Summary of project description:

Aims: To understand simple AI techniques and algorithms and how they can be applied to solve puzzles and play games.

Puzzle games and solving them with AI:

* Many puzzles and games require logical thinking to choose a best move, or to find a hidden pattern. AI can do this
* Most often use back-tracking based searches to find best solution
* Can employ algorithms to improve back-tracking
* Pay off can be optimised via pruning
* Allows us to solve puzzles much quicker than people

Constraint Satisfaction:

* Problem instance of related questions
* Cannot answer any particular question without answers to related questions

Early Deliverables (copied):

* Proof of concept programs: Simple colourful GUI with an active button
* Proof of concept programs: Relevant Data structures including a priority queue.
* Proof of concept programs: Solving the eight queens problem using backtracking
* Proof of concept program which solves a simple example of your puzzle.
* Reports: Design Patterns for AI and Search.
* Report on Backtracking and Recursion.
* Reports: Constraint Satisfaction, particularly consistency techniques.
* Reports: Techniques and Algorithms used by human solvers.
* Reports: User Interface design for a solver
* Reports: Complexity, NP hardness and the big O notation. Estimating problem size and hardness of your puzzle.

Final Deliverables (copied):

* Full object-oriented design, full implementation life cycle using modern software engineering principles
* Splash screen and at least two other user interaction screens.
* Graphical User Interface that can be used to generate, load, save, solve and help with puzzle solving/game playing.
* Produce puzzles that have only one correct solution, or games that have a single best strategy.

The report will:

* describe the software engineering process involved in generating your software.
* describe interesting algorithms and programming techniques used (or able to be used) on the project.
* discuss the choice of data structures and their performance impact on the solving and generation algorithms.
* describe at least a backtracking and recursion solving algorithm and will compare them (including benchmark).
* describe the CSP and algorithms useful for solving the CSP including generalised arc consistency, Forward Checking, DVO and Backjumping.

Possible extensibles:

* Most games have web-based implementation, can extend program to play automatically.
* Most games can be read from picture files or using image processing, another form of AI.
* Porting to mobile
* Program gives hints for users when stuck.
* Program supports printing, different screen size, timers etc.

Project plan to do:

* Constraint satisfaction problem
* Research chronological back tracking a DFS algorithm.

Abstract:

* To create an AI capable of solving a puzzle nonograms using backtracking and recursion solving algorithms.
* Constraint programming will be very suited towards nonograms as the numbers for how many squares are in a row are a constraint.
* Possibly use reinforcement learning to optimise.

Timeline:

* Proof of concept program: Display given or stored nonogram puzzles
* Proof of concept program: Solve a simple nonogram using back-tracking

Bibliography:

1. Yu, CH., Lee, HL. & Chen, LH. An efficient algorithm for solving nonograms. Appl Intell 35, 18–31 (2011). https://doi.org/10.1007/s10489-009-0200-0
2. Dandurand, F., Cousineau, D., & Shultz, T. R. (Year). Solving nonogram puzzles by reinforcement learning.
3. I. . -C. Wu et al., "An Efficient Approach to Solving Nonograms," in IEEE Transactions on Computational Intelligence and AI in Games, vol. 5, no. 3, pp. 251-264, Sept. 2013, doi: 10.1109/TCIAIG.2013.2251884.

Risk Assessment: