P, NP

- P, NP and the Computational Complexity Zoo
- P vs NP
- A problem is in class `P` if its solution may be **found** in polynomial time.
- A problem is in class `NP` if its solution may be **verified** in polynomial time.
- A problem in `P` is in `NP` by definition, but the converse may not be the case.
- `NP`-complete is a family of `NP` problems for which you know that if one of them has a polynomial solution then everyone of them has.
    - The most notable characteristic of `NP`-complete problems is that no polynomial-time algorithms are known for solving them (although they can be verified in polynomial time).
- Once you've reduced a problem to `NP`-complete, you know to give up on an efficient fast algorithm and to start looking at approximations.
- The `P` vs `NP` approach in Russian Math circle was to check whether a given problem can be solved without exploring (almost) all the possibilities.
    - `Q1`: Does there exist `x` in set `{1, 2, ..., 1000}` such that `x * x = 300`?
    - `Q2`: Do there exist `x, y, z` in set `{1, 2, ..., 10}` such that `x * y * z + x * y^3 * z + 17 * x * y * z^2 = 4564`?
    - `Q3`: Do there exist `a, b, c` in set `{1, 2, ..., 10}` such that `17 * a * b * c^2 + a * b * c + a * b^3 * c = 4564`?
        - `Q1` is in `P` (and also in `NP`) and `Q2` and `Q3` are in `NP` (but may not be in `P`). `Q2` / `Q3` is in `NP`-complete as once you solve that, `Q3`/`Q2` is easy.
- `NP`-hard
    - A lot of times you can solve a problem by reducing it to a different problem. I can reduce Problem `B` to Problem `A` if given a solution to Problem `A`, I can easily construct a solution to Problem `B`. (In this case, "easily" means "in polynomial time".)
    - If a problem is `NP`-hard, this means I can reduce any problem in `NP` to that problem. This means if I can solve that problem, I can easily solve any problem in `NP`. If we could solve an `NP`-hard problem in polynomial time, this would prove `P = NP`.
    - The Travelling salesman problem is a classic example. Suppose you wanted to visit some cities, but only wanted to visit each city once. What's the shortest path (order of cities) that allows you to do so? This question is `NP`-hard, but it has a `NP` brother - if someone gives you such a path, is there a shorter path that allows to visit each city once? Now, if someone gives you a solution to the second question, it is easy to check that it is right - all you have to do is calculate the distances and compare. So the second problem is `NP`, but the first is `NP`-hard.

Problem Type	Verifiable in P time	Solvable in P time
`P`	Yes	Yes
`NP`	Yes	Yes or No *[1]*
`NP`-complete	Yes	Unknown
`NP`-hard	Yes or No *[2]*	Unknown *[3]*

Note

- *[1]*: An NP problem that is also P is solvable in polynomial time.
- *[2]*: An NP-hard problem that is also NP-complete is verifiable in polynomial time.
- *[3]*: NP-complete problems (all of which form a subset of NP-hard) might be. The rest of NP hard is not.