5:

This diagram shows interaction between player and dependent objects. GameManager updates a level and gets GraphicGrid. After that, GameManager updates EntityManager acording to the pressed key and CollisionGrid. Entity manager redirects this data to the player object and gets PlayerData. This returned data is used with CollisionGrid to return response of dependent objects. DependentObject returns data with respect to the fulfillment of requirements. Finally, RenderData goes to Platform which will render these data on the screen

6:

Here, the combination of all sequence diagrams is displayed, all options except StartGame player have chosen in the main menu do not require any other class except Platform, Platform uses RenderMenu method to display these options. Other objects are called when actual game is initialized. When the game is started, the system will be in loop which is the Run() method. This loop is used in order to keep the game updated/rendered. Firstly, GameManager updates LevelManager, EntityManager and CollisionManager in order, and gets them data which is required for updating another manager. In this sequence, EntityManager is shown more detailed, EntityManager updates player, enemies, etc. and returns combined data to the GameManager. At the last step, GameManager returns the data to the Platform to display new situation on the screen

Activity:

Activity diagram shows behavior of the system at the gameplay. When the game is opened, system waits for mouse input. If user presses Play Game, system will initialize game, then user could make two choices( either customing his/her own level or starting default game). After that, system updates/renders the gameplay. While user is playing the game, system will check for collision. If collsion occurs, system will check if this is between projectile and entity, between door and player or entity and player. If collision is between projectile and entity, system will decide if entity is an enemy or a player. When entity is an enemy, system takes one health of the enemy and decides destroying an enemy if enemy has 0,and updates the new data. When entity is a player, system takes one health of the player and decides displaying gameover message and returns to the main menu if player has no life, else updates the data. On the other hand, if collision occurs between player and entity, entity decides updating new data if entity is not an enemy. If entity is an enemy, system will process this collision as a collision between projectile and a player. Lastly, if player collides with a door and has a key, system will move player to the next level. The system will make these decisions iteratively until the game is finished or player does not have any life