\*\*\* = Things that factor into the final grade.

>> = Possible subtasks necessary for functionality.

E = Everyone.

Y/N = yes or no

|  |  |  |
| --- | --- | --- |
| Done? | Person | Task |
| N | E | \*\*\*Write up proposal  Mostly, the “purpose” part. The UML diagram, with maybe a couple more sentences, can serve as implementation and this document is the “roles” bit. |
| N | E | \*\*\*Documentation of all classes |
| N | E | \*\*\*Create test methods for all classes |
|  |  |  |
| Y | Mengjing | Create a Map for the Game  >> Creation of a Point class that will serve as the point for our entities and items in the Map |
| Y | Mengjing | Implement Graphics  >> Creation of a Game class that will use either the Canvas class or the Jpanel class |
| Y | Ahmad | Creation of Player and Enemy classes  Possibility: An Entity class that is abstract and will be used as template for both classes since they will be very similar. |
| Y | Jacob | Implement Items  Possibility: since there are different types of items, there could be multiple subclasses such as EquippableItem, ConsumableItem, etc. |
| Y | Winston | Implement an ChestMap and EnemyMap  Possibility: an ArrayList that will contain that object |
| Y | Mengjing | Implement Player movement  >> algorithm that will check for walls, enemies, etc. so that player will not be drawn over those things |
| Y | Ahmad | Implement Enemy movement  >>same type of wall and enemy checker  Possibility: The enemy will move randomly while it cannot see the player but once the player is within its field of view it will have a chance of following or randomly trying to attack. |
| Y | Jacob | Implement Status class  Possibility: use regex to check for type of effect status will have. |
| Y (Up for debate) | Ahmad | Implement Random Monster Spawning  Class EnemyMap  public void spawn(int level, Map m)  >> Creation of a monster spawning algorithm in EnemyMap given a level number and Map object. The algorithm should be spawning more enemies/more difficult enemies at higher levels.  As of this task, the Game class should no longer be handling monster spawning or adding anything to EnemyMap. |
| N | Winston | Implement Random Item and Chest Spawning  Class ChestMap  public void spawn(int level, Map m)  >> Creation of an Chest spawning algorithm in ChestMap given a level number and Map object.  >>use of Chest class |
| N | Jacob | Implement ConsumableItems  Possibility: This may require some finagling in the Game class  >> use of Status |
|  | Mengjing | Implement a graphical potion display and relevant keystrokes  >>this *definitely* will require the Game class |
|  | Winston | Implement Loot/dropped items  Possibility: when clearDead() is called it accepts the itemMap as a parameter and will add drops to the itemMap.  >>use of Drop method that all entities have |
|  | Jacob | Implement EquippableItems  >> use of equip and unequip methods that all entities have |
| Y | Ahmad | Implement Slime class(other specific enemies will be done by whoever)  >>change of Enemy instantiation in Game  As of this task, the Enemy class should be abstract |
|  | Winston | Implement Chest, Normal Chest, and Special Chest Classes |
|  |  | ... |