# Course Project Instructions

**Please read the following instructions very carefully, and let me know as soon as possible if you have any questions! I am here to help.**

**Overview:** The course project is an opportunity to explore AI applications and concepts that go beyond what we will learn in class or read in the textbook. Your project should focus on comparing AI algorithms across a range of novel problems. By completing the project, you will develop the skills necessary to choose the right AI algorithm for whatever problem you are asked to solve.

**Getting Started:** To get started, please use this link to create a GitHub repository for your code. You must create this repository before submitting your project proposal (see below). Please use this link to register your GitHub ID, so we know whose project is whose.

**Topic:** Selecting a topic requires: **(i)** choosing the AI algorithm(s) you will use and **(ii)** choosing the problem(s) you will apply them to. When making these choices, bear in mind that the primary aim of your project will be to compare the performance of algorithms across problems to test an interesting hypothesis. This might take the form of comparing a single algorithm across multiple problems, comparing multiple algorithms across multiple problems, or comparing multiple algorithms on a single problem. A good rule of thumb is that the fewer algorithms you cover, the more problems I will expect (and vice versa).

The following rules also apply: **(i)** All students must select at least one algorithm that **is** covered in the course. **(ii)** All students must select at least one problem **not** covered in the course. ***(iii)*** *Graduate students must also select at least one algorithm* ***not*** *covered in the course. Undergraduates may complete the graduate requirement for extra credit.*

- Note: You are not required to implement your chosen algorithms right away, so don't be afraid to choose algorithms covered later in the course. See timeline below for more details.
- Note: By an "interesting" hypothesis, I mean more than predicting which algorithm will perform best on different applications. You should predict that specific algorithms will work better in specific ways for specific reasons grounded in the properties of the algorithms and applications.

**Timeline:** The project involves **three** separate submissions: (1) a project proposal, (2) a progress report, and (2) a final report. There are are drop boxes for these submissions under "Assignments." Exact due dates can be found on the syllabus, but (1) will be due around the end of the first full month of the course, (2) will be due around the end of the second full month of the course, and (3) will be due at the end of the course.

**Part 1, Project Proposal:** For the project proposal, you will **(i)** identify the algorithms and problems you intend to cover, **(ii)** identify the comparisons you plan to make, **(iii)** explain (at least roughly) what kinds of hypotheses you might test with these comparisons. Proposals are graded on effort, but I will provide feedback about the direction, scope, and feasibility of your project. You should expect to revise your project plans in light of this feedback. Your proposal should be 1-2 double-spaced pages.

- Note: It may help to start by thinking about your project as answering a question you find interesting and choose your topic accordingly. For example, when are the computational costs of using a more complicated heuristic justified by the increased efficiency of search? In this case, you might compare an algorithm with no heuristic, a simple heuristic, and a complicated heuristic across different problem types and problem difficulty levels. Your hypothesis will be an educated guessing about whether and when the costs of the more complicated heuristic will be justified.

**Part 2, Progress Report:** Once you have read and incorporated my feedback on your proposal, your next goal will be to implement the code needed for your chosen problem(s). As such, your project report will

consist of **(i)** a revised version of your proposal and **(ii)** a link to a GitHub repository with working problem code. This code should be set up with *random agents* and visually represent the problem in some way. Your final project code will replace these *random agents* with agents that use your chosen algorithms. (If this doesn't sound like something that makes sense given your topic, please consult with me.) You will be graded on the basis of whether your code appears to work as expected and whether you have made appropriate revisions to your proposal. Here are two examples:

- Suppose you want to compare the performance of several algorithms on the game checkers. You will write code for this game and for an *random agent* class that selects legal moves at random. When I run your code, I should see a simple visual representation of a game between two random agents.
- Suppose you want to compare the performance of several algorithms on the traveling salesperson problem (i.e., the problem of finding the shortest route that passes through all cities on a map and returns to the starting position). For this problem, you will probably want to generate several test maps, each with a set of cities and a network of roads connecting them. These could be generated randomly, based on real world maps, or hand-crafted. Next, you will want to write code for a *random agent* class that selects the next city to visit at random until all cities have been visited and the starting city is reached. When I run your code, I should see some visual representation of these random routes (e.g., an ordered sequence of cities visited on the route or a map with a line representing the agent's route).

**Part 3, Final Report:** Your final report should **(i)** briefly introduce your project, **(ii)** explain the AI algorithms and problems you have chosen, **(iii)** explain the specific question you hope your project will answer, **(iv)** describe the comparisons you will make to answer that question, **(v)** explain what results you initially expected from those comparisons, and **(vi)** explain why you expected those results. For these sections (i-v), you may copy any relevant material from your project proposal. Next, your final report should **(vi)** describe the results you **actually** observed and **(vii)** try to explain any results that surprised you.

Your final report will **not** be graded on whether you made the correct prediction. It **will** be graded on whether you appear to have implemented a sufficient number of problem(s)/algorithm(s) correctly and whether you have offered a plausible and thoughtful explanation of your results. Your final report should be approximately 6-7 double-spaced pages in length and include a link to a GitHub repository with the final version of your project code.

- Note: You should write your report on the assumption that I have never heard of the algorithms you are comparing. I need not be able to implement them based on your explanation, but I should fully understand (i) the basic idea of how they work and (ii) any differences between them that are relevant to the comparisons you plan to make or the results that you predict.

**Technical:** Your code should be set up to generate **all** comparison results when I run it. If you generate figures with matplotlib, the code should also generate those figures. If you generated the data in Python and created your figures in another program (e.g., Excel)--that is fine so long as the data is generated by your code when I run it (e.g., in the form of a .csv file). You should plan to write **all** of your code from scratch. This means implementing both the code for your problem *and* your algorithms. Please annotate your code with comments that will help me understand it. This is vital in assigning partial credit if something is wrong with your implementation. Visual representations of the problems/solutions may be as simple as they need to be to clearly convey the relevant information (e.g., a grid of characters printed to the terminal to represent a checkerboard). I will run all code within a Python environment that includes **only** NumPy, SciPy, matplotlib, and OpenCV (for displaying images). If your project requires another library, you must clear it with me in advance.

**Academic Integrity:** You are **not** permitted to use any code that you did not personally write--whether it comes from another human being or an AI system, like ChatGPT. You may read articles on algorithms, converse with an AI system about key concepts, or gather general information from any source so long as you clearly **cite it** in the references of your final report (e.g., by linking directly to your AI chat log). You must make

[redacted] [redacted]

**regular updates** to your GitHub repository and **describe** any important changes you have made. I expect **at least** 10 commits before your progress report and 10 additional commits before your final report. Failure to do so will result in **(i)** a significant reduction in your project grade and **(ii)** a careful inspection of your code for plagiarism. If plagiarism is found, the sanctions described in the syllabus will apply.

[redacted]