

## COP2800C Module 7 Graded Programming Assignment – Species Data Class

In this assignment, we will build on our Module 6 work by transitioning from using arrays and methods to grouping related data and functionality into a class. This will reinforce our understanding of objects, classes, and abstraction while improving our program's organization.

### Key Points Recap:

- Group related data and functionality into a class (SpeciesCounter).
- Use private fields and public methods to practice encapsulation.
- Apply abstraction by hiding internal data handling from main().
- Build upon your existing work from Module 6 by refactoring into an object-oriented design.

### Learning Objectives

Upon completing this assignment, you will be able to:

- Define and use a class to group related data and operations.
- Implement private fields and public member methods.
- Apply abstraction and information hiding to simplify program design.

### Instructions

#### 1. Accept the GitHub Classroom Invite

- Clone the provided repository and open the PalmerPenguinsM7.java and SpeciesCounter.java starter files.
- Complete the author and submission date lines in the ID headers for both files.

#### 2. Complete the SpeciesCounter class (in SpeciesCounter.java)

This class encapsulates the species data and related operations.

- **Private Fields:**

Find the comment which begins "// Private fields for encapsulation" and add the following private fields:

- speciesData – A String[] array containing species names.
- speciesCount – An int[] array representing counts for each species.

- **Public Member Methods:**

There are 5 public "stub" methods in this file which currently do nothing useful. Complete these methods as described below. Note that the code in these methods is derived from the Module 6 PalmerPenguins.java file.

- `readSpeciesData()` – Calls `CSVReader.readFile(FILE_NAME, 1)` and assigns the result to `speciesData`. Returns nothing.
- `initializeSpeciesCount()` – Creates a new integer array of size `NUM_SPECIES` and assigns it to `speciesCount`. Returns nothing.
- `isEmpty()` – Returns true if `speciesData` is empty, otherwise false.
- `countSpecies()` – Iterates through `speciesData` and updates `speciesCount`.
- `printSpeciesCount()` – Prints the count of each species using the predefined constants.

### 3. Modify `main()` in `PalmerPenguinsM7.java`

- Find the comment beginning `// Create an instance of SpeciesCounter` and create an instance of `SpeciesCounter` named `counter`.
- Find the comment beginning `// Read species data` and call `counter.readSpeciesData()`;

### 4. No other modifications should be needed in the code; now test your program

- Verify that your program produces the expected output:

**Chinstrap count = 68**

**Gentoo count = 123**

**Adelie count = 151**

### 5. Screenshot Your Output

- Take a screenshot of your program's output in the "Run I/O" pane of the jGrasp IDE.
- DO NOT include the source code in the screenshot.

### 6. Submission

- Submit your completed `PalmerPenguinsM7.java` and `SpeciesCounter.java` files to the GitHub repository.
- Upload your output screenshot to the repository.
- Ensure your code follows our class's coding conventions (indentation, comments, line length, meaningful variable names).