The course planner program is a Python-based tool designed to manage and search for college courses using a Binary Search Tree (BST). Initially written in C++, the program was converted to Python for enhancement one, ensuring that its core functionality—loading courses from a CSV file, storing them in a BST, and enabling course searches—remained intact. Enhancement Two, however, introduced significant improvements, making the program more efficient and user-friendly.

The artifact was chosen for my ePortfolio because it showcases my proficiency in data structures and algorithms, particularly with BSTs. The ability to efficiently insert, search, and retrieve course data demonstrates my understanding of hierarchical data storage. Enhancement two also included sorting capabilities, allowing courses to be displayed in order of prerequisites and an interactive search feature that enables users to find classes dynamically based on keywords. These updates highlight my ability to improve data accessibility and enhance user experience.

Throughout these enhancements, I successfully met the course outcomes I had planned for in Module One. The BST structure ensures optimized searching, and the added sorting functionality provides structured insights into course prerequisites. The interactive search feature further enhances the program’s usability by allowing users to search using partial course names or IDs. Moving forward, additional improvements could include balancing the BST for even greater efficiency and integrating a graphical user interface (GUI) for improved usability.

Throughout the enhancement process, I learned that usability is just as critical as algorithmic efficiency. While the BSTs are highly effective for storing and retrieving data, making that data easily searchable and navigable is crucial for user engagement. One challenge I encountered was sorting BST data dynamically since BSTs naturally store data in order but not by prerequisite count. This required additional sorting logic to properly display courses. Another challenge was ensuring CSV file integrity, as missing or malformed data could cause runtime errors. Despite these challenges, the enhancement process significantly improved my problem-solving skills in data structure implementation, algorithm optimization, and user interaction design.

**Status Checkpoints for All Categories**

|  |  |  |  |
| --- | --- | --- | --- |
| **Checkpoint** | **Software Design and Engineering** | **Algorithms and Data Structures** | **Databases** |
| **Name of Artifact Used** | Source.cpp | Source.cpp | Courses.csv |
| **Status of Initial Enhancement** | Completed (Converted from C++ to Python) | Completed (Updated algorithms for new sorting and display) | In Progress |
| **Submission Status** | Submitted | Submitted | Not Submitted |
| **Status of Final Enhancement** | Completed | Pending | Pending |
| **Uploaded to ePortfolio** | No | No | No |
| **Status of Finalized ePortfolio** | Pending | Pending | Pending |

**Enhancement One Evidence:**

**A computer screen shot of a course planner

Description automatically generatedA screenshot of a computer program

Description automatically generated**

**Enhancement Two Evidence:**

**A screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generated**