# Question I

A Tamaguchi in a normal state is not hungry for a while (called autonomy time). At the end from this time, Tamaguchi is hungry and crying. To feed him, the user of Tamaguchi puts him at the table and the Tamaguchi stops crying. A Tamaguchi eats for a while (called time restoration). At the end of this time, he starts crying again. He cries until he leaves the table. When he comes out of the table, the Tamaguchi returns to the normal state ... and so on as long as the Tamaguchi does not die. If the Tamaguchi cries for more than 5 minutes in a row, he dies.

Draw a state transition-state diagram that models the behavior of Tamaguchi. We use the names "not hungry not cry", "hungry crying", "at table not crying ", "At table crying" and "Dead" for these 5 states.

# Question II

Draw a state / transition diagrams summarizing the possible states of a "contract" object as described in the following statement.

A group of people decide to establish a contract. To do this they write a project by successive iteration. The contract is then informally accepted by the parties, and becomes what is called a pre-agreement. At this point it can still be changed and returned to the project state. Once the pre-agreement has been finalized, the contract is signed by the parties. From this moment the partners are linked. Once signed the contract can be made enforceable by a decision of one of the parties. A contract in execution may be the subject of discussions which are settled by an arbitrator appointed for that purpose. The contract once executed ends.

# Question III

We consider a digital watch with two buttons mode and set. The watch can be in three modes: time display, change of hours and change of minutes. When the watch is in time display mode, pressing mode switches the watch to the hour change mode and the hour portion of the display (and only it) flashes. Pressing the mode again changes the watch to minute change mode, in which the minute portion of the display (and only it) flashes. A new mode press returns the watch to the time display mode and no part flashes.

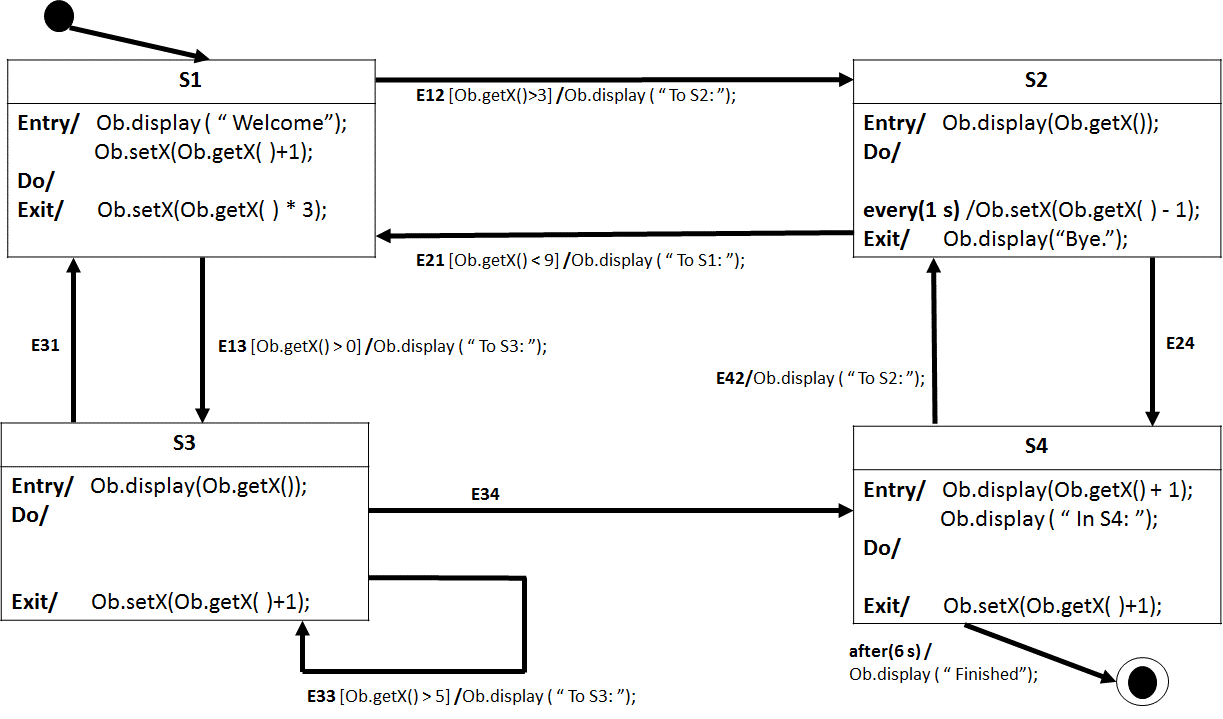
In time change mode, pressing set increments the hours of a unit and the watch remains in time change mode. Operation is similar in minute change mode. Finally, pressing set in time display mode is allowed but has no effect. Initially the watch is set to 0h00mn and in the time display state.

Initially the watch is set to 0: 0 i.e. 0h and 0mn and in the time display state.

Represent the transitions state diagram associated with the digital watch described above.

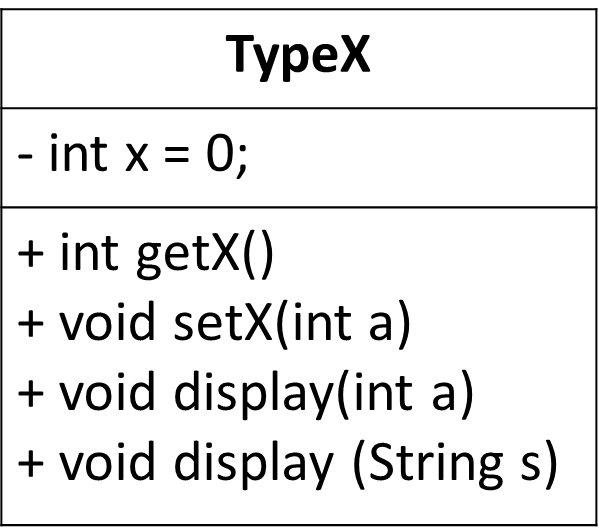
# Question IV

Let us have the following state-transition diagram



It is assumed that the execution time of the methods is negligible as well as the time of passage from one state to another.

The diagram corresponds to a set of states-transitions of an object **Ob** of type **TypeX** defined as follows:



## Part A

Compete the following table (**only the colored cases**) according to the diagram described above.

When an event has occurred and applied, put the new state in the same row of the table. In the column “on Screen”, put only which is new following an event.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Time (seconds) | Event | Display On Screen | Value of x | State |
| 0 |  | Welcome |  | S1 |
| 1 |  |  |  |  |
| 2 |  |  |  |  |
| 3 | E13 |  |  | S3 |
| 4 |  |  |  |  |
| 5 | E33 |  |  |  |
| 6 |  |  |  |  |
| 7 | E31 |  |  |  |
| 8 |  |  |  |  |
| 9 |  |  |  |  |
| 10 |  |  |  |  |
| 11 |  |  |  |  |
| 12 | E12 |  |  |  |
| 13 |  |  |  |  |
| 14 | E33 |  |  |  |
| 15 | E24 |  |  |  |
| 16 | E34 |  |  |  |
| 17 | E24 |  |  |  |
| 18 | E42 |  |  |  |
| 19 |  |  |  |  |
| 20 |  |  |  |  |
| 21 |  |  |  |  |
| 22 | E21 |  |  |  |
| 23 | E13 |  |  |  |
| 24 | E34 |  |  |  |
| 25 |  |  |  |  |
| 26 |  |  |  |  |
| 27 |  |  |  |  |
| 28 |  |  |  |  |
| 29 |  |  |  |  |
| 30 |  |  |  |  |

## Part B

For each of the following cases, give the name of the reached state, the value of the attribute **x** of the **Ob** object and determine what will be displayed on the screen:

1. At the beginning of the program, when the **Ob** object is initialized.

|  |  |  |
| --- | --- | --- |
| **Reached State** | **Ob.x** | **Display** |
|  |  |  |

1. When **Ob** is in state s1, with x = 2, and event E12 arrives:

|  |  |  |
| --- | --- | --- |
| **Reached State** | **Ob.x** | **Display** |
|  |  |  |

1. When **Ob** is in state s1, with x = 4, and event E12 arrives:

|  |  |  |
| --- | --- | --- |
| **Reached State** | **Ob.x** | **Display** |
|  |  |  |

1. Two seconds after **Ob** reaches s2 with x = 10:

|  |  |  |
| --- | --- | --- |
| **Reached State** | **Ob.x** | **Display** |
|  |  |  |

1. When **Ob** is in s3, with x = 6, and event E33 arrives:

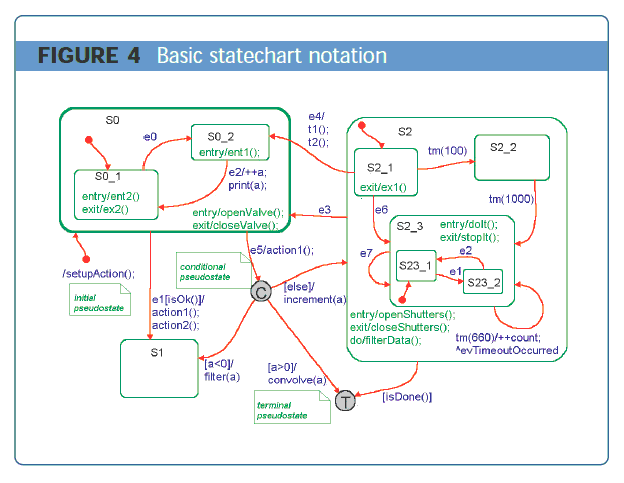
|  |  |  |
| --- | --- | --- |
| **Reached State** | **Ob.x** | **Display** |
|  |  |  |

1. 6 seconds after Ob arrives at s4 with x = 20:

|  |  |  |
| --- | --- | --- |
| **Reached State** | **Ob.x** | **Display** |
|  |  |  |

# Question V

Consider the following state transition diagram.



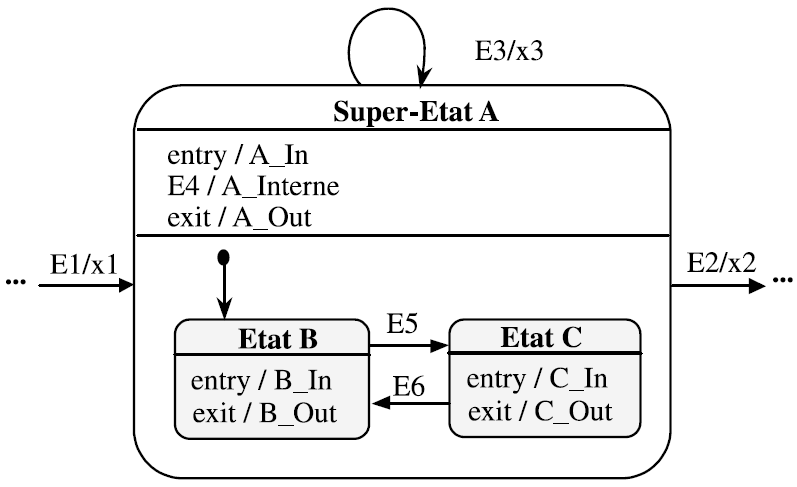


Complete the following Table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Current State** | **Event** | **Executed Actions** | **New State** |
| S0\_1 | e0 |  |  |
| S0\_2 | e0 |  |  |
| S0\_2 | e2 |  |  |
| S0\_2 | e1 &  isOK() == true |  |  |
| S0 | e5 & a == 0 |  |  |
| S23\_1 | tm(660) |  |  |
| S23\_1 | No event |  |  |
| S23\_2 | e3 |  |  |

# Question VI

Consider the following state - transition diagram.



Complete the following table :

|  |  |  |  |
| --- | --- | --- | --- |
| **Starting State** | **Event** | **Consequences** | **New State** |
| … | E1 |  |  |
|  | E5 |  |  |
|  | E4 |  |  |
|  | E6 |  |  |
|  | E3 |  |  |
|  | E5 |  |  |
|  | E3 |  |  |
|  | E2 |  |  |

# Question VII

Consider a class Party whose responsibility is to manage the course of a game of chess. This class can be in two states:

* the turn of the whites;
* the turn of the blacks.

The events to be considered are

* a movement of pieces by the black player;
* a movement of pieces by the white player;
* the request to take account of a checkmate by a player;
* A checkmate ensures the victory of the last player. In this case, a "blackWinning" or "WhiteWinning" activity is triggered;
* The request to take into account a tap that leads to an end game, with a tie. In this case, an "equality" activity is triggered.

Represent the state / transition diagram associated with the Part class.