**[Project #2: ChipARoonie!](https://blackboard.andrew.cmu.edu/webapps/assignment/uploadAssignment?content_id=_1279496_1&course_id=_8912816_1&assign_group_id=&mode=view)**

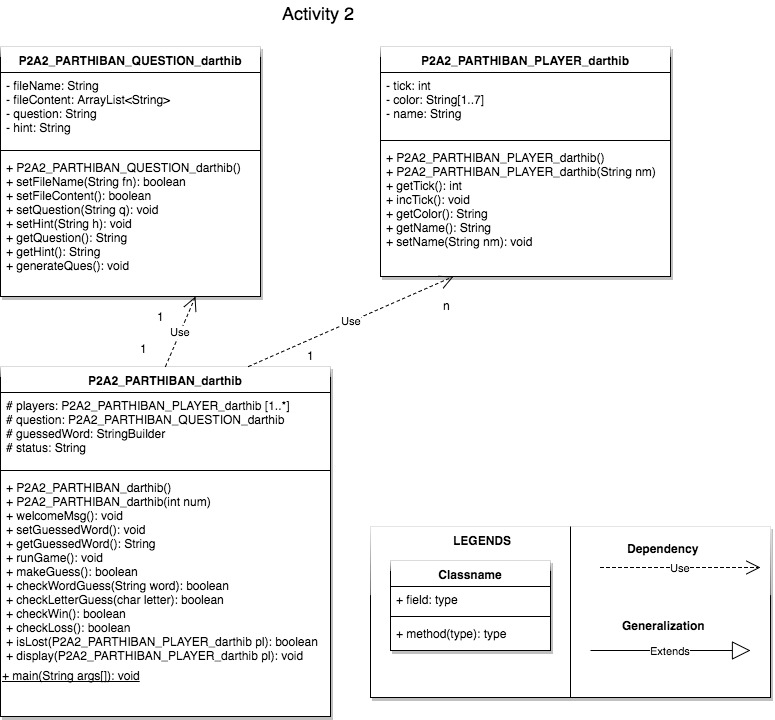
**Assumptions**

These are same assumptions as listed in the comments while submitting the game.

* Each player has his own individual bomb.
* The game is not case sensitive.
* If the user enters an already entered word then it is considered as incorrect guess.
* In autoplay mode the computer can only guess a letter and not a word.
* All the game in autoplay mode can have the same number of users in a single batch.

**Activity 2**

UML Diagram



Rationale:

The design choices for what goes into these three classes as listed below,

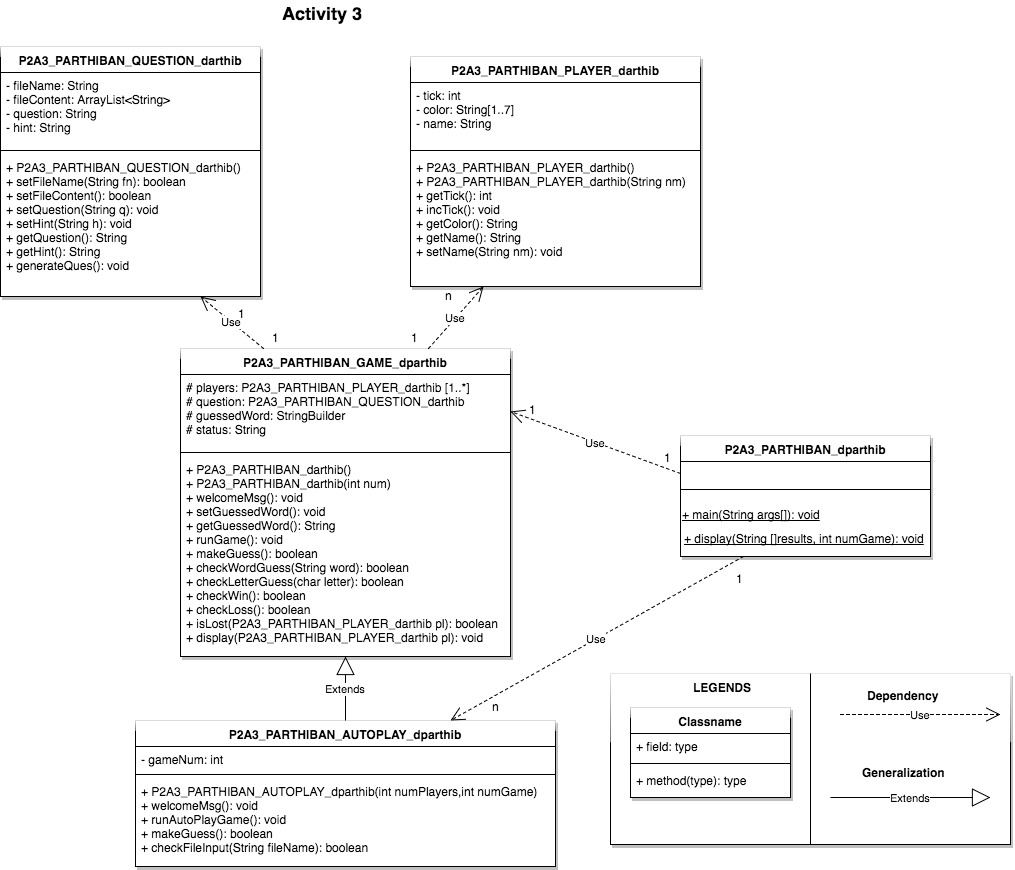
* The question class stores only question related information consists of methods to handle any operations on questions. So, the major responsibility for this class is listed below,
  + Initialize the file with given file name by the user.
  + Randomly generate a question.
  + Initialize and access the question and hint corresponding to the question.
* The player class stores only player related information consists of methods to handle any operations on players. So, the major responsibility for this class is listed below,
  + Initialize the player name with the given name by user.
  + Increment incorrect guesses (tick counter).
  + Maintain the bomb color for each player.
  + But there was a method called isLost() (which is used to identify if the player has lost) could have placed in player but moved to Game. The reason Win/Loss depends on the game you play more than the player.
* The main class here has also the functionality a game needs. This was designed in a way such that the entire class can be reused as “game” class for activity 3. This class just used the question and player does not qualify for any other relationship (like Generalization – “is a” or Aggregation - “parent child”). The fields of this class were made protected so when the “autoplay” class inherits the game class then it will be would able to access the fields too. So, the major responsibility for this class is listed below,
  + Initialize the numbers of players and question for the game.
  + Maintain the game logic by allowing the end user to make a guess.

Extensibility:

All the three classes I was able to reuse for the activity 3. The game class alone I had to remove the main function from it. Since the game logic is not embedded in question and player class it can be used in similar scenarios for other use cases too.

**Activity 3**

UML Diagram

****

Rationale:

The design choices for what goes into these 5 classes as listed below,

* Question class – completely reused apart from changing the class name.
* Player class – completely reused apart from changing the class name.
* Game class – the main class of activity 2 was converted into game class by removing the main function and changing the name.
* Autoplay class- this inherits the game class. So, the major responsibility for this class is listed below,
  + Initialize the game with number of players and game in autoplay mode.
  + overrides 2 functions from the game class such that it is automatically able to simulate the user’s input.
  + runAutoPlayGame() is used to run the game in autoplay mode. Why I did not override here because this method had to return a string of results which was used to display in the main function.
  + checkFileInput() is used to handle the input file validation logic i.e. it could not prompt again to the user to re-enter correct file name instead it had to stop the simulation mode.
* Main class- It is a mere selector of Game/Autoplay class. It does certain validations to check for valid input from the user. It also has function to display the results in the autoplay mode.