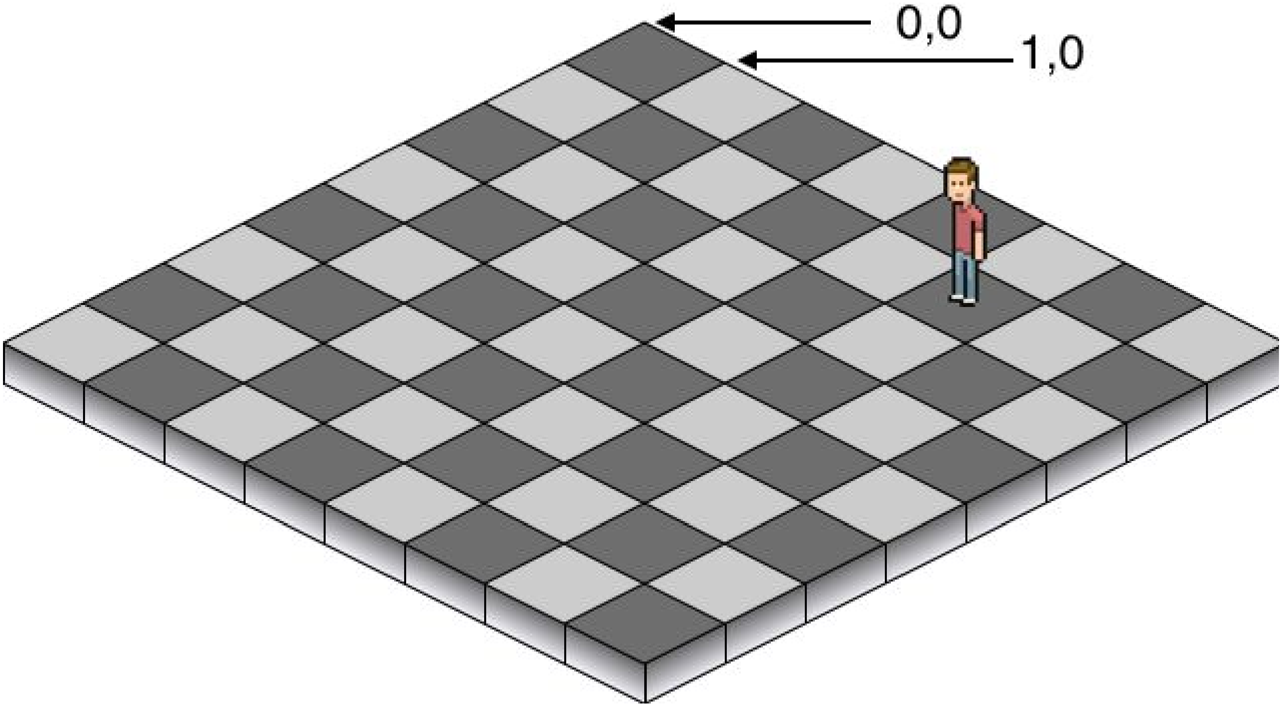
Game Development - Revaluation Examination - February 2nd 2018

YOUR FULL NAME: **Solution provided by the teacher**

- You have 2 hours to complete the assignment.
- Only valid text will be the one inside each box, everything else will be ignored by the teacher

1. **(3 points)** Given an map with isometric view, were the tiles are 128 x 64, write down the formula to find the screen coordinates of the tile the character is in. Also, calculate the tile the character would be in if it were drawn at 200,300. Both screen coordinates and tile coordinates start at 0,0 top vertex and increase to the right-bottom. Hint: world to screen on x coordinate formula *(x-y) * (tile_width / 2)*



- The character is in the tile **5,1**.

- To calculate the screen coordinates of tile 5,1:
    - X Screen Formula: (x-y) * (tile_width / 2)
    - X Screen Math: (5-1) * (128/2) -> 4*64 -> 256
    - Y Screen Formula: (x+y) * (tile_height / 2)
    - Y Screen Math: (5+1) * (64/2) -> 6 * 32 -> 192
    - Answer: **256,192**

- To calculate the square where we can find 200, 300:
    - X Tile Formula: (x / (tile_width/2) + y / (tile_height/2)) / 2
    - X Tile Math: ((200 / 64) + (300 / 32) / 2) -> 6
    - Y Tile Formula: (y / (tile_height/2) - x / (tile_width/2)) / 2
    - Y Tile Math: ((300 / 32) - (200/64)) / 2 -> 3
    - Answer: **6,3**

2. **(2 points)** Explain the main differences between the Dijkstra Algorithm and the A* algorithm.

The base for both is actually the same: A* is an improvement on top of Dijkstra were we take in account also an heuristic (approximation) in the form of the euclidean distance to the goal ignoring any obstacles (for optimization reasons we could also use Manhattan distance)

This transforms the Dijkstra algorithm in a Greedy Algorithm, optimizing the way it expands and minimizing its steps.

3. **(2 points)** Define and explain the pros and cons of **Fixed Time Step** versus **Variable Time Step**. Why would it be interesting to limit the framerate in a video game ?

Fixed Time Step assumes that every frame takes the same amount of time. This is only predictable if we know exactly our target hardware (console):

Pros: simpler calculations, more predictable outcome

Cons: cannot deal with unexpected variation of frame time that would produce a bad user experience

Variable Time Step calculates every frame the time it takes and uses it as a factor for most movement and animation calculations.

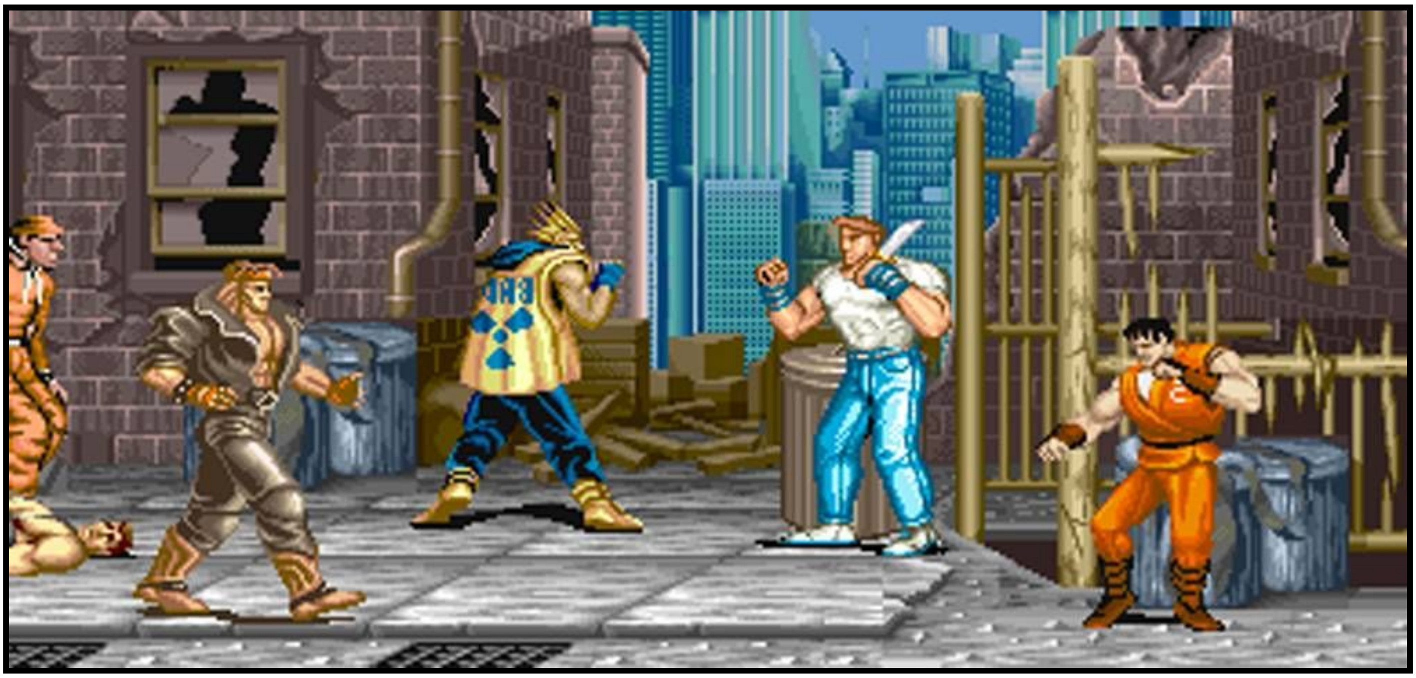
Pros: smooth experience even with frame rate changes

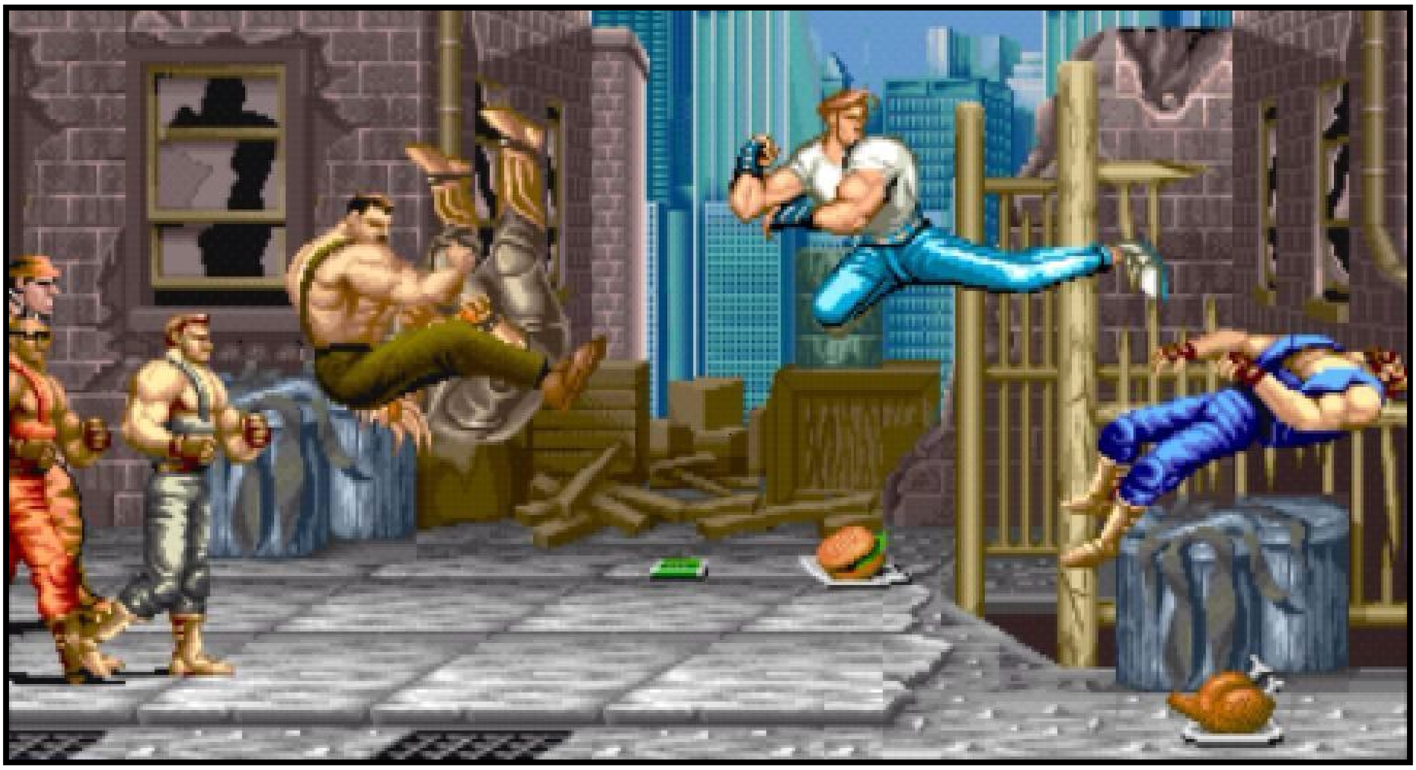
Cons: non-deterministic, weird behaviour in very small/large steps

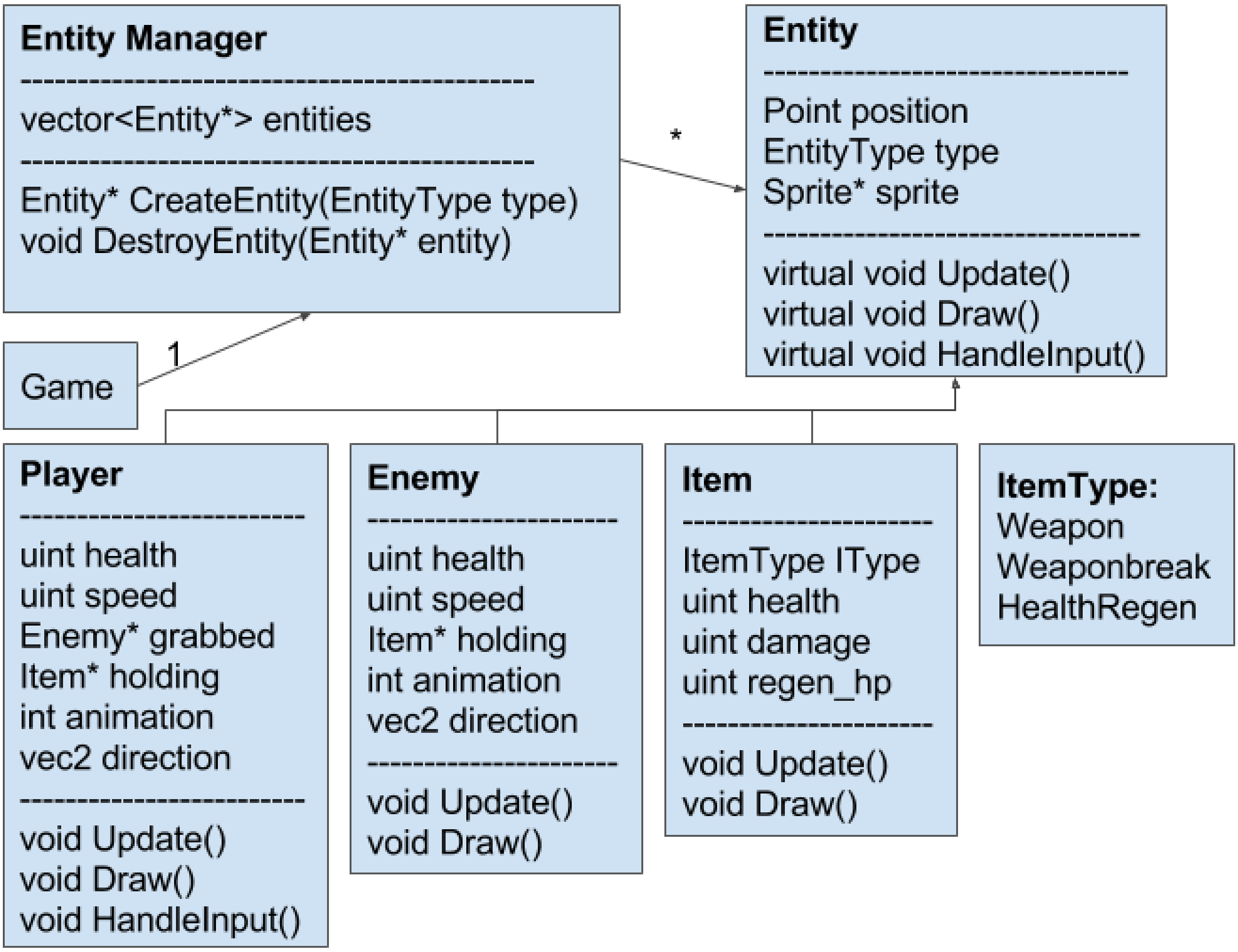
Limiting frame rate allows us to hide peaks of CPU usage, code becomes more predictable and the general experience is smoother for the user.

4. **(3 points)** Given can be seen in those two screenshot from *Final Fight*, draw the UML for an entity system that would have all those elements we can see in the picture. Take in account that:

- The three characters are active players (Cody, Guy and Haggar).
- Players and AI can grab and throw knives.
- Trash cans can be grabbed and thrown, as well as broken by attacking them.
- All NPCs have the same behaviour, attacks and animations, only changes the art.
- On the bottom screenshots, the items on the ground regain health.







```
Entity Manager
----------------------------------------------
vector<Entity*> entities
----------------------------------------------
Entity* CreateEntity(EntityType type)
void DestroyEntity(Entity* entity)
```

```
Entity
---------------------------------
Point position
EntityType type
Sprite* sprite
----------------------------------
virtual void Update()
virtual void Draw()
virtual void HandleInput()
```

```
Game
```

```
Player
--------------------------
uint health
uint speed
Enemy* grabbed
Item* holding
int animation
vec2 direction
--------------------------
void Update()
void Draw()
void HandleInput()
```

```
Enemy
-----------------------
uint health
uint speed
Item* holding
int animation
vec2 direction
-----------------------
void Update()
void Draw()
```

```
Item
-----------------------
ItemType IType
uint health
uint damage
uint regen_hp
-----------------------
void Update()
void Draw()
```

```
ItemType:
Weapon
Weaponbreak
HealthRegen
```

Use this page for your own notes. You cannot use any other page. Do not remove this page from the set.