CSCI 204 – Data Structures & Algorithms

Can you help a drone escape a battlefield using recursion?

Phase 1 deadline: Monday, 2/27. 11:55 pm

Phase 2 deadline: Monday, 3/20. 11:55 pm

Policies

Late policy, you are allowed to use two 24 hours extensions, one for each phase, without any penalty. You should not plan to use it from the beginning; the non-penalty extension is for unforeseen circumstances. Please put a note at the beginning of your code stating why you used an extension if you did so.

If you work with a partner, the non-penalty extension of both of yours is applied simultaneously. Any delay beyond the allowed extension would incur a 10% penalty for each extra day you use unless you have made a specific request and the instructor approved it over email.

The collaboration policy, Unless specified otherwise, you can work with a partner on any project. However, you cannot work with the same partner on more than one project. While you can discuss strategies for completing the assignment with anyone in the class, you may only share the details of your code with TAs, professors, and your partner. You and your partner must fully participate in and understand all project parts. You can't say, "I'll do this half, and you do this half."

When working with a partner, you will fill out a provided partner-assessment form after completing the assignment. At the end of the semester, grades may be adjusted if a student consistently contributes less than equal portions of project work.

Using the Internet

- 👍 You **can** use online resources to help you *understand* a topic in this course.
- ❗ Any online sources you use **must be cited as a comment in your code.**
- 🚫 You cannot search for *codes or solutions to your assigned problems*.

Defining appropriate collaboration

- 👍 You **can** *talk* to students about approaches to a problem and the resources they use to help them solve it. You probably should - that’s one of the advantages of being at a school like Bucknell.
- 🚫 On individual assignments, you are **not** allowed to show another student your code on an individual project or assignment (in group assignments, don’t show another group!). You should not let another student “borrow” the code. This is the *most common case* we (regretfully) bring to the Board of Review.
- 🚫 You are also not allowed to publicly post your code to assignments in this class on GitHub or in a *publicly* accessible cloud storage folder.

Main objectives

The main objective of this project is to master the **design and implementation of a program that can help a drone escape a battlefield.**

- **Phase 1:** Design a battlefield solver that operates recursively
- **Phase 2:** Design a battlefield solver that uses an explicit stack

Grading rubric

Phase 1 is worth 60% of your grade; phase 2 is worth 40%. While grading, we will look for the program's quality from various aspects, as described below.

1. **Functionality:** 60%, which is distributed as follows.
    1. User-friendly and well-formatted input and output (10%)
    2. Correctness of computation (35%)
    3. Program and readmes satisfy the problem specifications (15%)
2. **Organization and style:** 30%, which is distributed as follows.
    1. Program follows an object-oriented design approach and contains at least classes for **Stack**, **Battlefield**, and **Drone** (10%)
    2. Python style, particularly variable/function names, comments, and docstrings; each code file has comments on the top, including name, date, and file description (10%)

    3. Code is structured, clean, organized, and efficient. Files generally only have one class (10%). Avoid leaving commented-out legacy code in your submission to earn a full grade in this category.

3. **Extraordinary elements, creativity, and innovation:** 10%
   This includes anything beyond the provided specification that makes your program stand out.

Submission

Must run: make sure you test all your programs before submission. The submission will only be graded if the code runs**.** It is to your advantage to work iteratively, building and testing features so that whatever you submit compiles successfully.

Separate Repls: Phase 1 and Phase 2 repls should be submitted separately as we did in Project 1 on their respective deadline. The repls must remain the same after the submission deadlines. Otherwise, they will incur a late penalty.

Readme: the readme is the most underrated and ignored file. Please make sure you submit one. It would be a plain text file named **README** for each phase.

- Name the file **README1.md** for phase one and **README2.md** for phase two.
- The **README** file should contain:
  - A Couple of paragraphs describing one or two points in your program that you like most
  - One or two points that you had the most challenges.
  - Any point of "extraordinary element, creativity, and innovation" you feel you added if you are going for those points.
  - Notes on how you incrementally tested your code
  - Note: **README.p2** should also include the output of at least one small test case you created called **battlefield4.dat** (in the same format as other battlefields), designed to test something interesting. Include a few sentences describing why it is an interesting test. Make sure that **battlefield4.dat** is included in your replit repository.
  - The written portions of **README** files should not be longer than a page.

To whom the code belongs: make sure that your and your partner's names are marked in a comment in the readme and a comment in all your .py files (main.py, battlefield.py, dronerecursion.py).

Partnership log: fill in the following log if you worked with a partner. **Please upload the filled in log to the Project Repl for each phase**. A format is given on the next page. **Y**ou fill it only if you worked with someone else; otherwise, ignore.

Pair Programming Log for Project 2 Phase —

Date/Time	Task	Time spent	Driver	Navigator	Notes

We understand that teamwork is a collaborative process where both partners work together. Each partner is expected to code ("drive") roughly 50% and observe/suggest (“navigate”) 50% of the time the team is working together. The objective is to work together, learning from each other, not to divide the work into two pieces, with each partner working on a different piece. Therefore, at most, 20% of an individual’s effort for an assignment should be spent working alone. By signing below, we certify that the above log accurately reflects how we spent our time on this project.

Member1: Skillset:

Member2: Skillset:

Skillset examples include logic, programming, debugging, writing, timeliness, problem solver, etc.