**Pseudocode Binary Search Tree**  
 (Opening the file, reading data, parsing lines and checking for errors)  
  
FUNCTION loadCourses(filePath)

OPEN file at filePath FOR reading

IF file cannot be opened THEN

PRINT "Error: File cannot be opened."

RETURN

WHILE there are more lines in the file DO

READ a line from the file

CALL parseCourseLine(line)

CLOSE the file

FUNCTION parseCourseLine(line)

SPLIT line into tokens based on delimiter (e.g., comma)

IF number of tokens < 2 THEN

PRINT "Error: Line format is incorrect, missing parameters."

RETURN

courseNumber = tokens[0]

courseTitle = tokens[1]

prerequisites = tokens[2:]

IF courseNumber IS EMPTY OR courseTitle IS EMPTY THEN

PRINT "Error: Course number or title is missing."

RETURN

CALL validatePrerequisites(prerequisites, courseList)

CREATE a Course object

CALL insertCourseIntoBST(bst, Course)

**Creating Course Objects and Inserting Into Binary Search Tree**  
FUNCTION createCourseObject(courseNumber, courseTitle, prerequisites)

INITIALIZE a new Course object

SET Course.number = courseNumber

SET Course.title = courseTitle

SET Course.prerequisites = prerequisites

RETURN Course

FUNCTION insertCourseIntoBST(bst, course)

IF bst.root IS NULL THEN

bst.root = course

ELSE

CALL insertNode(bst.root, course)

FUNCTION insertNode(node, course)

IF course.number < node.course.number THEN

IF node.left IS NULL THEN

node.left = course

ELSE

CALL insertNode(node.left, course)

ELSE

IF node.right IS NULL THEN

node.right = course

ELSE

CALL insertNode(node.right, course)

**Printing Course information**FUNCTION printCourseInfo(bst)

CALL inOrderTraversal(bst.root)

FUNCTION inOrderTraversal(node)

IF node IS NOT NULL THEN

CALL inOrderTraversal(node.left)

PRINT courseInfo(node.course)

CALL inOrderTraversal(node.right)

FUNCTION courseInfo(course)

PRINT "Course Number: " + course.number

PRINT "Course Title: " + course.title

PRINT "Prerequisites: " + course.prerequisites