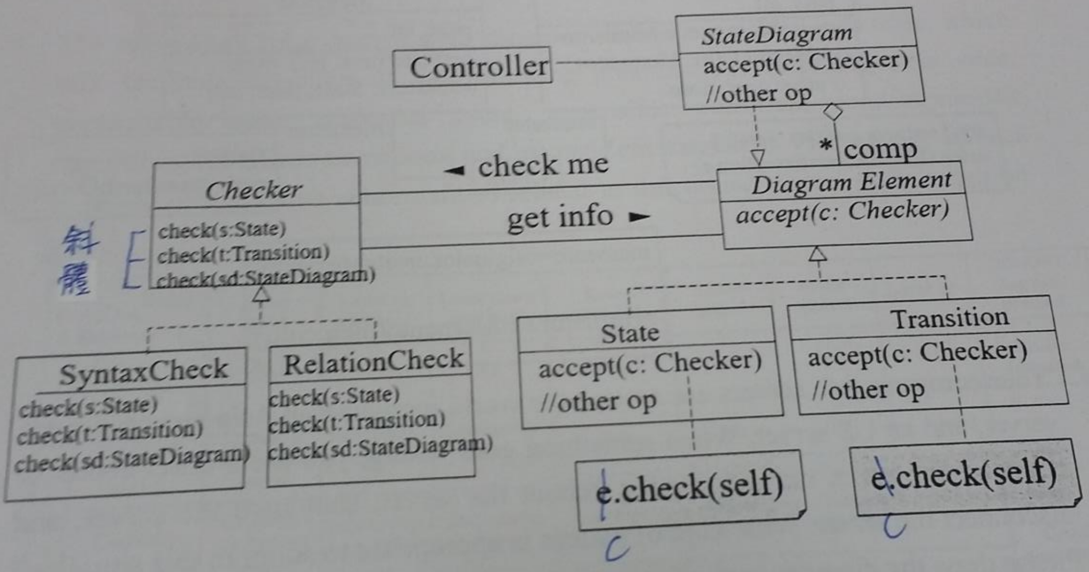
OOSE

Quiz2 05/18/2018

Name: 黃瑋琬 Anne ID: B10423036

1. The state diagram editor provides analysis capabilities such as checking the syntax and relation of state and transition specifications. We may also add additional operations later on. The problem is that the classes could be overloaded with unrelated responsibilities. An application of the visitor pattern is suggested to decouple these type-dependent operations from the classes of the structure. Please write the visitor pattern in Java code based on the provided diagram. **20%**



2. Fill in the appropriate answer for each question. **10%**

1) ____________________. This pattern decouples an algorithm from its implementation(s). It serves the same purpose as the Adapter and Bridge patterns, except that the encapsulated unit is a behavior.

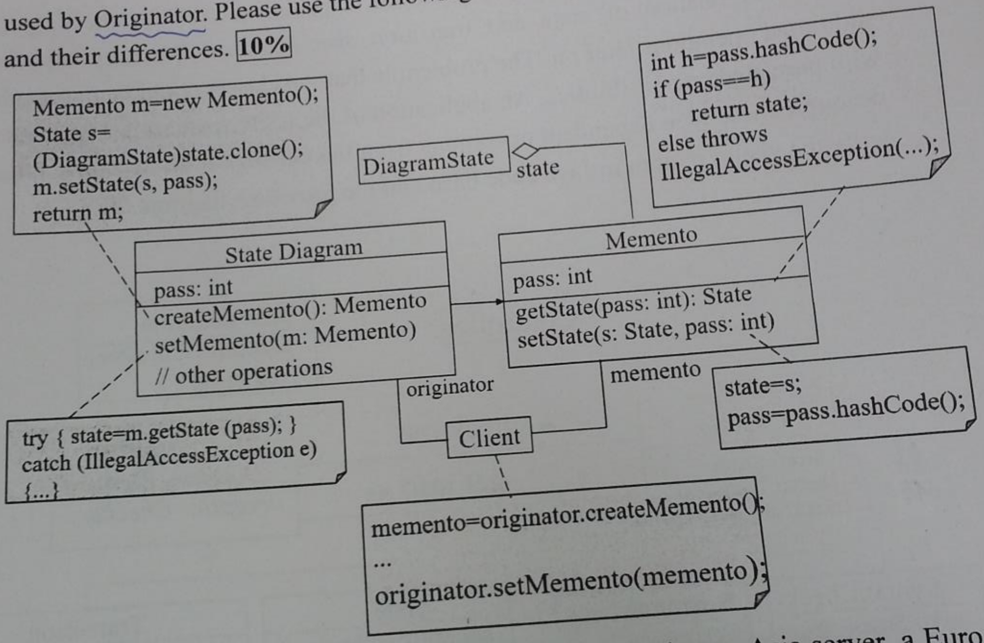
2) ____________________. This pattern provides a way to access the elements of an aggregate object sequentially without exposing its underlying representation.

3) ____________________. This pattern can be used to prevent unwanted concurrent update to a concrete observable class.

4) ______________. What are the types of software maintenance of Adapter patterns? (Note: A=Adaptive, C=Corrective, E=Enhancement, P=Perfective)

5) ______________. What are the types of software maintenance of Decorator patterns?

3. Mementos have effectively two interfaces. One is used by Caretaker, the other is used by Originator. Please use the following diagram to describe the two interfaces and their differences. **10%**



```
Memento m=new Memento();
State s=
(DiagramState)state.clone();
m.setState(s, pass);
return m;
```

```
int h=pass.hashCode();
if (pass==h)
    return state;
else throws
IllegalAccessException(...);
```

```
try { state=m.getState (pass); }
catch (IllegalAccessException e)
{...}
```

```
state=s;
pass=pass.hashCode();
```

```
memento=originator.createMemento();
...
originator.setMemento(memento);
```

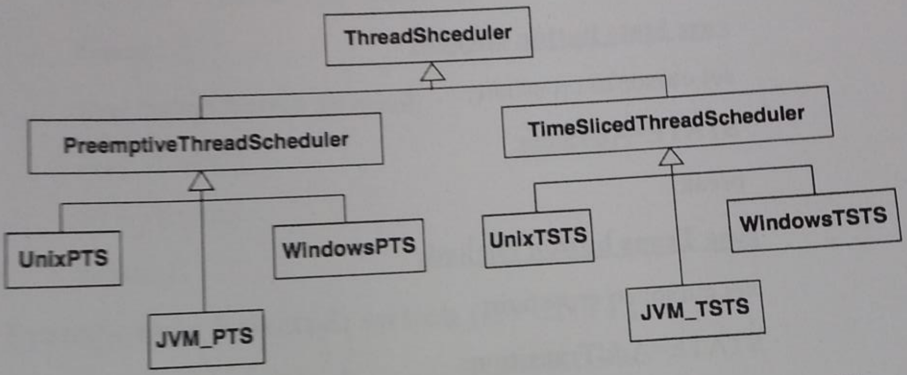
4. Your company web servers are around the world including an Asia server, a Euro server, and an US server. When something goes wrong, you use a crisis center to connect the server, run diagnostics, reboot the server, shutdown the server, and disconnect the server. What kind of pattern is appropriate to adopt in this situation? Please draw the diagram and annotate its participants. **10%**

5. Given a household switch controlling lights, ceiling fans, TV, etc. The purpose of the switch is to turn a device on or off. The actual switch can be implemented as a pull chain, simple two position switch, or a variety of dimmer switches. An application of the bridge pattern and command pattern is suggested to implement in this example. Which pattern would you adopt? Please draw the diagram of adopted pattern and write the main pieces of skeleton Java code. **20%**

人使用開關

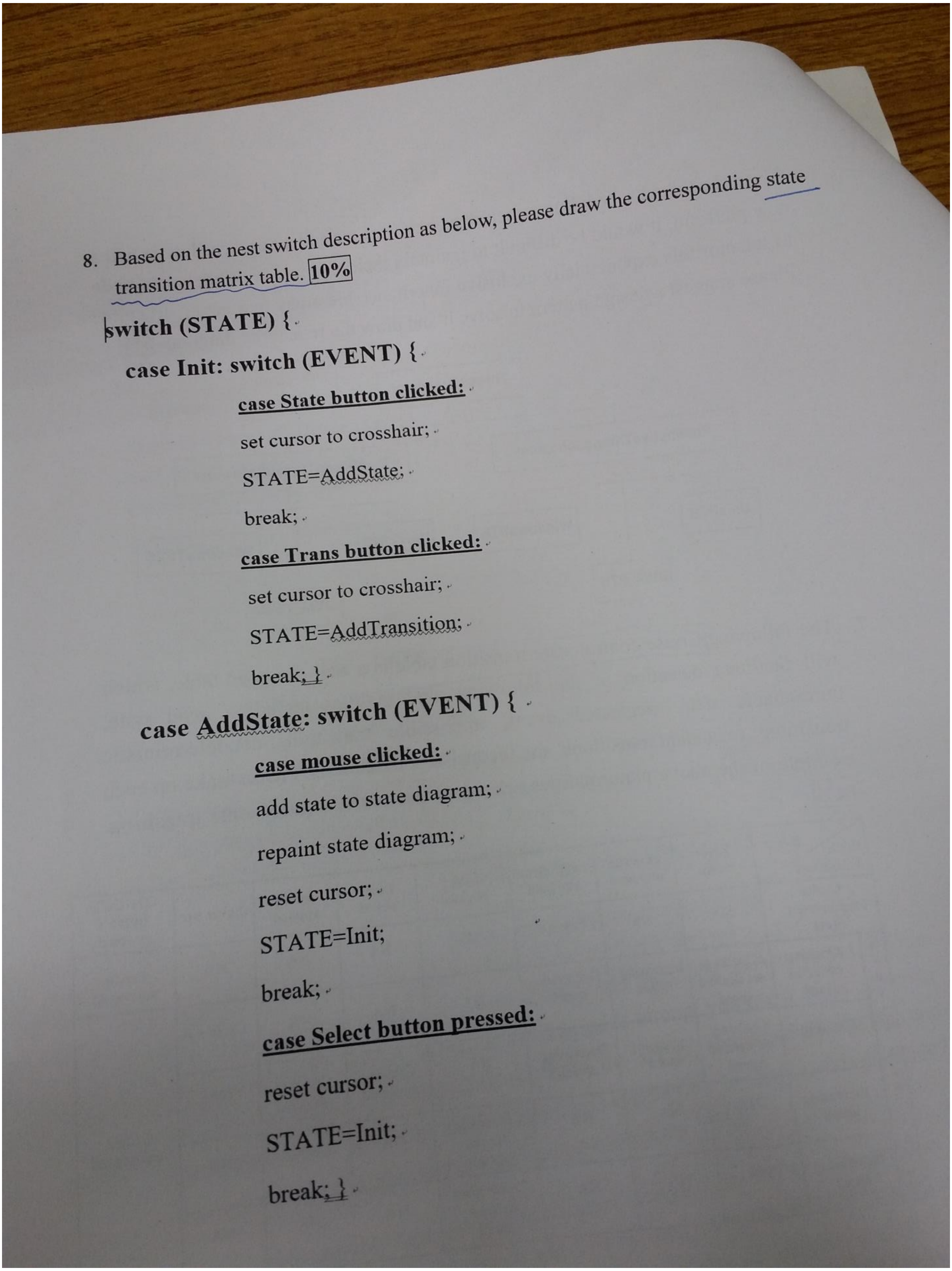
2個pattern選1個

6. Given the thread scheduling structure as below, if we add a new schedule or add a new platform, it would be difficult to maintain their association. What would to do to refactor this exponentially explosive inheritance hierarchy to a better structure? Please suggest a design pattern to solve it and draw the refactored diagram. **6%**



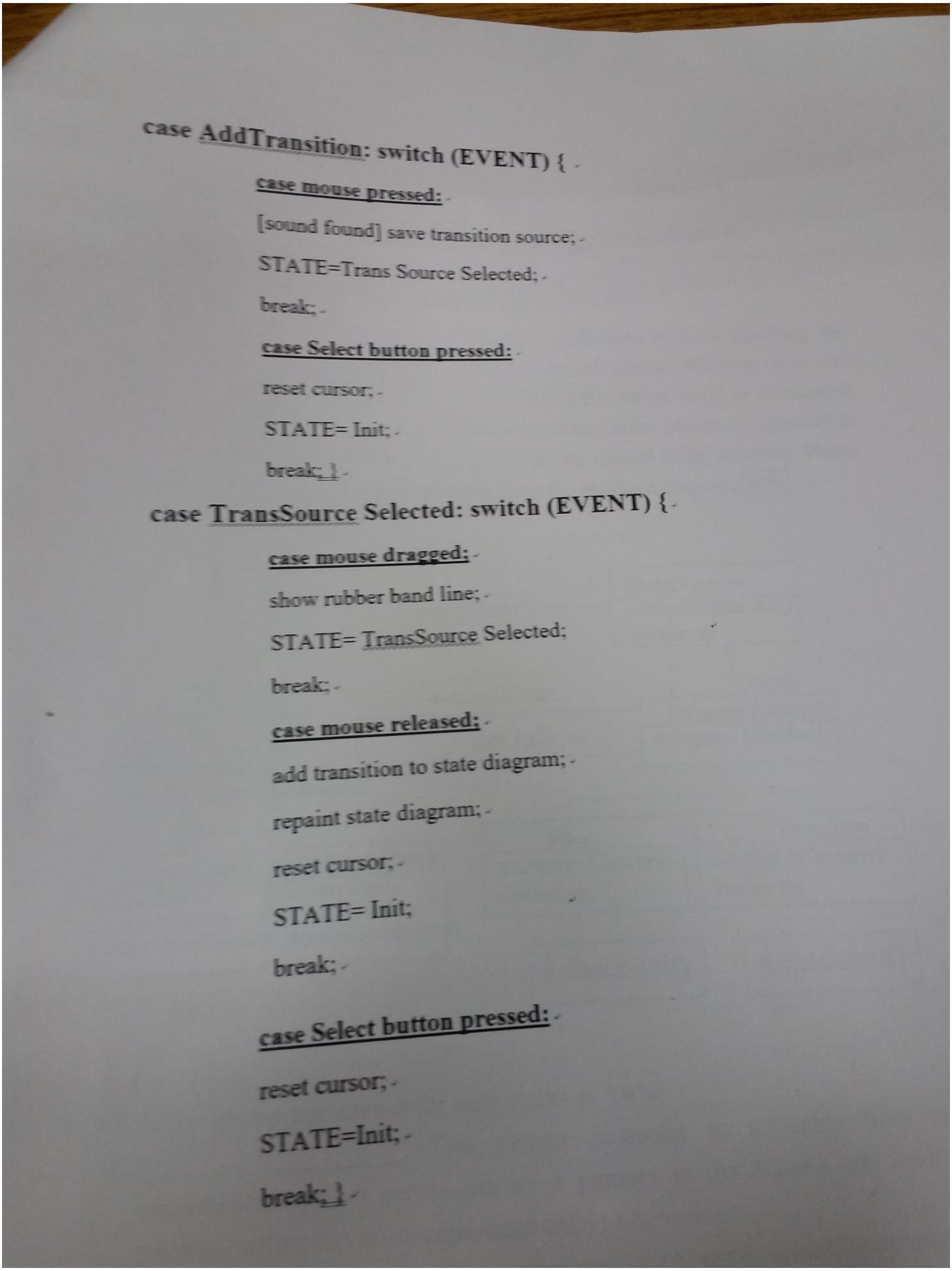
7. The following Cruise control state transition table is a well-designed table, which will facilitate detection of the following abnormalities including dead state, unreachable state, neglected event, impossible transition, nondeterministic transitions, redundant transitions, and inconsistent transitions. Please make up each example of the above abnormalities related to this cruise control state transition. **14%**

State & Substate		Lever down	Lever up and hold	Lever down and hold	Lever released	Lever pulled	Brake applied	Lever up	ON-OFF button pressed
Cruise deactivated (init)		NA	NA	NA	NA	NA	NA	NA	Cruise activated
Cruise Activated	Cruising canceled (init)	Cruising/ set desired speed	Increasing speed	Decreasing speed	NA	NA	NA	Cruising	Cruise deactivated
	Cruising	Cruising/ set desired speed	Increasing speed	Decreasing speed	NA	Cruising canceled	Cruising canceled	NA	
	Increasing speed	NA	NA	NA	Cruising/set desired speed	NA	NA	NA	
	Decreasing speed	NA	NA	NA	Cruising/set desired speed	NA	NA	NA	



8. Based on the nest switch description as below, please draw the corresponding state transition matrix table. 10%

```
switch (STATE) {
  case Init: switch (EVENT) {
          case State button clicked:
          set cursor to crosshair;
          STATE=AddState;
          break;
          case Trans button clicked:
          set cursor to crosshair;
          STATE=AddTransition;
          break; }
  case AddState: switch (EVENT) {
          case mouse clicked:
          add state to state diagram;
          repaint state diagram;
          reset cursor;
          STATE=Init;
          break;
          case Select button pressed:
          reset cursor;
          STATE=Init;
          break; }
```



```
case AddTransition: switch (EVENT) {
        case mouse pressed:
        [sound found] save transition source;
        STATE=Trans Source Selected;
        break;
        case Select button pressed:
        reset cursor;
        STATE= Init;
        break; }
case TransSource Selected: switch (EVENT) {
        case mouse dragged;
        show rubber band line;
        STATE= TransSource Selected;
        break;
        case mouse released;
        add transition to state diagram;
        repaint state diagram;
        reset cursor;
        STATE= Init;
        break;

        case Select button pressed:
        reset cursor;
        STATE=Init;
        break; }
```