword: 11100 01101 11110 Codeword: 111000 011010 111101 100000``` Check: ```Type: Two-Dimensional-Odd Insert Codeword: 111000 011010 111101 100000 PASS :)```
2	Generate: ```Type: Two-Dimensional-Odd Insert Dataword: 101 100 001 111 111 Codeword: 1011 1000 0010 1110 1110 1110``` Check: ```Type: Two-Dimensional-Odd Insert Codeword: 1011 1000 0010 1110 1110 1110 PASS :)```
3	Generate: ```Type: Two-Dimensional-Odd Insert Dataword: 10111 01011 00111 00110 00100 01010 00011 01111 11100 Codeword: 101111 010110 001110 001101 001000 010101 000111 011111 111000 111000``` Check:

	Type: Two-Dimensional-Odd Insert Codeword: 101111 010110 001110 001101 001000 010101 000111 011111 111000 111000 PASS :)
4	Generate: Type: Two-Dimensional-Odd Insert Dataword: 1011110 0011011 1110100 0101111 1110100 0100000 0001000 0111011 1101011 Codeword: 10111100 00110111 11101001 01011110 11101001 01000000 00010000 01110110 11010110 11011010 Check: Type: Two-Dimensional-Odd Insert Codeword: 10111100 00110111 11101001 01011110 11101001 01000000 00010000 01110110 11010110 11011010 PASS :)
5	Generate: Type: Two-Dimensional-Odd Insert Dataword: 10111 00110 00001 10111 11101 Codeword: 101111 001101 000010 101111 111011 001011 Check: Type: Two-Dimensional-Odd Insert Codeword: 101111 001101 000010 101111 111011 001011 PASS :)
6	Generate: Type: Two-Dimensional-Odd Insert Dataword: 011 110 001 110 001 000 111 Codeword: 0111 1101 0010 1101 0010 0001 1110 0111 Check: Type: Two-Dimensional-Odd Insert Codeword: 0111 1101 0010 1101 0010 0001 1110 1111 FAIL :(
7	Generate: Type: Two-Dimensional-Odd Insert Dataword: 01000 01111 00101 00111 10100 Codeword: 010000 011111 001011 001110 101001 011100 Check: Type: Two-Dimensional-Odd Insert Codeword: 010000 011111 001011 001110 101001 011101 FAIL :(
8	Generate:

	Type: Two-Dimensional-Odd Insert Dataword: 1110010 1110110 0001000 0101000 0110101 1100011 1000001 Codeword: 11100101 11101100 00010000 01010001 01101011 11000111 10000011 10011000 Check: Type: Two-Dimensional-Odd Insert Codeword: 11100111 11101100 00010000 01010001 01101011 11000111 10000011 10011000 FAIL :(
9	Generate: Type: Two-Dimensional-Odd Insert Dataword: 101 110 001 101 001 111 000 Codeword: 1011 1101 0010 1011 0010 1110 0001 1101 Check: Type: Two-Dimensional-Odd Insert Codeword: 1011 1101 0010 1011 0010 1110 1001 1101 FAIL :(
10	Generate: Type: Two-Dimensional-Odd Insert Dataword: 11101 11110 01011 10100 00100 00101 00001 00000 01110 11111 00110 Codeword: 111011 111101 010110 101001 001000 001011 000010 000001 011100 111110 001101 101001 Check: Type: Two-Dimensional-Odd Insert Codeword: 111011 111101 010110 101011 001000 001011 000010 000001 011100 111110 001101 101001 FAIL :(

CRC

	CRC-8
1	Generate: ``` Type: CRC-8 Insert Dataword: 11011 Codeword: 1101111010001 ``` Check: ``` Type: CRC-8 Insert Codeword: 1101111010001 Syndrome: 00000000 PASS :) ```
2	Generate: ``` Type: CRC-8 Insert Dataword: 1001111000101 Codeword: 100111100010111110100 ``` Check: ``` Type: CRC-8 Insert Codeword: 100111100010111110100 Syndrome: 00000000 PASS :) ```
3	Generate: ``` Type: CRC-8 Insert Dataword: 111000110101010010010010010000011 Codeword: 1110001101010100100100110010000011110100111 ``` Check: ``` Type: CRC-8 Insert Codeword: 1110001101010100100100110010000011110100111 Syndrome: 00000000 PASS :) ```
4	Generate: ``` Type: CRC-8 Insert Dataword: 00100001 Codeword: 0010000101110001 ``` Check:

5	Generate:

```
Type: CRC-8
Insert Dataword: 01010110010000101010101000000110111000101011100011000011100000100000111101010001
Codeword: 01010110010000101010101000000110111000101011100011000011100000100000111101010001001111101
```

Check:

```
Type: CRC-8
Insert Codeword: 01010110010000101010101000000110111000101011100011000011100000100000111101010001001111101
Syndrome: 00000000

 PASS :)
```

6	Generate:

```
Type: CRC-8
Insert Dataword: 1111001010101010000010100111111101010000001101111011010
Codeword: 1111001010101010000010100111111101010000001101110110100010111
```

Check:

```
Type: CRC-8
Insert Codeword: 1111001010101010000010100111111101010000001101110110100011111
Syndrome: 00010000

 FAIL :(
```

7	Generate:

```
Type: CRC-8
Insert Dataword: 001100011101111001110011
Codeword: 0011000111011110011100111101011111
```

Check:

```
Type: CRC-8
Insert Codeword: 1111000111011110011100111010101111
Syndrome: 10101100

 FAIL :(
```

8	Generate:

```
Type: CRC-8
Insert Dataword: 01111
Codeword: 0111101111101
```

Check:

```
Type: CRC-8
Insert Codeword: 011111011111111
Syndrome: 00000010
```

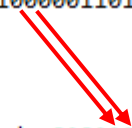|	
9	Generate: ``` Type: CRC-8 Insert Dataword: 01000010001101101010 Codeword: 0100001000110110101010110100 ``` Check: ``` Type: CRC-8 Insert Codeword: 0100001000110110101010010100 Syndrome: 00100000 FAIL :(```
10	Generate: ``` Type: CRC-8 Insert Dataword: 01010000011 Codeword: 0101000001101010011 ``` Check: ``` Type: CRC-8 Insert Codeword: 0101110001101010011 Syndrome: 00111001 FAIL :(```

CRC-16
1   Generate: ``` Type: CRC-16 Insert Dataword: 010111110101110101111110000101000010100 Codeword: 0101111101011101011111000010100001010010010010100111011 ``` Check: ``` Type: CRC-16 Insert Codeword: 0101111101011101011111000010100001010010010010100111011 Syndrome: 0000000000000000   PASS :) ```
2   Generate:

```
Type: CRC-16
Insert Dataword: 111100
Codeword: 11110000000000010001000
```

Check:

```
Type: CRC-16
Insert Codeword: 11110000000000010001000
Syndrome: 0000000000000000

 PASS :)
```

3	Generate:

```
Type: CRC-16
Insert Dataword: 1110000101111011101
Codeword: 111000010111011101100111100111111011
```

Check:

```
Type: CRC-16
Insert Codeword: 111000010111011101100111100111111011
Syndrome: 0000000000000000

 PASS :)
```

4	Generate:

```
Type: CRC-16
Insert Dataword: 1100
Codeword: 11000000000000101000
```

Check:

```
Type: CRC-16
Insert Codeword: 11000000000000101000
Syndrome: 0000000000000000

 PASS :)
```

5	Generate:

```
Type: CRC-16
Insert Dataword: 000111011101000011011110011001011010101010100110111001010
Codeword: 000111011101000011011110011001011010101010100110111001010100101110000011000
```

Check:

```
Type: CRC-16
Insert Codeword: 000111011101000011011110011001011010101010100110111001010100101110000011000
Syndrome: 0000000000000000

 PASS :)
```

6	Generate:

```
Type: CRC-16
Insert Dataword: 0011101110
Codeword: 001110111000000001001100100
```

Check:

```
Type: CRC-16
Insert Codeword: 001110111000000001001100111
Syndrome: 0000000000000011

  FAIL :(
```

7	Generate:

```
Type: CRC-16
Insert Dataword: 0011010111110001001010
Codeword: 0011010111110001001010100011001110
```

Check:

```
Type: CRC-16
Insert Codeword: 0011011111110001001010100011001110
Syndrome: 1000000000000110

  FAIL :(
```

8	Generate:

```
Type: CRC-16
Insert Dataword: 101110
Codeword: 101110000000011100100
```

Check:

```
Type: CRC-16
Insert Codeword: 111110000000011100100
Syndrome: 1000000001100011

  FAIL :(
```

9	Generate:

```
Type: CRC-16
Insert Dataword: 0101000101010000001101001010100010000011
Codeword: 010100010101000000110100101010010000110110101100010101
```

Check:

```
Type: CRC-16
Insert Codeword: 010100010100000000110100101010010000110110101100010101
Syndrome: 1101000000000011

  FAIL :(
```

10	Generate:

```
Type: CRC-16
Insert Dataword: 11000
Codeword: 1100000000001010000
```

Check:

```
Type: CRC-16
Insert Codeword: 1111100000000001010000
Syndrome: 1000000000010001

  FAIL :(
```

CRC-32

1 Generate:

```
Type: CRC-32
Insert Dataword: 01110111
Codeword: 0111011111111110111100110100110111100010
```

Check:

```
Type: CRC-32
Insert Codeword: 0111011111111110111100110100110111100010
Syndrome: 00000000000000000000000000000000

  PASS :)
```

2 Generate:

```
Type: CRC-32
Insert Dataword: 100111010001010110011010010011000101
Codeword: 1001110100010101100110100100110000101000001111010111001111111001110000
```

Check:

```
Type: CRC-32
Insert Codeword: 1001110100010101100110100100110000101000001111010111001111111001110000
Syndrome: 00000000000000000000000000000000

  PASS :)
```

3 Generate:

```
Type: CRC-32
Insert Dataword: 010111001101010001000
Codeword: 0101110011010100010011111111100001110000111111000011
```

Check:

4	Generate:
	Type: CRC-32 Insert Dataword: 00000101100000111011000001111100 Codeword: 00000101100000111011000001111100001110001001101000101010000001011
	Check:
	Type: CRC-32 Insert Codeword: 00000101100000111011000001111100001110001001101000101010000001011 Syndrome: 00000000000000000000000000000000 PASS :)
5	Generate:
	Type: CRC-32 Insert Dataword: 01100 Codeword: 011000011010100001100100110110100
	Check:
	Type: CRC-32 Insert Codeword: 011000011010100001100100110110100 Syndrome: 00000000000000000000000000000000 PASS :)
6	Generate:
	Type: CRC-32 Insert Dataword: 100010101010101011010000011 Codeword: 100010101010101011010000011000101111101001111000010100010010
	Check:
	Type: CRC-32 Insert Codeword: 100011111111111111010000011000101111101001111000010100010010 Syndrome: 0010010100010011100100001011011010 FAIL :(
7	Generate:
	Type: CRC-32 Insert Dataword: 10110 Codeword: 101100101011010010111100101011011000010
	Check:

8	Generate:

```
Type: CRC-32
Insert Dataword: 0101101111101111111110000100001
Codeword: 01011011111011111110000100001011000011110000011111100111011010
```

Check:

```
Type: CRC-32
Insert Codeword: 00011011111011111110000100001011000011110000011111100111011010
Syndrome: 11111001111010101001100000001010
```

9	Generate:

```
Type: CRC-32
Insert Dataword: 1110011000010001111010
Codeword: 11100110000100011110100010000100010100101010001000101111
```

Check:

```
Type: CRC-32
Insert Codeword: 11100110000000011110100010000100010100101010001000101111
Syndrome: 01000101001001000010000110101001

  FAIL :(
```

10	Generate:

```
Type: CRC-32
Insert Dataword: 0001000111001
Codeword: 00010001110010101011000001001000000111100100000
```

Check:

```
Type: CRC-32
Insert Codeword: 11010001110010101011000001001000000111100100000
Syndrome: 10011010000110011101100001000001

  FAIL :(
```

Checksum

1	Generate:
| | ```
Insert Dataword: 0110 1100 1111 0001 1110 0001
6 Word

sum: 0011 0001 (49)
wrapped sum: 0100
actual checksum: 1011
Codeword: 0110 1100 1111 0001 1110 0001 1011
``` |
| | Check: |
| | ```
Insert Codeword: 0110 1100 1111 0001 1110 0001 1011
7 Word

add: 0011 1100 (60)
wrapped sum: 1111
calcuated checksum: 0000

 PASS :)
``` |
| 2 | Generate: |
| | ```
Insert Dataword: 111100 001010 001001 100101 000101 110100 001111
7 Word

sum: 000010 111100 (188)
wrapped sum: 111110
actual checksum: 000001
Codeword: 111100 001010 001001 100101 000101 110100 001111 000001
``` |
| | Check: |
| | ```
Insert Codeword: 111100 001010 001001 100101 000101 110100 001111 000001
8 Word

add: 000010 111101 (189)
wrapped sum: 111111
calcuated checksum: 000000

 PASS :)
``` |
| 3 | Generate: |
| | ```
Insert Dataword: 1100001010 1110101010 0001011101 1101000000 0011100101
5 Word

sum: 0000000010 1100110110 (2870)
wrapped sum: 1100111000
actual checksum: 0011000111
Codeword: 1100001010 1110101010 0001011101 1101000000 0011100101 0011000111
``` |
| | Check: |

	``Insert Dataword: 1100001010 1110101010 0001011101 1101000000 0011100101 0011000111
6 Word

sum: 0000000010 1111111101 (3069)
wrapped sum: 1111111111
actual checksum: 0000000000
Codeword: 1100001010 1110101010 0001011101 1101000000 0011100101 0011000111 0000000000``` |
| 4 | Generate:

```Insert Dataword: 1110101011001010 1111011010101010 0001011101101101 0001101010110010 0001010101111010
5 Word

sum: 0000000000000010 0010100100001101 (141581)
wrapped sum: 0010100100001111
actual checksum: 1101011011110000
Codeword: 1110101011001010 1111011010101010 0001011101101101 0001101010110010 0001010101111010 1101011011110000```

Check:

```Insert Codeword: 1110101011001010 1111011010101010 0001011101101101 0001101010110010 0001010101111010 1101011011110000
6 Word

add: 0000000000000010 1111111111111101 (196605)
wrapped sum: 1111111111111111
calcuated checksum: 0000000000000000

 PASS :)``` |
| 5 | Generate:

```Insert Dataword: 00110 10111 11011 00110 11011 11011 01000 00100 01100 00011 00001 00101 11011 10011 00011 11000 00011
17 Word

sum: 00111 00001 (225)
wrapped sum: 01000
actual checksum: 10111
Codeword: 00110 10111 11011 00110 11011 11011 01000 00100 01100 00011 00001 00101 11011 10011 00011 11000 00011 10111```

Check:

```Insert Codeword: 00110 10111 11011 00110 11011 11011 01000 00100 01100 00011 00001 00101 11011 10011 00011 11000 00011 10111
18 Word

add: 00111 11000 (248)
wrapped sum: 11111
calcuated checksum: 00000

 PASS :)``` |
| 6 | Generate:

```Insert Dataword: 1110100101 0001011011 0110100101 0111011011 1101000001
5 Word

sum: 0000000010 1011000001 (2753)
wrapped sum: 1011000011
actual checksum: 0100111100
Codeword: 1110100101 0001011011 0110100101 0111011011 1101000001 0100111100```

Check: |

```
Insert Codeword: 1110100101 0001011011 0110100101 0111011011 1101000001 1100111100
6 Word

add: 0000000011 0111111101 (3581)
wrapped sum: 1000000000
calcuated checksum: 0111111111

 FAIL :(
```

7 | Generate:

```
Insert Dataword: 00110 01110 01110 01111 11010 00001 00000 01110 11000
9 Word

sum: 00011 10010 (114)
wrapped sum: 10101
actual checksum: 01010
Codeword: 00110 01110 01110 01111 11010 00001 00000 01110 11000 01010
```

Check:

```
Insert Codeword: 00110 01110 01110 01111 11010 00001 00000 01111 11000 01010
10 Word

add: 00011 11101 (125)
wrapped sum: 00001 00000
calcuated checksum: 11110011111

 FAIL :(
```

8 | Generate:

```
Insert Dataword: 1000000011 1101011011 0001010100 0001111111 0110100011 1111010000
6 Word

sum: 0000000010 1110100100 (2980)
wrapped sum: 1110100110
actual checksum: 0001011001
Codeword: 1000000011 1101011011 0001010100 0001111111 0110100011 1111010000 0001011001
```

Check:

```
Insert Codeword: 1000000011 1101011000 0001010100 0001111111 0110100011 1111010000 0001011001
7 Word

add: 0000000010 1111111010 (3066)
wrapped sum: 1111111100
calcuated checksum: 0000000011

 FAIL :(
```

9 | Generate:

```
Insert Dataword: 01111111 00010101 10010110 00011011 11001011 00000111 000100011 11111010 01110011 11010111
10 Word

sum: 00000100 01111110 (1150)
wrapped sum: 10000010
actual checksum: 01111101
Codeword: 01111111 00010101 10010110 00011011 11001011 00000111 000100011 11111010 01110011 11010111 01111101
```

Check:

```
Insert Codeword: 01111111 00010101 10010110 00011011 11001011 00000111 000100011 00000010 01110011 11010111 01111101
11 Word

add: 00000100 00000011 (1027)
wrapped sum: 00000111
calcuated checksum: 11111000

 FAIL :(
```

10	Generate:

```
Insert Dataword: 1101000100 0001110101 1110001111 0001000111 0101110010 1110001010 0001110101
7 Word

sum: 0000000011 0100000000 (3328)
wrapped sum: 0100000011
actual checksum: 1011111100
Codeword: 1101000100 0001110101 1110001111 0001000111 0101110010 1110001010 0001110101 1011111100
```

Check:

```
Insert Codeword: 1101000100 0001110101 1110001111 0001000111 0101110010 1110001010 0001110101 1011111111
8 Word

add: 0000000011 1111111111 (4095)
wrapped sum: 0000000001 0000000010
calcuated checksum: 1111111111001111111101

 FAIL :(
```

Hamming Code

1	Generate: ``` Insert Dataword: 1011101111101 Codeword: 101111011101101100 ``` Check: ``` Insert Codeword: 101111011101101100 PASS :) ```
2	Generate: ``` Insert Dataword: 00100111000001111110 Codeword: 001001110000001111111010 ``` Check: ``` Insert Codeword: 001001110000001111111010 PASS :) ```
3	Generate: ``` Insert Dataword: 000011110010001101111110010100001 Codeword: 0000110110010001101111110101010000100 ``` Check: ``` Insert Codeword: 0000110110010001101111110101010000100 PASS :) ```
4	Generate: ``` Insert Dataword: 0011000010001100110011110000110010110000110111100000011000010111 Codeword: 00110000010001100110011110000110010110000110111100000011000010111100 ``` Check: ``` Insert Codeword: 00110000010001100110011110000110010110000110111100000011000010111100 PASS :) ```
5	Generate: ``` Insert Dataword: 111100010100001011010 Codeword: 1111000101100010111011001 ``` Check: ``` Insert Codeword: 1111000101100010111011001 PASS :) ```
6	Generate:

```
Insert Dataword: 1001101
Codeword: 10011100101
```

Check:

```
Insert Codeword: 10010100101

  FAIL :( at position 7
```

7	Generate:

```
Insert Dataword: 100111100011
Codeword: 110011110000011111
```

Check:

```
Insert Codeword: 010011110000011111

  FAIL :( at position 17
```

8	Generate:

```
Insert Dataword: 0001110101110101011010011001
Codeword: 00001110101110101011101001110000110
```

Check:

```
Insert Codeword: 00001110001110101011101001110000110

  FAIL :( at position 26
```

9	Generate:

```
Insert Dataword: 110001101001100001000100101
Codeword: 110001101001010000100001000100111
```

Check:

```
Insert Codeword: 110001101001010000100001000100011

  FAIL :( at position 3
```

10	Generate:

```
Insert Dataword: 001110110
Codeword: 0011110110011
```

Check:

```
Insert Codeword: 0011111110011

  FAIL :( at position 7
```