#1 In your code for the Gauntlet game of HW1 explain how you initialized the rival players and restricted their position to either side of the screen.

I made another constructor for the rivals that took a second parameter int that represented how many rivals were on the screen. I then took mod 2 of the int to determine if the rival should spawn on the left or right. If it's supposed to spawn on the left I ~~added~~ subtracted 2/3 the windows x size from the hi point along the x-axis when calling set move box. If the rival is supposed to spawn on the right I add 2/3 windows x size to the lo point along the x-axis.

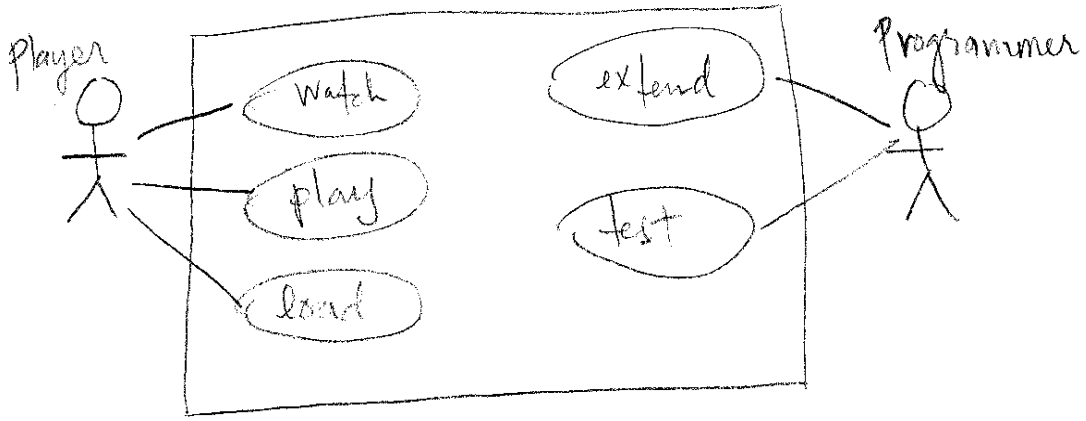
Resolution to practice midterm problem #2.

Player Movement:
For example, Pac Man is restricted to 1.25 dimensions of movement because he is usually restricted to one axis of movement except at intersections where he's allowed to change to the other axis of movement.

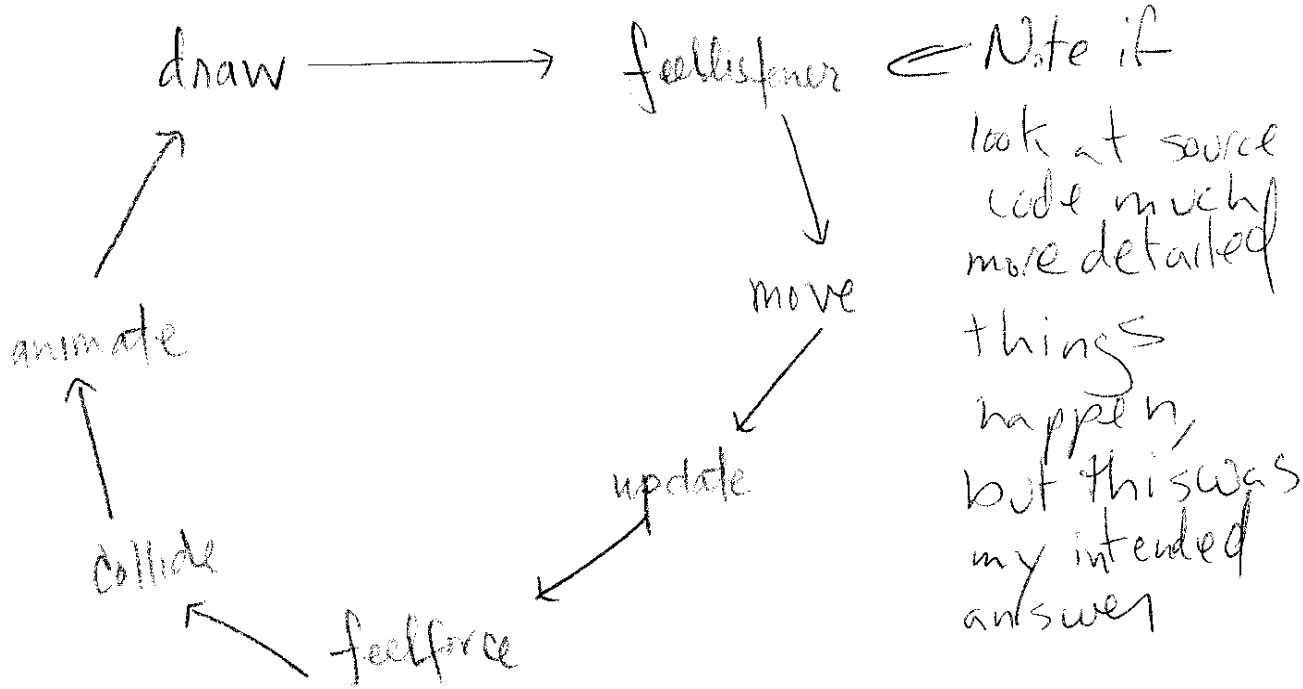
Dimensionality of the world:
The depth of the terrain of the game world and its inhabitants, such as 2D or 3D sprites and walls, dictates the amount of dimensionality of the world. For example, the original Mario Bros. game allows walking left/right & vertical jumping, but Mario Sunshine allows movement in a fully 3D world where you can walk in all four cardinal directions + jump up.

Game's viewport's dimensionality:
The angle, position, and "zooming" capability of your "camera" on the game world influences the dimensionality of your viewport. Doom allows a fully changeable 3D viewport while games like the original Metroid only have 2D scrolling, or even have no change of viewport in games like Pong.

3) A use case diagram is a visual tool to help better understand the needs of the end user. Actors are represented by stick figures. The program is represented as a big rectangle. Ovals are use cases.



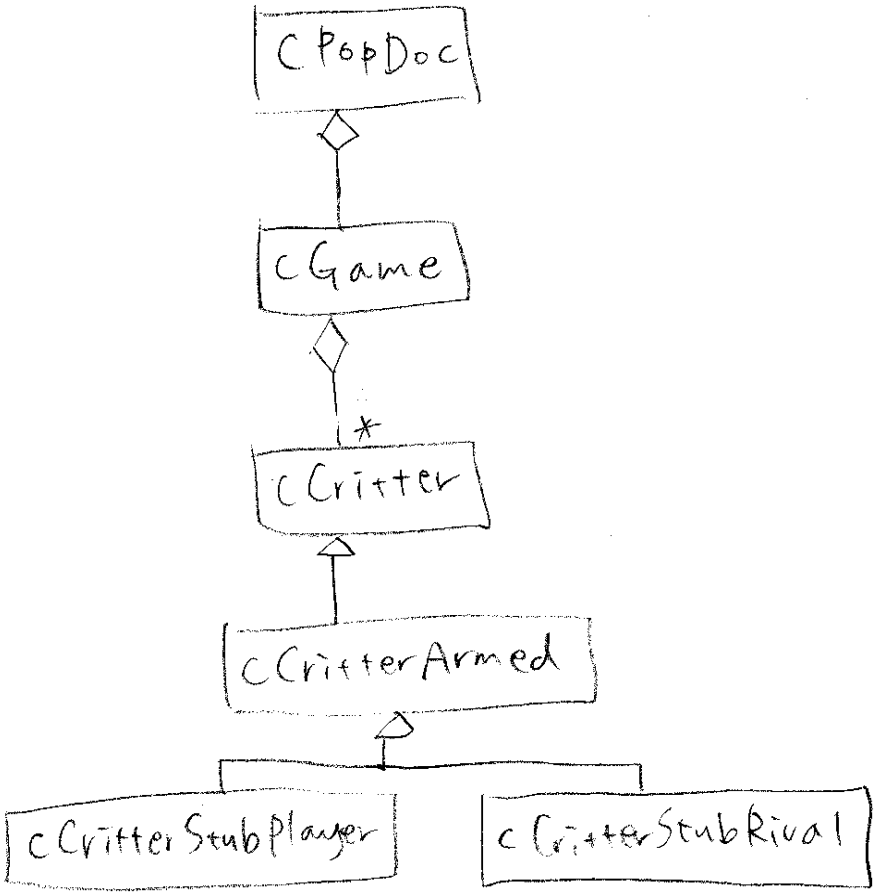
10)



Note if look at source code much more detailed things happen, but this was my intended answer

Asami Enomoto
Hai Lin Wu

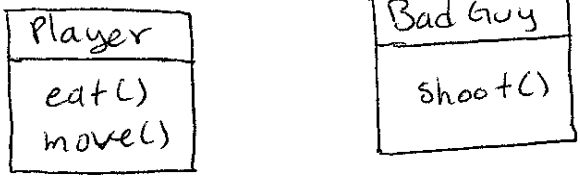
④



#5 Noun-Verb Analysis

Map nouns in the requirement description to class names.
Map verbs in the requirement description to member functions.

ex.
the Player must move back & forth to avoid bullets shot by bad guys. The Player may eat food to regain health

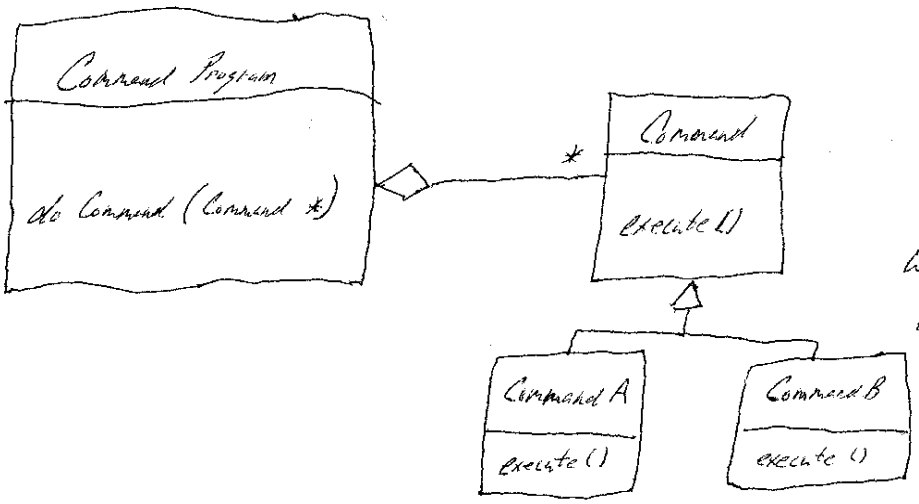


(6) Draw the UML diagrams of:

Command, Singleton, and Bridge

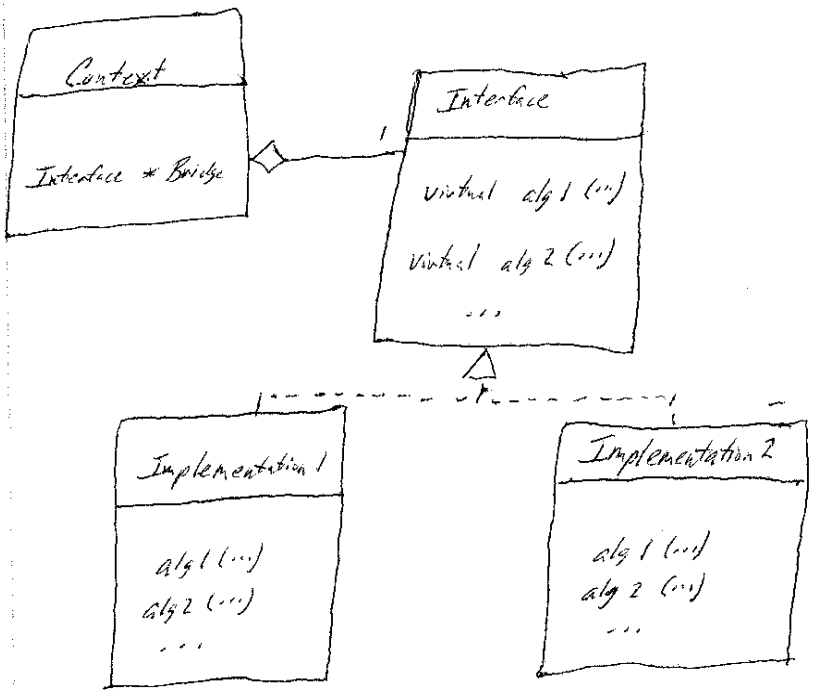
and explain how they are used in Pop.

Command



Pop uses the cServiceRequest struct, which passes off command execution to cBrota.

Bridge



Pop has a cGraphics Interface, to bridge to either MFC or Open GL graphics.

(6)

Singleton

```
Singleton
private static Singleton *_pinstanceSingleton
private singleton();
public static Singleton * pinstance()
{
  if(_pinstanceSingleton == NULL
    _pinstanceSingleton = new Singleton();

  return _pinstanceSingleton.
}
```

Pop uses the Singleto pattern in the cRandomizer class.

8) update()
Calculates all forces on critters in an array.
move(dt)
loops through the array and changes the critters position based on the total forces on it.

It appears to move in parallel because all these calculations and movements are done before it is drawn to the screen and when it is drawn all the movements have already occured.

Since update calculates forces w/o moving the creatures or changing velocities and presumably the new forces only depend on current values of positions, velocities, order of calculating critters in update does not matter.

Since move calculates new positions/velocities based only on the already updated forces, order in this case doesn't matter.

The fact that both these operations are order independent gives rise to the feeling of parallelism in the game, because we can't see observing the game what order things were updated internally.

Niki Sardjono
Ted Gomez

Problem 9 - Draw an x & y playing field as in create world ~~..~~ like  .

- randomly chosen x, y [illegible], so you create rectangles
- Rectangles are created & randomly chosen & removed if it's too small.
- Randomly created doors in the rectangles.
- Start with a room connect the room to unconnected rooms using edges (from rectangle to connect to the edges of the other rooms (graph like).
  This is how you connect the rooms
- After that connect the doors.