The following is a case study (and my solution) from the course on Logic and Computational Thinking offered by Microsoft through edX. I strongly propose it to anyone interested in getting into programming!

Inductive Reasoning: A Case Study

Read

To begin read the article "How to Help Self-Driving Cars Make Ethical Decisions" by Will Knight on the *MIT Technology Review*. This article will present the scenario you will analyze using the tools from this lesson and other parts of the course.

The Problem Statement

You will use parts of the entire article to help you formulate the exact problem to solve but focus on this particular scenario in the article:

“When you ask a car to make a decision, you have an ethical dilemma,” says Adriano Alessandrini, a researcher working on automated vehicles at the University de Roma La Sapienza, in Italy. “You might see something in your path, and you decide to change lanes, and as you do, something else is in that lane. So, this is an ethical dilemma.”

The Exercise

To practice applying the skills taught in this lesson and throughout the course, do the following:

1. Clarify the problem statement. What problem or challenge is the researcher attempting to solve? Can you put the dilemma in the form of an inductive or deductive argument so you can figure out which premise or premises to test? Can you reword the scenario so it makes more sense to you?
2. Create an analogy. Come up with an analogy that you can use to clarify the problem. Start with "This problem is like . . ." and fill in the blank with something the dilemma is similar to. Describe why you think the analogy works. If you have trouble coming up with analogy, you may not fully understand the problem and may need to spend more time thinking about it.
3. Create scenarios that describe a solution and how to test those solutions.

- Now that you have clarity about what the problem is, how would you solve this using software? You don't need to write code of course but describe what the software might do. For example, you might talk about creating software that, "When the car senses two objects its path and cannot go in either direction, the car attempts to come to a complete stop." Of course, that's one option but you also have to account for scenarios where that option isn't available and what might happen to the driver if the car came to a sudden stop from high speed (in other words, does your solution create a new problem?).

- Use Enumerative Induction to further describe your scenarios. How many tests would you need to determine that the scenario solves the dilemma. For example, how many times would the car coming to a complete stop and not hitting either object show that the objects are "safe"?
- Use the confirmation and disconfirmation method to test each scenario. For example:

  Taking the scenario above, you can apply the confirmation method to it like this:

  1. If the car coming to a complete stop would solve the dilemma is true, then neither object in either lane is struck by the car should be observed.
  2. Neither object is struck by the car when it comes to a complete stop is observed.
  3. Therefore, the car coming to a complete stop would solves dilemma is probably true.

  You can also apply the disconfirmation method in the same way.