Writer: s56gao h63lu

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

Answer: We plan to use inheritance relationship. We design a base call Player, which is a subclass, and all classes – Human, Dwarf, Elves, and Orc – are subclass of Player. All of these subclasses inherit the increaseHP(), decreaseHP(), etc. methods from Player, which dramatically reduce the amount of codes. Also, it is not hard to add new classes from Player. You just need to write the methods which are unique to the new class and inherit all other methods from Player.

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

Answer: We use the same method to generate enemy character as we do with player character. Both of them are inherited from Character Class. The reason is that both player and enemy have some same property, for example, both of them have HP, Atk, Def. Thus, designing them as the subclasses of Character Class reduce the amount of repeated code we need to write.

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

Answer: We will write a pure virtual method called specialAbility() in the super class Enemy. And then we override each method based on the property of each enemy separately in each subclass. This can make the whole classes more intact and the relationship between them more obvious.

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?**

Answer: We decide to use decorator pattern to implement this part of code. Every time the player has a potion, like BA, BD, WA, and WD, we create a new player with the same curHP, curAtk and curDef as the original one and put the effect of these potions on the newly created player. Also, every time the player got attacked, both the HP of original and newly created players will be reduced. And when the player goes to the next floor, these newly created player will be destroyed and leave the original player intact.

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 duplicated code?**

Answer: I will put both Treasure and Potion as subclass of a super class. And put the same filed in the super class. Then when I construct Treasure and Potion, I can use member initialization list to first initialize the super class and then initialize the unique part of each subclass.

Plan of Attack

First, on July 10 (Sunday), we will discuss how to design this project and have a big idea about different parts of this project. We decide that first two days of this week we will focus on the design this project, find the relationship between different parts and have a draft of the UML diagram.

Second, starting from July 12 (Tuesday), we will start work on the .h files and make adjustment to the draft of the UML. And at the end of July 13 (Wednesday), we will have a final UML (still needed to be adjusted) of this game.

Third, starting from July 14 (Thursday), we will start work on the implementation of this project and make adjustment of .h files and UML. The following is the specific spilt of the process of implementation:

<table>
<tr><th colspan="2">Split of Work</th><th>Time</th></tr>
<tr><td>Hongyi Lu</td><td>Shang Gao</td><td>Expected finish Date</td></tr>
<tr><td>Item
(Potions, Treasures)</td><td>Characters
(Player, Enemy)</td><td>Thursday July.14 – Saturday July 16</td></tr>
<tr><td colspan="2">Meeting
have a discussion to improve the code
Debug</td><td>Sunday July. 17</td></tr>
<tr><td>Floor
(Construction of the floor)</td><td>Floor
(Random Generation of the items of floor)</td><td>Monday July 18 –
Tuesday July 19</td></tr>
<tr><td colspan="2">Meeting
Have a discussion to improve the code
Debug</td><td>Wednesday July 20</td></tr>
<tr><td colspan="2">Work on the attack part together
Work on the generation of different level together</td><td>Thursday July 21</td></tr>
<tr><td colspan="2">Finish the display and finish most of this project
(Hope to see the actual result after run the code)</td><td>Friday July 22</td></tr>
<tr><td>Bonus:
Work on the inventory system to hold potions</td><td>Bonus:
Add ability to merchant to sell</td><td>Saturday July 23</td></tr>
<tr><td colspan="2">Meeting
Have a discussion to improve the code
Debug
have a final version of UML and improve all of the answers to questions</td><td>Sunday July 24- Monday July 25</td></tr>
</table>