1. A) **Convert 623_{10} to base 3.**

```
3|623 2
  ---
  207 0
  ---
  69  0
  ---
  23  2
  ---
  7   1
  ---
  2   2
  ---
   0
```

Final Result:
$623_{10}= 212002_3$

1. B) **Convert 2341_{10} to base 7.**

```
7|2341 3
  ---
  334  5
  ---
  47   5
  ---
  6    6
  ---
  0
  ---
```

Final Result:
$2341_{10}= 6553_7$

2. A) **Convert 43.856 to binary.**

```
2|43  1
  ---
  21  1
  ---
  10  0
  ---
  5   1
  ---
  2   0
  ---
  1   1
  ---
  0
```

43 = 101011 in binary

0.85600000*2= 1.71200000 ⇒ **1**

0.71200000*2= 1.42400000 ⇒ **1**

0.42400000*2= 0.84800000 ⇒ **0**

0.84800000*2= 1.69600000 ⇒ **1**

0.69600000*2= 1.39200000 ⇒ **1**

0.39200000*2= 0.78400000 ⇒ **0**

0.85600000 = .110110 in binary

Final Result:

43.856 equals = 101011.110110 in binary

2. B)**Convert 299.5656 to binary.**

```
2|299 1
  ---
  149 1
  ---
  74  0
  ---
  37  1
  ---
  18  0
  ---
  9   1
  ---
  4   0
  ---
  2   0
  ---
  1   1
  ---
  0
```

299 = 100101011 in binary

0.56560000*2= 1.13120000 ➡ **1**

0.13120000*2= 0.26240000 ➡ **0**

0.26240000*2= 0.52480000 ➡ **0**

0.52480000*2= 1.04960000 ➡ **1**

0.04960000*2= 0.09920000 ➡ **0**

0.09920000*2= 0.19840000 ➡ **0**

0.56560000 = .100100 in binary

Final Result:

299.5656equals = 100101011.100100 in binary

3. A) **01101110**

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	1	0	1	1	1	0

= 0+64+32+0+8+4+2

= 110

Final Result:

$(01101110)_2 = (110)_{10}$

B) **01101110**

01101110

10010001 $\Longrightarrow$ One’s complement (Switching the 0s and 1s)

+1 $\Longrightarrow$ Two’s complement (Adding 1)

1001001

128	64	32	16	8	4	2	1
1	0	0	1	0	0	1	0

= -128+16+2

=|-110|

Final Result:

110

4.A) **1010 X 10**

```
      1010
     X  10
     ------
       0000
   +
      10100
     -------
      10100
```

Final Result:

10100

B)**11011 X 111**

```
       11011
      X  111
      -------
         11011
+       110110
       1101100
----------------
     10111101
```

Final Result:

10111101

5.A) **67.25 in floating point model**

67 =

```
    2|67 1
      ---
      33 1
      ---
      16 0
      ---
      8  0
      ---
      4  0
      ---
      2  0
      ---
      1  1
      ---
      0
      ---
```

0.25=

0.25*2= 0.50 ⇒ **0**

0.50*2= 1.0 ⇒ **1**

$(1000011.01)_2 = 0.100001101 \times 2^7$

Offset exponent = 7+15 = 22

Offset exponent in binary: 22 = 10110

Final bit string: |0|10110|10000110|

47.98 in floating point model

47 =

```
    2|47 1
      ---
      23 1
      ---
      11 1
      ---
      5  1
      ---
      2  0
      ---
      1  1
      ---
      0
```

0.98=

0.98*2= 1.96 ⇒ **1**

0.96*2= 1.92 ⇒ **1**

0.92*2= 1.84 ⇒ **1**

0.84*2= 1.68 ⇒ **1**

0.68*2= 1.36 ⇒ **1**

0.36*2= 0.72 ⇒ **0**

$(101111.111110)_2 = 0.101111111110 \times 2^6$

Offset exponent = 6+15 = 21

Offset exponent in binary: 21 = 10101

Final bit string: |0|10101|10111111|

B) **Adding the two numbers together**

67.25 + 47.98= 115.23

115.23 in binary = 1110011.01

Now we normalize = 0.11100110 x 2^7

7+15 = 22 which in binary is 10110 and this is our exponent

Final bit string: |0|10110|11100110|

6.A) **34.55 in IEEE-754**

34 =

```
2|34 0
  ---
  17 1
  ---
  8  0
  ---
  4  0
  ---
  2  0
  ---
  1  1
  ---
  0
  ---
```

0.55=

0.55*2= 1.1	⇨	**1**
0.1*2= 0.2	⇨	0
0.2*2= 0.4	⇨	**0**
0.4*2= 0.8	⇨	0

0.8*2= 1.6 ⟹ **1**
0.6*2= 1.2 ⟹ 1
Etc..

Write it in binary:

100010.10001100110011001101

Convert into base 2 scientific notation:

100010.10001100110011001101 x 2^0

1.0001010001100110011001101 x 2^5

Exponent bias is 127 for single-precision, so we add 127 to 5 and we get 132. The binary value of 132 = 10000100

Sign= 0 since the number is positive

Exponent: The binary value of 132 is our exponent so 10000100

Mantissa= number after the dot 00010100011001100110011

So we get:

Final bit string: |0|10000100|00010100011001100110011|

B) **-201.601 in IEEE-754**

201 =

```
2|201 1
  ---
  100 0
  ---
  50  0
  ---
  25  1
  ---
  12  0
  ---
  6   0
```

---
3 **1**
---
1 **1**

0

0.601=

0.601*2= 1.202 ⇒ **1**
0.202*2= 0.404 ⇒ **0**
0.404*2= 0.808 ⇒ **0**
0.808*2= 1.616 ⇒ **1**
0.616*2= 1.232 ⇒ **1**
0.232*2= 0.464 ⇒ **0**
Etc..

Write it in binary:

11001001.1001100111011011001

Convert into base 2 scientific notation:

11001001.1001100111011011001 x 2^0

1.10010011001100111011011001 x 2^7

Exponent bias is 127 for single-precision, so we add 127 to 7 and we get 134. The binary value of 134 = 10000110

Sign= 1 since the number is negative

Exponent: The binary value of 134 is our exponent so 10000110

Mantissa= number after the dot 10010011001100111011011001

So we get:

Final bit string: |1|10000110|10010011001100111011011|

7. **Minimum hamming distance of the four code words**

A-B= 8 code words differ

1110001110010111

1001011010001110

A-C= **9** code words differ

1110001110010111

0010111101101111

A-D= **5** code words differ

1110001110010111

1100000000011111

B-C= **9** code words differ

1001011010001110

0010111101101111

B-D= **7** code words differ

1001011010001110

1100000000011111

C-D= **10** code words differ

0010111101101111

1100000000011111

Final result: Minimum hamming distance is **5**.

8. **Locate the bit that has been altered.**

Bit string: |0|1|1|1|1|0|1|0|1|0|1|

|0(D11)|1(D10)|1(D9)|1(P8)|1(D7)|0(D6)|1(D5)|0(P4)|1(D3)|0(P2)|1(P1)|

P1, p2, p3 ,p4 = The 4 check bits

P1: Check bits 1,3,5,7,9,11

1 1 1 1 1 0 $\Longrightarrow$ Odd

P1= 1

P2: Check bits 2,3,6,7,10,11

0 1 0 1 1 0 $\Longrightarrow$ Odd

P2= 1

P4: Check bits 4,5,6,7

0 1 0 1 $\Longrightarrow$ Even

P4= 0

P8: Check bits 8,9,10,11

1 1 1 0 $\Longrightarrow$ Odd

P8= 1

Converting we get:

P8 P4 P2 P1

1 0 1 1 = 11 in Decimal

Now we know that there is an error in D11 so we correct it

Corrected bit string: **|1|1|1|1|1|0|1|0|1|0|1|**

9.A) **Find the quotient and remainder of $100010_2/1011_2$**

```
      101
1011|100010
     1011
     ----
     001110
       1011
     ------
       0101
```

Final Result: The quotient is 101_2 and the remainder is 101_2.

B) **Find the quotient and remainder of $1110111_2/100_2$**

```
     11101
100|1110111
    100
    ----
    0110
     100
    ----
     0101
      100
     ----
      00111
        100
      -----
        011
```

Final Result: The quotient is 11101_2 and the remainder is 11_2.

10. **Reverse calculation for the receiver.**

```
       11100
1101|10001111
     1101
     ----
     1011111
     1101
     -----
      110111
      1101
      -----
        00011
        0000
       -----
        0011
        0000
        ----
         011
```

CRC code word is now 10001111011 since we added the remainder at the end of the information word.

For the reverse calculation, we divide the received CRC code by 1011 to find the remainder.

```
         1110001
1101|10001111010
     1101
     ----
      1011
      1101
     -----
       1101
       1101
       -----
        0001101
           1101
           -----
            00000
```

Final result:

Since the remainder is 0, then the receiver can be certain there's no error occurring in transmission and that he got the correct information.