CS 4810/6810 Fall 2017 Midterm Name: _____

You have 70 minutes

Please turn off all cellphones, pagers, etc.

You may ask questions about the problems, but not about your answers.

1. NetLogo:

Given the following NetLogo code, write:

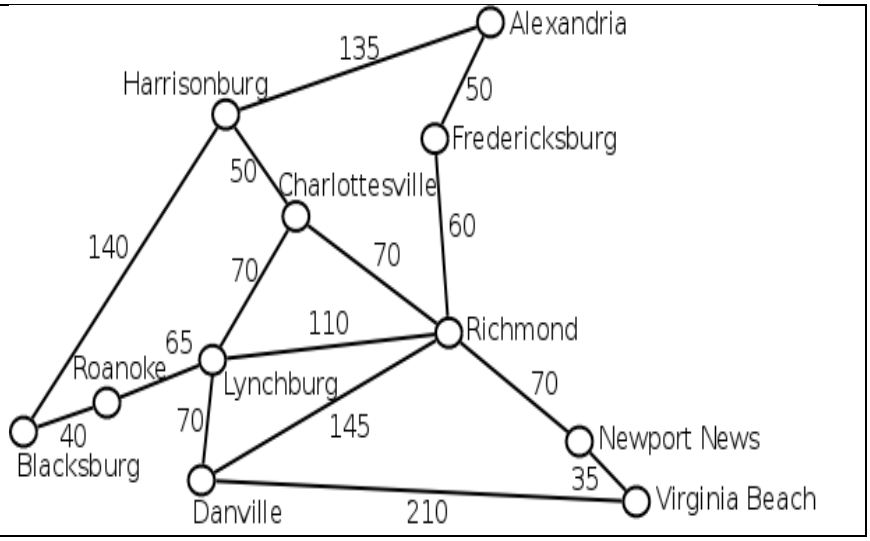
- A NetLogo reporter *breezy* for an individual turtle that returns true if there is a pit in any neighboring patch (above, right, below or left). For the sake of tidiness, you do not have to check if the neighboring patch is valid.
- a NetLogo command *move* for an individual turtle to move it in the chosen heading. Assume your turtle has integer coordinates.
    - Display "you feel a breeze" if any neighboring patch has a pit. You should use *breezy* to help you do this – assume it exists even if you have not written it yet
    - Display "you fell into a pit and died" if the new patch has a pit

```
patches-own [has-pit]
; not required check if coordinates are valid
to-report valid-xycor [x y]
  report x >= min-pxcor and x <= max-pxcor
    and y >= min-pycor and y <= max-pycor
end

; check if above, below, right and/or left patches have a pit
to-report breezy
  report any? neighbors4 with [has-pit]
;; for a more traditional solution, with the not required checks for valid squares
thrown in
;  report (valid-xycor (pxcor - 1) pycor and [has-pit] of patch (pxcor - 1) pycor)
;         or (valid-xycor pxcor (pycor - 1) and [has-pit] of patch pxcor (pycor - 1))
;         or (valid-xycor (pxcor + 1) pycor and [has-pit] of patch (pxcor + 1) pycor)
;         or (valid-xycor pxcor (pycor + 1) and [has-pit] of patch pxcor (pycor + 1))
end

; move in a direction (0, 90, 180, 270, ...) and update status
to move [head]
  set heading head
  forward 1
  if [has-pit] of patch pxcor pycor [
    show "you fell into a pit and died"
    stop ; game over
  ]
  if breezy [
    show "you feel a breeze"
  ]
end
```

2. (A*) For going from Danville up to Alexandria on the following map, show the status of the A* algorithm after each of the first 3 nodes is chosen for expansion: include the search tree and the values of the *f, g* and *h* functions, and the candidates to expand after the first 3 nodes. Use the straight-line distance as the heuristic function.

Straight line distances to Alexandria (if you are having a hard time reading the map, the version for the last problem is larger)

Blacksburg	249
Charlottesville	110
Danville	230
Fredericksburg	45
Harrisonburg	122
Lynchburg	176
Newport News	161
Richmond	99
Roanoke	223
Virginia Beach	214



1st node to expand: Danville $f = g + h = 0 + 230$

Lynchburg $f = 70 + 176 = 246$ Richmond $f = 145 + 99 = 244$ Virginia Beach $f = 210 + 214 = 424$

2nd node = Richmond

Lynchburg $f = 255 + 176 = 431$ Charlottesville $f = 215 + 110 = 325$
Fredericksburg = 205 + 45 = 250 Newport News = 215 + 161 = 376

3rd node = Lynchburg (246 < 250)

For 6810 students, do all remaining questions. For 4810 students, you may pick 2 of the last 3 questions, or you can do all 3. If you only pick 2, each question is worth 25 points. If you do all 3, each question is worth 20 points.

3. (Agents) Describe how you might represent the states and possible actions for the following problem.

You are asked to implement a program to control a coffee machine. The machine should be able to be set to start making coffee at some time in the next 24 hours (ex: at some time in the morning so the coffee is waiting for you when you get up)

Indicate any assumptions you are making (this problem description is quite incomplete).

States: machine is idle, making coffee, programming, waiting to make coffee or warming the coffee
(assume powered off = program is not running)

Actions:
Can initiate programming to move the machine from idle to programming
Can set the timer to move the machine from programming to waiting.
Can cancel to move to idle from any other state.
Can start making coffee to move the machine from either idle (no delay) or waiting to making
Can stop the coffee making process and shift to warming

4. (Constraint Satisfaction)

[This is a version of the photo arrangement problem, but with hard constraints instead of soft constraints.]
Abby, Bob, Carl, Diane, Edna and Felix are going to line up for a photo. Here are the constraints:

- Diane can be anywhere to the left of Carl
- Bob and Edna need to be next to each other
- Abby insists on being at one end of the line
- Felix cannot be next to Abby
- The starting order is, left to right: Bob, Carl, Diane, Edna, Abby, Felix

Step through the first iteration of min-conflicts towards an ordering that satisfies all constraints– you may not actually complete the solution. On each iteration, you will swap the chosen person with another. Explain how you choose the first person to move and where to move them.

For starting order: # violations = 4, individual violations are:
Bob 1
Carl 1
Diane 1
Edna 1
Abby 2
Felix 1

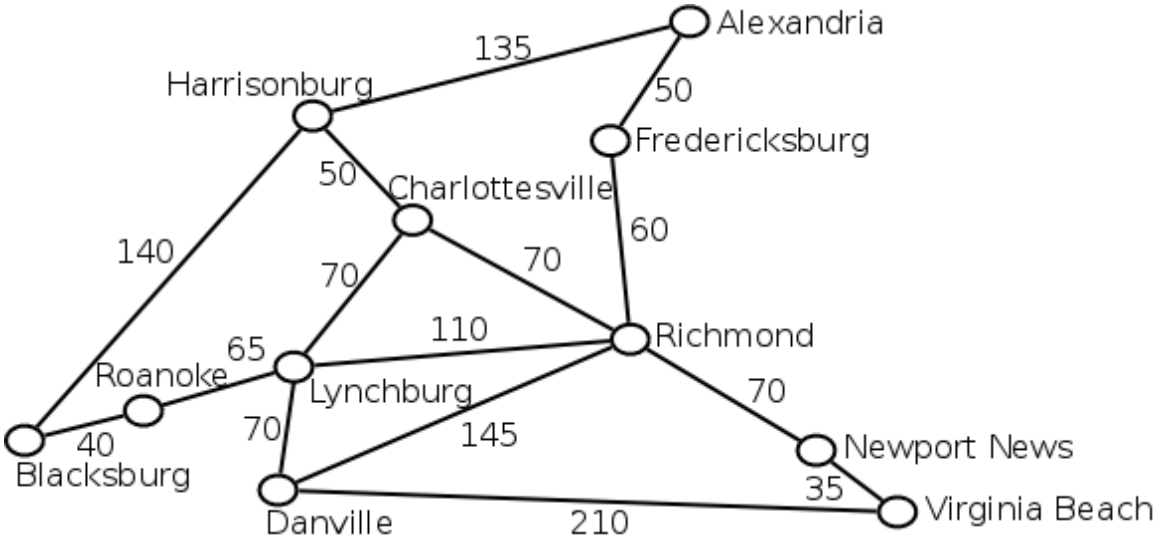
So move Abby. If we swap with each person, here are the total # violations:

Bob 1
Carl 2
Diane 3
Edna 3
Felix 3

So swap Abby with Bob

5. (Iterative deepening)

For traveling from Danville to Alexandria, show the order of nodes being explored to a depth of 2



Depth = 0: Danville

Depth = 1: Danville and its neighbors
Lynchburg, Richmond, Virginia Beach

Depth = 2: Danville and its neighbors
Lynchburg, Richmond, Virginia Beach and (depth 1)
Lynchburg's neighbors: Roanoke, Charlottesville, Richmond
Richmond's neighbors: Lynchburg, Charlottesville, Fredericksburg, Newport News
Virginia Beach's neighbors: Newport News (depth 2)