Project Overview
(15% of course grade)

Introduction

Your course project will allow you to experience several phases of designing and processing a simple task-oriented programming language, which we'll call the **Cleaning Language**. This language is used to write programs that direct a simple **Agent** to walk through a two-dimensional **Cleaning World**, represented as a grid, performing a cleaning task. The rules governing the **Cleaning World** are described below. Your responsibility in this project is to design a programming language that allows you to represent the Cleaning World and program the Agent to perform the cleaning task.

Project Contents and Structure

You will be responsible for different aspects of the language and stages of its processing:

- Designing the allowable elements of the language, including defining the syntax of lexemes and programs
- Writing the lexical analyzer
- Writing the parser
- Writing a minimal static semantics analyzer
- Generating code from the Abstract Syntax Tree

The expectations for each phase will be detailed as we get to each one. You will turn in work for each stage according to the schedule below. *Note that dates past Nov 5th may be changed a little.*

Delivery	Due Date	Points
Part 0: Team formation	Wed, Oct 29th, 10 am	--
Part 1: Language design	Wed, Nov 5th at 10 am	30
Part 2: Lexical analyzer	Wed, Nov 5th at 10 am	20
Part 3: Parser	Tue, Nov25th at 10 am	35
Part 4: Static semantics analyzer	Tue, Nov25th at 10 am	15
Part 5: Code generator / Interpreter, & Final Submission	Tue, Dec 9th at 10 am	50
	TOTAL	15%

Note: Code generation is a topic that we have not covered in this course, and so this part will not be complex, also because the task targeted by the project does not require a complex language.

Iteration: As you go through the phases of the project, you will find that you need to go back to previous phases and make some changes. So, you will be able to resubmit all phases of the project with the last submission, including a description of

what you changed, if anything, and why. The final grade for the project will consider the improvements you made.

Groups

Group membership: The project is to be done in **groups,** which will ideally consist of three members. I will consider teams of two and teams of four under exceptional circumstances. I will not accept 1-person "teams". With the current number of students across the two sections, there will be approximately 13 teams of three members. You can understand why I don't want smaller teams and my experience is that larger teams have a tendency to fall apart. Team members can be in different sections. Form your teams **here**. Consider whether your personalities and schedules favor the team.

Team Captain: For each stage, I will appoint a Team Captain, alternating among team members, whose responsibility, in addition to participating in the technical work, will be to keep the team on track, schedule meetings, agree on how the work will be divided and scheduled, make sure every member of the team does their job according to the agreed upon schedule, and intervene to resolve conflicts (if necessary, consult the instructor about team coordination issues).

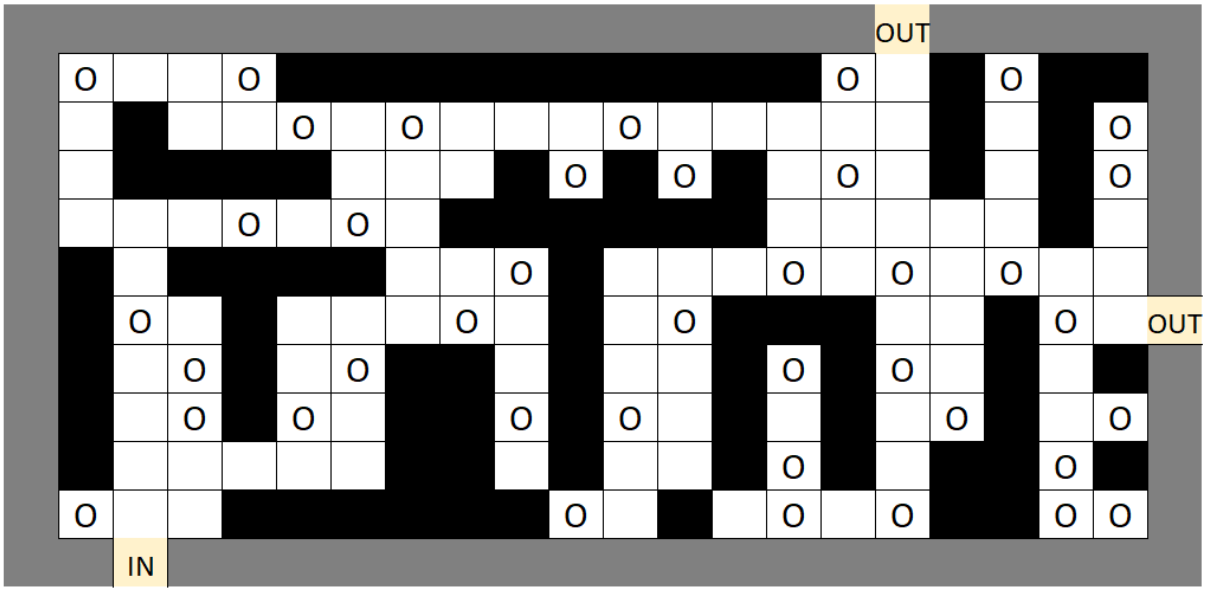
Team Captain Report: With each delivery, the Team Captain will turn in a brief (no more than one page) summary of how the team organized itself for that delivery, the issues it faced, and how they were resolved. Once the Teams are formed, I will use the Teams formation sheet to appoint the captain for each phase of the project. If you see a **C*n*** next to your name, it will mean you are the captain for delivery ***n***.

The Cleaning World

The figure below shows a two-dimensional Cleaning World. Its *height* and *width* don't have to be the same, but neither can be smaller than 3. These will be values that should be set as program constants.

There isn't just one Cleaning World. Cleaning worlds contain obstacles, paths, and dirt, an entry point and one or more exit points, which can be anywhere along the borders of the world except where the entry is. You design the placement of the obstacles, the paths, and the dirt (in one or more locations). The Cleaning World thus needs to be initialized by your code to reflect the conformation you desire.

The figure below shows what a sample Cleaning World might look like. The gray wall all around shows the limits of the Cleaning World. This world has height 10 and width 20, and two exits. The black blocks represent obstacles; the white blocks are part of the path. The 'O's are the dirt to pick up (you can use another symbol, it doesn't have to be an 'O').



The Task

An Agent enters the Cleaning World, and its task is to exit after having cleaned up everything it can. Obviously, if you put the dirt in an inaccessible place, it won't be able to clean it up. The Agent needs to keep track of how much dirt it has picked up because it will need to report it when it exits. In the sample World above there are 40 dirt items and they are all reachable.

The Agent doesn't know what the Cleaning World looks like. It only knows what its dimensions are, where the entry is, and in which direction it points when it enters. In the sample World above, the agent would be placed at the cell labeled 'IN' and it points North. You can assume that the Agent is always positioned so that, if it just moves forward, it will enter the World. Once inside the Cleaning World, the Agent must find its way around the world by sensing its environment.

As the agent moves around the Cleaning World, it should incrementally build its own map of the world so that, if it ever needs to go back over locations it has already been, it can do so without sensing but using the knowledge it has gathered during exploration.

The Cleaning World is fundamentally a maze where, in addition to finding an exit, you also need to clean up as much as possible and simultaneously build a map of the maze.

The Agent's Knowledge and Actions

The following is an initial list of the Agent's knowledge and the actions it can take. You may need to expand it, if you think it's not sufficient.

Directions: North, East, South, and West (you can abbreviate them to N, E, S, W).

Sensing: The Agent can sense:

- ***Obstacles***, if it is facing them, i.e., it can only sense in the direction in which it is moving.
- ***Dirt***, if it is in location where the Agent is.
- ***Entrance*** and ***Exits***, if they are in front of it or at its sides, but it is not allowed to exit where it entered. It must find an exit.

Moving: The Agent can move only one location at a time.

Turning: The Agent can turn right by 90° or turn left by 90°.

Cleaning: The Agent can pick up dirt, but only in the location where the Agent is.

What Is Next?

The **Project Parts 1&2** handout tells you what you need to do for the first deliverables of the project.

Questions?

Do you have questions? Ask them by mail, I will answer them **here**.