AI – Assignment 2 [Due: Feb 27, 3:30pm]

Rules:

- Plagiarism of any kind will result in failure for the course. No exceptions. If you are not sure whether or not you are plagiarizing from the web, classmates, or any other person/source, it is your responsibility to clarify it with the professor before submitting the code.
- This is an individual assignment. You are not allowed to work with any other student.
- You are allowed to use the web to learn more about simulated annealing, MC, queens problem. You are not allowed to use or copy source code from the web.
- You are allowed to write the code in any programming language. You should submit all your source files and a README.txt file which needs to describe how to compile and run the code. The README.txt file should also contain the names and email address of each team member.

  You will receive 0 points if the code does not compile, i.e., generates errors during the compilation.
- Submit a printout of your code in class. Submit the source code and the README file as one zipped directory via Blackboard.

1 Solving Queens-and-Knights Problem with Hill Climbing

Given an $N \times N$ board, Q queens, and K knights, the objective is to place the queens and the knights on the board so that

- no queen attacks another queen
- no knight attacks another knight
- no knight attacks a queen

Note that it is ok for a queen to attack a knight.

The program will take the following arguments from the command line: `N Q K tmax solution.txt` where `N` is the board size, `Q` is the number of queens, `K` is the number of knights, `tmax` is an upper bound on the runtime (in seconds), and `solution.txt` is the name of the output file.

The program should use hill climbing to solve the problem. Specifically,

```
s <-- random placement of queens and knights
while s is not a solution and time < tmax
  s <-- neighbor with the smallest number of conflicts
print s
```

At the end, the program should write the solution to the file `solution.txt`. The solution should show print the board. For example, the board below

```
|K|K| |Q|K|
|K|Q| | | |
| | | | |Q|
| | |Q| |K|
|Q|K| | | |
```

shows a solution with 5 queens and 6 knights.

`tmax` is an upper bound on the runtime (in seconds) for your program. If your program does not find a solution within this time limit, it should stop the computation and print -1 for the column positions.

Make sure to cite the sources that you are using for determining an appropriate annealing schedule (how you change the temperature).

2 Requirements for Graduate-Course Enrollment

If you are taking AI as a graduate course, you also need to implement Simulated Annealing. See

`https://en.wikipedia.org/wiki/Simulated_annealing`

Your program should take an additional argument, `method`, which should be HC for Hill Climbing or SA for Simulated Annealing. So, the list of arguments would be

```
N Q K tmax solution.txt method
```