ypety//add beetle starving counter++

//add beetle eating ant action

Main.cpp

- Initialize 2D array to store initial gridStore filename and number of turns into

-      find size of row and set it = to number of columns

-      find number of lines in input and set it = to number of rows

-          Initialize 2D array of creatures with size of grid to store initial grid

-     store each spot in the map into the 2D array

Nested for loop going through each spot in array to check for ants, make ant’s move, then check for beetle, make beetle move, then call starve method for beetle

Creature.h

* Public:
  + Move() =0;
  + Breed() =0;
* Private
  + int turnsSurvived;

Ant.h

Ant.cpp

* Move(characterArray[][], rowIndex, columnIndex)
  + Check location of beetle in relation to ant at a distance of 1
  + if beetle to north check if south empty and try going south
  + Otherwise if beetle to east check if west empty and try going west
  + Otherwise if beetle to south check if north empty and try going north
  + Otherwise if beetle to west check if east empty and try going east
  + if none, repeat process with distance increased. Repeat until edge of board reached.
    - If no beetles found, don’t move
* Breed(characterArray[][], rowIndex, columnIndex)
  + If turns survived = 3
    - Then check if adjacent spot north is empty
      * If empty, add ant there and reset survival counter
    - But if adjacent north spot not empty, check east, then south, then west
    - End checking when spot found( break;) and reset counter.
    - If spot not found, reset counter for turns survived and don’t breed
  + Otherwise if the number of turns survived is less than 3, increment turns survived counter

Beetle.h

* Public:
* Starve(characterArray[][], rowIndex, columnIndex);
* Private:
  + int northAnts;
  + int eastAnts;
  + int southAnts;
  + int westAnts;
  + int turnsSurvived;
  + int turnsStarved;

Beetle.cpp

* Move(characterArray[][], rowIndex, columnIndex)
  + Check location of ant in relation to beetle at a distance of 1
    - If multiple directions have ant, check for number of ants in each direction(include diagonal and orthogonal directions in relation to selected ant)
  + If no ants available, repeat with distance increased by 1.
    - Try again until distance reaches edge of map
  + If no ants found at max distance
    - Move towards farthest edge of map (if tie, use order of priority N, E, S, W)
* Breed(creatureArray[][], rowIndex, columnIndex)
  + If turns survived = 8
  + Then check if adjacent spot north is empty
    - If empty, add ant there and reset survival counter and end method
  + If adjacent north spot not empty, check east, then south, then west
    - End checking when spot found (break;) and reset counter.
  + If spot not found, reset counter for turns survived and don’t breed
  + If turns survived is less than 8, increment the counter by 1
* Starve(creatureArray[][], bool ateAnt)
  + If turnsStarved is equal to or greater than 5, remove beetle from array(leave spot empty)
  + If ateAnt is true, then reset turnsStarved to 0
  + Otherwise increment turnsStarved by 1