# Pseudocode: ABCU Course Management System

STRUCT Course  
 STRING courseNumber  
 STRING courseTitle  
 LIST of STRING prerequisites  
  
STRUCT Node  
 Course course  
 Node\* left ← NULL  
 Node\* right ← NULL  
  
CLASS BinarySearchTree  
 PRIVATE:  
 Node\* root ← NULL  
  
 METHOD addNode(Node\* node, Course course)  
 IF course.courseNumber < node.course.courseNumber THEN  
 IF node.left IS NULL THEN  
 node.left ← new Node(course)  
 ELSE  
 addNode(node.left, course)  
 ELSE  
 IF node.right IS NULL THEN  
 node.right ← new Node(course)  
 ELSE  
 addNode(node.right, course)  
  
 METHOD inOrder(Node\* node)  
 IF node IS NOT NULL THEN  
 inOrder(node.left)  
 PRINT node.course.courseNumber, node.course.courseTitle  
 inOrder(node.right)  
  
 METHOD searchNode(Node\* node, STRING courseNumber) RETURNS Node\*  
 IF node IS NULL OR node.course.courseNumber = courseNumber THEN  
 RETURN node  
 IF courseNumber < node.course.courseNumber THEN  
 RETURN searchNode(node.left, courseNumber)  
 ELSE  
 RETURN searchNode(node.right, courseNumber)  
  
 PUBLIC:  
 METHOD Insert(Course course)  
 IF root IS NULL THEN  
 root ← new Node(course)  
 ELSE  
 addNode(root, course)  
  
 METHOD InOrderPrint()  
 CALL inOrder(root)  
  
 METHOD Search(STRING courseNumber) RETURNS Course  
 node ← searchNode(root, courseNumber)  
 IF node IS NOT NULL THEN  
 RETURN node.course  
 ELSE  
 RETURN empty Course  
  
FUNCTION loadCoursesFromFile(filePath, BinarySearchTree\* bst)  
 OPEN file at filePath  
 IF file NOT opened THEN  
 DISPLAY error message  
 EXIT  
  
 SET allCourses ← empty map of courseNumber → Course  
  
 FOR EACH line in file DO  
 SPLIT line into tokens by comma  
 IF number of tokens < 2 THEN  
 DISPLAY format error  
 CONTINUE  
  
 courseNumber ← tokens[0]  
 courseTitle ← tokens[1]  
 prerequisites ← empty list  
  
 FOR i FROM 2 TO tokens.size - 1 DO  
 APPEND tokens[i] TO prerequisites  
  
 CREATE new Course object  
 SET course.courseNumber ← courseNumber  
 SET course.courseTitle ← courseTitle  
 SET course.prerequisites ← prerequisites  
  
 ADD course to allCourses map with key as courseNumber  
  
 FOR EACH course IN allCourses DO  
 FOR EACH prereq IN course.prerequisites DO  
 IF prereq NOT IN allCourses.keys THEN  
 DISPLAY error: "Missing prerequisite: " + prereq  
  
 FOR EACH course IN allCourses DO  
 bst.Insert(course)  
  
FUNCTION printCourseInfo(BinarySearchTree\* bst, STRING courseNumber)  
 course ← bst.Search(courseNumber)  
 IF course.courseNumber IS EMPTY THEN  
 DISPLAY "Course not found."  
 ELSE  
 PRINT "Course: " + course.courseNumber + ", " + course.courseTitle  
 IF course.prerequisites IS EMPTY THEN  
 PRINT "Prerequisites: None"  
 ELSE  
 PRINT "Prerequisites: "  
 FOR EACH prereq IN course.prerequisites DO  
 PRINT prereq  
  
FUNCTION printAllCourses(BinarySearchTree\* bst)  
 CALL bst.InOrderPrint()  
  
MAIN FUNCTION  
 CREATE new BinarySearchTree bst  
 CALL loadCoursesFromFile("courses.txt", bst)  
  
 LOOP UNTIL user chooses to exit  
 DISPLAY menu:  
 1. Print All Courses  
 2. Search for a Course  
 9. Exit  
  
 PROMPT user for choice  
  
 IF choice = 1 THEN  
 CALL printAllCourses(bst)  
 ELSE IF choice = 2 THEN  
 PROMPT user for course number  
 CALL printCourseInfo(bst, course number)  
 ELSE IF choice = 9 THEN  
 EXIT PROGRAM