Unit 8 - Lesson 1
Project Planning

Computer Science A

🎯 Lesson Objectives

By the end of this lesson, you will be able to . . .

- Break down a large project into manageable chunks
- Identify project requirements, tasks, and priorities

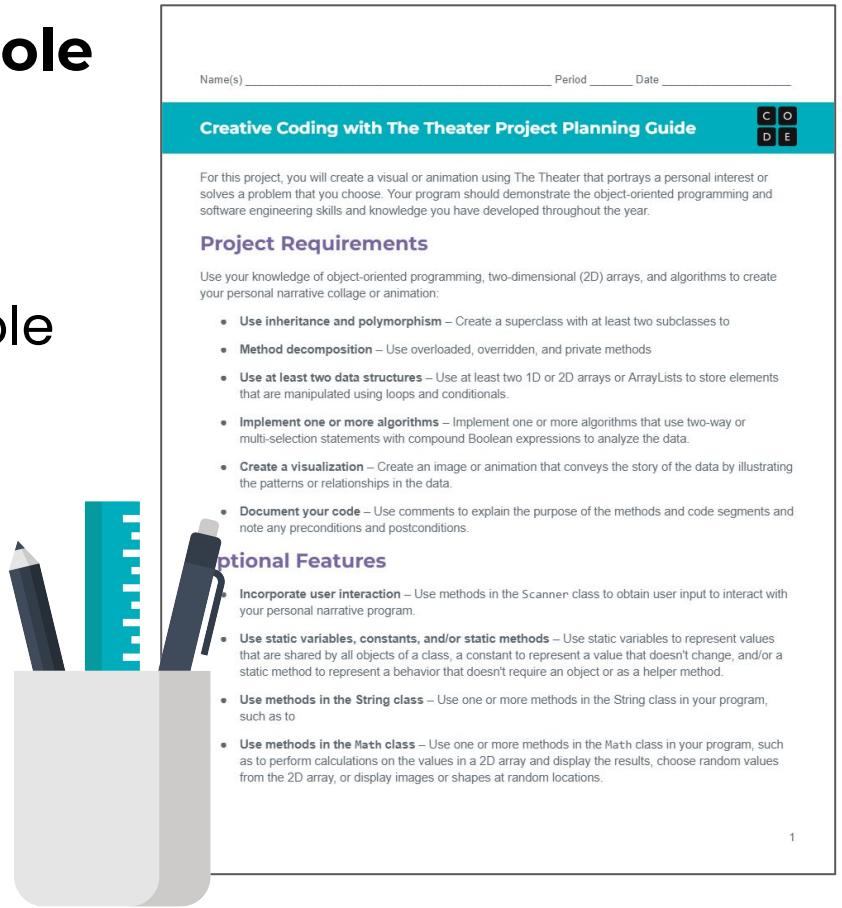
🚀 Question of the Day

Why is it important to start a project by first identifying requirements, tasks, and priorities?

Creative Coding with the Console Project Planning Guide

📍 You should have:

- Creative Coding with the Console Project Planning Guide
- pen / pencil



Name(s) _____ Period _____ Date _____

Creative Coding with The Theater Project Planning Guide

For this project, you will create a visual or animation using The Theater that portrays a personal interest or solves a problem that you choose. Your program should demonstrate the object-oriented programming and software engineering skills and knowledge you have developed throughout the year.

Project Requirements

Use your knowledge of object-oriented programming, two-dimensional (2D) arrays, and algorithms to create your personal narrative collage or animation:

- **Use inheritance and polymorphism** – Create a superclass with at least two subclasses to
- **Method decomposition** – Use overloaded, overridden, and private methods
- **Use at least two data structures** – Use at least two 1D or 2D arrays or ArrayLists to store elements that are manipulated using loops and conditionals.
- **Implement one or more algorithms** – Implement one or more algorithms that use two-way or multi-selection statements with compound Boolean expressions to analyze the data.
- **Create a visualization** – Create an image or animation that conveys the story of the data by illustrating the patterns or relationships in the data.
- **Document your code** – Use comments to explain the purpose of the methods and code segments and note any preconditions and postconditions.

ptional Features

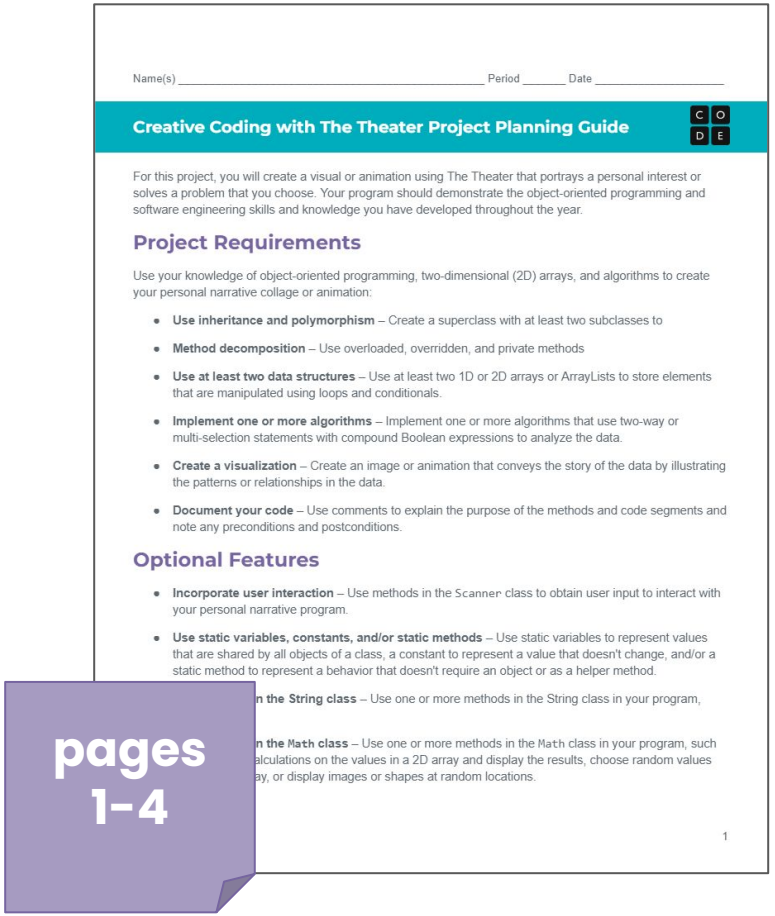
- **Incorporate user interaction** – Use methods in the `Scanner` class to obtain user input to interact with your personal narrative program.
- **Use static variables, constants, and/or static methods** – Use static variables to represent values that are shared by all objects of a class, a constant to represent a value that doesn't change, and/or a static method to represent a behavior that doesn't require an object or as a helper method.
- **Use methods in the String class** – Use one or more methods in the String class in your program, such as to
- **Use methods in the `Math` class** – Use one or more methods in the `Math` class in your program, such as to perform calculations on the values in a 2D array and display the results, choose random values from the 2D array, or display images or shapes at random locations.

1

The Creative Coding with the Console Project is an open-ended project that allows you to choose the topic for your project and the type of program you want to create.

✅ Do This:

Read through the **Project Description** on the first page of the **Creative Coding with the Console Project Planning Guide** and the **rubric** on pages 3-4.



Name(s) _____ Period _____ Date _____

Creative Coding with The Theater Project Planning Guide

For this project, you will create a visual or animation using The Theater that portrays a personal interest or solves a problem that you choose. Your program should demonstrate the object-oriented programming and software engineering skills and knowledge you have developed throughout the year.

Project Requirements

Use your knowledge of object-oriented programming, two-dimensional (2D) arrays, and algorithms to create your personal narrative collage or animation:

- **Use inheritance and polymorphism** – Create a superclass with at least two subclasses to
- **Method decomposition** – Use overloaded, overridden, and private methods
- **Use at least two data structures** – Use at least two 1D or 2D arrays or ArrayLists to store elements that are manipulated using loops and conditionals.
- **Implement one or more algorithms** – Implement one or more algorithms that use two-way or multi-selection statements with compound Boolean expressions to analyze the data.
- **Create a visualization** – Create an image or animation that conveys the story of the data by illustrating the patterns or relationships in the data.
- **Document your code** – Use comments to explain the purpose of the methods and code segments and note any preconditions and postconditions.

Optional Features

- **Incorporate user interaction** – Use methods in the `Scanner` class to obtain user input to interact with your personal narrative program.
- **Use static variables, constants, and/or static methods** – Use static variables to represent values that are shared by all objects of a class, a constant to represent a value that doesn't change, and/or a static method to represent a behavior that doesn't require an object or as a helper method.
- **n the String class** – Use one or more methods in the String class in your program,
- **n the `Math` class** – Use one or more methods in the `Math` class in your program, such alculations on the values in a 2D array and display the results, choose random values ay, or display images or shapes at random locations.

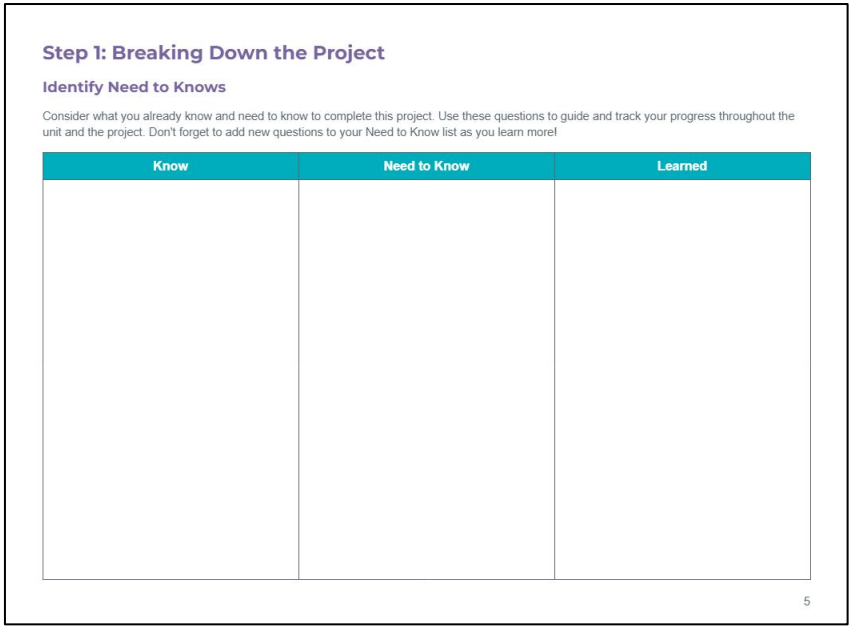
1

pages 1-4

✅ Do This:

Go to **page 5** of the **Project Planning Guide**.

Fill in the **Know** column with what you already know about Java that you can use to complete this project.



Step 1: Breaking Down the Project

Identify Need to Knows

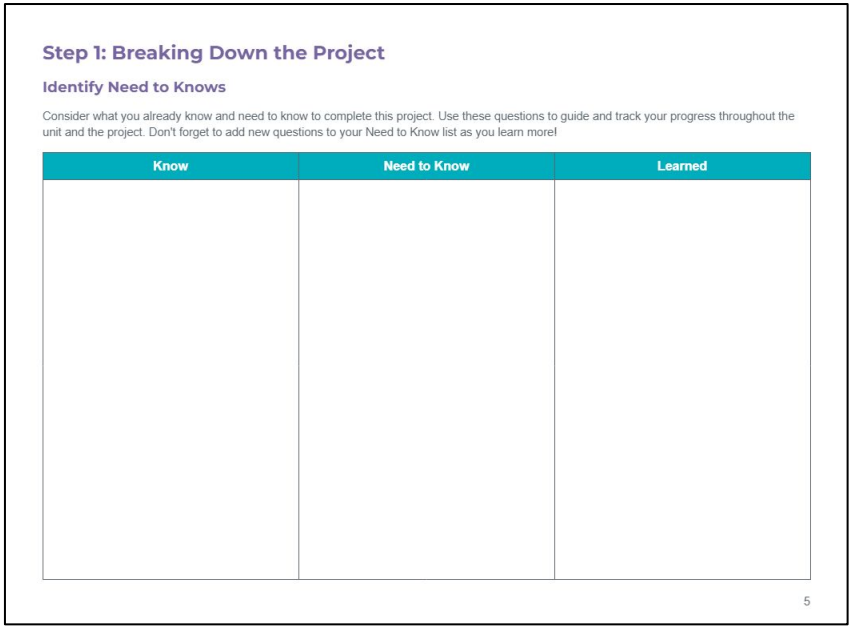
Consider what you already know and need to know to complete this project. Use these questions to guide and track your progress throughout the unit and the project. Don't forget to add new questions to your Need to Know list as you learn more!

Know	Need to Know	Learned

5

✅ Do This:

Fill in the **Need to Know** column with questions you have or things you don't know that you will need to learn to complete this project.



Step 1: Breaking Down the Project

Identify Need to Knows

Consider what you already know and need to know to complete this project. Use these questions to guide and track your progress throughout the unit and the project. Don't forget to add new questions to your Need to Know list as you learn more!

Know	Need to Know	Learned

5

🎯 Benchmark #1: Due Lesson 6

- Brainstorm project ideas and goals
- Decompose the problem to identify the classes and methods you will need to implement
- Obtain and implement feedback from peers

🎯 Benchmark #2: Due Lesson 8

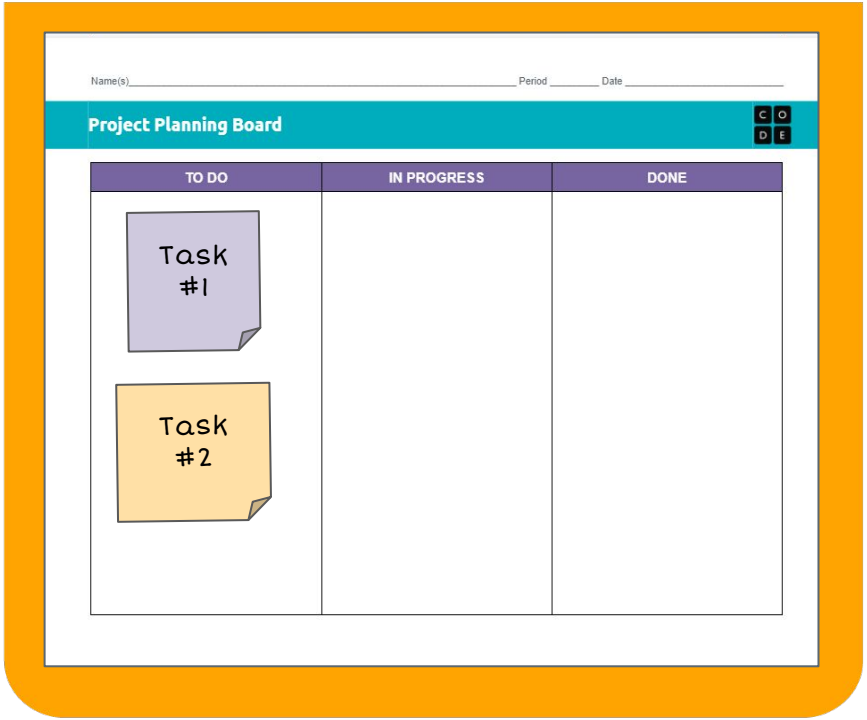
- Develop one or more classes and data structures
- Implement one or more algorithms
- Obtain and implement feedback from peers

🎯 Benchmark #3: Due End of Project

- Self-assess your work using the program requirements and rubric
- Finalize your program using the feedback you received and your self-assessment
- Showcase your work!

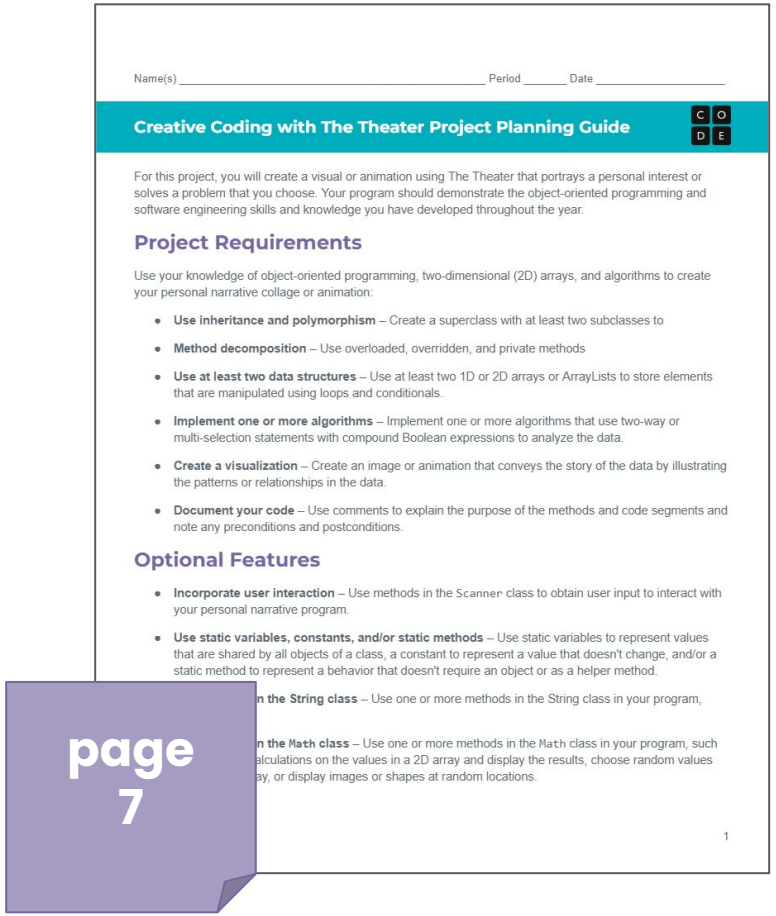
✅ Do This:

Post the tasks for the first benchmark in the **TO DO** column of your Project Planning Board.



✅ Do This:

Respond to the brainstorming prompts on **page 7** of your Project Planning Guide.



Unit 8 - Lesson 2
Searching

Computer Science A

🎯 Lesson Objectives

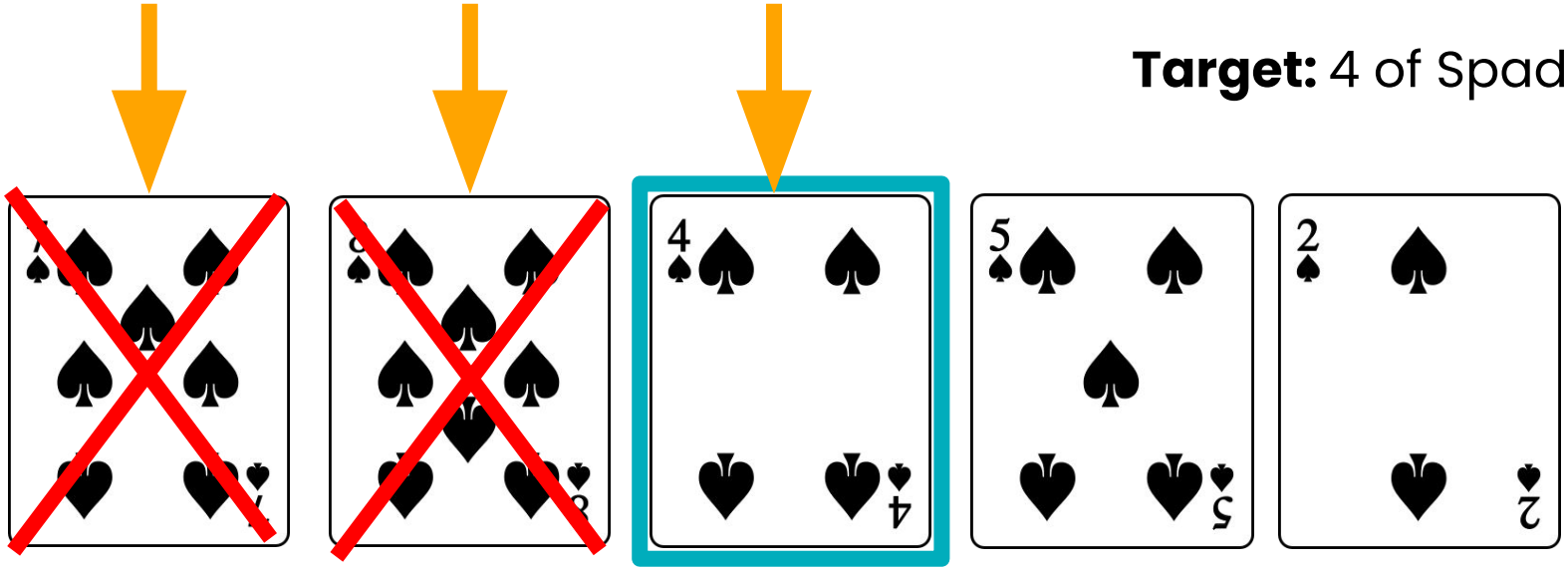
By the end of this lesson, you will be able to . . .

- Analyze the efficiency of the linear search algorithm using execution counts
- Differentiate between best case, average case, and worst case
- Identify the benefits and limitations of the linear search algorithm

🚀 Question of the Day

How can I determine the efficiency of an algorithm?

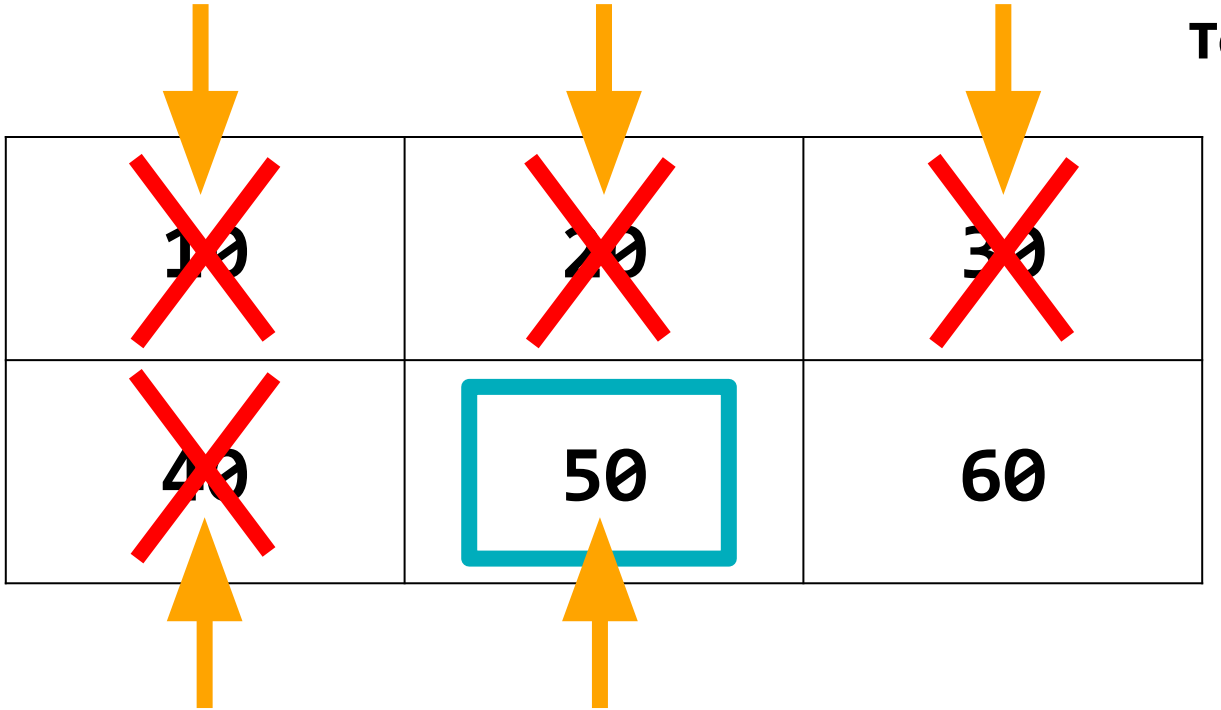
A **linear** (or **sequential**) **search** is a search algorithm that **checks each item in order** until the target is found.



Target: 4 of Spades

When a **linear search** is used on a **2D array**, it checks **each element** in **each row** one at a time.

Target: 50



We can analyze an algorithm to determine its:

- **Best case** : the **least** number of executions an algorithm can take to complete its goal
- **Average case** : the **average** number of executions an algorithm can take to complete its goal
- **Worst case** : the **most** number of executions an algorithm can take to complete its goal

Retrieve
your knowledge and ideas and write it down silently

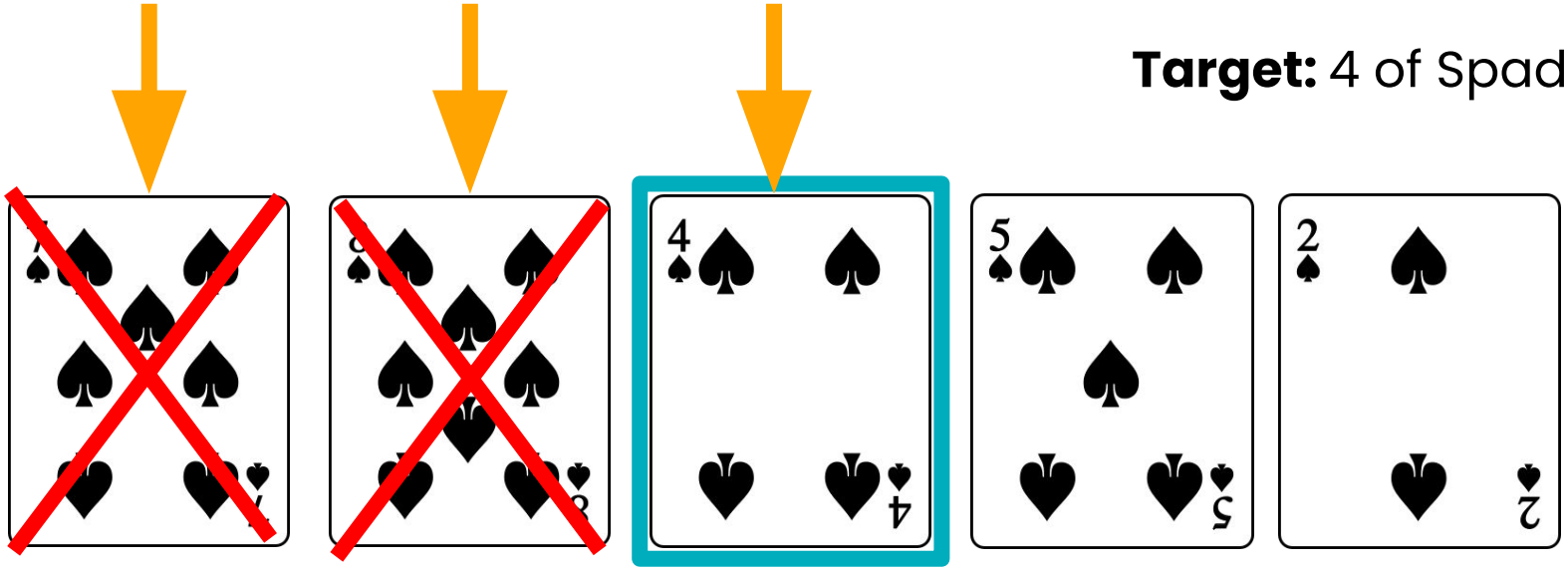
Pair
up with a neighbor and talk about your reflections

Share
your thoughts in a class discussion

Discuss:

- What is the **best case** for the World's Worst Magician?
- What is the **average case** for the World's Worst Magician?
- What is the **worst case** for the World's Worst Magician?

By counting the **number of guesses**, you've been counting the **number of executions** of the algorithm.



Target: 4 of Spades

You've been tracking the **execution count** , or **number of times** a code segment runs.

An **end user** is the person who will use the program.

A **user story** is an informal explanation of a program feature written from the perspective of the user.

✅ Do This:

Write **user stories** for your project on **page 8** of your Project Planning Guide.

Step 3: User Stories

A **user story** is a short explanation of a program feature that is written from the end user's point of view to describe how the feature will provide value to the user. User stories are often expressed in short sentences, like **"As a [persona], I [want to], [so that]."** You should write a user story for each large step or component of your overall project. For each user story:

- Identify the criteria for what it means for it to be considered "done".
- Decide which specific steps need to be completed for it to be considered "done".

User Story	Criteria	Steps to Complete

8

🚀 Question of the Day

How can I determine the efficiency of an algorithm?

💼 Key Vocabulary

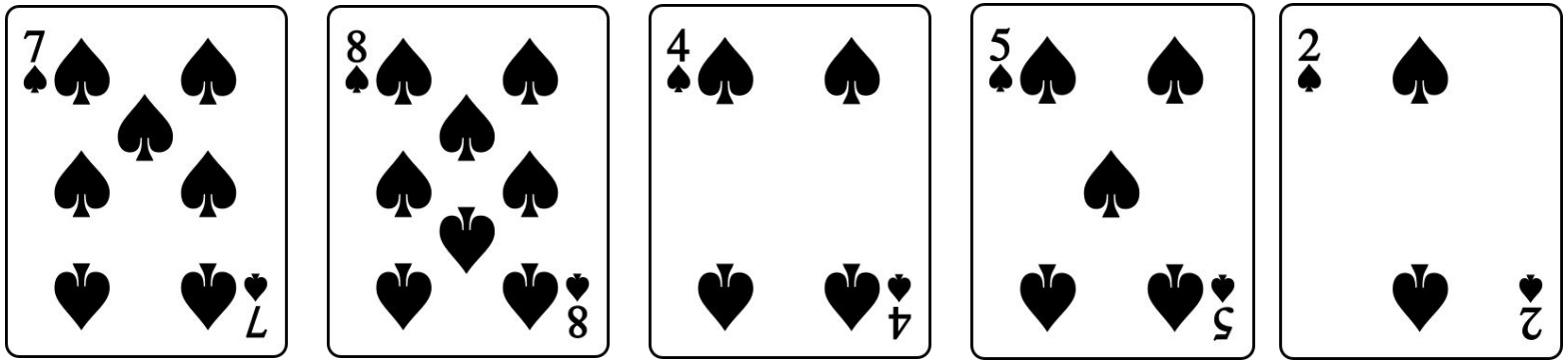
- **average case:** the average number of executions an algorithm can take to complete its goal
- **best case:** the least number of executions an algorithm can take to complete its goal
- **execution count:** the number of times a code segment runs
- **linear/sequential search:** a search algorithm that checks each item in order until the target is found
- **worst case:** the most number of executions an algorithm can take to complete its goal

Unit 8 - Lesson 3
Binary Search

Computer Science A

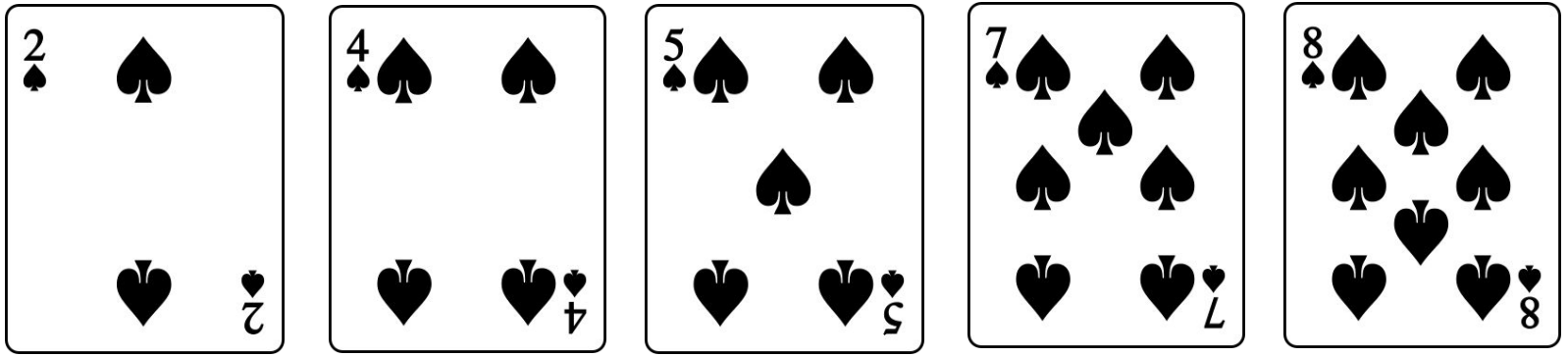
💡 Discuss:

What would the **worst** case execution count be if we used a **linear search** to find a specific card?

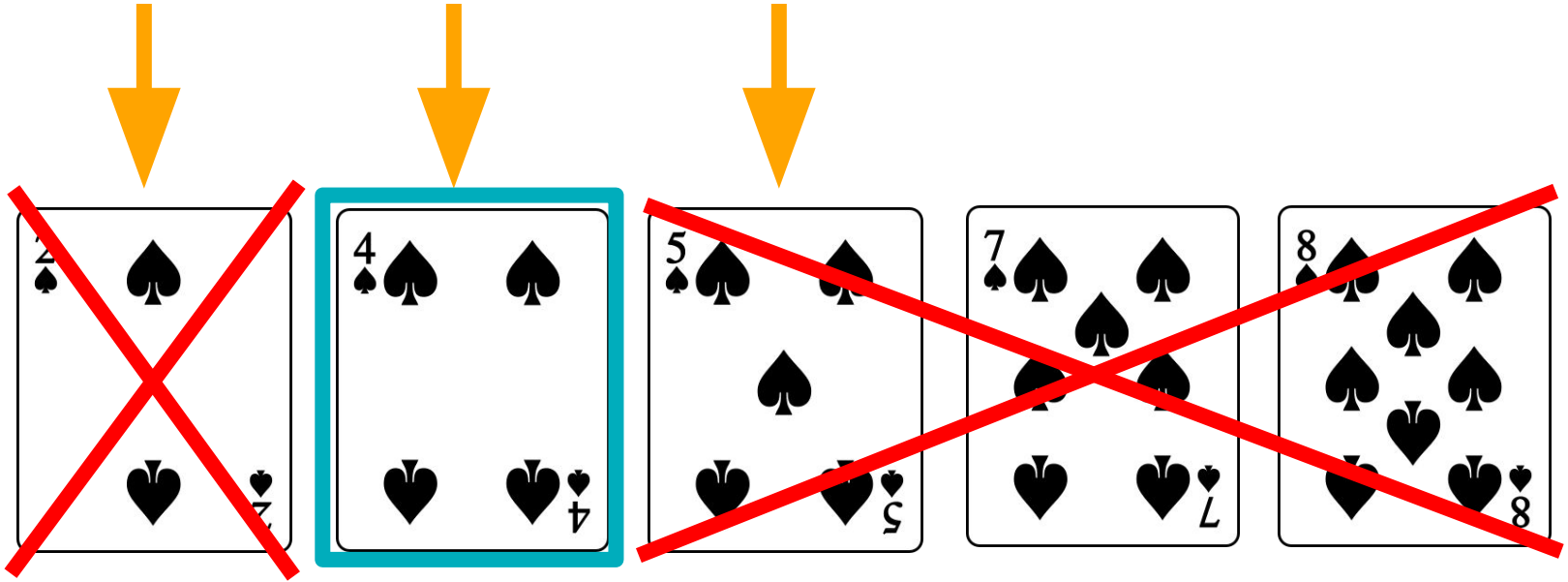


💡 Discuss:

What if this was the deck we needed to search?



Target: 4 of Spades



🎯 Lesson Objectives

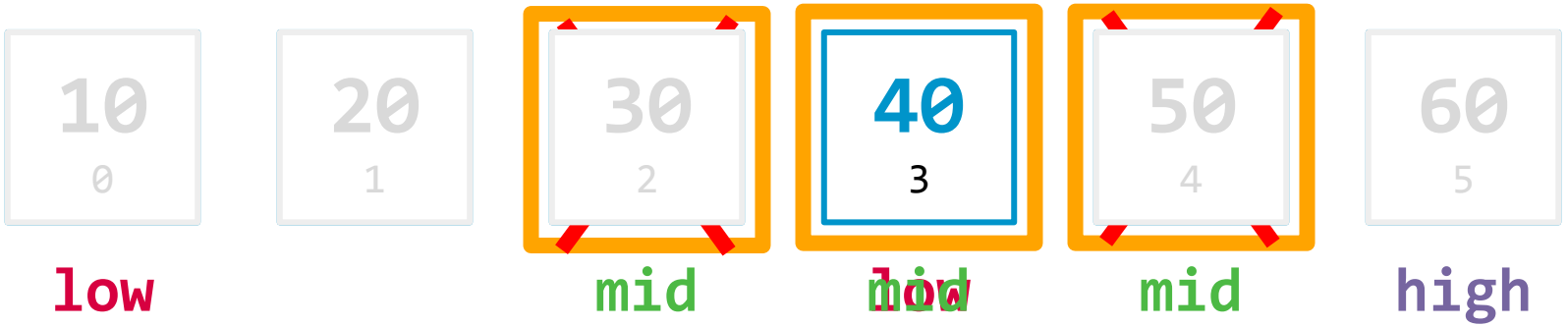
By the end of this lesson, you will be able to . . .

- Compare the efficiency of the binary search algorithm with the linear search algorithm using execution counts
- Differentiate between an iterative and recursive implementation of the binary search algorithm
- Identify the benefits and limitations of the binary search algorithm

🚀 Question of the Day

How is a binary search more efficient than a linear search?

The **binary search** algorithm finds a target element in a **sorted list** by **dividing the list in half** in each iteration.



Target: 40

💡 Discuss:

How does the execution counts for the **binary search** algorithm **compare** to the execution counts for the **linear search** algorithm?

```
public int recursiveSum(int num) {
  if (num <= 1) {
      return num;
  }

  return num + recursiveSum(num - 1);
}
```

The **base case** is the instance where a recursive method will return a value rather than calling itself.

The **recursive case** is the instance where the recursive method calls itself.

Recursion is when a method calls itself.

Searcher.java

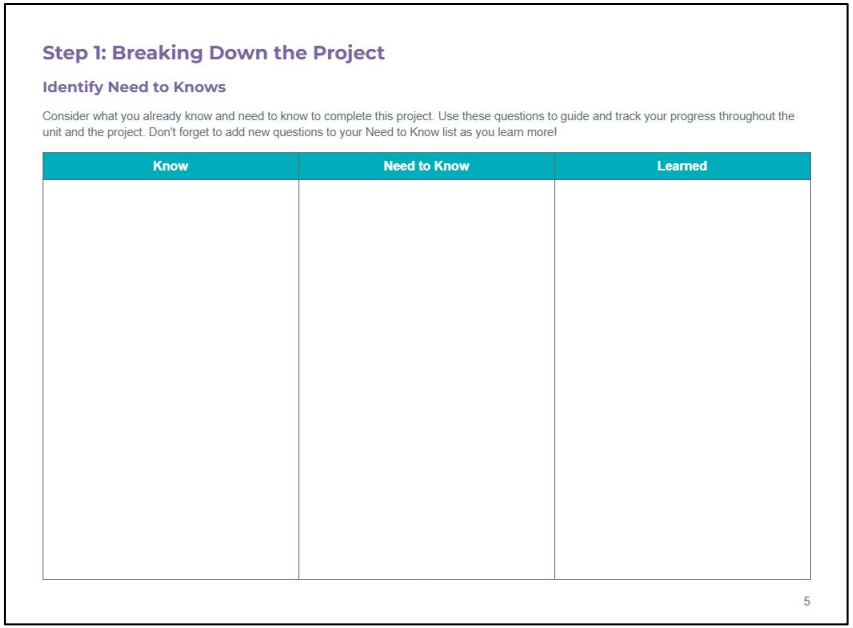
```
public static int binarySearch(int[] nums, int left, int right, int target) {
  if (left > right) {
    return -1;
  }

  int middle = (left + right) / 2;

  if (target < nums[middle]) {
    return binarySearch(nums, left, middle - 1, target);
  }
  else if (target > nums[middle]) {
    return binarySearch(nums, middle + 1, right, target);
  }
  else {
    return middle;
  }
}
```

Otherwise, the value to find is equal to the middle position so we return the middle position.

✅ Do This:

Revisit your **Need to Knows**!

- Check off **answered questions** in the **Need to Know** column.
- Add what you have **learned** and **answers to any questions** in the **Learned** column
- Add any **new questions** to the **Need to Know** column



Step 1: Breaking Down the Project

Identify Need to Knows

Consider what you already know and need to know to complete this project. Use these questions to guide and track your progress throughout the unit and the project. Don't forget to add new questions to your Need to Know list as you learn more!

Know	Need to Know	Learned

5

💼 Key Vocabulary

- **binary search:** a search algorithm that finds a target element in a sorted list by dividing the list in half in each iteration