**Joao Soveral - COMP40070 – Lab1 Report**

For this lab we saw that there is different ways of playing but all of them had the same pattern – 1. Initialization 2. Executing the play 3. Outputting the result. After realizing this we decided that the Template pattern would make sense.

We applied here that pattern, by creating a Super class Play with a general Play method and having sub classes that implement/override the inner Play methods, prepare\_play/execute\_play/return\_result.

This goes in line with the Strategy Pattern as we then decided to apply here as well. By doing this the Play class receives on the initialize constructor method the play\_type which is an instance of the specialized play. This way we only need to give the initialize method the “strategy” we want to follow. The code becomes more elegant and simpler to comprehend. Also, makes it very easy to add a new strategy like we did – reverse linear strategy.

**def** *initialize*(*play\_type*, *lower*, *upper*, *oracle*)  
 **@play\_type** = *play\_type* **@lower** = *lower* **@upper** = *upper* **@oracle** = *oracle***end  
  
def** *play* **@play\_type**.prepare\_play(**self**)  
 **@play\_type**.execute\_play(**self**)  
 **@play\_type**.return\_result  
**end**