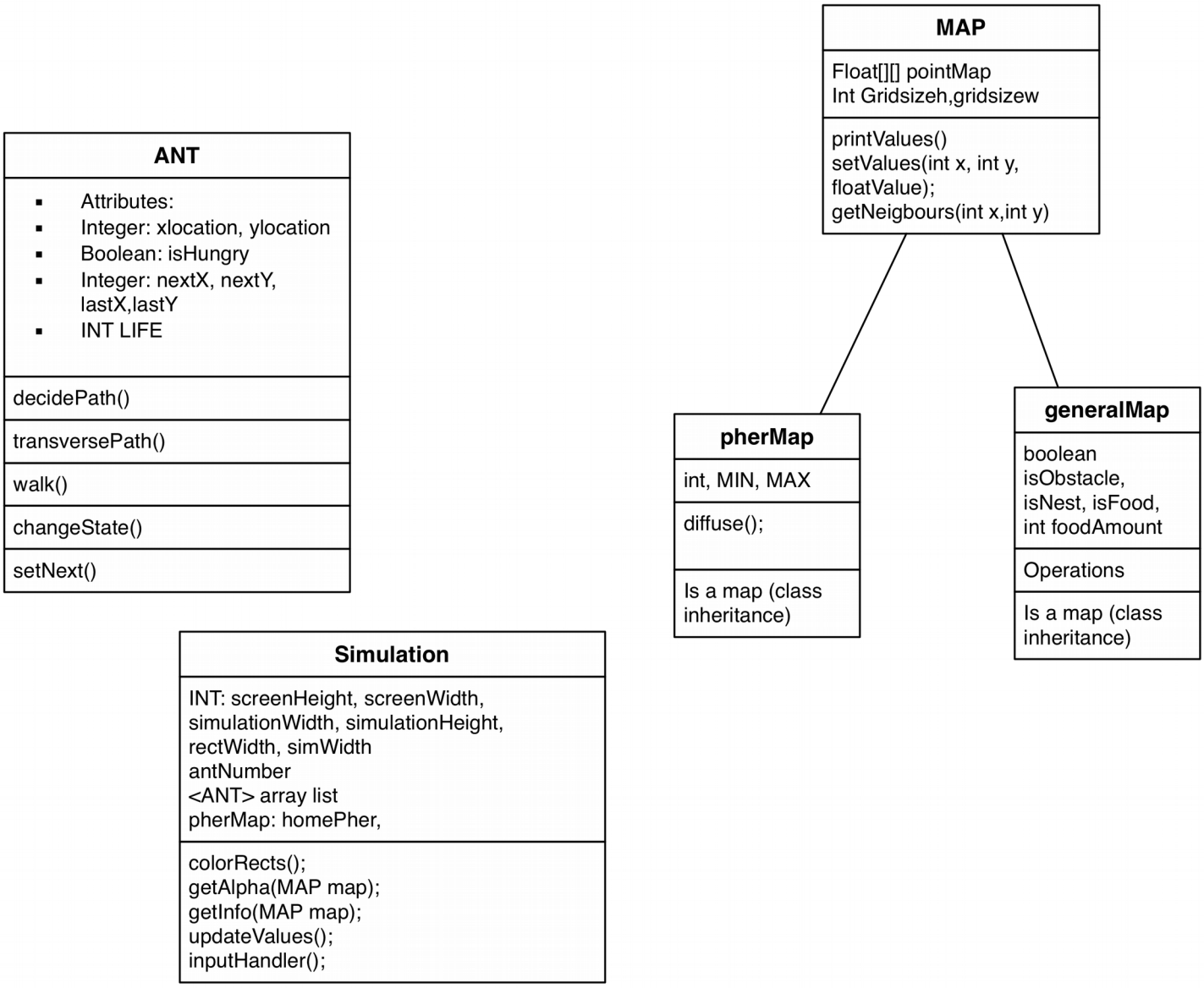
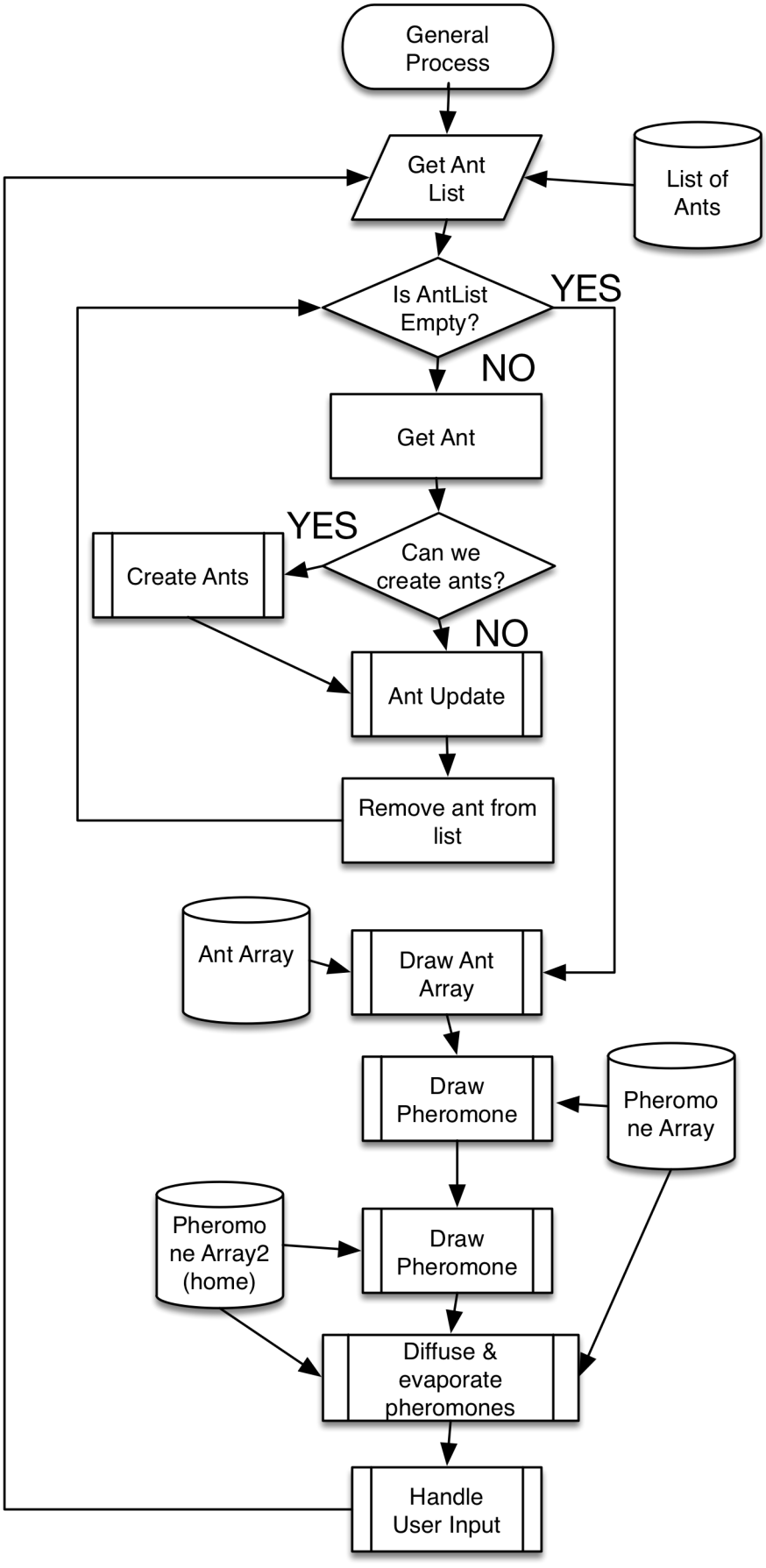
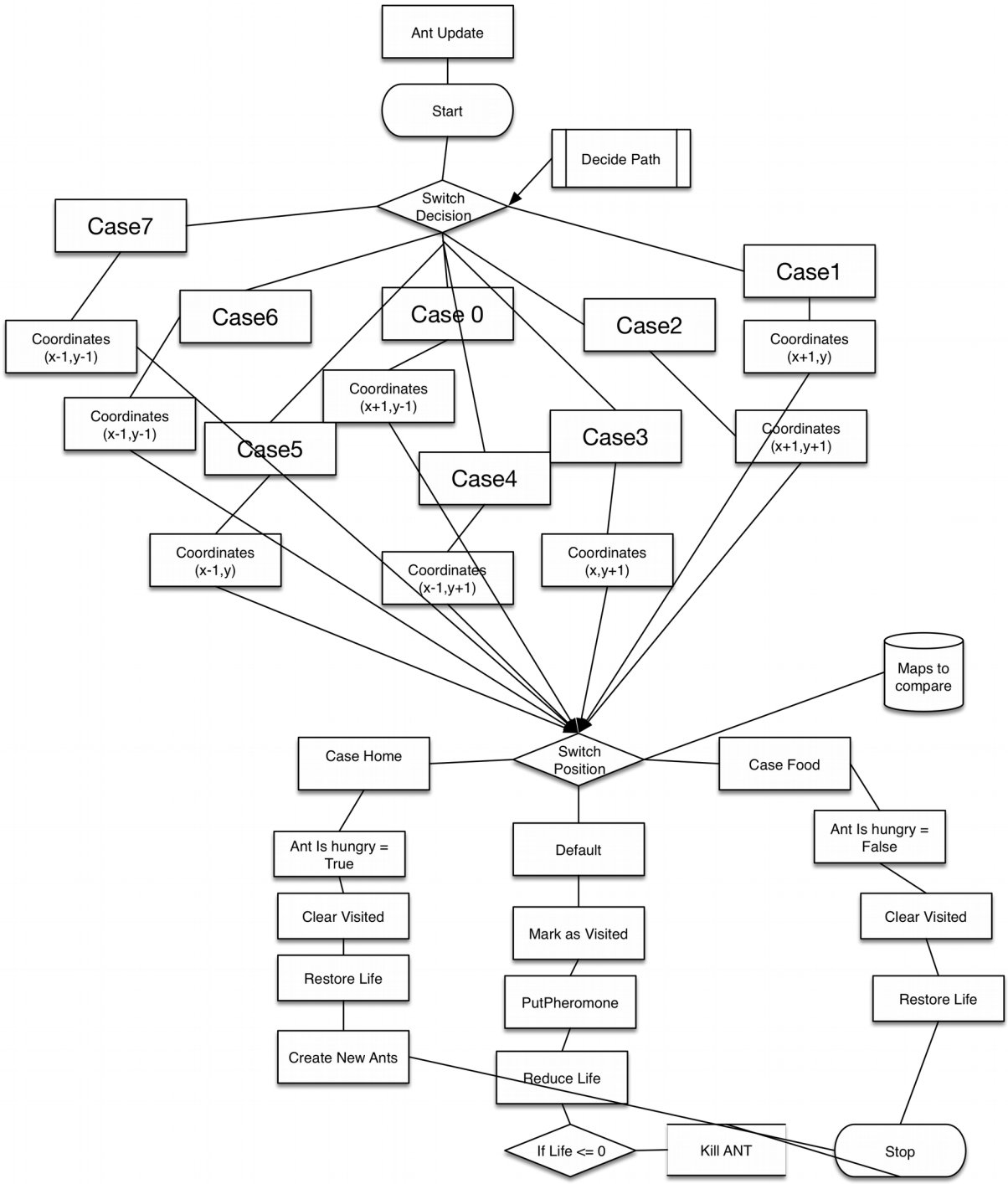
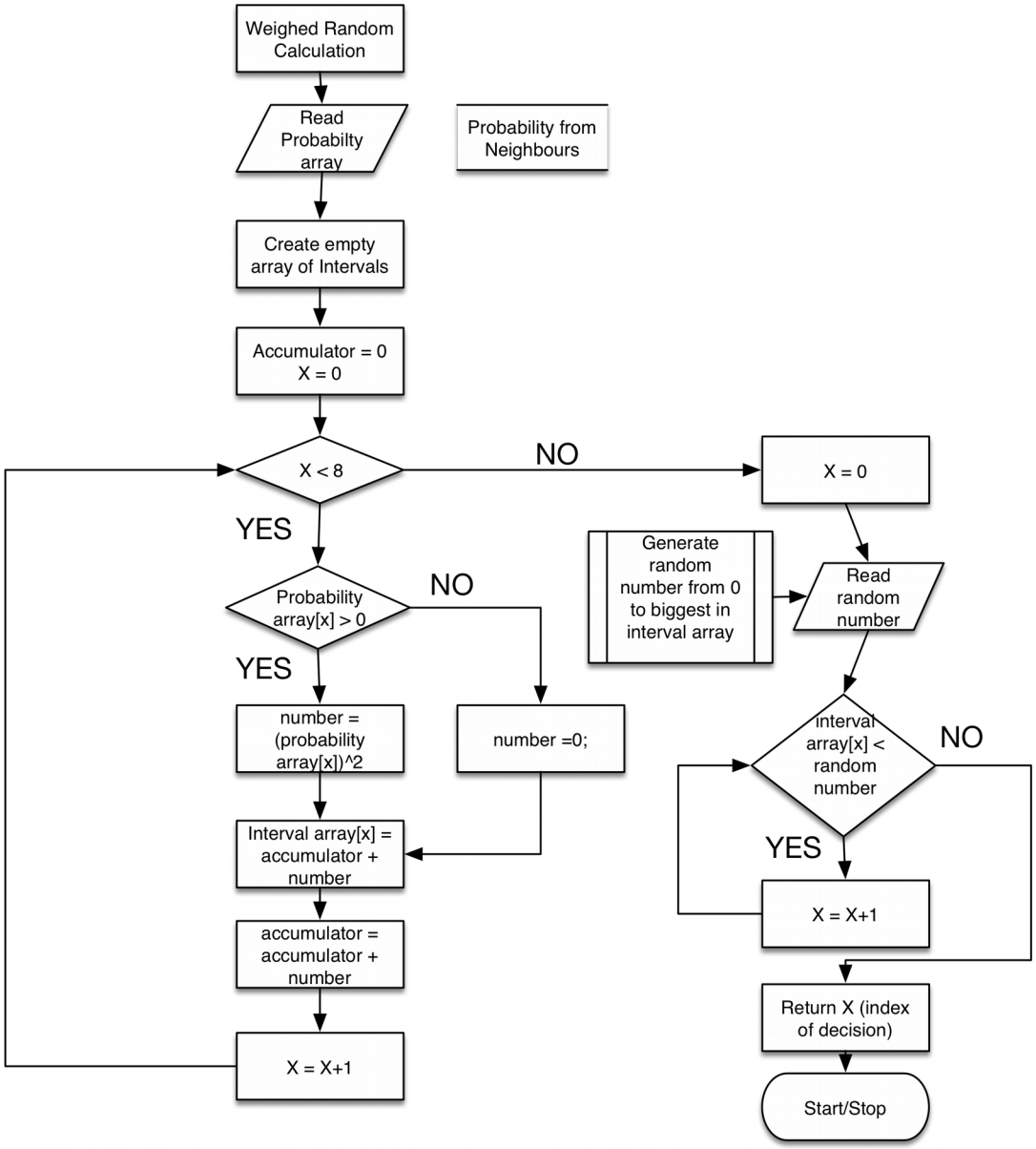
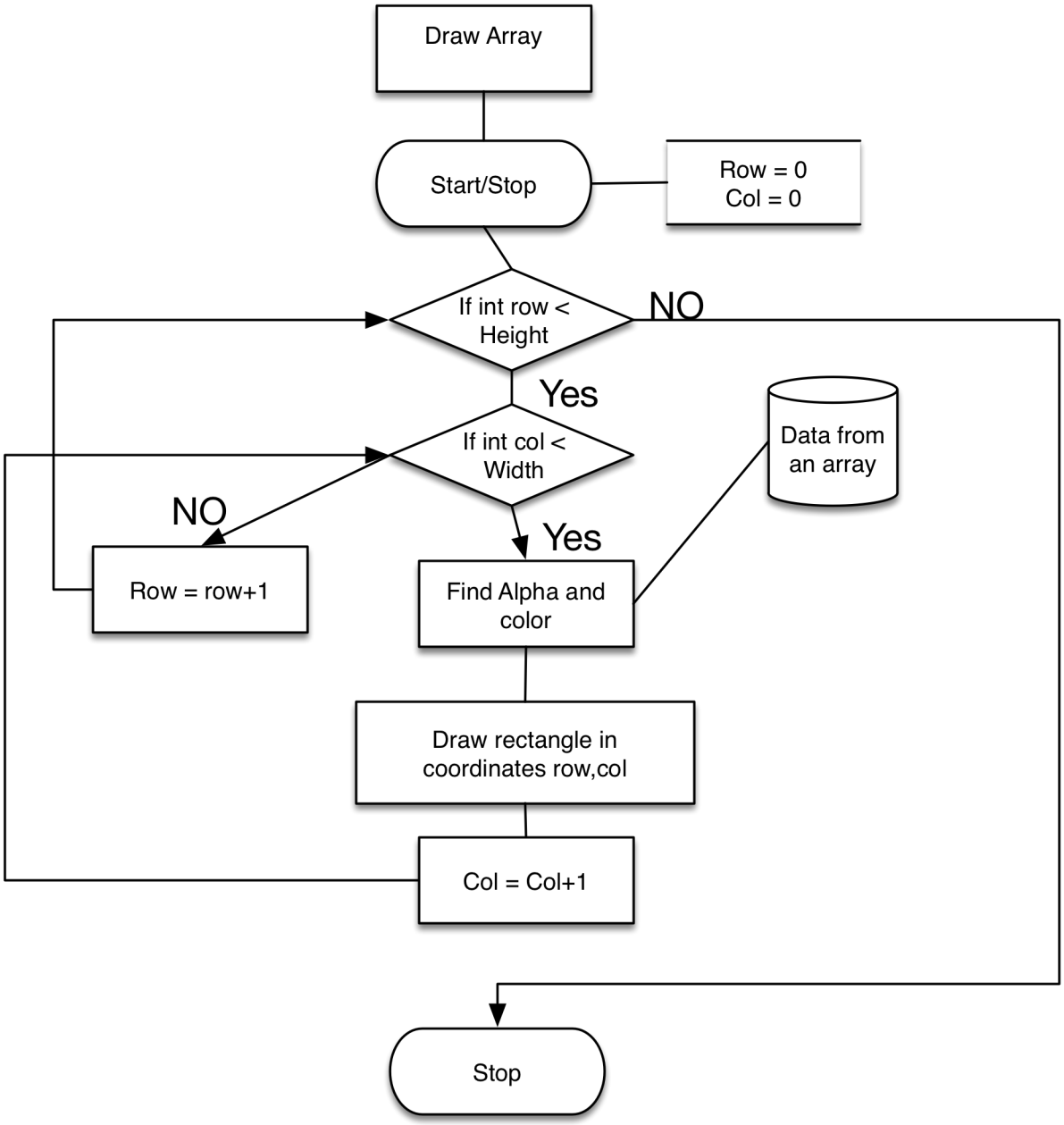
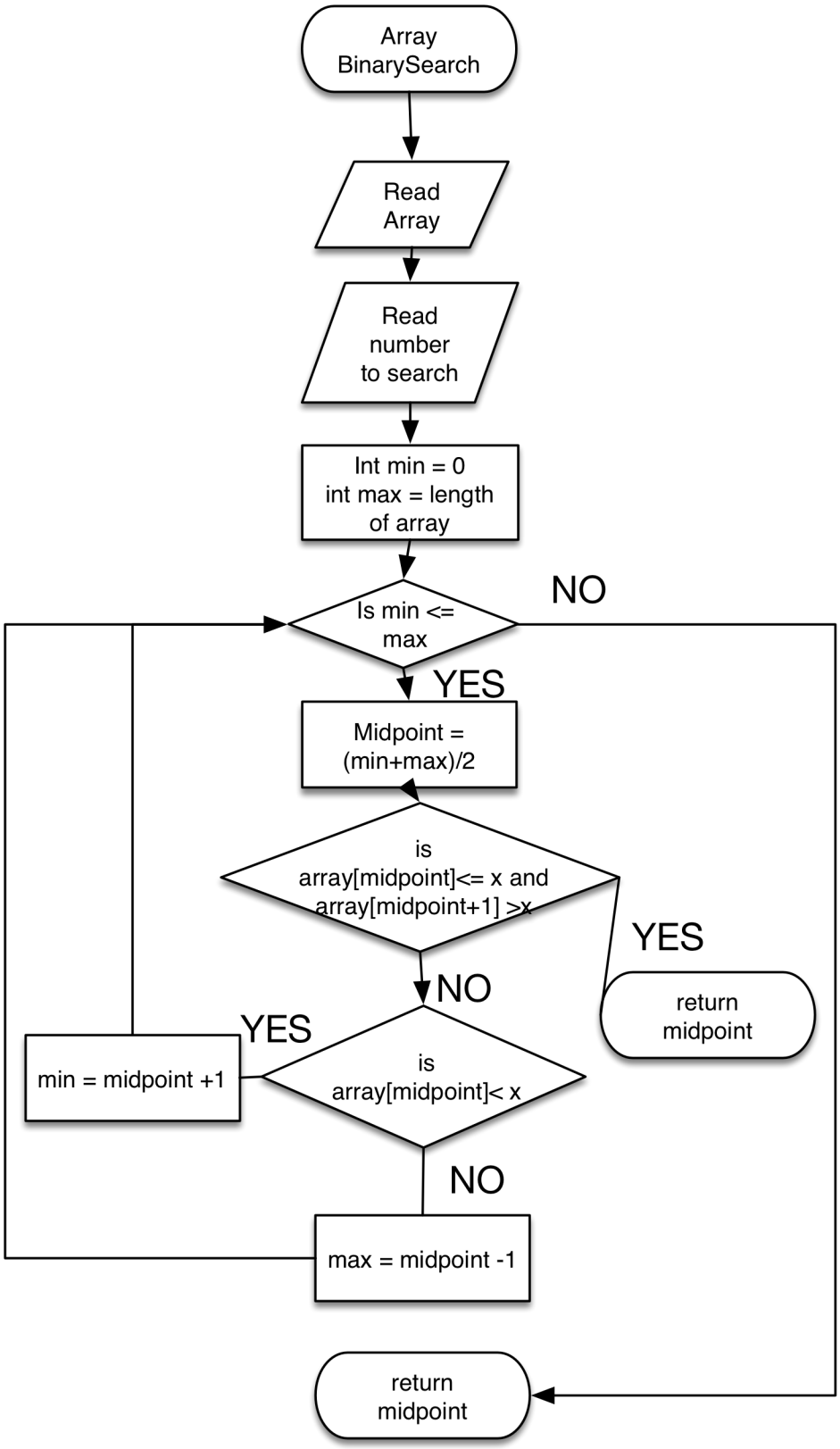

MAP
Float[][] pointMap
Int Gridsizeh,gridsizew
printValues()
setValues(int x, int y, floatValue);
getNeigbours(int x,int y)
ANT
Attributes:
Integer: xlocation, ylocation
Boolean: isHungry
Integer: nextX, nextY, lastX,lastY
INT LIFE
decidePath()
transversePath()
walk()
changeState()
setNext()
pherMap
int, MIN, MAX
diffuse();
Is a map (class inheritance)
generalMap
boolean isObstacle, isNest, isFood, int foodAmount
Operations
Is a map (class inheritance)
Simulation
INT: screenHeight, screenWidth, simulationWidth, simulationHeight, rectWidth, simWidth
antNumber
<ANT> array list
pherMap: homePher,
colorRects();
getAlpha(MAP map);
getInfo(MAP map);
updateValues();
inputHandler();

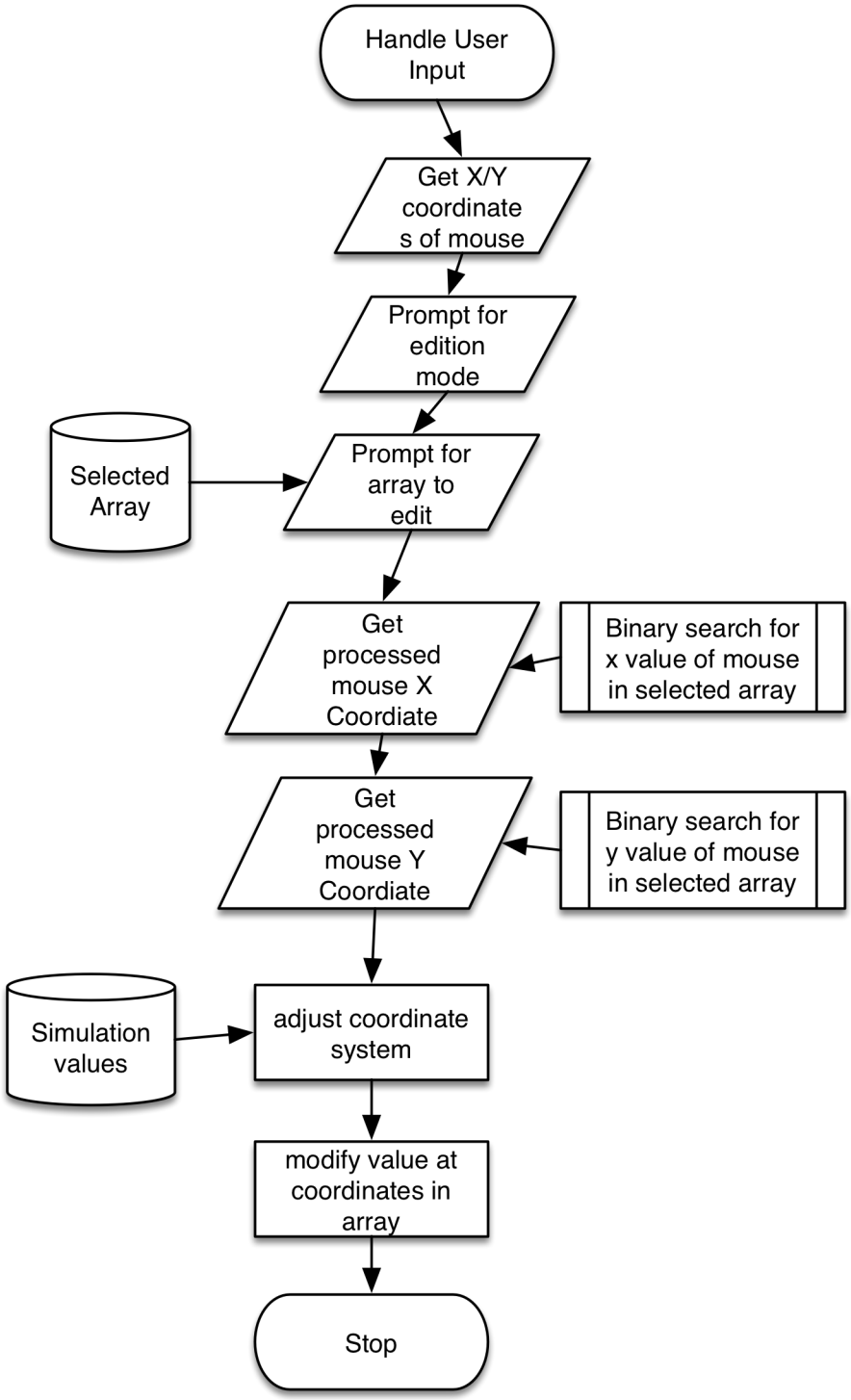

General Process
Get Ant List
List of Ants
Is AntList Empty?
YES
NO
Get Ant
Can we create ants?
YES
Create Ants
NO
Ant Update
Remove ant from list
Ant Array
Draw Ant Array
Draw Pheromone
Pheromone Array
Pheromone Array2 (home)
Draw Pheromone
Diffuse & evaporate pheromones
Handle User Input


Ant Update
Start
Decide Path
Switch Decision
Case7
Case1
Case6
Case 0
Case2
Coordinates (x-1,y-1)
Coordinates (x+1,y)
Coordinates (x+1,y-1)
Coordinates (x-1,y-1)
Coordinates (x+1,y+1)
Case5
Case3
Case4
Coordinates (x-1,y)
Coordinates (x-1,y+1)
Coordinates (x,y+1)
Maps to compare
Case Home
Switch Position
Case Food
Ant Is hungry = True
Default
Ant Is hungry = False
Clear Visited
Mark as Visited
Clear Visited
Restore Life
PutPheromone
Restore Life
Create New Ants
Reduce Life
If Life <= 0
Kill ANT
Stop


Weighed Random Calculation
Read Probabilty array
Probability from Neighbours
Create empty array of Intervals
Accumulator = 0
X = 0
X < 8
NO
X = 0
YES
Generate random number from 0 to biggest in interval array
Read random number
Probability array[x] > 0
NO
YES
number = (probability array[x])^2
number =0;
interval array[x] < random number
NO
YES
Interval array[x] = accumulator + number
X = X+1
accumulator = accumulator + number
X = X+1
Return X (index of decision)
Start/Stop


Draw Array
Start/Stop
Row = 0
Col = 0
If int row < Height
NO
Yes
If int col < Width
Data from an array
NO
Yes
Row = row+1
Find Alpha and color
Draw rectangle in coordinates row,col
Col = Col+1
Stop


Array
BinarySearch
Read
Array
Read
number
to search
Int min = 0
int max = length
of array
Is min <=
max
NO
YES
Midpoint =
(min+max)/2
is
array[midpoint]<= x and
array[midpoint+1] >x
YES
return
midpoint
NO
YES
min = midpoint +1
is
array[midpoint]< x
NO
max = midpoint -1
return
midpoint


Handle User Input
Get X/Y coordinate s of mouse
Prompt for edition mode
Selected Array
Prompt for array to edit
Get processed mouse X Coordiate
Binary search for x value of mouse in selected array
Get processed mouse Y Coordiate
Binary search for y value of mouse in selected array
Simulation values
adjust coordinate system
modify value at coordinates in array
Stop

| Actions to test | Method of testing |
|---|---|
| Ants converge on efficient path (the main algorithm). | Letting the simulation run |
| Ants avoid obstacles. | Placing obstacles and observing ants react by choosing other route |
| The user is able to interact with the simulation by dropping objects. | Ants respond to dropped stimuli, such as food, obstacles or pheromones.<br>The objects placed by the user appear on the map |
| The number of ants is reasonable as well as its born/death rate. | Given enough time, and sufficient food ants do not disappear or fill the screen |
| The graphical representation is relevant and updated correctly. | Check that the level of pheromones and ants in a given space, corresponds to its transparency or intensity. |
| Check edge cases | If no food is available, the simulation should either end gracefully or continue without crashing. |
| Objects interact with each other only in expected ways | Making sure that pheromones, ants or obstacles in a given space do not produce unexpected results, such as crashing the simulation |
| The probability distribution works adequately | Isolating the code and testing it with dummy inputs with known outputs. |