**P, NP-Complete, NP, and NP-Hard**

NP problems have their own significance in programming, but the discussion becomes quite importants when we deal with differences between NP, P , NP-Complete and NP-hard.

P and NP- Many of us know the difference between them.

**P**- Polynomial time **solving** . Problems which can be solved in polynomial time, which take time like O(n), O(n2), O(n3). Eg: finding maximum element in an array or to check whether a string is palindrome or not. so there are many problems which can be solved in polynomial time.

**NP**- Non deterministic Polynomial time solving. Problem which can't be solved in polynomial time like TSP( travelling salesman problem) or An easy example of this is subset sum: given a set of numbers, does there exist a subset whose sum is zero?.

but NP problems are **checkable** in polynomial time means that given a solution of a problem , we can check that whether the solution is correct or not in polynomial time.

So till now you have got what is NP and what is P.

Now we will discuss about NP-Complete and NP-hard.

but first we need to know what is **reducibility** .

Take two problems A and B both are NP problems.

**Reducibility**- If we can convert one instance of a problem A into problem B (NP problem) then it means that A is reducible to B.

**NP-hard**-- Now suppose we found that A is reducible to B, then it means that B is at least as hard as A.

**NP-Complete** -- The group of problems which are both in NP and NP-hard are known as NP-Complete problem.

Now suppose we have a NP-Complete problem R and it is reducible to Q then Q is at least as hard as R and since R is an NP-hard problem. therefore Q will also be at least NP-hard , it may be NP-complete also.

