|  |  |
| --- | --- |
|  | Yes. |
|  | Yes; I imagine that in this simulation fox populations change in relation to the amount of rabbits they’re around. |
|  | No, it looks like at least in the first 100 and 150 steps that the rabbits ar vastly out growing the foxes. |
|  | It seems like both populations wax and wane in response to each other. |
|  | Not the same every time. During a long enough simulation the same patterns to appear. |
|  | It doesn’t appear that this simulation will ever wind up in a situation where one side has taken over completely. Rabbits are reproducing quickly and often and more than replenish the food supply for foxes until fox population gets to a certain level. Then foxes die and the rabbits catch back up. |
|  |  |
|  | Steps:  0.  1.  2.  5.  10.  25.  50.  100.  200.  500. |
|  | Because the static reset method is assigning it a “random track” using the setSeed method. Using this method all the calls will be a random number, but they will be the same set of random numbers each time. |
|  |  |
|  | Obviously leaving out any variable will cause some innaccuracy, but for the most part rabbits behavior doesn’t change between gender and with the high rate of reproduction it probably wouldn’t make much of a difference. |
|  | * According to some quickly googled information about Rabbits, our simulator has max litter size at 4 when in reality litter’s range from 4-12. * In our simulation reproduction age is set to 5 turns when in reality it’s 3-8 months. That would mean that each turn is .6 to 1.6 months. Max age is set to 40 turns which is between 2 years and 5.33 years which is much less than the 9-12 years rabbits can expect to live. |
|  | It seems to change the amplitude of the population wave. |
|  | Well it’s pretty unlikely the foxes would just die due to overcrowding. It would be more likely that they would move more times. I would also be concerned about getting exact birthing rates and life cycles. |
|  | It seems like most foxes are dying from hunger rather than age at double life. |
|  | There are a few scenarios that can cause populations to die out, but not many. Nearly all scenarios cause a large variation between populations. If we adjusted fox rabbit consumption to be closer to rabbit birth rates we can get closer to stability. |
|  | Increasing the size of the area doesn’t seem to have a huge amount of change overall, although the futher apart they get the easier it is for stray rabbits to repopulate. |
|  | As long as the area isn’t made to become too small it doesn’t seem to have that much of an effect. |
|  | Yes, decreasing the size of the field can cause problems if it’s made to be too small. |
|  | This just seemed to shorten the amplitude of the wave length |
|  | Considering there wasn’t much change by making the eat all the rabbits around them, I’m not sure how much it will change. I guess that if we make changes such that the foxes will not hunt again until they’re hungry again and allow the hunger to continue without a cap the foxes could find themselves overrun by rabbits. |
|  | The first thing off the top of my head would be if the foxes did a thorough job of eating the bunnies and pushed them into corners where they couldn’t reproduce. This would cause a seemingly stable simulation to fail. |
|  | Ran almost 100 simulations to 250 turns and it doesn’t appear to be catastrophic very often if at all. In mine the average number of foxes was around 300-350 which means that the fox population will initially start higher, but the rabbit population brings it back into line. |
|  | Looks  Looks like it balances back out to me. |
|  | It looks like when animals are created in the simulator class they’re added to an array list that holds their type of object.  The field object has a 2d array set up and stores objects in specific positions of the array.  The hunt method asks field if there are object’s in adjacent rooms and run works similarly. |
|  | Why does simulator even need a copy of the list? Can’t it just iterate over the list created by field? I think that generally storing data isn’t simulator’s job and a separate class should be in control of holding a single list for the other two classes. |
|  | So I set up a way to compare the two, and it took several hours. It’s returning “true” which means that there’s an inconsistency with the numbers. I’m pretty much done with this problem though, forever, until I’m burning in hell and Satan uses it as my eternal punishment. Here are some pictures of my hard work. |
|  | Methods:  isAlive  getLocation  setLocation  incrementAge  giveBirth  breed  canBreed  setDead |
|  | See 28 |
|  | Well yeah sort of, although realistically I’m looking for at function than name. |
|  | We could make a test for each variable and method to make sure that it’s functioning correctly when we’re done. With a test for each item made up, they could all be put into a single test class that ran them. |
|  |  |
|  | Well it’s just about halving the code for our animals. It will obviously make the test class work better because we can just use animal objects. Additionally if we decide to add more animals later it will be no problem |
|  | I would imagine that because everything is technically an Object object that everything would need the same method in them. As it is we know that all Animals have the same the same method for this occasion. |
|  | According to the reading that was right before this, abstract methods can only be on abstract classes. |
|  | Yes, as long as we don’t want to create it directly that shouldn’t be an issue. |
|  | See above. |
|  | I don’t know, probably really important ones. |
|  | Yeah there’s the keyword abstract in front of the methods. Additionally, there’s a tab for abstract methods. |
|  | Because we’re adding methods that MUST be overridden. |
|  | Totes right dawg |
|  | The top line is the rabbit population and the bottom line is the fox population. |
|  | I wouldn’t say that it’s given me any new insights, but it's nice to have the visual representation of the change over time. |
|  | Ok |
|  |  |
|  |  |
|  |  |
|  | Yes it can be moved if the variable names are changed to accessor methods. Then make those two methods abstract in the Animal class and give fox/rabbit actual methods for it. |
|  | Fuck da visibility, I mean half of my answers are probably fucking wrong and nobody’s going to look at my code so why bother. This chapter is taking an absurd amount of time and I still have > 20 questions to go. |
|  | Yeah for the most part. I think there might have been like one thing but I can’t really remember; it’s late. They did a pretty good job. Or there was a lot of changes and I’m too tired to remember so they should have been better encapsulators ffs. |
|  |  |
|  | Yeah I definitely think it is. |
|  | We have to change several instances of a list variable and change the type in a few loops. |
|  |  |
|  | They would be class fields and by definition they would have public, static, and final can be in an interface. |
|  | Can’t have private variables,  Can’t have non finalized variables that aren’t static,  No constructer so no instance variabes,  Can’t make method calls or have methods with fields, |
|  |  |
|  | ArrayList has ways to interact with independent objects in a list of undefined size.  LinkedList has ways to interact with a list where items are linked together. |
|  |  |
|  | Comparable, RandomAccess.  replaceAll, sort, Spliterator |
|  |  |
|  | Used for assigning colors to the actors and the board.  Used to simulate movement.  Used to clear the board. |
|  | I don’t understand why the author would want me to implement simulator view. The outcome he wants are the animals name and numbers at the time in black ink. Therefore there’s no reason to inheret setColor as it causes lots of issues. It seems like this could easily be done with a new class separate from simulator view. |
|  | Yeah, why not? |
|  | Time- based has many more steps without significant action. More accurate but much more data to process.  Event based is much more time efficient because you’re only seeing the events you want. There’s a significant amount of potential variations between steps, so important things may be missed. |
|  | Yah |
|  | No thanks, this assignment is late enough as it is. |
|  | Yes, our animal class is like that.  Not it must be abstract to have abstract methods.  Yes, as long as you don’t ever want to make one. |
|  |  |
|  |  |
|  | I imagine that one is used in most cases and the adapter is used to adapt to different scenarios. Otherwise I don’t know. I mean the they’ll both pass down the methods and the interface will force the next class to override them whereas the adapter wouldn’t. I’m not really sure how useful that is though. I guess if you wanted PrintJobAdapter’s to be a PrintJobListener and wanted to immediately override the methods and not use them. But again, I’m not really sure how useful that would be. |
|  |  |
|  |  |