Number bases: (10 always equals value of base)

- Decimal (base 10)
- Binary (base 2)
- Hexadecimal (base 16)

Numbers Conversion Table

Decimal	Binary	Octal	Hex
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	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

The power of 10 expansion for a decimal

72081.203

$10^4\ 10^3\ 10^2\ 10^1\ 10^0\ 10^{-1}\ 10^{-2}\ 10^{-3}$

$7x10^4 + 2x10^3 + 0x10^2 + 8x10^1 + 1x10^0 + 2x10^{-1}\ 0x10^{-2} + 3x10^{-3}$

Converting from base “b” into decimal (base 10)

*Write the powers of the base under each digit

1. Write the power of “b” expansion
2. Simplify the multiplication and addition

2615_{seven} or $(2615)_{seven}$

$7^3\ 7^2\ 7^1\ 7^0$

$2x343 + 6x49 + 1x7 + 5x1 = 992_{ten}$

$(10)_b$ = base

Converting out of base 10 into base “b”

1. Write the powers of b till you reach a value larger than the given number
2. Create the same number of place value slots as the exponent on the base from step 1
3. Divide the given number by the largest power of b smaller than the number
4. Write the quotient in the left most open slot (could be zero)
5. Repeat steps 3-4 using the remainder as the “given number” till the remainder is zero
6. Right fill zero if needed

Converting between bases (without going to base 10; special cases): When converting bases that are multiples of each other, they can be directly translated without going to base 10 in between

- Because 16 is equivalent to 2^4, 4 bits of binary is equivalent to one unit of hex
  - Ex. $1101\ 0110_{ten} = D6_{sixteen}$
  - Ex. $6D5_{sixteen} = 0110\ 1101\ 0101_{two}$

$a \pmod b \equiv c$

C = the remainder of a/b “Remainder when divided by the mod”

Mod arithmetic tie into bases

- Congruence mod m of a given number is the same as the value of the one’s digit when written in base m

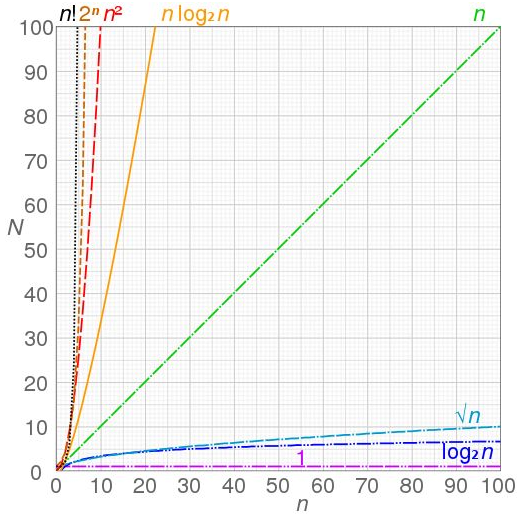
Sorting algorithm

- sieve of eratosthenes

String operations

- ❖ MSB or LSB: Most significant bit/byte (leftmost) or Least significant bit//byte (rightmost)
  - ➢ We will assume bit unless otherwise told
    - MSB (100 110) = 1
    - LSB (100 110) = 0
    - MSB (0110 1101) = 0
    - LSB (0110 1101) = 1
- String length
  - Input a string, outputs/returns a number equal to the number of bits/characters/digits
    - Ex. string length(digits) = 6
    - String length (101011) = 6
- Checksum
  - Input a binary string, output the sum of the bit values
    - Checksum (1011 0010 0110) = 6
- Reverse/invert a string
  - Input a string, outputs a string of equal length with the characters in the opposite order
    - Reverse (100110) = 011001
- Complement (two's complement)
  - Input a binary string, outputs a string of the same length where every 1 in the original string becomes a 0 and all the 0's become 1's
- Concatenate
  - Input two or more strings, outputs a string where the second string follows the first with a string length equal to the sum of the conjoining string lengths
    - Concatenate (1011 with 01) = 101101
    - Concatenate (01 with 1011) = 011011
- Pad a string
  - Input a string and a number (representing the new string length, number must be greater then the number of the current string length), outputs a string where 0's are left-filled till the desired string length is reached
    1. Desired string - original length = n
    2. Create a string of n zeros and concatenate it with the original string
    - Pad 01101 to 8 bits = 00001101
- Transpose
  - Input a string and an amount and sometimes a direction or diameter, outputs a string of the same length
    - If a direction is not stated we assume it is to the left
    - Transpose (10110101) three places = 10101101
      - The first three digits are brought to the end
- Split a string
  - Input a string of length a, outputs n strings of length b, where b is the least integer greater than a/n

Time complexity or computational time: measures the rate the time an algorithm takes to run relative to the complexity of the input (rate comparing time to input complexity). Length and complexity have linear relationship



$$\sum_{J=1}^{n} J \quad j = [n(n+1)]/2$$

Fastest to slowest

- Constant O(1)
- Log O(log n)
- Roots O($\sqrt[x]{n}$)
- Linear O(n)
- Linlog (linear x log) O(n x log n)
- Polynomial O(n^x)
- Exponential O(2^n)
- Factorial O(n!)

Sorting algorithms:
A list (or grid) of data is known as an array in computer science

- Linear/sequential/series search
    - Inspects each element of the list one at a time (no prep work)
    - Used for smaller/shorter lists
    - Linear time
- Binary search
    - Order data (ranking - often lowest to highest if numbers)
    1. Order your data
    2. Halve the data set and determine which half to keep
    3. Half the keeper set and determine which half of it to keep
    4. Repeat until complete
        a. Steps 2 to 4 = O(log n) -- log time
    - Not compatible for adding new data (not scalable)
    - Log time (not including sorting)
- Hash table/hash algorithm
    - Takes data set and relates all data points to an index by a hash function
    - More scalability than binary search
    - % means mod (mod arithmetic)
    - Best case: constant time; worst case: Linear time
    - Hash collision: when a hash function gives a value that fits a slot that is already filled