What does it mean to be NP-Complete?

To explain this first I want to explain what it means to be P, NP, and NP-Hard so that I can give a more thorough explanation.

- For a problem to be P that means that it has a deterministic polynomial runtime algorithm.
- For a problem to be NP that means that it has a non-deterministic polynomial runtime algorithm. This effectively means that there exists an algorithm that contains non-deterministic steps that if solved would create a deterministic polynomial runtime algorithm.
- For a problem to be NP-Hard the problem needs to be as difficult as every NP problem OR it needs to be reduced to an already NP-Hard problem.
- For a problem to be NP-Complete it needs to be both NP-Hard and have a proven non-deterministic polynomial runtime algorithm, such that if the non-deterministic parts of the algorithm were solved all NP problems would be solved.

Describe why or how you know that your favorite NP complete problem is actually NP complete

I am going to be describing why the 1/0 knapsack problem is NP-complete and I will be doing this by first showing that it is NP-Hard by reducing it to the SAT problem. I will then show it is NP-complete by showing the non-deterministic polynomial runtime algorithm.

- The 1/0 knapsack problem can be reduced to the SAT problem because both problems can be shown as an Identical binary state tree.
- *The 0–1 knapsack decision problem:
    - Given a set of N objects.
    - Each object O has a specified weight and a specified value.
    - Given a capacity, which is the maximum total weight of the Complexity of Problems knapsack, and a quota, which is the minimum total value that one wants to get.
    - The 0–1 knapsack decision problem consists in finding a subset of the objects whose total weight is at most equal to the capacity and whose total value is at least equal to the specified quota.

*Algorithm collected from
http://webpages.iust.ac.ir/yaghini/Courses/RTP_882/Complexity%20Theory_02.pdf

Other research, https://youtu.be/e2cF8a5aAhE