Assignment 3

Choose one design pattern among those that we saw in class. For this design pattern, you must have a corresponding implementation in your code. If not, refactor your code to include it. Then, complete the following points:

1. Write a natural language description of why and how the pattern is implemented in your code.
2. Make a class diagram of how the pattern is structured statically in your code.
3. Make a sequence diagram of how the pattern works dynamically in your code.

1 Natural description

We have decided to implement the **Composite Pattern**. The reason is the following: consider a possible evolution of the game in which you can use also "diagonal" movement. Since, for instance, doing "up" and then "right" is different than doing "right" and then "up", a convention is chosen:

$$\nearrow = \uparrow \ then \ \rightarrow$$

$$\nwarrow = \uparrow \ then \ \leftarrow$$

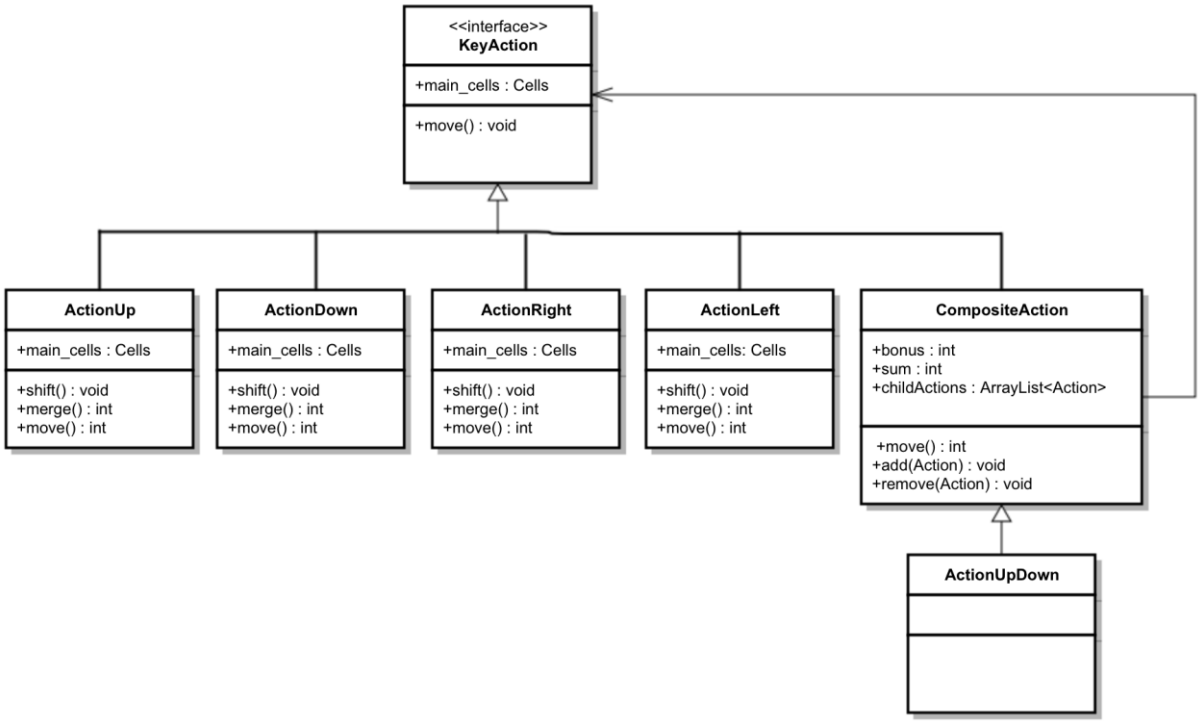
$$\searrow = \downarrow \ then \ \rightarrow$$

$$\swarrow = \uparrow \ then \ \leftarrow$$

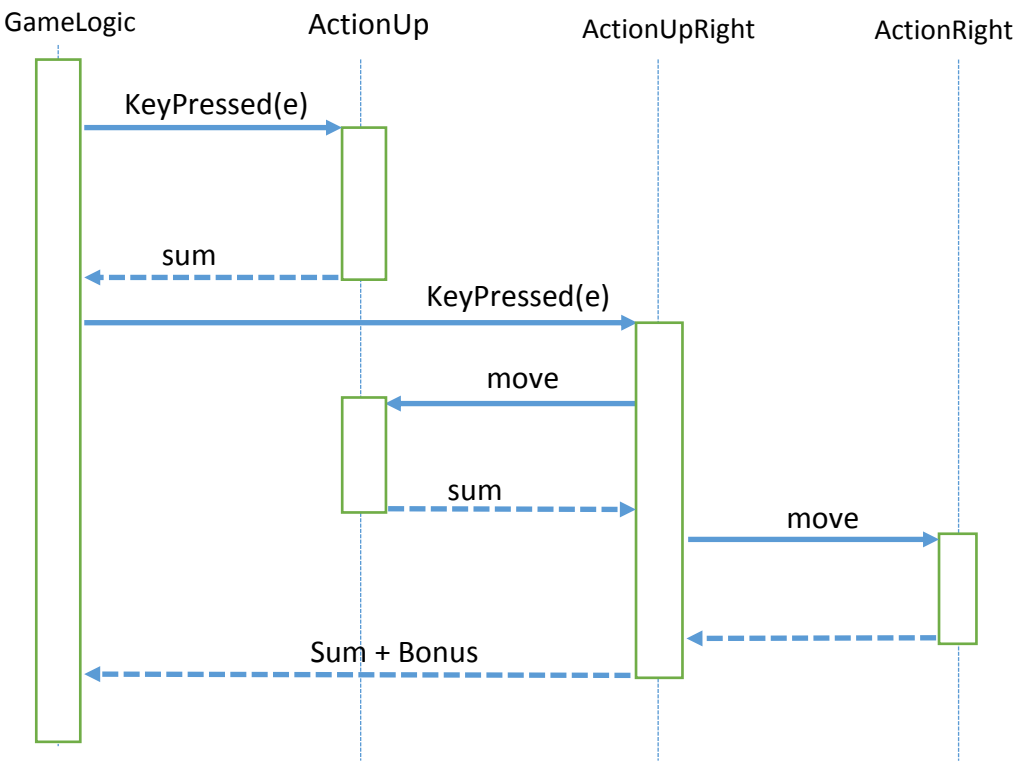
Since there is no difference in doing "up" and "right" instead of the correspondent diagonal movement, we have decided to reward the use of a diagonal movement with some bonus points. We made this choice because the player cannot visualize the intermediate moves and so he has to simulate the moves in his mind.

Implementing the Strategy Pattern, we created 4 classes which correspond to the four action (up, down, left, right). In order to implement a diagonal movement, a class CompositeAction is created; this class implements the same interface of the four initial movements. Then for every composition of two movements we create a new class which extends the CompositeAction class and it has an ArrayList of initial movements as attribute.

2 Class diagram



3 Sequence diagram



Case that has be taken into account: ↑ then ↗