Complexity Classes:

- Polynomial Time Algorithms
  - Linear Search O(n)
  - Binary Search O(log n)
  - Merge Sort O(n log n)
  - Insertion Sort O(n^2)
  - Matrix Multiplication O(n^3)
- Non-Polynomial Time Algorithms
  - 0/1 Knapsack Problem O(2^n)
  - Graph Coloring O(2^n)

n	**n^2**	**n^3**	**2^n**
10	100	1000	1024
100	10000	1000000	1.27E+30
200	40000	8000000	1.61E+60

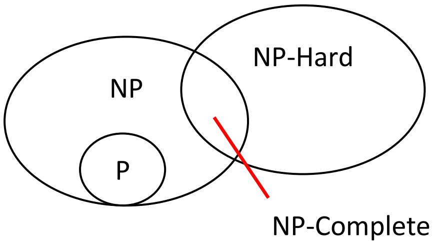
PROBLEM: For the algorithms with exponential complexity, there is a need to write algorithms in such a way that the complexity comes under Polynomial time.

- Deterministic: No Choices
- Non-Deterministic: Choices, Success, Failure

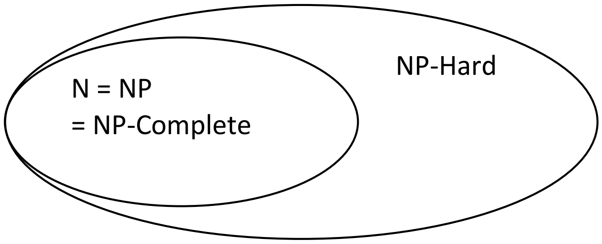
NOTE: Numerous Complexity classes.

- P: The complexity class P, which stands for polynomial, consists of problems that can be solved with known polynomial-time algorithms. In other words, for any problem in the class P, an algorithm of time complexity O(n^k) exists, where k is a constant. Deterministic.
- NP: The nondeterministic polynomial or NP complexity class involves the concept of a nondeterministic computer. Every problem in P is also contained in NP, because deterministic calculations can be emulated on a nondeterministic machine. So P is a subset of NP.
- NP-hard: For many computationally hard problems, the best algorithms known so far have exponential time complexity. Class of problems which are at least as hard as the hardest problems in NP.
- NP-complete: Class of decision problems which contains the hardest problems in NP.

P: Polynomial Time + Deterministic
NP: Polynomial Time + Nondeterministic
NP-Complete: NP-Hard problem solved using nondeterministism and converted to polynomial time.



P ≠ NP



P = NP = NP-Complete