BEGIN

// Define Course structure

STRUCT Course

STRING courseID // Unique identifier for each course

STRING courseTitle // Title of the course

LIST<String> prerequisites // List of prerequisites (other courses)

END STRUCT

// Function: Load course data from file into a list

FUNCTION loadCourseData(FILENAME)

DECLARE LIST<Course> courseList

OPEN FILENAME FOR READING

IF file cannot be opened THEN

PRINT "Error: Unable to open file."

RETURN EMPTY LIST

END IF

FOR each line in file DO

SPLIT line by ","

IF number of tokens < 2 THEN

PRINT "Warning: Invalid line skipped"

CONTINUE

END IF

CREATE Course newCourse

newCourse.courseID = first token

newCourse.courseTitle = second token

FOR each remaining token in line DO

APPEND token TO newCourse.prerequisites

END FOR

APPEND newCourse TO courseList

END FOR

CLOSE file

RETURN courseList

END FUNCTION

// Function: Verify all course prerequisites exist

FUNCTION verifyCourseData(LIST<Course> courseList)

DECLARE SET<String> courseIDs

// Collect all course IDs into the set

FOR each course IN courseList DO

ADD course.courseID TO courseIDs

END FOR

// Validate prerequisites for each course

FOR each course IN courseList DO

FOR each prereq IN course.prerequisites DO

IF prereq NOT IN courseIDs THEN

PRINT "Error: " + prereq + " does not exist as a prerequisite for " + course.courseID

END IF

END FOR

END FOR

END FUNCTION

// Function: Search for a course by ID and display details

FUNCTION findCourse(LIST<Course> courseList, STRING courseID)

FOR each course IN courseList DO

IF course.courseID == courseID THEN

PRINT "Course ID: " + course.courseID

PRINT "Course Title: " + course.courseTitle

PRINT "Prerequisites: "

IF course.prerequisites IS EMPTY THEN

PRINT "None"

ELSE

FOR each prereq IN course.prerequisites DO

PRINT prereq

END FOR

END IF

RETURN

END IF

END FOR

PRINT "Error: Course with ID " + courseID + " not found."

END FUNCTION

// Main execution loop

FUNCTION main()

STRING filename = "course\_data.txt"

LIST<Course> courseList = loadCourseData(filename)

IF courseList IS EMPTY THEN

RETURN

END IF

CALL verifyCourseData(courseList)

WHILE TRUE DO

PRINT "Enter a course ID to search (or type 'EXIT' to quit): "

STRING userInput = GET USER INPUT

IF userInput == "EXIT" THEN

BREAK

END IF

CALL findCourse(courseList, userInput)

END WHILE

END FUNCTION

CALL main()

END

Looking at how the program handles course data, both loading the file and checking prerequisites run in O(n) time, where *n* is the number of courses. Using a vector is simple and works well for storing the data, but searching through it takes longer as the list grows. A tree keeps things in order and speeds up searches a bit, but it’s more complex to manage. A hash table, on the other hand, makes finding courses almost instant with average O(1) lookup time. Since the goal is to quickly access course information by ID, the hash table feels like the smartest and most efficient choice overall.