Celestial Distance Calculator

**CSCI E-10b Term Project Proposal**

**LeAnne Branch**

**I. Project Description**

The Celestial Distance Calculator is an application that would allow users to determine the distance to celestial objects using various astronomical methods. The program will feature an interactive GUI representing a very small section of the night sky, where users can select some objects and view detailed calculations of their distances from Earth.

**II. Purpose and Utility**

My inspiration for this project is an excellent YouTube channel that I follow called ‘How Far Away Is It?’ by David Butler: [https://www.youtube.com/@howfarawayisit](https://www.youtube.com/@howfarawayisit%20)   
This channel breaks down very math-heavy astronomy concepts in an understandable way (with excellent narration, I might add).  
  
In general, astronomy enthusiasts, students, and educators often need to understand the different methods used to calculate astronomical distances. This application would serve several purposes:

1. **Educational Tool**: Demonstrates how astronomers determine cosmic distances through different methodologies
2. **Reference Application**: Provides quick access to distance information for selected celestial objects
3. **Visualization Tool**: Helps users understand the relative positions and distances of objects in our cosmic neighborhood
4. **Calculation Aid**: Shows the step-by-step calculations behind each distance determination method

This application bridges the gap between theoretical astronomy concepts and their practical application, making celestial distance calculations accessible to those with an interest in astronomy.

**III. Input/Output Design**

**Input:**

* The primary input will be user interaction with the GUI through mouse clicks on celestial objects displayed in a small star field
* Additional inputs will include dropdown menus or buttons to select different calculation methods
* For some calculation methods, users may input specific parameters (like apparent magnitude)

**Output:**

* Visual representation of a section of the night sky with labeled celestial objects
* Detailed distance calculations displayed when an object is selected
* Multiple measurement units (light-years, parsecs, astronomical units)
* Visual representation of the selected calculation method (e.g., parallax animation)

**Sample User Interaction:**

1. User launches the application and sees a star field with several labeled celestial objects
2. User clicks on "Proxima Centauri"
3. The application displays:
   * Distance: 4.246 light-years
   * Calculation Method: Parallax
   * Formula Used: d = 1/p, where p = 0.7687 arcseconds
   * Step-by-step calculation
   * Additional information about Proxima Centauri
4. User selects a different calculation method from a dropdown menu
5. Updated calculations appear based on the selected method

**IV. Algorithms and Data Structures**

**Key Algorithms:**

1. **Parallax Distance Calculation**:

distance (in parsecs) = 1 / parallax (in arcseconds)

1. **Spectroscopic Parallax**:

distance = 10^((apparent\_magnitude - absolute\_magnitude + 5)/5)

1. **Standard Candle Method** (for Cepheid Variables):

absolute\_magnitude = -2.43 \* (log10(period\_in\_days) - 1) - 4.05

distance = 10^((apparent\_magnitude - absolute\_magnitude + 5)/5)

1. **Hubble's Law** (for distant galaxies):

distance = redshift \* c / H₀

Where:

* + c is the speed of light
  + H₀ is the Hubble constant

**Data Structures:**

1. **Star Class**: Contains star properties (name, coordinates, parallax, apparent magnitude, absolute magnitude, etc.)
2. **Galaxy Class**: Contains galaxy properties (name, coordinates, redshift, etc.)
3. **ArrayList/Vector of celestial objects**: Stores all objects displayed in the star field
4. **HashMap of calculation methods**: Maps method names to their corresponding calculation algorithms
5. **Coordinate System Class**: Handles conversion between screen coordinates and celestial coordinates

**V. Template Classes**

The application will make use of the following template classes:

1. **CelestialObject (Abstract Class)**:
   * Base class for all celestial objects
   * Attributes: name, coordinates (RA/Dec), apparent magnitude
   * Methods: getDistance() (abstract), getInfo(), draw()
2. **Star (Extends CelestialObject)**:
   * Additional attributes: parallax, absolute magnitude, spectral type
   * Methods: getParallaxDistance(), getSpectroscopicDistance()
3. **Galaxy (Extends CelestialObject)**:
   * Additional attributes: redshift, type (spiral, elliptical, etc.)
   * Methods: getRedshiftDistance(), getHubbleDistance()
4. **DistanceCalculator**:
   * Methods for different calculation techniques
   * Static utility methods for astronomical conversions
5. **SkyMap**:
   * Manages the celestial objects collection
   * Handles user interaction with the star field
   * Methods: findObject(), getVisibleObjects()
6. **AstronomicalConstants**:
   * Contains constants used in calculations (Hubble constant, speed of light, etc.)
   * Conversion methods between different units (parsecs to light-years, etc.)
7. **UserInterface**:
   * Manages the Swing components and user interaction
   * Methods: displayObjectInfo(), updateCalculation()

**VI. GUI Design**

The application will feature an event-driven GUI using Java's Swing and AWT classes with the following components:

1. **Main Window**:
   * ~~Menu bar with File, View, and Help options~~
   * Star field panel (main display area)
   * Information panel (displays selected object details)
   * ~~Status bar~~
2. **Star Field Panel**:
   * Interactive display of celestial objects
   * Mouse event handling for object selection
   * Optional grid lines and constellation markers
   * Zoom functionality
3. **Information Panel**:
   * Object name and basic information
   * Distance calculation with formula display
   * Method selection dropdown
   * Calculation breakdown
   * Visual representation of the calculation method
4. **Dialog Windows**:
   * About dialog
   * Help dialog with explanations of calculation methods
   * Preferences dialog for display settings

The GUI will be designed with usability in mind, featuring clear labels, intuitive controls, and responsive feedback to user actions.

**VII. Implementation Plan**

The project will be implemented in stages:

1. **Foundation:**
   * Define and implement the template classes
   * Set up the basic GUI framework
   * Create data structures for celestial objects
2. **Core Functionality:**
   * Implement distance calculation algorithms
   * Create the star field display
   * Add object selection functionality
3. **Enhanced Features:**
   * Implement additional calculation methods
   * Add visual representations of methods
   * Improve the user interface with more information and options
4. **Testing and Refinement:**
   * Test with various celestial objects
   * Debug and optimize
   * Add any remaining features and polish

**VIII. Conclusion**

The Celestial Distance Calculator would be a useful tool for understanding and visualizing the methods astronomers use to determine cosmic distances. The project will make good use of the concepts covered in class while throwing in a few challenges ~~(such as the proposed HashMap implementation~~). All in all, it would bring together object-oriented programming, GUI development with Swing/AWT, and practical algorithms from astronomy.