Teacher's Name: Ticey Little

Domain: Exploring Computer Science

Date Range: February 03, 2025 – February 07, 2025

ACOS Standard:

4 - Use and adapt classic algorithms to solve computational problems.
31 - Create interactive data visualizations using software tools to help others understand real-world phenomena.
39 - Identify a problem that cannot be solved by either humans or machines alone and discuss a solution for it by decomposing the task into sub-problems suited for a human or machine to accomplish.

Student Friendly Outcome:

I CAN describe the linear search algorithm.
I CAN describe the binary search algorithm.
I CAN explain conditions under which each search might be appropriate.
I CAN explore the consequences of search algorithms.
I CAN define sorted and unsorted lists.
I CAN describe various sorting algorithms.
I CAN compare various sorting algorithms.
I CAN solve a minimal spanning tree.
I CAN draw a node edge graph to solve a problem.
I CAN reinforce my problem-solving strategies and solutions from one context to another.
I CAN name the basic terms used in Scratch.
I CAN create the beginning of a simple program in Scratch.

Monday	Tuesday	Wednesday	Thursday	Friday
Continued from last week Tower building algorithm (20 minutes) Model Linear and Binary searches (15 minutes) Comparison of linear and binary search (10 minutes) Journal entry (10 minutes)	Continued from last week Day 1 Journal entry (10 minutes) Students understand the problem. ECS Problem solving Step 1. (10 minutes) Students develop a strategy. ECS Problem solving Step 2. (10 minutes) Students execute their strategy and refine it. ECS Problem solving Step 3. (15 minutes)	Continued from last week Students repeat the problem-solving process either using more elements or at minimum follow the unplugged activity using another strategy. (20 minutes) Students present their sorting algorithms to the class. (15 minutes) Journal entry (20 minutes)	Continued from last week Journal entry (15 minutes.) CS Unplugged: The Muddy City (15 minutes) CS Unplugged: The Muddy City—extension (15 minutes.) Journal entry (10 minutes)	Day 1 Journal entry (10) Complete Scratch KWL chart (15) KWL Chart discussion (10) Journal entry (10) Name discussion (10)

	Students reflect upon their outcomes and reflect. ECS Problem solving Step 3. (10 minutes)			

Instructional Lesson # 6. Day 11

Topic Description: Apply the problem-solving process. Algorithms and algorithm bias.

Objectives

The student will be able to:

- Describe the linear search algorithm.
- Describe the binary search algorithm.
- Explain conditions under which each search might be appropriate.
- Explore the consequences of search algorithms.

Outline of the Lesson

Segment	Reason/Purpose
Tower building algorithm (20 minutes) Model Linear and Binary searches (15 minutes) Comparison of linear and binary search (10 minutes) Journal entry (10 minutes)	Hand on problem-solving using physical models Connecting Tower problem to algorithms Connecting tower problem to searches Understanding the differences and limitations of solving problems with computers. Reinforce the part reflection of the problem-solving process. Develop the transitive process of problem solving that has different contexts.

Student Activities

- In pairs complete the Tower Building Activity. Have students complete the Tower Building Activity with their elbow partner and write their solutions in their journals.
- Model the tower building algorithm.
- Share their solutions with another elbow partner pair.
- Participate in the activity modeling linear and binary searches and their comparison.
- Complete Journal entry.

Teaching and Learning Strategies

- The Tower Building Problem
    - Teacher Introduces the Tower Building Problem.
        - Students in pairs use the four steps of the problem-solving process to create an algorithm for the solution to the given problem.
        - Have students use the physical model strategy for solving the problem.
    - Have students share their algorithms to the Tower Building Problem

    - Ask students how could they have solved the Tower problem another way using other strategies?

  Note: The solution is to start by taking half of the height of the tower and create that number of stacks of two. Continue halving the number of stacks and doubling the height (plus one stack of any remainder) until the desired height is reached. This foreshadows binary search.

- Journal entry: *What has been your experience when searching for specific information on the internet (e.g., cafe near me, what is computer science)? How does the search engine provide results for your search? How do you think computers search information?*
- Linear and Binary Searches Exercise
  - Linear—start at the beginning, look at each item until you find it or there is no more data. Data can be sorted or not.
  - Binary—look at the middle item, eliminate the half where the value is not located. Find the new middle element and continue the process until you find it, or there is no more data. Ask students to describe what is necessary in order to use a binary search—the list must be sorted.
  - Have students provide examples of where each type of search is appropriate and why. When the data is very large, which search type would be better?
  - Note that decisions often need to be made about whether to maintain lists in sorted order, provide an option for sorting should it be necessary, etc. based on the types of searches that are expected to be performed on the data.
- Journal Entry**:** *Is the computer doing searches like linear or binary search? Search engines use all types of algorithms to find information for people. If the algorithm includes personal data, previous searches, previous purchases, etc. how could this affect the results of a search? Should the person searching know what parameters are being used for their search?*

Resources

- Tower Building Activity Instructions
- Some type of tower manipulatives such as lego, building blocks, wood tiles, or a square taffy candy similar to Starburst.
- At least two dictionaries.

Teacher Reflection Notes

Tower Building Activity

A construction company wants to build a 100-meter-high tower as quickly as possible. The company has unlimited resources and an unlimited budget and is willing to spend any amount to get the job done.

The tower is to be built with blocks that are 100 meters long and 100 meters wide, but only 1 meter tall. The blocks interlock on top and bottom (like legos). They cannot be stacked sideways.

Using special lifters, putting one stack on top of another stack takes one week regardless of how high the stacks are.

What is the shortest amount of time that it will take to build the tower?

Suggestions:

- Use something like legos or a graph to help solve this problem.
- Start with a smaller tower of 5 or 10—solve a smaller problem.
- Extend that knowledge to the larger problem.

Instructional Lesson # 7. Days 12-13

Topic Description: Explore sorted and unsorted lists and various sorting algorithms. Reinforce the problem-solving process. Privacy concerns. There are some problems that cannot be solved by computers. Compare algorithms (solutions) Why are we so interested in searching and sorting?

Objectives

The student will be able to:

- Define sorted and unsorted lists.
- Describe various sorting algorithms.
- Compare various sorting algorithms.

Outline of the Lesson

Segment	**Reason/Purpose**
Day 1 Journal entry (10 minutes) Students understand the problem. ECS Problem solving Step 1. (10 minutes) Students develop a strategy. ECS Problem solving Step 2. (10 minutes) Students execute their strategy and refine it. ECS Problem solving Step 3. (15 minutes) Students reflect upon their outcomes and reflect. ECS Problem solving Step 3. (10 minutes)	Connect previous knowledge. Clarify constraints of the problem. Constraints are difficult for students to understand.
Day 2 Students repeat the problem-solving process either using more elements or at minimum follow the unplugged activity using another strategy. (20 minutes) Students present their sorting algorithms to the class. (15 minutes) Journal entry (20 minutes)	Exploration of AlgorithmExplore additional Algorithms. There is not only one way. Emphasizes the importance of reflection in the problem-solving process. Consider limitations of algorithms. Students start to think about loops.

Student Activities

Day 1

- Complete journal entry.
- Lightest and Heaviest activity.

- Complete journal entry

Day 2

- CS Unplugged: Lightest and Heaviest—Alternate method
- Complete journal entry.

Teaching and Learning Strategies

- Journal Entry: *What is your experience using sorted lists of information on your phones or computers? (e.g., a list of songs on Spotify, list of videos on YouTube or Netflix, etc.) Why does it matter to sort the items you listed?*
    - Teacher facilitates journal entry discussions of lists of sets of things and data. Why is sorting important? What are the different ways we may sort things? What attributes of the things in the set you are ordering that one wants to use? Are there several attributes you might be interested in when sorting?
- CS Unplugged: Lightest and Heaviest
    - This activity can be downloaded from http://exploringcs.org/curriculum. Download and unzip CSUnplugged files, then open. Note there are many additional resources listed that you may wish to explore.
    - It will be helpful to read through the entire activity in advance, so that you can revise questions, add your own questions, and think about how you might want to structure each part of the activity. The goal is for students to be actively involved in some way and for all students to be able to describe the various types of sorting.
        - Each group has a set of elements. The first round could have four elements. Each student would be responsible for creating strategies and their group's algorithm. One student could be the code writer and another student could be the code verifier that makes sure the code is understandable to others outside of their group. Another student could be the evaluator that compares two elements. And the last student could be the evaluator verifier to make sure that the evaluator and evaluator verifier agree on the results of the comparison of two elements.
        - There are many possible ways to make the weighted elements. One would be to use bags with varying numbers of pieces of candy. Or you could use takeout boxes and fill them with sand. If you don't have balance scales, you can help students come up with a strategy that will simulate a scale. For example, if the weights are clearly different in weight, the students could do this by feel. The students could be divided into groups of four with different jobs.
    - Lead discussion of what the problem is and the constraints of the problem. Emphasize that the computer does not remember results of a comparison once the comparison is made. Therefore, their instructions need to have the computer do something or not according to what the results of the comparison is. Also emphasize that the computer

only knows locations of elements and not their value at any time. Field students for any questions or clarifications.
    - Have students develop a strategy for solving the sorting problem. Prompt students about things to think about when developing their strategy. Such as how will the algorithm "know" when it is done and all the elements are sorted? What details need to be in the algorithm instructions so that someone else may follow the instructions without asking questions? Commands for the algorithm may be: evaluate element # to element # with a conditional statement of an action command. Action Commands: swap, place at # position, place at beginning, place at end, copy list, create empty list, copy element to another named list.
    - Teacher circulates to different groups while students carry out their plans and asks questions about the algorithm they are creating. When students feel they are finished the teacher could follow their instructions, read to them or have other groups follow their instructions to see how well their algorithm works. Once students have a working algorithm, the teacher has each student confer with their group and write their own reflection of the process.
- Class Discussion
    - Teachers facilitate students discussing other ways of sorting that were not explored by the students. Teachers may point out other ways of sorting that were not explored by the students. .
- Journal Entry: *Reflect on your group's strategies algorithm. How did your algorithm know when to stop? What was the attribute you used to sort? How many reiterations did your group have to do before you had an algorithm that worked?*
    - Teacher facilitates journal entry discussions of algorithms and reflection of the entire process and the importance of reiterations of the process.
- CS Unplugged: Lightest and Heaviest—Alternate method
    - Teacher has students refine their working algorithm with eight elements or create another method. Teacher ensures that the students repeat the entire ECS problem-solving process taking their reflections into consideration.
- Journal Entry: *How does the size of the data being sorted affect the sorting process? How do you compare these sorting criteria to your experiences of sorting on different platforms such as Netflix, Google search, Spotify? What are the effects of sorting and searching on our society when the process and attributes are not shared with the user? Energy is needed to keep computers running. What are the environmental costs of sorting and searching large data sets?*

Resources

- Bell, Tim, Ian Witten and Mike Fellows. Computer Science Unplugged, New Zealand: 2002. Computer Science Unplugged Activity 7: Lightest and Heaviest—Sorting Algorithms, pp. 64–70
- Containers of the same size with different weights. Or takeout boxes with different amounts of sand in bags.
- Balance scales or have a group of four students. Have student one a code writer, student two a code verifier, student three a evaluator, and have student four a evaluator verifier. All four students create their sorting algorithm.

Teacher Reflection Notes

Instructional Lesson # 8. Day 14

Topic Description: Introduces minimal spanning trees and how graphs can be used to help solve problems.

Objectives

The student will be able to:

- Solve a minimal spanning tree.
- Draw a node edge graph to solve a problem.
- Students reinforce their problem-solving strategies and solutions from one context to another.

Outline of the Lesson

Segment	Reason/Purpose
Journal entry (15 minutes.) CS Unplugged: The Muddy City (15 minutes) CS Unplugged: The Muddy City— extension (15 minutes.) Journal entry (10 minutes)	Students apply the problem-solving process to The Muddy City. Students explore different representations of their information and their advantages. Students reflect upon their process and representations. Students develop transitive applications of problem solving with different contexts. Some problems are not easily solved by computers.

Student Activities

- Complete journal entry.
- Work in groups to solve The Muddy City
- Students share their solutions with the class.
- Reiterate the problem-solving process using the edge node graph to find other solutions that are better than their previous solution.
- Journal Entry

Teaching and Learning Strategies

- Journal Entry: *Many Problems have different contexts, but the problems are similar regarding constraints and types of solutions. How could the following scenarios be similar? Shaking hands with a group of people and connecting computers together as a network? How are the two situations different?*
    - Teacher leads student discussion about their journal entries making the connections between the two seemingly different problems. Teacher helps students see the possible differences of the two different contexts of the problems and their constraints. Teacher

then makes the connections that there are other problems that may not seem similar, but have connections that are helpful to solve more problems.

- CS Unplugged: The Muddy City
    - This activity can be downloaded from http://exploringcs.org/curriculum. Download and unzip CSUnplugged files, then open. Note there are many additional resources listed that you may wish to explore.
    - It will be helpful to read through the entire activity in advance, so that you can revise questions, add your own questions, and think about how you might want to structure each part of the activity. The goal is for students to be actively involved in some way and for all students to be able to describe shortest path strategies. What follows is the minimal suggestion. Follow the directions in The Muddy City Problem on p. 78. Introduce the Muddy City Problem.
    - Have students work with their elbow partners. Teacher ensures that students use the four steps of the problem-solving process.
    - Have students share their solutions and lead the follow-up discussion p.77. Students have a class discussion on their reflections of their solutions.
- CS Unplugged: The Muddy City extension
    - If the node edge graph is not presented, then the teacher suggests other ways to represent the problem and find solutions using a node edge graph. Have students repeat the Muddy City Problem with the abstract representation in the figure on p. 79 and answer the questions on p 79. U
    - Discuss the students' reflections of the process and reiterations. How do you know you have the best solution? Is there an algorithm to give us the best answer or is it only a strategy that we are using? Can computers solve such problems?
    - Discuss various applications of this problem in the industry sector of Computer Science (p. 80). Why do we have different values for different paths? How does this relate to networking computers? What resources of material and energy are needed for networking? Given that different paths have different values and determine certain paths being used, how might that affect different groups of people depending on where they live?
    - Emphasize the idea of the shortest path and how some representations may be more advantageous than others for a given context.
- Journal Entry: *Is there a formula we could use to figure this problem out? What other ways could we represent the paths, houses, and stones? How does this problem relate to computer science?*
    - Discuss various applications of this problem in the industry sector of Computer Science (p. 80). Why do we have different values for different paths? What would it mean to apply the problem-solving strategies for a data or network situation? How does this relate to networking computers or digital divide issues? What resources of material and energy are needed for networking? Given that different paths have different values and determine certain paths being used, how might that affect different groups of people depending on where they live?

Resources

Computer Science Unplugged Activity 9: The Muddy City— Minimal Spanning Trees, pp.76–80. Materials

Teacher Reflection Notes

Instructional Lesson # 1. Days 1-4

Topic Description: This lesson introduces the Scratch programming language, including the basic terms utilized in the language, through student's names, identities and cultures.

Objectives:

The student will be able to:

- Name the basic terms used in Scratch and create the beginning of a simple program in Scratch
- Build upon previous units to explore the concept of identity through personalized projects and demonstrate how programming can be used to express and explore cultural and personal identity.
- Revise name projects responding to comments & feedback (Iterative process of programming)

Outline of the Lesson

Segment	Reason/Purpose
Day 1 Journal entry (10) Complete Scratch KWL chart (15) KWL Chart discussion (10) Journal entry (10) Name discussion (10)	Establish the different knowledge levels of Scratch Discuss names & how they represent/tie to a person's identity
Day 2 Journal entry (10) Investigate features of Scratch (25 min) Design name project (20 min)	Explore Scratch
Day 3 Complete name project (55 min)	Implement first program using Scratch features
Day 4 Gallery walk of name projects (15 minutes) Revise programs responding to comments/feedback (30 min) Journal entry (10 min)	Give feedback about name projects and how accurately they reflect identities Learn the iterative process of programming by revising programs in response to feedback

Student Activities

Day 1

- Complete journal entry
- Complete Scratch KWL

- KWL Chart Discussion
- Complete journal entry
- Name discussion

Days 2

- Complete journal entry
- Investigate features of Scratch
- Design name project

Day 3

- Complete name project

Day 4

- Gallery walk of name projects
- Revise programs responding to comments/feedback
- Journal entry

Teaching/Learning Strategies:

Day 1

- Journal Entry**:** *Reflect on the various themes you've encountered in this class so far- community, identity and technology- how do those themes help us understand our place in the world?*
    - Explain that Scratch is a block-based programming language. Students will learn the different features of Scratch to build increasingly complicated programs. Students will use the pair programming model to design, program & revise their programs. The themes of identity & community will be explored and represented in the programs created throughout the unit.
    - Discuss what it means to program a computer. Remind students that in Unit 1 they learned about following directions and in Unit 2 they learned about algorithms. Discuss the ways in which programming can overcome constraints or create additional ones. Remind them that in the previous unit they used a markup language to provide instructions to the computer on the layout and content of web pages. Programming languages are used to translate algorithms into a language that a computer can execute.
- Complete KWL chart
    - Students meet in groups of 2-3 and each group completes a KWL chart. (Know, Want to Learn, Learned) related to Scratch
- KWL chart discussion
    - Students take turns sharing out their K's and W's orally. Encourage them not to repeat anything that has already been said. Put KWL charts up in the classroom; tell students that they will finish the L towards the end of the unit.
    - Wrap up the discussion by previewing their first Scratch project and a conversation about the meaning of names.
        - You are going to create a Scratch program to animate the letters of your preferred name, to reflect the meaning of your name and how it relates to your identity. Remind students of their unit 1 discussion and website they created about identity.
        - What does the name Scratch suggest about how it might be used to create and explore?
        - Who makes up names and gives them meaning?

- PBS- Say It Loud- "Black Sounding" Names and Their Surprising History
    - Suggested Clips:
        - Introduction: 1:00-2:25
        - Middle Passage 2:54-4:22
        - Civil Rights & Islam: 4:31-5:56
        - Creole & French: 7:12-7:57
        - Creating new names: 7:58-8:35
        - How the hosts got their names: 8:48-9:48
        - Why is this important?: 10:52-11:56
- PBS- Origin of Everything- "Why Do We Have Middle Names?
        - Suggested Clips:
        - History of: 0:00-2:11
        - Paternal & Roman Names: 2:20-3:25
        - Not in the Middle: 3:36-4:04 (Korean middle names)
        - Not in the Middle: 4:10-4:43 (Spanish middle names)
        - MIddle Initial: 4:50-5:20
        - Why is it important?: 5:20-5:49
- Journal Entry: *What is the story of your given name (either first/middle/last name): Who chose the name? Why was this name chosen? What is your preferred name? Do you use your given name or another name (nickname,..)? Why/why not?*
    - Students elbow share, then share in groups of 4-5

Day 2

- Journal Entry**:** *Choose 2 letters from your preferred name. What would these 2 letters look like /do if you were to turn each into an animation that represents your identity & interests?*
    - Sample answers: Letter T will turn into a bike representing my favorite mode of transportation, Letter E will say: Hola, Hi, Salaam,...
    - Each student shares one letter, its animation and how it relates to the student's identity/name meaning.
    - Bring it all together- help students make the connection between names, visual representations, symbolism, and how those can be combined to create animations of their own names.
    - Additionally, talk about the word abstraction. Students will see it in later lessons and should become familiar with it.
    - Remind students that they are going to create a Scratch program to animate the letters of their preferred name, to reflect the meaning of their name and how it relates to their identity. Scratch has many features that they can use to create the animation.
- Investigate features of Scratch
    - Prior to lesson: Address how sound will be handled in the classroom.
        - Ensure that each student has access to a computer with Scratch cloud access or installed. (Check with IT and make sure not blocked. A couple weeks may be needed.
        - Scratch lends itself to playing sounds so it can get noisy. Headsets with microphones are one possible way to address it.
    - Begin the Scratch interface exploration by demonstrating how to navigate and use key features. This helps students feel more comfortable before engaging in hands-on activities.
    - Assign students in pairs. Instruct students that in pair programming one person is the "driver" and does the clicking and typing. The other person is the "navigator" and describes to the driver what to do at each step. Students should trade roles every 5–10 minutes. Keep track of the time