UNIVERSITY OF TEXAS AT ARLINGTON
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

4308/5360 **ARTIFICIAL INTELLIGENCE I** **FALL 2021**

PROGRAMMING ASSIGNMENT (100 POINTS)
ASSIGNED: 10/7/2021 DUE: 11/4/2021

This assignment constitutes 30% of the course grade. You must work on it individually and are required to submit the Lisp programs described below along with a PDF report.

1 Introduction

Figure 1: Get all the missionaries and cannibals to the other side of the river without the cannibals devouring the missionaries!

In this first part of this assignment, you will design a Lisp program that solves the missionaries and cannibals problem (Figure 1). The problem is defined as follows. A group of missionaries and cannibals must cross a river using a boat which can carry at most two people. Crossing the missionaries and cannibals from one side of the river to the other is subject to the constraint that the number of cannibals cannot outnumber the number missionaries present on either side of the river bank (otherwise the cannibals will eat the missionaries). In addition, the boat cannot cross the river by itself with no people on board.

In the second part of this assignment, you will design a Lisp program that solves the 8-puzzle problem (Figure 2) using the A* algorithm. The 8-puzzle problem (described in the lecture notes) was invented and popularized by Noyes Palmer Chapman in the 1870s. It is played on a 3×3 grid with 8 square blocks labeled 1 through 8 and a blank square. Your goal is to rearrange the blocks so that they are in order by sliding blocks horizontally or vertically into the blank square.

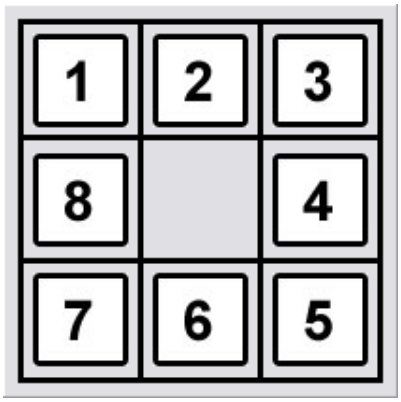

Figure 2: The 8-puzzle problem.

Files to submit: The programs in this assignment must be written in Common Lisp (see the class lecture notes on Lisp). The development of the programs can be done on the machine of your choice, however the source code must compile and run using CLISP. You should create a Lisp script that invokes the necessary functions to solve the problems and displays the output to the terminal.

You must submit your source code and report as follows. Create a directory with your last name followed by an underscore followed by your student ID number, e.g. 'smith_1010101010'. Then, place the files in this directory, zip the directory, and upload the zip file to Canvas.

Evaluation: Your code will be graded for technical correctness. The report must contain the following items: your name and student ID number, an implementation description of the programs, known bugs and deficiencies, a description of how to run the programs, and the terminal session script output. The weight distribution for the assignment is 80% for correctness and 20% for the report.

Academic dishonesty: We will be checking your code against other submissions in the class for logical redundancy. If you copy someone else's code and submit it with minor changes, we will know. These cheat detectors are quite hard to fool, so please don't try. We trust you to submit your own work only; please don't let us down. If you do, we will pursue the strongest consequences available to us.

2 Missionaries and Cannibals

Question 1 (50 points)

Write Lisp code that solves the missionaries and cannibals problem assuming that you have one boat, 15 cannibals, and 15 missionaries. Assume that the boat can carry at most six people and there is a single boat available. You cannot have the boat moving with no one on board. The cannibals should never outnumber the missionaries (either on the boat or on both sides of the river). You can write your own code or modify the code from the course textbook website to solve the problem. You need to print out the sequence of moves leading to a correct solution. Try the same problem with 20 cannibals and 20 missionaries. **You must not hardcode your solution**.

Sample Lisp code for this problem can be found at: https://github.com/aimacode/aima-lisp

3 8-Puzzle

Question 2 (50 points)

Implement the A* algorithm for searching trees in Lisp. Do not use the course textbook website code or other code from the Internet. Implement a counter that counts the number of nodes expanded and prints this number at the end of the search. Use your code to solve the 8-puzzle problem with the heuristic being the number of misplaced tiles and start state: $((E,1,3),(4,2,5),(7,8,6))$. The goal state is: $((1,2,3),(4,5,6),(7,8,E))$. Print the number of nodes expanded. You only need to show the states generated during the search process. Your code should detect **infeasible puzzles**.