**Number System**

In computers we normally use **four** different **numbering systems** - **Decimal**, **Binary**, **Octal** and **Hexadecimal**

In **Decimal** system we use the symbols 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 to denote various numbers.

In a **Binary** number system we use 0's (zeros) and 1's (ones) as the only symbols to represent numbers of all magnitudes (sizes).

A number in a particular base is written as (Number)base of number  for example (34)10 is a **decimal** number (Thirty Four) and (11)2 is a **binary** number 11 (we will read it as One One and not Eleven) which actual represents a decimal number whose value is **3**.

Since we normally use the decimal number system the decimal number (124)10 is simply written as **124**. However, if we want to represent a binary **One Zero One**, we will write it as 1012.

Similarly we have **octal** number system which uses **8** as the base. It is usually used in digital displays and in representing file permissions under UNIX/Linux operating systems.

**Hexadecimal** or **Hex** is a number system that uses **16** as the base to represent numbers.

# Decimal System (Base – 10 system)

It is called base-10 because it uses 10 unique digits from **0** to **9** to represent any number in this number system.

A base (also called **radix**) is the number of unique digits or symbols (including 0) used to represent a given number.

# Binary System (Base – 2 system)

The numbering system which uses base-2 is called Binary System. In Binary System (base-2) a total of 2 digits (0 and 1) are used to represent a number of any size (magnitude).

**In Java, Binary numerals are prefixed with a leading 0b (or 0B) (digit zero followed by char 'b')**. For example, to store an binary value of seven into a variable binarySeven, we write

int binarySeven = 0b111;

For example:

**Zero** is represented as 0, | 0 = (0 \* 20) = (0 \* 1)

**One:** 1 = (1 \* 20) = (1 \* 1)

**Two:** since 0 or 1 are the only digits we can use to represent 2, let us divide 2 by 2 and write down [quotient][reminder], i.e.: [1][0] :: 2 = (1 \* 21) + (0 \* 20) = (2) + (0)

**Three**: since 0 or 1 are the only digits we can use to represent 3, let us divide 3 by 2 and write down [quotient][reminder], i.e.: [1][1] :: 3 = (1 \* 21) + (1 \* 20) = (2) + (1)

**Fourteen:** let us divide 14 by 2 and write down [quotient][reminder], i.e.: [7][0], by repeating the above logic for 7 (7 = [3][1], and 3 = [1][1]) we finally get [1][1][1][0] ::

14 = (1 \* 23) + (1 \* 22) + (1 \* 21) + (0 \* 20)

14 =       (8)     +     (4)     +     (2)     +     (0)

[Note: we can also use “8,4,2,1” method]

Computers have circuits (logic gates) which can either be in off or on state. These two states are represented by 0 (zero) and 1 (one).  
It is for this reason computation in computers is performed using a binary number system (base-2) where all numbers are represented using 0's and 1's.

Each **binary digit** i.e., a single **0** (zero) or **1** (one) is called a **bit**. A collection of **8** such bits is called a **Byte**.

1 byte = 8 bits

1 kilobyte = 1024 bytes

1 megabyte = 1024 kilobytes

1 gigabyte = 1024 megabytes

1 terabyte = 1024 gigabytes

1 petabyte = 1024 terabytes

# Octal System (Base – 8 system)

 In Octal System (base-8) a total of 8 digits (0, 1, 2, 3, 4, 5, 6 and 7) are used to represent a number of any size (magnitude).

 representing **Twenty One** (21), since 0 to 7 are the only digits we can use to represent 21, let us divide 21 by 8 and write down [quotient][reminder], i.e.: [2][5] :: 21 = (2 \* 81) + (5 \* 80) = (16) + (5)

representing **Four Hundred and Twenty One** (421), let us divide 421 by 8 and write down [quotient][reminder], i.e.: [52][5] (further dividing 52 by 8 we get [6][4]), which is [6][4][5]  
421 = (6 \* 82) + (4 \* 81 + (5 \* 80) = (384) + (32) + (5)

**In Java, Octal numerals are prefixed with a leading 0 (zero)**

if we want to store a octal representation of decimal number 9 into a variable numberNine, we write

int numberNine = 011;

In **octal** to **binary** conversion, we will have to use **three bits** to represent each **octal digit**.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Octal** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| **Binary** | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |

For example, an octal number 0246 is converted into binary as

Octal Number -> 2 4 6

Binary Number -> 010 100 110

# Hexadecimal System (Base – 16 system)

The numbering system which uses base-16 is called Hexadecimal System or simply Hex. In Hexadecimal System (base-16) a total of 16 symbols are used.

Digits 0(zero) to 9(Nine) are used to represent values from 0 to 9 respectively and alphabets A, B, C, D, E and F (or a, b, c, d, e and f) are used to represent values from 10 to 15 respectively.

**In Java and many other programming languages ‘0x’ is used as a prefix to denote a hexadecimal representation.**

For example, in hexadecimal the value of **Zero** is represented as 0x0, where  
0 = (0 \* 160) = (0 \* 1)

Similarly 1, 2 ...up to 9:  
1 = (1 \* 160) = (1 \* 1)  
2 = (2 \* 160) = (2 \* 1)  
...  
9 = (9 \* 160) = (9 \* 1)  
10 = A = (10 \* 160) = (10 \* 1)  
15 = F = (15 \* 160) = (15 \* 1)

Now, lets try representing **Twenty One** (21), remember we can use 0 to 9 and the alphabets A to F to represent 21, let us divide 21 by 16 and write down [quotient][reminder], i.e.: [1][5]  
21 = 0x15 = (1 \* 161) + (5 \* 160) = (16) + (5)  
Similarly, lets try representing **One Hundred and Sixty** (160), let us divide 160 by 16 and write down [quotient][reminder], i.e.: [10][0], [A][0] (since 10 is represented by A)  
160 = 0xA0 = (10 \* 161 + (0 \* 160) = (160) + (0)

**In Java, both uppercase and lowercase letters can be used when representing Hex values.** For example:

int hexHunderedAndSixty = 0xA0; // or 0Xa0, however 0xA0 is preferred

Question…

To convert a **hex** number to an **octal**, we will first convert the **hex** number to **binary** and then to **octal**.  
  
Below are the detailed steps followed for the above mentioned process:

1. Convert each **hex digit** to its corresponding **four binary digits** (bits) from right to left.
2. Combine all the **binary digits**.
3. Separate the **binary digits** into groups of **3 bits** each from right to left.
4. Ensure the left most group has **3 bits** by prefixing necessary zeros.
5. Find the **octal** equivalent of each group and combine the resultant digits together.

For example, hexadecimal number 0x5AF6 is converted into binary as

Hex Number -> 5 A F 6

Binary Number -> 0101 1010 1111 0110

Binary Number -> 0101101011110110

Binary Number -> 0 101 101 011 110 110

Binary Number -> 000 101 101 011 110 110 // After prefixing **zeros** in the left most group

Octal Number -> 0 5 5 3 6 6

Octal Number -> 055366

Hence, the **octal** equivalent of the given hexadecimal number is (55366)8

Similarly an **octal** can be converted into a **hex** following the below steps:

1. Convert each **octal digit** to its corresponding **three binary digits** (bits) from right to left.
2. Combine all the **binary digits**.
3. Separate the **binary digits** into groups of **4 bits** each from right to left.
4. Ensure the left most group has **4 bits** by prefixing necessary zeros.
5. Find the **hex** equivalent of each group and combine the resultant digits together.

For example, an octal number 0246 is converted into hexadecimal as

Octal Number -> 2 4 6

Binary Number -> 010 100 110

Binary Number -> 010100110

Binary Number -> 0 1010 0110

Binary Number -> 0000 1010 0110 // After prefixing **zeros** in the left most group

Hex Number -> 0 A 6

Hex Number -> 0X0A6

In **hex** to **binary** conversion, we will have to use **four bits** to represent each **hex digit**.

The following table shows the conversion of each **hex digit** into their corresponding **binary digits**.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Hexadecimal** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| **Binary** | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |

For example, hexadecimal number 0x5AF6 is converted into **binary** as:

Hex Number -> 5 A F 6

Binary Number -> 0101 1010 1111 0110

Hence, 0x5AF6 is (0101101011110110)2