PS : Differentiated Learning 9-12

K-8 We encourage K-8 teachers to read at least the Turing Test discussion and then complete the subsequent engagement activity on the next page; many K-8 students find this to be a fun activity .

9-12 9-12 teachers are expected to read all discussions and then complete the subsequent engagement activity on the next page.

Problem Solving: The 9-12 Deeper Dive

There is a strong theoretical side to Computer Science Problem Solving. This area intersects with mathematics, and includes some content and considerations students often find engaging. In this deeper dive, we consider a Turing Test, a re-visit to Polya's general problem solving strategies, and a reflection on what makes a problem solvable.

1. The Turing Test

Alan Turing was a man seemingly born out of time. He suffered significant persecution in his personal life, and ultimately died by cyanide poisoning under what some historians have labelled suspicious circumstances. He was astonishingly brilliant, making major contributions to mathematics, computer science, logic and code-breaking during the war effort. He is credited by many as developing the first work in Artificial Intelligence. Numerous articles, textbooks and movies have been made about Turing.

One of Turing's many musings was how to identify intelligent behavior. He proposed what is referred to as the "Turing Test" or "imitation game", in which both a human being and a computer respond to input and answer questions. If an independent evaluator, within a set time frame, can not tell which respondent is the human being and which is the computer, then the computer has demonstrated intelligent behavior.

Watch the following TedTalk about the Turing Test:

https://www.youtube.com/watch?v=3wLqsRLvV-c (https://www.youtube.com/watch?v=3wLqsRLvV-c)

(https://www.youtube.com/watch?v=3wLqsRLvV-c)

Next, have a short conversation with Eliza, a Rogerian therapist chatbot: **http://psych.fullerton.edu/mbirnbaum/psych101/Eliza.htm (http://psych.fullerton.edu/mbirnbaum/psych101/Eliza.htm)**

By the way, years before Turing, Ada Lovelace posed similar questions about artificial intelligence and creativity that you can explore here:

Can robots be creative? - Gil Weinberg (https://www.youtube.com/watch?v=Rh9vBczqMk0)

(https://www.youtube.com/watch?v=Rh9vBczqMk0)

These questions are an interesting look at determining where a line might be "drawable" between humans and the computing technology they create, and offer endless student engagement around AI, the social rights of an AI agent, and more.

2. Re-visiting general problem solving techniques

Let's formally identify three problem solving strategies:

- **Brute force enumeration** is an approach in which all possible solutions are considered. For example, if you know a solution to a math problem is a number between 3 and 6, in brute force enumeration, you would check 3, then 4, then 5, then 6. This approach may seem inelegant, but sometimes a plodding strategy can win the race.
- **Trial and error** is an approach similar to brute force enumeration. The "Secret Number Guessing Game" is an example of trial an error. Is the secret number higher than 20? Less than 30? Here you can try solution *ranges* first, and then use brute force enumeration to hone in on a final answer.
- A **heuristic** is a solution approach that is particularly useful when a problem is not well defined, or a more deliberate approach would take too much time, or a "close enough" estimate will suffice. Heuristics can be contrasted with algorithms. While often successful, heuristics are vulnerable to bias, such as representativeness, base rates, availability, etc. The "travelling salesman problem" ("TSP") is a famous computer science problem that can be approached from various solution perspectives, including heuristics, which you can read about here: **here (https://www.khanacademy.org/computing/ap-computer-science-principles/algorithms-101/solving-hard-problems/a/using-heuristics)** in Kahn Academy's free course support for the Computer Science Principles.

3. Is a problem solvable?

A lot of attention has been given to the solvability of problems. After all, if we knew in advance that a problem couldn't be solved exactly, or solved in a "reasonable" amount of time, we could go ahead and adopt other strategies, such as heuristics, to at least approximate a solution. Or, from a different perspective, we could feel that a "computationally hard" to crack password strategy provides reasonable protection from hackers. This branch of computational theory considers problem solutions from a theoretical perspective, using the mathematical vocabulary of something called NP problems. You can read about NP problems **here** **(http://news.mit.edu/2009/explainer-pnp)** and then watch these short **Kahn academy tutorials on NP** **(https://www.khanacademy.org/partner-content/bjc/2018-challenge/2018-challenge-mathematics/v/breakthrough-junior-challenge-2018-the-wizarding-world-of-mathematics-explores-the-p-vs-np-concept)**. (Note that the College Board's Computer Science Principles course includes high level treatment of the NP concept.)

K-8 K-8 teachers are encouraged to complete the Turing Test engagement project on the next page.

9 12 9-12 teachers should complete the Turing Test project on the next page to acquire some hands on experience with these advanced ideas.