REFACTORING DOCUMENTS

1. REFACTORING TARGETS

Refactoring is a process of removing bad code smells from a software application without changing its behavior. As part of the build 2, the targets for refactoring were:

- Map Related Functionality
- Phases (Re-enforce, Fortification) Related functionality

A. Reasons for Refactoring

1) **Phases (Re-enforce, Fortification) Related functionality**
   - To Implementation of the reinforcement, attack and fortification as methods of the Player class. In Build one, the phases were implemented separately under the Phase package.

2) **Map Related Functionality**
   - **Redundant Methods and Classes**
     i) The create map, edit map were implemented as separate classes and were redundant in nature since methods implemented for create map scenarios were already implemented in MapController
   - **Hard Coding**
     i) The map path for loading the map file was hardcoded and every time the user wanted to load a new file, either the existing file was changed, or the default file path had to be changes in the code.

B. Refactoring Techniques

1) **Extract Method**
   - This technique involves grouping of code fragments to ensure better readability of code, code isolation and to remove redundancy. This technique is applied to GameContoller wherein the functionality written in main method is divided into different methods and those methods are called in the main method.

2) **Parameterized Method:**
   - This technique involves combining methods that perform similar functionality but with different internal values by parametrizing the method.
   - In the application, create map, edit map were implemented as separate classes and were redundant in nature since methods implemented for create map scenarios were already implemented in MapController. Also, the load map related functionality was implemented separately in class Game Controller.
   - As part of refactoring, these functionality were moved to a single Class CustomMapGenerator and an init() method was defined to differentiate between the different scenarios.