Order of the project
1. Character Class(including the player and the enemies)
2. Potion Class
3. Treasure Class
4. Subclasses of Object class is done
(Visitor Pattern including objects visit each others)
5. Floor Class(Client)
6. Factory Class(Factory Pattern)
7. main function

Estimated completion dates: 29 Nov. 2013

Partner responsibility:
w34ma: points 1,2,3,4(Content of the Game)
y497li: points 5,6,7(Random Creation Process)

Questions Answers:

Question 1 How could your design your system so that each race could be easily generated? Additionally, how difficult does such a solution make adding additional classes?

We will create new classes for each race as a subclass of the Player class. If a new class is to added, we need to create a new subclass of Player class, and overload certain functions that represents the special ability of the new race.

Question 2 How does your system handle generating different enemies? Is it different from how you generate the player character? Why or why not?

We will use factory pattern to generate enemies since there is a probability distribution of the enemies. Because the player is unique and the race of a player is chosen by the client, and the enemies are not uniqe and the race of the enemy is randamly chosen, they should be generated differently.

Question 3 How could you implement special abilities for different enemies. For example, gold stealing for goblins, health regeneration for trolls, health stealing for vampires, etc.

The special abilities can be implemented by adding some features to the specfic function fo the specific race(subclass of Enemy).

Question 4 What design pattern could you use to model the effects of temporary potions (Wound/Boost Atk/Def) so that you do not need to explicitly track which potions the player character has consumed on any particular floor?

The decorator pattern can be used to solve this problem. However, giving field called Temporary*** to the Player class can solve this problem pretty well as well.

Question 5 How could you generate items so that the generation of Treasure and Potions reuses as much code as possible? That is, how would you structure your system so that the generation of a potion and then generation of treasure does not duplicate code?

Use factory pattern and let Create function take a parameter that determines the thing that it returns.