CS 481 — AI — Camel Cover Puzzle

Project — Camel Cover Puzzle

Introduction

This project is to write a Lisp program to find a sequence of moves for a chess Camel that covers (visits all the squares) on a 7x7 doughnut chess board: with the center square (at the row-column location (3 3)) removed.

A Camel moves almost like a Knight (Horse): using an L-path for the move, but 3 squares in one direction (instead of the Knight's 2 squares) horizontally or vertically and then 1 square at right angles. You may place the Camel on any square on the board to start with: part of the problem is to find a good square to start from.

Your state should be the board with a camel on it and each square marked if it has been visited or not.

Your path is a sequence of square coordinates (2 dimensions): in the range from (0 0) up to (7 7).

The goal is to find a path that provides the largest coverage: visited squares.

A constraint is that the Camel is not allowed to visit a previous square (kind of like the Konigsberg Bridge problem).

This search should be done recursively.

As a hint, we recommend that if a Camel has no legal move, then you should 1) replace your prior best coverage path with the current one, if it is better, and 2) back up to the prior path square (unvisiting the current square) and pick a different move to extend the path from that prior square in a new direction.

Team

The team may contain up to three members. Pick a three-letter name for the team based on the first letters of the members' last names. If fewer than three members are on the team, pick a third letter of 'X'. Each team must email to me your team name and your member names.

Readme File

You should provide a README.txt text file that includes the class and section, your (team) name, the project/program name, instructions for building, instructions for use, any extra features, and any known bugs to avoid. Be clear in your instruction on how to build and use the project by providing instructions a novice programmer would understand. If there are any external dependencies for building, the README must also list them and how to find and incorporate them. Usage should include an example invocation.

A README would cover the following:

- Program name
  - Your Name (authors, team)
  - Contact info (email)
  - Class number (481)
- Intro (see the Introduction section, above)
- External Requirements
- Build, Installation, and Setup
- Usage
- Extra Features
- Bugs

Academic Rules

Correctly and properly attribute all third party material and references, if any, lest points be taken off.

Submission

Your submission must, at a minimum, include a plain ASCII text file called **`README.txt`** (e.g., title, contact info (of all team members), files list, installation/run info, bugs remaining, features added) all necessary source files to allow the submission to be built and run independently by the instructor. [For this project, no unusual files are expected.] Note, the instructor doesn't necessarily use your IDE or O.S.

All source code files must include a comment header identifying the author, author's contact info (please, no phone numbers), and a brief description of the file.

Do not include any IDE-specific files, object files, binary **executables**, or other superfluous files.

Place your submission files in a **folder named** `X-pY_lastname-firstinitial`. (If working in a team, use the team's 3-letter acronym: `X-pY_RAK`, for team RAK.) Where X is the class course number (e.g., for course CS-123 you should use "123") and Y is the project number (eg, 9 for Project #9) For example in CS-123 for Project #9, if your name were Alice Ghost, then you should use this:

```
123-p9_Ghost-A
```

Then zip up this folder. Name the .zip file the **same as the folder name**.

Turn in by 11pm on the due date (in the bulletin-board post) by **sending me email** (see the Syllabus for the correct email address) with the zip file attached. The email subject title should also include **the folder name**. [NB, If your emailer will not email a .zip file, then change the file extension from .zip to .zap, attach that, and tell me so in the email.] Please include your name and campus ID (for each team member) at the end of the email (because some email addresses don't make this clear). If there is a problem with your project, don't put it in the email body – put it in the README.txt file. Do not provide a link to Dropbox, Gdrive, or other cloud storage.

Grading

- 75% for compiling and executing with no errors or warnings
- 10% for clean and well-documented code
- 10% for a clean and reasonable **`README`** file
- 5% for successfully following Submission rules