Critter Class

Protected:

Col – col number of critter

Row – row number of critter

Age – age of critter

Public:

Virtual move – move the critter according to rules specified

Increment age – add 1 to age of critter (call at end of day)

Virtual breed – breed according to rules specified

Virtual starve (I do not think this needs to be in critter since it is doodlebug specific but the rubric asks for it to be overridden). – deletes doodlebug if starve conditions met

SetAge – sets age of critter to specified (will probably use to reset age after breeding)

virtual printType – prints the type of critter, useful for printBoard function

Doodlebug:

Protected:

lastMeal – days since the doodlebug ate

Public:

move – read values of spaces around doodlebug, call eat function if applicable, otherwise move randomly, stay in place random direction is invalid.

eat – moves into first detected adjacent cell with an ant in it, eats (deletes) the ant [optional function]

breed – if age >= 8, create another doodlebug in random, empty adjacent cell, reset doodlebug age to 0, if no adjacent cell, proceed as normal

Starve – if lastMeal >= 3, delete doodlebug

incrementLastMeal – adds 1 to lastMeal

setLastMeal – sets last meal to specified integer (maybe -1, and then gets incremented to 0 at end of day)

printType – prints a 'X'

Ant:

Public:

Virtual move – move to random direction, stay in place if direction is invalid

Virtual breed – if age >= 3, create another ant in random, empty adjacent cell, reset doodlebug age to 0, if no adjacent cell, proceed as normal

printType – prints an 'O'

Board

Private:

numRows – maximum number of rows (this will help with EC)

numCols – maximum number of columns (this will help with EC)

numAnts – starting number of ants on board

numDoodlebugs – starting number of doodlebugs bugBoard - 2D array (must be dynamic, will be pointer to a critter (use printType to determine ‘X’ or ‘O’ and NULL to determine ‘ ‘)

Public:

placeDoodlebug - (for loading initial doodlebugs, take numDoodlebugs as parameter)
placeAnt – (for loading initial ants, take numAnts as parameter)
moveDoodlebugs – loop through board, moving doodlebugs, incrementing age and lastMeal
moveAnts – loop through board, moving ants, incrementing age
breedDoodlebugs – breed eligible doodlebugs
breedAnts – breed eligible ants

Menu:

Get number of steps
EC get number of ants, doodlebugs, rows, and columns (verify ant and doodlebug numbers based on row and column input, limit row and column size as we decide is appropriate) Use overloaded board constructor to initialize these values