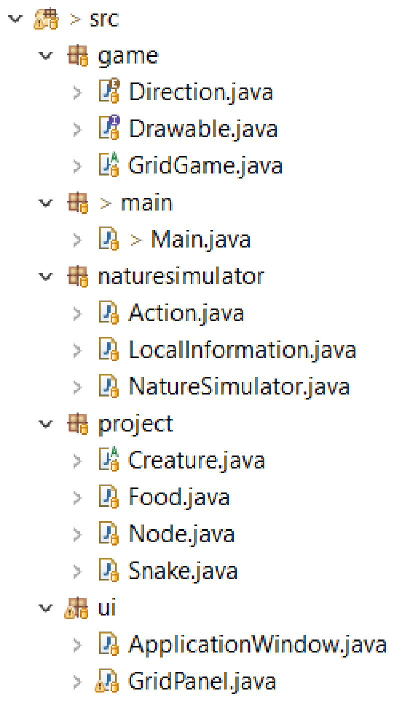
CMPE160 SNAKES PROJECT REPORT

Name-Surname: Emre Girgin   Student Number: 2016400099

Introduction:

This project is a simulation that contains snakes and food. Snakes eat food and grow 1 unit. If a snake reaches 8 unit length, a new snake is produced from the second half of parent snake. That means there will two snakes which have 4 unit length each. Every time a food is eaten, game logic puts a new food into game randomly.



```
> src
  game
    Direction.java
    Drawable.java
    GridGame.java
  > main
    > Main.java
  naturesimulator
    Action.java
    LocalInformation.java
    NatureSimulator.java
  project
    Creature.java
    Food.java
    Node.java
    Snake.java
  ui
    ApplicationWindow.java
    GridPanel.java
```

Outline of packages

Implementation:

game Package:

This package includes classes that determine what will the user see inside the application window.

Direction Enum:

Contains 4 enum representing the directions in grid world.

Drawable Interface:

Classes that implement this interface promises that they will implement “draw” method.

GridGame Class:

It is an abstract class and classes that extends this class can start and stop the game, draw components in panel.

Some Important Methods:

start() -> starts the game

stop()-> stops the game

addDrawable(Drawable)-> adds a component into the list whose elements are going to be drawn.

removeDrawable(Drawable)-> removes a drawable form list

redraw()-> invokes draw method for all drawables.

ui Package:

ApplicationWindow Class:

This class determine the boundaries and frame of window.

GridPanel:

This class draws inside of the window

Some Important Methods:

drawSquare(x,y,color)-> Fills the square with given color. Used for Snake.

drawSmallSquare(x,y,color)-> Fills little inside of the square with given color. Used for Food.

project Package:

Creature Class:

This class is superclass of components of game which are Node, and Food. Contains coordinates and isHead field which determines whether this creature is an instance of Node and head of any snake.

Food Class:

This class represents food and extends creature.

Node Class:

This class represents segments of snake. Contains an extra method for nodes -whose isHead attribution true, that returns a list containing whether any neighbor of head is food or not.

Snake Class:

This class represents snakes. Every Snake contains a LinkedList made of Nodes. 0th element represents head of the snake.

Some Important Methods:

Snake()-> Creates the first snake.

Snake(List)-> takes parent's list as parameter and while removing parents nodes from end of the list, adds the removed node into the new snake's list.

chooseAction(Localinformation, Food)-> if size of the snake is 8, it returns REPRODUCE, if is there any food neighbor, it returns ATTACK, else, if there are directions that are moveable to and these directions overlaps with the direction to the food it MOVES that way, if it does not overlap it moves randomly if its head is not stuck.

move(Direction)-> Creates a new Node and sets its coordinates according to given direction and adds it to the LinkedList at 0th index. Then makes former head's isHead attribution to false and makes new 0th element's as true.

findDirection(Food)-> Creates a direction towards food. If the road is diagonal by 50 percent chance it chooses one of the directions. If directions are not free then returns a random direction.

attack(Creature)-> Creates a new node which has the same coordinates with attacked creature's. Then adds it as the first element of LinkedList.

reproduce()-> returns a new snake and constructor takes parent snake's LinkedList as parameter. (see: Snake(List))

naturesimulator Class:

Action Class:

This class represents the actions of snake. Each action has type and move and attack has direction also.

LocalInformation Class:

This class provides the information about a Node's neighbors.

getFreeDirections()-> returns a list containing the empty neighbor directions of a Node.

NatureSimulator Class:

This class looks like the brain of the game. It stores list of creatures, representative map of the game as the 2d-array, list of snakes and the current food.

Some Important Methods:

timertick()->This method is one turn of the game. First it checks whether game contains a food, if not puts a new food. Then takes every snake in the snakes list respectively. When a snake is chosen, first, chooses an action for snake, second, removes this snake's nodes from map, then executes the action. While executing action, if action's type is reproduce, invokes reproduce method from snake and adds it into snakes list. If type is move, invokes move method of snake and -with addCreature method- adds new added head of snake's list to the game. Then removes tail of snake from game and then from list of the snake. If type is attack, invokes attack method from snake and removes food from game and adds new node into the game. Then loads snake's nodes into the map again.

addCreature(Creature)-> Adds given creature into the creatures list, 2-d array map and drawables list.

createLocalInformationForCreature(Creature)-> creates a localinformation instance of a creature (in our case it is always head of a snake). Therefore, neighbors of head can be reachable in order to execute proper action.

main Package:

Main Class:

This class is the main. In this class, a new game is created as an instance of NatureSimulator, first snake is created and added then window of application is created as an instance of ApplicationWindow. Then the game starts.

Conclusion:

Program runs just like it is supposed to do. After a while snakes get stuck and no of them can move, so game ends.