1. **What changes you needed to make to accommodate the addition of one extra state?**
   1. Add a new state in Enum
   2. Add case in switch statement for current state
   3. Add method for handling the new state
   4. Add logic which trigger the transition from the new state to others
2. **How are the changes dependent on the number of transitions from/to this newly added state?**

Any state can transition into the new state under certain conditions, and it can get into other states from the new state according to different conditions. Thus, more transitions from/to the new state can cause changes in the scripts, mostly showing in the conditions statement which trigger the incoming and outcoming transition of the new state. If the new state connects with a certain number of other states, the restructure of FSM may be required.

1. **What changes you predict you need to make to accommodate the removal of one state?**
   1. Remove the state in Enum
   2. Remove the corresponding case in switch statement for current state
   3. Remove method for handling the state
   4. Remove or update the logic which involve the state being removed
2. **How are these changes you predict dependent on the states you start with and number of transitions from/to the state to be removed?**

As more transitions from/to the corresponding state can cause changes in the scripts, so more transitions connected with the state being removed, more changes needed. Also, if the state is connected with a certain number of other states, it could cause a restructure of the whole FSM.

1. **Is this method sustainable if you have a prior knowledge that the lead game designer is fond of changing (adding/removing) states a lot?**

No. Since every change depends on the number of transitions to/from the added or removed state, a large number of transitions would require numerous updates to the script and FSM manually. This increases the risk of bugs and errors, making the game development process inefficient.