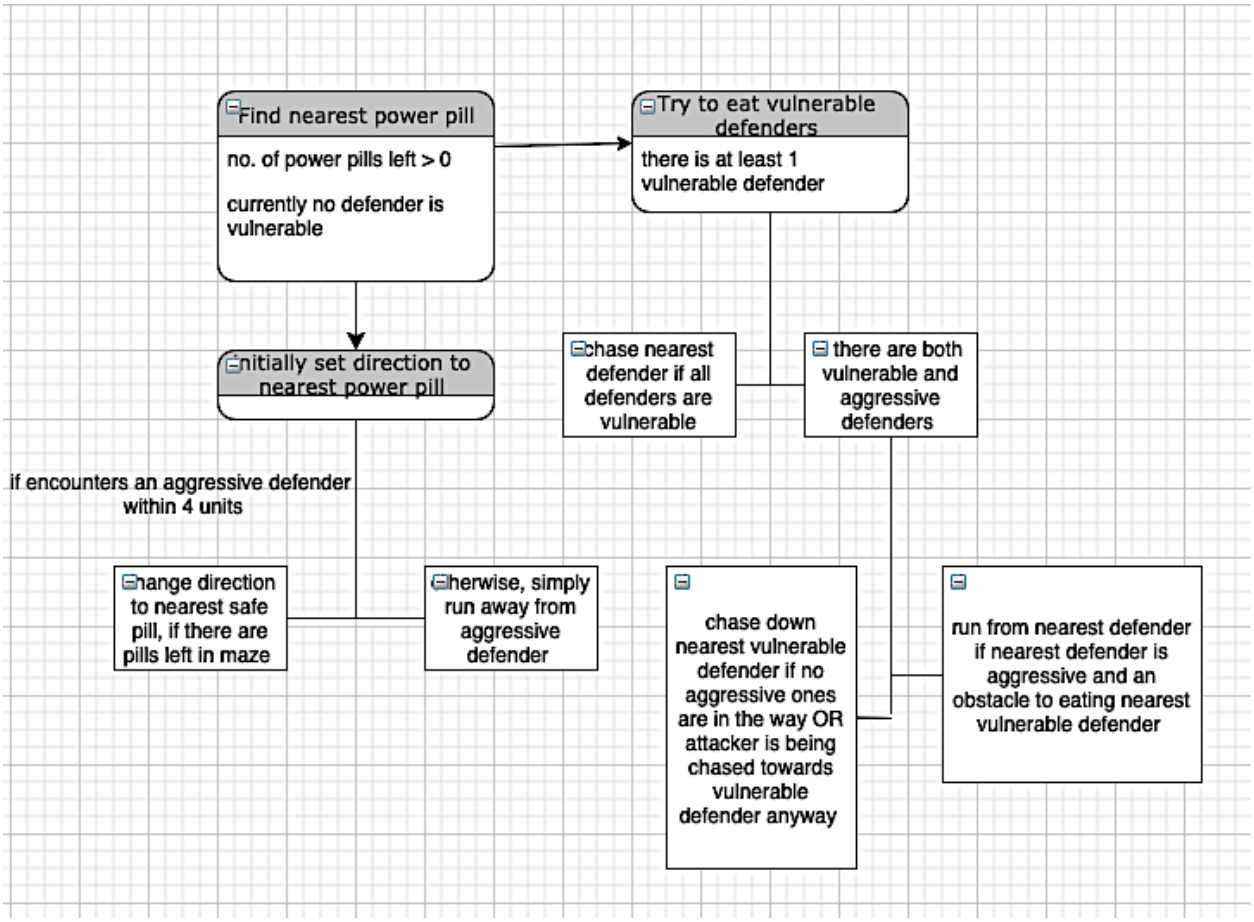
Caijun Qin
12/02/2019

Design and Post-Mortem

A. The top priority overall for the attacker is to gather all the power pills, preferably in the order of nearest distance. Let defenders that are not vulnerable be called aggressive. Any time after a power pill is consumed and there is at least one (1) vulnerable defender, the attacker will seek out and destroy the nearest vulnerable defender. There are specific precautions that are taken first. If there is an aggressive defender in between the attacker and the nearest vulnerable defender, the attack will ditch and run in the opposite direction. If there is no hostile obstacle, then the attacker has the new top priority of chasing down the nearest vulnerable defender. This continues until all defenders are aggressive, and the top priority switches back to eating power pills. Under the condition that the attacker encounters the nearest aggressive defender within a set gap of 4 units, the attacker goes for the nearest safe pill. There is a safe pill list that is a sub-list of the pill list only containing pills that poses a safe direction for the attacker to turn to. Lastly, if being actively chased by an aggressive defender within the short preset distance gap, the top priority shifts from finding power pills to simply running away to safety.



B. The first major and noticeable problem I encountered from the beginning was the seizures that gator was suffering from. After many hours, I discovered several factors. Firstly, if either 2 or more power pills, regular pills, or closest aggressive defenders were equidistant to the attacker, the code has trouble picking the nearest one. This causes indecisiveness to the direction to turn next. This was fixed by sorting the pill and power pill lists by the distance to the attacker, and then the

first element was picked (using get(0) method from List) as the closest element. Also, the attacker was wasting time not chasing any vulnerable defender after eating only one (1) vulnerable defender. The problem lied in an erroneous if statement that makes the attacker pursue only the nearest defender after eating a pill and nothing more. This led to the attack not eating other vulnerable defenders and sometimes die by mistaking aggressive ones as vulnerable. This was fixed by divvying up the if statement into an if-else-if chain. This time, attacker chooses whether to pursue the next closest vulnerable defender depending on whether the nearest defender overall is aggressive or not and its relative location. Those are the major issues encountered. From the start, there were several things that were already beneficial for the attacker. A copy of the pill list was shortened, via stream and filtering operations, to a smaller list for safe pills. The next direction from the attacker to each safe pill does not collide with going towards an aggressive defender. This protective measure allows the attacker to flee the nearest aggressive defender while simultaneously having a chance to gain points along the way. The best part about the implementation of the update method of the student controller is the prioritization order of actions based on current conditions. The attacker always attempts to eat a vulnerable defender, therefore using vulnerable time wisely. Otherwise, try to eat more power pills and repeat. As a last countermeasure, avoid the nearest defender and eat safe pills.

C. The project became a great practice for logical reasoning and algorithm design in a way. I was able to focus on finding an optimal order of actions and priorities to gain as much points as possible for the attacker. This was also a solid time for practice with debugging. The most important part of the project was increasing my expertise in highly useful standard Java functions, especially using stream to sort, filter, and manipulate lists of objects. For example, the safe pill list is formed by filtering out any regular pill such that the next direction from the attacker to said pill leads to an aggressive defender. Since I had quite some code for a single function, I also seized the chance to use Java 8 parallelization on streams to reduce the strain on the processors. This allowed each tick in the game to be updated seamlessly despite a lot of processing needed.

```
 1 /Library/Java/JavaVirtualMachines/jdk-12.0.1.jdk/Contents/
   Home/bin/java "-javaagent:/Users/fennec2000/Library/
   Application Support/JetBrains/Toolbox/apps/IDEA-U/ch-0/192.
   7142.36/IntelliJ IDEA.app/Contents/lib/idea_rt.jar=52271:/
   Users/fennec2000/Library/Application Support/JetBrains/
   Toolbox/apps/IDEA-U/ch-0/192.7142.36/IntelliJ IDEA.app/
   Contents/bin" -Dfile.encoding=UTF-8 -classpath "/Users/
   fennec2000/Desktop/Files/College/Coursework/Fall 2019/COP
   3502 - Programming Fundamentals I/Projects/Proj4/
   gatorraider-master/bin/production/PakuPaku:/Users/
   fennec2000/Desktop/Files/College/Coursework/Fall 2019/COP
   3502 - Programming Fundamentals I/Projects/Proj4/
   gatorraider-master/lib/pakupaku-benchmark.jar" game.Exec -
   teststudent
 2 Trial #0 complete. Score: 17540
 3 Trial #1 complete. Score: 14770
 4 Trial #2 complete. Score: 16530
 5 Trial #3 complete. Score: 5670
 6 Trial #4 complete. Score: 9350
 7 Trial #5 complete. Score: 9670
 8 Trial #6 complete. Score: 17740
 9 Trial #7 complete. Score: 7590
10 Trial #8 complete. Score: 7770
11 Trial #9 complete. Score: 6260
12 Trial #10 complete. Score: 13940
13 Trial #11 complete. Score: 18940
14 Trial #12 complete. Score: 16740
15 Trial #13 complete. Score: 5680
16 Trial #14 complete. Score: 7470
17 Trial #15 complete. Score: 5790
18 Trial #16 complete. Score: 9220
19 Trial #17 complete. Score: 9620
20 Trial #18 complete. Score: 6880
21 Trial #19 complete. Score: 8890
22 Trial #20 complete. Score: 13900
23 Trial #21 complete. Score: 6100
24 Trial #22 complete. Score: 17800
25 Trial #23 complete. Score: 7820
26 Trial #24 complete. Score: 9480
27 Trial #25 complete. Score: 7830
28 Trial #26 complete. Score: 9570
29 Trial #27 complete. Score: 13900
30 Trial #28 complete. Score: 6690
31 Trial #29 complete. Score: 7500
32 Trial #30 complete. Score: 14940
33 Trial #31 complete. Score: 9350
34 Trial #32 complete. Score: 6370
35 Trial #33 complete. Score: 9130
36 Trial #34 complete. Score: 7590
37 Trial #35 complete. Score: 13020
```

```
38 Trial #36 complete. Score: 9270
39 Trial #37 complete. Score: 6020
40 Trial #38 complete. Score: 17220
41 Trial #39 complete. Score: 18140
42 Trial #40 complete. Score: 13020
43 Trial #41 complete. Score: 9210
44 Trial #42 complete. Score: 17380
45 Trial #43 complete. Score: 6090
46 Trial #44 complete. Score: 7930
47 Trial #45 complete. Score: 14830
48 Trial #46 complete. Score: 16950
49 Trial #47 complete. Score: 6420
50 Trial #48 complete. Score: 9320
51 Trial #49 complete. Score: 9400
52 Trial #50 complete. Score: 8520
53 Trial #51 complete. Score: 9700
54 Trial #52 complete. Score: 7670
55 Trial #53 complete. Score: 8930
56 Trial #54 complete. Score: 7000
57 Trial #55 complete. Score: 5750
58 Trial #56 complete. Score: 6390
59 Trial #57 complete. Score: 15140
60 Trial #58 complete. Score: 9580
61 Trial #59 complete. Score: 9320
62 Trial #60 complete. Score: 9300
63 Trial #61 complete. Score: 8700
64 Trial #62 complete. Score: 7280
65 Trial #63 complete. Score: 7230
66 Trial #64 complete. Score: 11700
67 Trial #65 complete. Score: 7590
68 Trial #66 complete. Score: 8850
69 Trial #67 complete. Score: 6240
70 Trial #68 complete. Score: 15940
71 Trial #69 complete. Score: 12260
72 Trial #70 complete. Score: 13900
73 Trial #71 complete. Score: 13900
74 Trial #72 complete. Score: 17430
75 Trial #73 complete. Score: 5670
76 Trial #74 complete. Score: 7300
77 Trial #75 complete. Score: 15860
78 Trial #76 complete. Score: 5170
79 Trial #77 complete. Score: 6260
80 Trial #78 complete. Score: 5750
81 Trial #79 complete. Score: 8470
82 Trial #80 complete. Score: 15090
83 Trial #81 complete. Score: 9700
84 Trial #82 complete. Score: 15540
85 Trial #83 complete. Score: 6540
86 Trial #84 complete. Score: 9610
87 Trial #85 complete. Score: 14190
```

```
 88 Trial #86 complete. Score: 7390
 89 Trial #87 complete. Score: 7480
 90 Trial #88 complete. Score: 9260
 91 Trial #89 complete. Score: 7590
 92 Trial #90 complete. Score: 7090
 93 Trial #91 complete. Score: 14590
 94 Trial #92 complete. Score: 11520
 95 Trial #93 complete. Score: 5860
 96 Trial #94 complete. Score: 7590
 97 Trial #95 complete. Score: 15540
 98 Trial #96 complete. Score: 8100
 99 Trial #97 complete. Score: 6800
100 Trial #98 complete. Score: 8070
101 Trial #99 complete. Score: 8930
102 10175.3
103
104 Process finished with exit code 0
105
```