Coder Army

Lecture - 001

By - Rohit Negi Bhaiya

Typed By – Nishit Mehta

Introduction To Programming

Decimal Number System (DNS):

In Decimal Number System there are **10** unique digits i.e. {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}. Therefore, DNS is **Base$_{10}$**.

Binary Number System (BNS):

In Binary Number System there are **2** unique digits i.e. {0, 1}. Therefore, BNS is **Base$_{2}$**.
In Binary Number System,

0 + 0 = 1; 0 + 1 = 1; 1 + 0 = 1; 1 + 1 = 10;

Octal Number System (ONS):

In Octal Number System there are **8** unique digits i.e. {0, 1, 2, 3, 4, 5, 6, 7}. Therefore, ONS is **Base$_{8}$**. In Octal Number System,

3 + 4 = 7; 4 + 5 = 11; 5 + 6 = 13; 23 = 27;

HexaDecimal Number System (HNS):

In HexaDecimal Number System there are **16** unique digits i.e. {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}. Therefore, HNS is **Base$_{16}$**. In HexaDecimal Number System,

3 + 4 = 7; 10 = A; 15 = F; 15 + 10 = 19; 15 + 12 = 1B;

Decimal Numbers	Binary Numbers	Octal Numbers	HexaDecimal Numbers
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12

To change number in any number system just check the **base** of that number system i.e., for decimal system the base is **10**, for binary system the base is **2**, for octal system the base is **8** and for hexadecimal system the base is **16**.

Decimal System to any Number System Conversion:

Formula:

Base	Quotient	Remainder

We'll divide the **Base** with the **Quotient** till the **Quotient** column becomes **0**, and write the number system form of the decimal number in the reverse order of the **Remainder** column.

- Suppose we need to find **27** in its **Binary** form.

2	27	Remainder (in reverse order)
2	13	1
2	6	1
2	3	0
2	1	1
2	0	1

Therefore, **27** in binary form is **11011**.

- Suppose we need to find **43** in its **Binary** form.

2	43	Remainder (in reverse order)
2	21	1
2	10	1
2	5	0
2	2	1
2	1	0
2	0	1

Therefore, **43** in binary form is **101011**.

Any Number System to Decimal System Conversion:

Let's consider a number **278**, we can write this number as:

$$2 \times 10^2 + 7 \times 10^1 + 8 \times 10^0 = 278$$

In the above example we can see that the base is **10**. Similarly, a binary number, octal number and hexadecimal number can be converted to its equivalent decimal from with base as **2, 8** and **16** respectively.

- Suppose we need to find **101** in its **Decimal** form.
  We can convert **101** as:
  $$1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 4 + 0 + 1 = 5$$
  Therefore, **101** in decimal form is **5**.

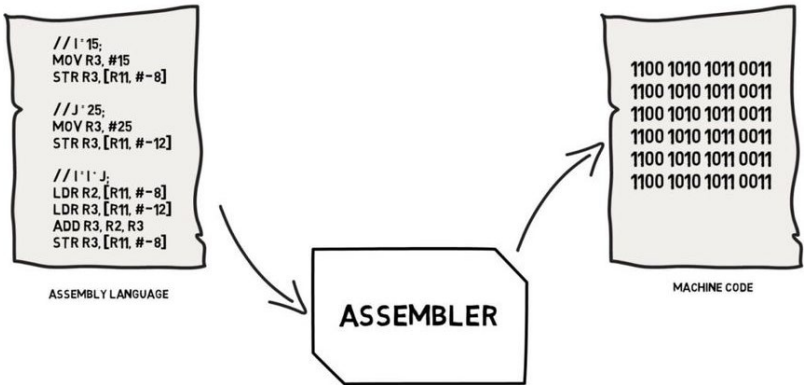
- Suppose we need to find **110101** in its **Decimal** form.
  We can convert **110101** as:
  $$1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 32 + 16 + 0 + 4 + 0 + 1 = 53$$
  Therefore, **110101** in decimal form is **53**.

- Suppose we need to find **AC2** in its **Decimal** form.
  We can convert **AC2** as:
  $$10 \times 16^2 + 12 \times 16^1 + 2 \times 16^0 = 2560 + 192 + 2 = 2754$$

Moore's Law:
It states that, in every 2 years the capacity of transistor will be increased by double.

History of Languages:

A computer only understands **binary form**. The language which computer understands is called **Machine Language**. A Machine Language is that language which consists of either **'0'** or **'1'**.
Eg: 010101000101;



To write Machine Language manually was a tough task and sometimes mistakes might happen. Therefore, to resolve this issue, an **assembler** was introduced which converts the **assembly language** submitted by user to computer into machine language (understandable to computer) i.e., **assembler** worked as a translator between user and computer.

Later on, more advancements were made to make a language which was similar to our speaking language which we termed as **high-level language**.
High-Level Language made our writing code convenient than before.
Eg: Java, Python, etc.

Machine Language is faster than **Assembly Language** and **Assembly Language** is faster than **High-Level Language**. This is because Assembly Language and High-Level Language are converted to Machine Language via some medium whereas Machine Language requires no conversion and is directly understandable to computer.

Data:
Programmers main goal should be to store maximum amount of data in minimum space, fetch data in less amount of time and arrange data in a sorted fashion.
