Good Neighbor APCS pd1

Alexia Leong, Lynne Wang, Lily Yan 5/28/2018

Final Project Proposal

Police Chase Simulator

"Police Chase: Will You Be There?"

This is going to make a police chase simulator where for a few nights the player will play as the police and try to solve various crimes throughout a city. There will be three levels of increasing difficulty. The crimes will all have different extremities, and the player will choose what order to deal with them by taking in various factors like how many police cars are needed, if there needs to be any other assistance, what equipment to bring, etc. The player will place each crime as it comes into a position in a priority queue.

To Do List / Timeline

- ❏ Get Woo.java to compile -- Day 1
- ❏ Implement all Crime classes -- Day 2-3
- ❏ Implement Busters -- Day 4
- ❏ Implement Equipment -- Day 4
- ❏ Implement CallCenter --Day 4-5
- ❏ Implement Jake --Day 5
- ❏ Implement Level -- Day 5
- ❏ Resolve issues -- Till the end

<u>Stretch Goals</u>

- ❏ Play in the perspective of a criminal
- ❏ Factoring distance of Busters and Crimes
- ❏ Graphics? (Processing?)

Term 2 Concept Incorporation:

- Queues: PriorityQueue used to store player's prioritization of crimes
- Heaps: Heap used to store the types of equipment the player can choose for officers to bring when solving a crime
- Stacks: Stack used to store interactions the player can have with his/her chief.
- LinkedList: used to store the available busters

Classes

- Level -- there will be different levels with different difficulties
- Call Center - receives info about crimes as they occur
- Crime (a **super** class crime)
    - Arson
        - Note: Later you have to call a fire truck and maybe ambulance
    - Robbery
    - Murder
        - Depending on extremity count, you might have to call FBI
        - Have to call a coroner, ambulance
    - Traffic
        - DUI

            - Might have to call an ambulance depending on extremity
        - Speeding
        - ParkingVio
        - CarCrash
            - Note: Have to call a towing company and ambulance
    - Drugs
        - Note: Depending on extremity count, you might have to call FBI
    - Vandalism
    - SexualAssault
        - Note: Have to call an ambulance
    - Kidnapping
        - Note: Depending on extremity count, you might have to call FBI and maybe an ambulance
    - Tax fraud
        - Note: Depending on extremity count, you might have to call the IRS
- Busters
    - FireDpt
    - Paramedics
    - FBI
    - IRS
    - TowingCompany
    - Coroner
- Equipment
    - Pencil
    - Handcuffs
    - Taser
    - Bike
    - Gloves
    - WalkieTalkie

    - FirstAidKit
    - Gun
    - Cruiser
- Jake (player)
    - A cadet with dreams of being the police chief so he overcompensates by trying to help the whole town by himself
    - Wears khakis
- Woo (driver class)

CallCenter variables + functions:

- PriorityQueue<Crime> crimes
    - This will put the crimes in order based off its severity (which depends on # of criminals, # of police officers needed, who/what kind of help needed, how far away the police are, how long it takes to solve the crime/how long until bad things happen). The crime on top will be presented to the player since it has the most priority but the player has a choice to pass the crime to solve the next one. If this happens, the crime that was passed will be moved to the bottom even though it has higher priority. Someone might choose to do this because there's not enough time in the say to solve a crime with higher extremity.
    - Each crime will have a time limit that determines how long the player has to solve the crime until the criminals run free.
    - When crimes come in then the player has a chance to put the crime anywhere in the queue and when they want to deal with it (through rank(). See below for details).
- ALHeapMin<Equipment> inventory
    - Stores the equipment the player can choose to assign to solve a crime. The higher the index, the more valuable the equipment is.
- void chooseBusters(Crime input)

    - Player chooses if they want assistance to solve their next crime, and if so, who they want to send
- void chooseEquipment(Crime input, int root)
    - Player chooses if they want to use the equipment at index root for their crime, or if they want to traverse deeper into the heap to use better equipment (the branch they travel down is chosen randomly).
    - The deeper they traverse, the less time they have to solve that crime.
- LinkedList<Busters>
    - Store the available busters that are ready to help Jake

Crime variables + functions:

- int extremity
- int priority
- boolean needParamedics
    - Potentially for Arson, Murder, CarCrash, Drugs, SexualAssault, Kidnapping
- boolean needCoroner
    - Same as above
- boolean needFireDpt
- boolean needFBI
- boolean needIRS
- boolean needTow
    - For CarCrash and ParkingVio
- boolean needBackUp
    - For more police cars if needed
- int timer
    - Determines how long player has to solve a crime
- void whenSolved(Crime input)

    - Once amount of time passed equals the amount of time it takes to solve the input crime, the Busters sent to help with the crime return back to the station (available = true), and the crime is removed from the queue crimes.

Busters variables + functions:

- boolean available
    - False if buster is already out dealing with a crime, true otherwise

** note: the variables below test if the buster in question is suited to solve that crime. Example: For FireDpt, solveArson = true, but solveDrugs = false.

- boolean solveArson
- boolean solveMurder
- boolean solveDUI
- boolean solveCarCrash
- boolean solveDrugs
- boolean solveSexualAssault
- boolean solveKidnapping
- boolean solveTaxFraud

Equipment variables + functions:

- int value
    - Represents how much the equipment will help on a mission (the greater the value of the equipment, the quicker it lets you solve the crime)

** note: the variables below test if the equipment in question will be useful for that crime.

- boolean helpArson
- boolean helpRobbery
- boolean helpMurder
- boolean helpTraffic
- boolean helpDrugs
- boolean helpVandalism

- boolean helpSA
- boolean helpKidnapping
- boolean helpTF

Jake variables + functions:

- int score
    - Keeps track of points Jake has accumulated so far. Negative points will be awarded for passed crimes. Points must meet/exceed a certain number for player to go on to next level / day.
- Stack<String> interactions
    - Throughout the level, interactions will be popped off the stack. Jake will be able to talk with his chief. Depending on his answers, his chief will either reward him (ex: time bonuses, more available officers) or punish him (ex: suspensions).
    - We chose to use a stack because the interactions build off of each other. It's more efficient to pop an interaction off the stack when needed instead of repeatedly calling remove(0) or something of that nature.
- void rank(Crime newCrime)
    - When a new crime comes in during the level, the player must give it a priority number. The crime is then added to the queue Crimes.
- void interact(String interaction)
    - Takes next interaction in the stack interactions for the player to play out.

Level variable + functions:

- There will be 3 different levels: 1 (easy), 2 (medium), 3 (hard)
    - A harder level means that the extremity value of crimes will go up, it takes longer to solve a crime, there's less time in the day to solve all the crimes, crimes are called in more frequently.
    - For each crime, the player will have to wait 5-15 seconds for a crime to be solved

    - An easy level will take 4 minutes, a medium level will take 3 minutes, and a hard level will take 2 minutes (subject to change).
- startDay()
    - This will kick off the start of the simulation

Woo variables + functions:

- long timePassed
    - For each level, there will be a timer displayed, in seconds, for a “day” of crime. Jake must resolve as many crimes as he can in the given time period
- void main(String[] args)
    - Creates three instantiations of Level, each with a higher difficulty than the last, and calls each of their startDay() methods to carry out the game