Exercise

Data Representation

1.2 Converting between Number Bases

Convert 01101001 to decimal =	Convert 10 001101 (binary) to hexadecimal =
Convert 151 (octal) to decimal =	Convert 123 (octal) to binary =
Convert 15B (hexadecimal) to decimal =	Convert 123 (octal) to hexadecimal =
Convert 156 (decimal) to binary =	Convert 1F3 (hex) to binary =
Convert 1234 (decimal) to octal =	Convert 1F3 (hexadecimal) to Octal =
Convert 689 (decimal) to hexadecimal =	
Convert 101101001 (binary) to octal =	

```
try {
      // read the number from Keyboard
      BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
      System.out.print("Enter the number in decimal:");
      String numberInStr = in.readLine();
      int num = Integer.parseInt(numberInStr);
      System.out.println();
      System.out.println("Number in Binary: " + Integer.toBinaryString(num));
      System.out.println("Number in Octal: " +  Integer.toOctalString(num));
      System.out.println("Number in Hexadecimal: " + Integer.toHexString(num));
   } catch (Exception e) {
      System.err.println("Error");
}
```

What is the output from the above program if user enters "46".

```
Exercise 1
Number in Binary:
Number in Octal:
Number in Hexadecimal:
```

What is the output of the following program? Why?

```
public class Ex1 {
  public static void main(String[] args) {
    int n1 = 10;
    int n2 = 010;
    int n3 = 0x10;
    System.out.println(n1);
    System.out.println(n2);
    System.out.println(n3);
  }
}
```


1.3 Arithmetic Operations

Base 2: 01011111 + 00010001 00111110	Base 8: 363 + 247 110 carries	Base 16: F3 + a7 100
Base 2: 01011111 - 00010001 00000000	Base 8: 363 - 247 010 borrows	Base 16: F3 - a7 10

1.4 Negative Numbers

What is the range of :
a) 4-bit unsigned number?
b) 4-bit Excess (biased)?
c) 4-bit Two's complement?

1.4.4 Conversion

(1) Convert 10110011 to Decimal if the number is represented as signed 8-bit binary in Excess representation

(2) Convert 10110011 to Decimal if the number is represented as signed 8-bit binary in two's complement

(3) Convert 10110011 to Decimal if the number is represented as unsigned 8-bit binary

(4) Convert Decimal –17 to 8-bit Binary using Excess representation

(5) Convert Decimal –17 to 8-bit Binary using Two's complement representation

1.4.5 Overflow

(1) 0111 + 0101	**(2)** 1001 + 1100
(3) 1100 + 0111	**(4)** 1001 + 0011

1.5 Subtraction by complement addition

(1) 01101011 – 00010110
(2) 00101100 – 01101001

1.6 Bitwise Logical Operations

Exercise 1010 0001 &0101 1111	Exercise 1010 0001 |0101 1111	Exercise 1010 0001 ^0101 1111

1.7 Shift Operations

A) What is the result of 00001100 << 3? =	B) 32-bit binary signed number: int x = 31 << 2;
A) 8-bit binary signed number: 10100001 >> 3 =	B) 32-bit binary signed number: int x = 21 >> 2;
A) 8-bit binary signed number: 10100001 >>> 3 =	B) 32-bit binary signed number: int x = -1 >>> 2;

```
public class BitOperators {
  public static void main(String[] args) {
    int x = 61, y = 31;
    System.out.println("x="+x);
    System.out.println("y="+y);
    System.out.println("x="+Integer.toBinaryString(x));
    System.out.println("y="+Integer.toBinaryString(y));
    System.out.println("x & y=" + (x & y));
    System.out.println("x | y=" + (x | y));
    System.out.println("x << 2=" + (x << 2));
    System.out.println("x >> 2=" + (x  >> 2));
    System.out.println("x >>> 2=" + (x >>> 2));
    System.out.println("x ^ y=" + (x ^ y));
  }
}
x=21
x=
y=56
y=
x & y
x | y
x << 2
x >> 2
x >>> 2
x ^ y
```


2.5 Conversion

Convert 0062 0073 0042 662E 0020 5DF6 from UCS-2 to UTF-8 coding.
Convert 21 E8 A2 93 C7 8E from UTF-8 to UCS-2 coding.