Computational Thinking for Educators

Google online course

Introducing Computational Thinking

After reading this section, brainstorm some ideas into the yellow fields of the table below based on what you are currently teaching.

Computational Thinking Concept	Subject Area Application (brainstorm one way to include each CT concept in any area you currently teach)
Example: *Break a problem into parts or steps*	*Example:* *Kindergarteners who want to build a fort at recess sort their materials out by type so they can see what they need more of (plywood here, long branches there, heavy rocks in this area, etc.).*
Break a problem into parts or steps	High School art students would need to do this all the time. If they were working on a multi-media piece, they would have to strategize what they would do first to act as the background and then build upon it after scrutinizing the best way to stabilize the piece and connect the various elements.
Recognize and find patterns or trends	Art is all about patterns, and trends. Both cultural trends throughout history and how they have either affected the art making or been the instigator of a trend. Visual patterns are very important in artwork and have a long history throughout time.
Develop instructions to solve a problem or steps for a task	If you were to make a repeating pattern, by hand, in order to create wallpaper, you would split the project up into a few simple steps. 1. Draw multiple graphics or pictures on a sheet of paper, careful not to touch the edges, in black and white. 2. When complete, cut the image in half. Flip the two sides over and then tape them back together in reverse. 3. Cut the image one more time in half the other direction, and also switch those two sides. You should now have your images bleeding off of the edges of the paper. 4. Fill in the rest of the papers blank space, without touching the edges. 5. You can now print off multiple copies of the image and assemble them in order to make a large wall of reoccurring wallpaper.
Generalize patterns and trends into rules, principles, or insights	The best patterns tend to be simple and clear (both in visual art and in computational thinking). Trends are reoccurring in history and across cultures. The best ideas are just mash-ups of other ideas that already exist out there.

Exploring Algorithms

Explore Traveling (Computing), Words Over Time (Humanities), Cellular Automata (Mathematics), and Genomics (Science). In two or three sentences below, tell what you are passionate about *and* where you see possibilities for algorithms to provide new ideas and opportunities to learn and explore. This does not have to be related to your classroom, just apply the concept generally, in life, to something you wonder about. I was really interested in the traveling activity because it is so practical in everyday life. While planning a trip, you don't even realize that you are naturally attempting to do find the most direct path in order to save money and time resources, but putting a label on it is really interesting. Words Over Time is really applicable to teaching Theater, or even studying art/cultural trends and potentially forecasting what is up and coming. For example, I typed in "YOLO," which is a really stupid slang word being used currently (meaning You Only Live Once). According to the graph, this word was very popular in the early 1800s and led me down a google search rabbit hole. I think this would be really fun to use with students when teaching Shakespeare and his influence over language (for example).

Finding Patterns

Experiment with the activities in this area (Data Compression, Music, Turtle Geometry, and Classification). As you experiment, try to figure out in what ways you are using decomposition, pattern recognition, abstraction/generalization, and algorithm design (write your thinking in the fields below). In the Findings Patterns (Lesson) you'll be able to confirm your predictions. Strikethrough (~~like this~~) any incorrect predictions and write the actual application after it following a semicolon (~~like this~~; actually, like this).

Data Compression

Decomposition: *The painting can be broken down into color, form, sharpness and light/shadow.*
Pattern Recognition: *Compare and contrast two paintings in order to find out what differs between them regarding bitmask and bytes.*
Abstraction/Generalization: *The computer program will be very similar for both paintings, with the difference being in a small part of the code specifying the paintings colors (for example).*
Algorithm Design: *Understanding how this data works would be very beneficial if approaching Photoshop or needing to alter an image for resolution quality.*

Music

Decomposition: *Music can be broken down into notes, meter, and A B A B form (for example).*
Pattern Recognition: *Patterns are key to music and easily recognizable in their meter, melody, song form and instrumentation. Music rarely works without some pattern or intentional interruption of a pattern.*
Abstraction/Generalization: *It would be easy to alter the song by simply changing it from a major to a minor key. The mood would greatly differ. It also would if you changed the speed.*
Algorithm Design: *Music is very mathematical and most sheet music requires one to become familiar with key symbols in order to sight read a piece. I'm sure that using a computer program to play the music is similar in it's use of repeating symbols (or stanzas in music).*

Turtle Geometry

Decomposition: *We learned how to simplify instructions by adding a repeat in the CODE quest earlier. I'm guessing this will do the same thing.*
Pattern Recognition: *Even without understanding the symbols full meaning, you can repeat a pattern after analyzing what it does already based off of those symbols instructions.*
Abstraction/Generalization: *With little familiarity of math or geometry, one can still use common sense to decode most of these problems. We are instructing the turtle to build the house.*
Algorithm Design: *It's helpful to know 90 degree angles and 180 degrees. Simple math will then allow you to break those angles into thirds, or even complete a full circle.*

Classification

Decomposition: I answered that it would take about ~~20 questions~~ (it was really 32) in order to confidently guess any species on Earth. I have no idea if that's right or not, but used the number based off of their example of the game 20 questions.

Pattern Recognition: This requires understanding basic principles of dividing or doubling a number.

Abstraction/Generalization: You can scientifically guess how many questions you would need in order to answer almost any problem. By dividing one number in half, you would realize how many questions were necessary to solve the question. There was a pattern to it and it was presented in a fun card game.

Algorithm Design: My math skills are lacking, but I could solve the problems while the numbers were small. I got overwhelmed when it asked to continually divide 8 billion down to the fewest amount of questions necessary. I guessed ~~20 once again, but really gave up about 5 minutes into the problem~~. The answer was about 32 instead. Still pretty remarkably small!

Developing Algorithms

Were you able to finish each of the algorithms successfully? What was difficult for you? Did you find that you became more capable over time, more frustrated, or something else? How might your students approach these challenges differently? I liked stacking the wooden blocks on the pins, and was able to move all three of the circles in 7 moves, but then I didn't reread the question that was using 64 as an example. This is where I fail at computational thinking…I move too quickly and don't tend to double check things. This characteristic flaw influences my recipe reading poorly as well. It is beneficial to read through the WHOLE problem first and then begin your task. This reminded me of playing solitaire. It's fun! I was able to predict the number of times you would need to do the puzzle correctly, but I did get frustrated following through the full puzzle myself when they got bigger.

In the first section, you reflected on areas of your curriculum where you could see the opportunity to apply algorithms (in the yellow fields of the table above). Before moving on to the final project, are there any new topics that you could see students learning more through the application or computational thinking and the development of algorithms? Add to your thinking below…

Computational Thinking Concept	Subject Area Application (brainstorm one way to include each CT concept in any area you currently teach)
Break a problem into parts or steps	*Study and create a line drawing from a figure model*
Recognize and find patterns or trends	The best design is the cleanest and therefore, the simplest of choices.
Develop instructions to solve a problem or steps for a task	1. Study the portrait model for at least a minute or two prior to drawing. Have pencils sharpened and at your side. 2. Break the face down into pronounced characteristics particular to your model. 3. Begin drawing from one corner of the face, either at the ear, an eye, hairline or neck and without picking up your pencil, continually follow the contour and details of the face in the most efficient manner possible.
Generalize patterns and trends into rules, principles, or insights	Analyze the angles and the length of each aspect of the body. What position is the model in? What angle is their torso? Their head? What is the length and texture of their hair? How symmetrical are their facial features? Moving slowly produces better, or more accurate results when trying to achieve a clean line drawing that resembles the figure model.

Applying Computational Thinking

In one or two sentences: How does computational thinking apply to your domain or subject area? There are a great many ties to both Visual Arts and Theatre Arts. In visual arts, it's easy to see how patterns and trends can hold significant value in the design process. Art has a fascinating cultural history as well through the lens of trending cycles and repeats (for example: look at the 30-year cycle of the fashion industry that continually borrows from the decades before them). I can see it applying when organizing a studio space or your materials before diving into a project. Art can be time sensitive and if you're attempting to glue a piece together and are not prepared with your materials beforehand, your piece can be ruined.

As far as theatre, I can see the benefit of computational thinking in teaching character analysis. Portraying different characters is often based on "stock characters" or archetypes that exist. Commedia dell'arte is a form of theatre characterized by masked "types" which began in Italy in the 16th century and involved improvised performances based on sketches or scenarios. The art of improvisation is really based on computational thinking in the way that it predicts outcomes based on past experience, usually resulting in sophisticated comedy. The best actors store so much profile information in their brain regarding humanistic trends and stories.

Choose to apply CT in either of these ways (or create a new way of your own!) and attach your final product in the submission area of 3D GameLab:

1. Curriculum plan integrating at least one computational thinking concept (decomposition, algorithms, pattern recognition, abstraction, etc.)

    a. Lesson Plan (road map of what students needs to learn and how it will be effectively implemented during the class, lecture, or period of time; it describes the title, keywords, objectives, activities, and assessments during class time)
    b. Assessment (an activity, test, or project that evaluates or measures if students met the objectives and to what degree)
    c. Lesson Activity (one action or a group of actions done by students with a goal of supporting lesson objectives)
    d. Project (a piece of work in which students gain skills by working for a specified time period to solve a problem, investigate, or respond to a complex challenge, problem or task)
    e. Unit Plan (or Module plan; a group of related lessons, activities, assessments, and projects that share a common larger objective, usually completed over the period of multiple days to multiple weeks)

2. Communication plan for your students, parents, or school community explaining

    a. what computational thinking is
    b. why computational thinking is important, and
    c. how and where you will begin integrating it (e.g. subject, term, curriculum, project, activity, lesson, project, after-school curricula, service project, module, etc