**Component Prototype**

**update()**- draws this *Component* on the canvas

**touchingObstacle()**- looks through *obstacles* array, returns true if this *Component* ought to be damaged by an obstacle, returns false if this *Component* should not be damaged by an obstacle

**changeArea()-** changes this *Component*’s area by the amount passed in the param

**Monster Prototype**

**findClosestPlayer()-** updates this *Monster*’s *distances* array and returns the index of the closest *Player* smaller than this *Monster*

**changePosition()-** takes in the identity of the *Player* closest to this *Monster*, calculates this *Monster’s* speed and moves it towards that nearest *Player*

**changeArea()-** increases this *Monster’s* area by the amount passed into the param, without allowing this *Monster*’s area to exceed a size cap

**updateGameArea()**

**computeDistance()-** very often-used helper method that computes the distance between two *Component*s

**movePlayer()-** calculates this *Player’s* new position, based off of their current position and the cursor position

**drawNewFrame()-** draws the new frame of the game to the screen

**moveMonsters()-** moves every *Monster* towards the *Player* closest to them, replaces every dead (tiny) *Monster* with a freshly spawned one

**dealDamage()-** handles the complex task of damage dealing by calling three sub-methods

* **obstaclesDamageMonstersAndPlayers()**
* **monstersAndPlayersDamageEachOther()**
* **playersDamageEachOther()**

**checkFood()-** checks to see if every player or monster can eat any of the food pieces, relies on a helper method

* **tryToEat()-** if the relevant *Component* is close enough to the relevant *Food* to be able to eat it, then grow the *Component*’s size and delete that *Food*

**updateIntervalCounter()-** regularly creates a new *Food* object (unless there are already the maximum allowed number of *Food* objects) and places it in a random place that is not on top of an *Obstacle*, relies on a helper method

* **getNewRandomCoordinates()**- returns a pair of random coordinates for a *Food* object so that it won’t spawn touching any *Obstacle*