**Aum Amriteswaryai Namah**

**19CSE205 – Program Reasoning – Assignment – 11**

**S. Abhishek**

**AM. EN. U4CSE19147**

**Construct the finite state diagram for the Dining Philosophers problem using Plant UML online.**

**Go to website** [**https://swaminathanj.github.io/jive/diagram.html**](https://swaminathanj.github.io/jive/diagram.html)

**Type the following text to generate the initial diagram showed in the class.**

|  |
| --- |
| note "State = [P1, P2, P3, P4, P5]" as n1  AllT : T,T,T,T,T  H1 : H,T,T,T,T  H2 : T,H,T,T,T  H3 : T,T,H,T,T  H4 : T,T,T,H,T  H5 : T,T,T,T,H  [\*] -right-> AllT  AllT --> H1  AllT --> H2  AllT --> H3  AllT --> H4  AllT --> H5 |

**Add more states and transitions to the best extent you can.**

**It should not contain bad states and blocking state.**

note "State = [P1, P2, P3, P4, P5]" as n1

AllT : T,T,T,T,T

H1 : H,T,T,T,T

H2 : T,H,T,T,T

H3 : T,T,H,T,T

H4 : T,T,T,H,T

H5 : T,T,T,T,H

H6 : H,T,H,T,T

H7 : T,H,T,H,T

H8 : T,T,H,T,H

H9 : H,T,T,H,T

H10 : T,H,T,T,H

[\*] -right-> AllT

AllT --> H1

AllT --> H2

AllT --> H3

AllT --> H4

AllT --> H5

AllT --> H6

AllT --> H7

AllT --> H8

AllT --> H9

AllT --> H10

E1 : E,H,H,H,H

E2 : H,E,H,H,H

E3 : H,H,E,H,H

E4 : H,H,H,E,H

E5 : H,H,H,H,E

E6 : E,H,E,H,H

E7 : H,E,H,E,H

E8 : H,H,E,H,E

E9 : E,H,H,E,H

E10 : H,E,H,H,E

H1 --> E1

H2 --> E2

H3 --> E3

H4 --> E4

H5 --> E5

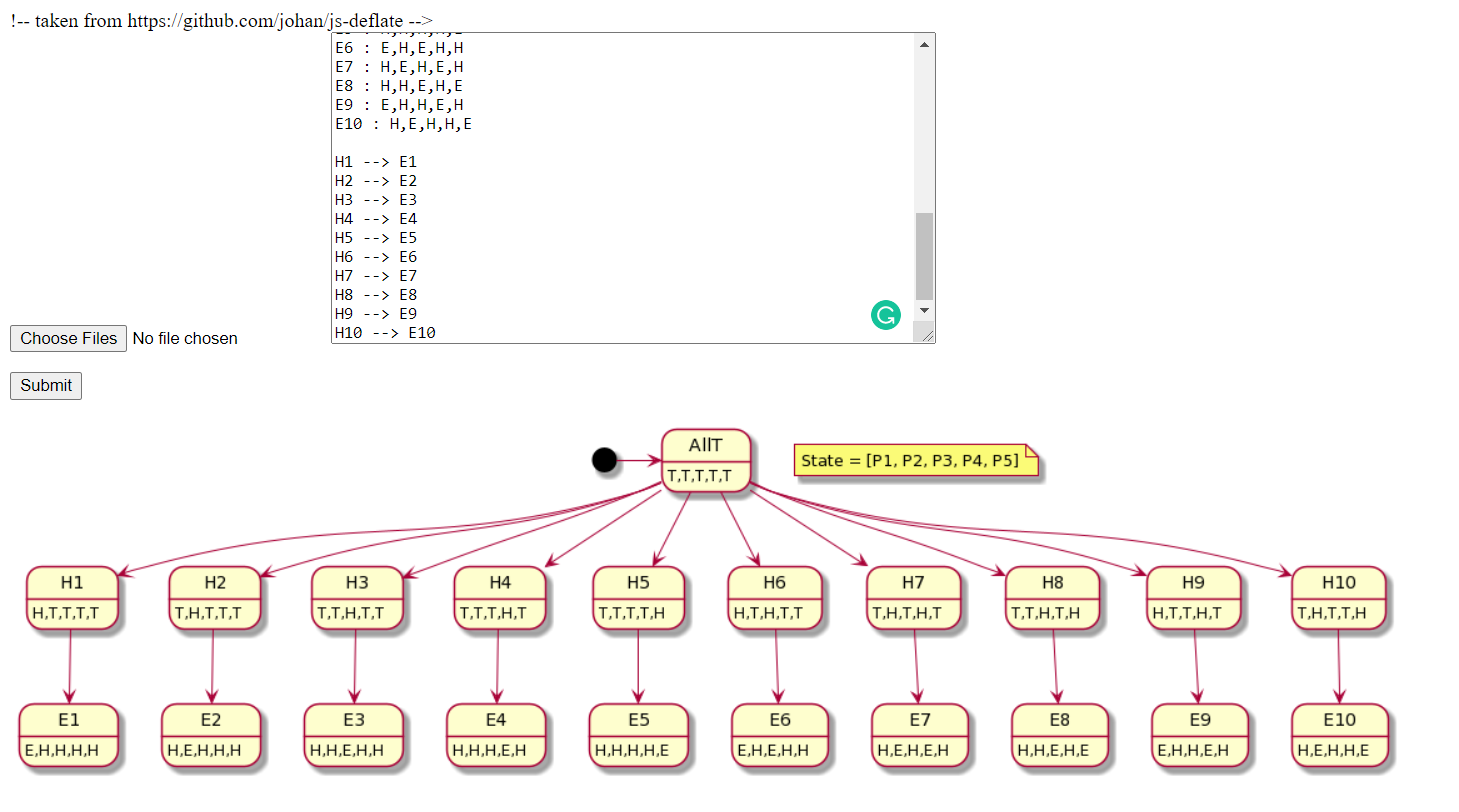
H6 --> E6

H7 --> E7

H8 --> E8

H9 --> E9

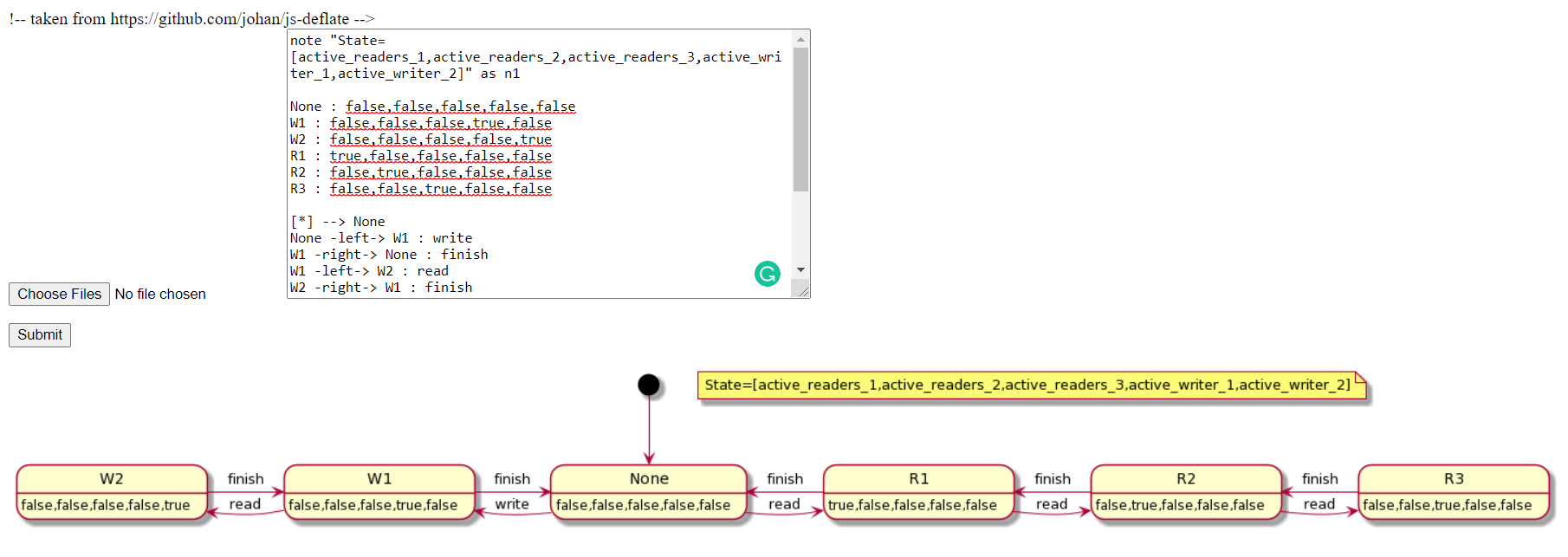
H10 --> E10

****

**Assume the case of Readers-Writers problem with 3 readers and 2 writers.**

**To be precise, state = [R1, R2, R3, W1, W2] with each process taking value true or false based on whether the process is active or not.**

* 1. Use the above online plant uml state-diagram generator to construct the complete state diagram with all possible states and transitions.



* 1. Comment on the following states based on the safety criteria.

|  |
| --- |
| 1. true, true, false, true, false  2. false, false, false, true, true  3. true, true, true, true, true |

**1. true, true, false, true, false**

* Write and read process cannot occur at same time.
* Because the there will be inconsistent information while reading each time.

**2. false, false, false, true, true**

* It is not safe to have multiple writers at a time.
* Because this too leads to inconsistent information.

**3. true, true, true, true, true**

* Write and read process cannot occur at same time.
* Because the there will be inconsistent information while reading each time.
* Multiple readers can read at a time because there is no possibility of updation in the data.

**How do you state the safety, liveness and fairness properties of the following problems in Linear-time Temporal Logic.**

**First write each property in English and then write it in LTL.**

**Traffic lights:**

**Safety criteria:**

P1: There should not be more than 1 Green signal at same time

G p1

**Liveness:**

P1: All signals should not be Red at same time

G p1

**Fairness:**

P1: No signal has to wait indefinitely to turn green.

G F p1

**Counter:**

**Safety:**

p1: Increment and decrement should not take place at same time & counter should not get to negative value.

**G p1**

**Liveness:** None

**Fairness:** None

**Readers-Writer:**

**Safety:**

P: When a writer is active

Q: No other process is active

P U Q

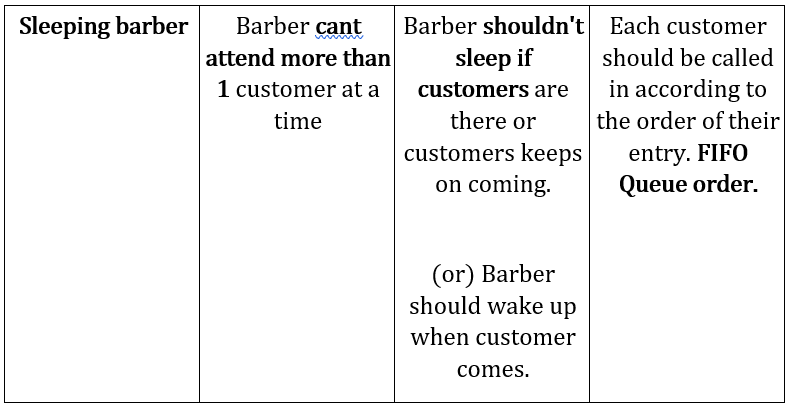
**Fairness:**

p: Reading process occurs

q: Writing process occurs

G (p -> F q)

**Sleeping Barber:**

****

**Safety:**

p: only one customer can get hair cut at a time.

G p

**Liveness:**

p: Barber is sleeping

q: Customer is waiting

p W q

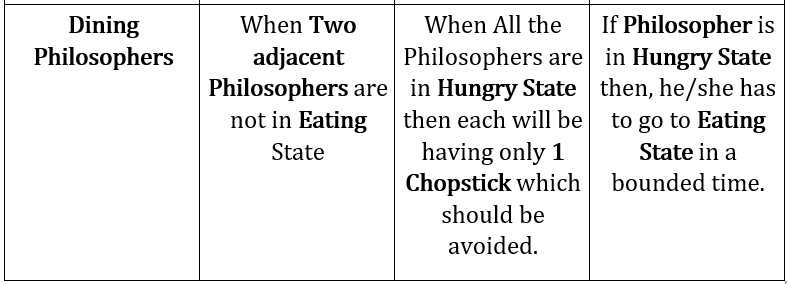
**Fairness:**

p: Customer waits.

q: Customer gets a Haircut.

X(p-> F q)

**Dining Philosopher**



**Safety criteria:**

p: No 2 adjacent philosophers can eat together.

G p

**Liveness:**

p: Not all philosophers are hungry with one fork.

G p

**Fairness:**

p: No philosophers waits indefinitely hungry.

G F p

**In the slides for each property X, F, G, U, W, R, F G, G F and conditional properties, one or two examples were demonstrated.**

**Provide two different examples for each property from any real-life systems around you.**

|  |  |  |
| --- | --- | --- |
| LTL  property | Real-life example 1  (evaluates to true) | Real life example 2  (evaluates to false) |
| X p | Desc:   * In traffic signal once we get yellow signal then the next signal should be green. * Stop, Get Ready, Go. * All lights should work simultaneously to make the traffic to be in control.   p:  p1: Yellow Color Signal Glows.  P2: Green Color Signal Glows. | Desc:   * Fund transfer between bank accounts. * If we transfer fund then, the account from which we need to transfer should be debited and the account to which the fund has to be credited should be credited. * But the false statement is that the source account gets credited. * The correct answer is it has to be debited.   p:  p1: Transferred Fund from Source Account.  P2: Source Account has to be debited. |
| F p | Desc:  When we drive an automobile if we find that there is no gasoline then out next task is to fill it eventually once it’s over.  When we encounter that there is no fuel then we need to fill it eventually to use the vehicle again.  p:  Fill gasoline in a Vehicle if its empty. | Desc:  If you are working as an inter trainee in a company and your training is for 1 year.  It’s not always necessary that after one-year training, he/she has to work in same company.  If he/she is not interested in the company then he/she can also leave from the company after training.  It need not be necessarily true always.  P: Trainee becomes permanent Employee of the company. |
| G p | Desc:   * Birthday date is true always in every year. * Though the day changes the date doesn’t change it remains the same. * Here my birthday date is 4th of February. * My birthday date remains same every year irrespective of year and day of the week.   p:  My Birthday Date is 4th of February | Desc:   * Sun rise and sun set is regular. * But its not necessary that sun should rise before 5.30 AM or 6 AM. * It depends on the climate and the time zone of each country. * We can’t predict the exact time.   p:  Sun rises before 5.30 AM. |
| p U q | Desc:   * In the same Traffic signal example, the green light glows until red light glows. * Vehicles can pass until they get the STOP signal.   p:  Traffic light signals Green light.  q:  Traffic light signals Red light. | Desc:   * When there is no charge in mobile phone its not always necessary that the mobile has to be active or in “on” state until it is recharged. * Even if we don’t use the mobile in low charge, the background process in mobile continues to run. * So, we can’t say that the mobile will stay active until we recharge it. * It can switch off after some time if the charge is too low.   p:  Mobile remains on or active.  q:  We recharge the mobile. |
| p W q | Desc:  Sun rise and sun set is the best example in this case.   * If the sun had raised once, then it has to be there showering the earth with its rays, until it sets. * Its always true that we get light until it sets. * We can’t say that sun doesn’t set, once it had raised up.   p: Sun rises and gives light.  q: Sun sets and Moon comes. | Desc:   * Consider that we are waiting for a bus in bus station. * We can’t surely say that bus will come within the expected time. * It may come or may not even come in future due to some reasons. * It’s not mandatory that we have to wait for a particular bus. * If any other bus or train is there, we can go for that. In this case the arrival of bus is not always true.   p:  I am waiting in Bus Station.  q:  Bus will come before 10 AM. |
| p R q | Desc:   * We need to work hard to get a good job. * Only if we work hard, we will get in to a good job. * Here Hard work continues till we get a good job as we expected. * Even after getting a good job u need not stop working hard.   p:  Get a good Job.  q:  Work Hard. | Desc:   * We can take the above Bus Example. * There it’s not mandatory that we have to wait in the bus station. * We can choose any other mode of transport other than waiting.   p:  Bus Comes at 10.00 AM.  q:  I am Waiting for Bus in Bus Station. |
| F G p | Desc:   * When the sports car starts ignition, it won’t initially itself run at top speed. * Only after 5 to 6 seconds of ignition and acceleration, the car reaches its top speed after few seconds. * The speed of the Super Car increases eventually.   p:  Speed of the Sports car. | Desc:   * But in the case of airplane only after reaching the max speed of 270 to 285 km/hour it should take off. * This is not the eventual event because only after reaching the top limit it is allowed to take off and not like taking off slowly.   p:  Airplane Take Off. |
| G F p | Desc:  Earth revolves around the sun again and again.  This action is true and continues to be true always.  Once its completes the rotation then the next 24 hours will come to effect which is probably the next day of the week.  p:  Revolving of Earth round the Sun. | Desc:   * We shall consider the example of clock where it keeps on ticking continuously. * Its always true that once the correct time is set it won’t show the wrong time. * But there also exists the situation where due to some internal issues in clock it won’t be able to display the correct time.   p:  Rotating Clock. |
| F (p → G q) | Desc:   * Whenever P is true eventually Q is always true. * When we give acceleration in super cars, as time goes on the speed of the car increases. * In this case eventual increase in acceleration results in the increase in the speed of the vehicle which is always true when acceleration is given.   p:  Acceleration is Given in Sports Car.  q:  Speed of the car increases. | Desc:   * We can consider the same case where the speed can’t increase more than that of the permitted of given limit. * Even if we keep on increasing the acceleration the speed of the car may not increase after reaching the particular limit.   p:  Giving Acceleration in Sports Car.  q:  Speed of the car increases. |
| G (p → F q) | Desc:   * Whenever p is true, q will be true eventually. * Whenever Red signal is given no vehicles are allowed to move further. * Means that they have to wait for some time to let the vehicles from opposite side to pass.   p:  Red Signal is Given in traffic Light  q:  Vehicles should stop for some time. | Desc:   * Though red signals are given ambulance, fire truck and other escorts need not wait until the green signal. * They are allowed to go without waiting in the signal.   p:  Red Signal is Given in traffic Light.  q:  All vehicle should stop for some time.  This is false because as mentioned, ambulance, fire truck and other escorts need not wait |

**Plantuml text for Traffic Lights**

note "State = [North South East West]" as n1

NorthFlow : Green Red Red Red

SouthReady : Green Yellow Red Red

SouthFlow : Red Green Red Red

EastReady : Red Green Yellow Red

EastFlow : Red Red Green Red

WestReady : Red Red Green Yellow

WestFlow : Red Red Red Green

NorthReady : Yellow Red Red Green

[\*] --> NorthFlow

NorthFlow -right-> SouthReady : ready

SouthReady -right-> SouthFlow : switch

SouthFlow -right-> EastReady : ready

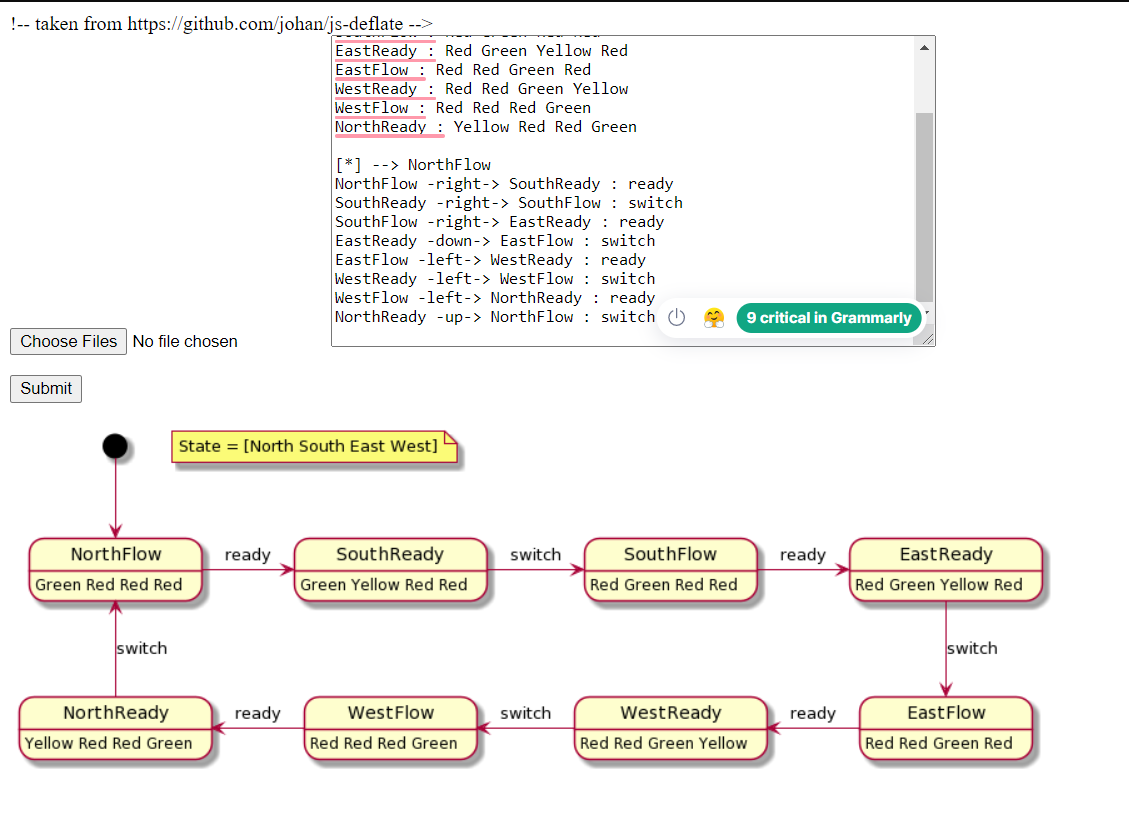
EastReady -down-> EastFlow : switch

EastFlow -left-> WestReady : ready

WestReady -left-> WestFlow : switch

WestFlow -left-> NorthReady : ready

NorthReady -up-> NorthFlow : switch



**Plantuml text for Counter**

note "State = [counter\_value]" as n1

zero : 0

one : 1

two : 2

three : 3

four : 4

five : 5

[\*] -right-> zero

zero -right-> one : incr

one -left-> zero : decr

one -right-> two : incr

two -left-> one : decr

two -right-> three : incr

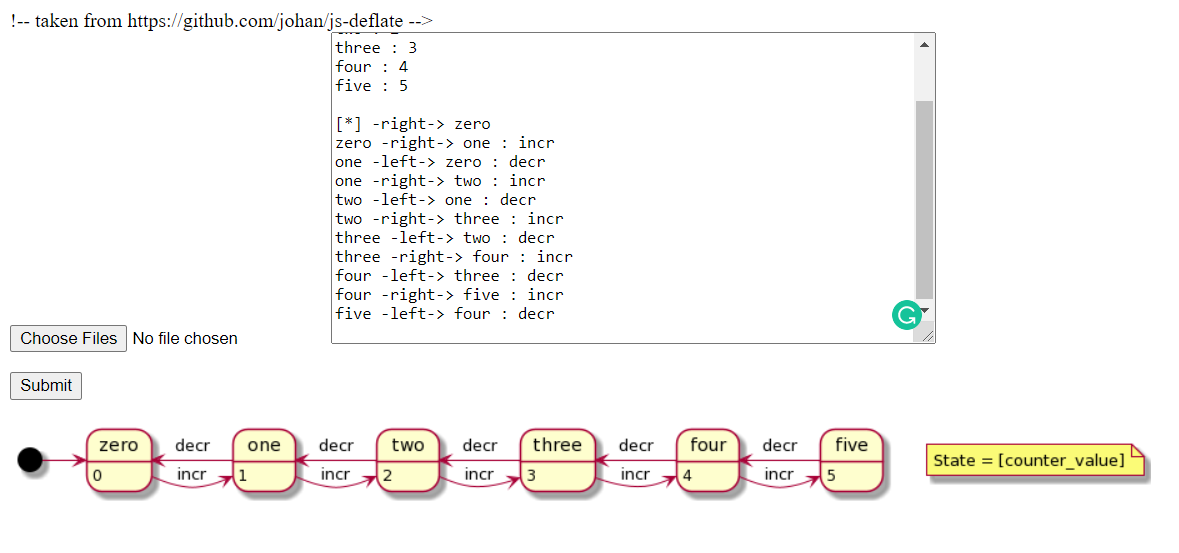
three -left-> two : decr

three -right-> four : incr

four -left-> three : decr

four -right-> five : incr

five -left-> four : decr



**Plantuml text for Readers-Writers**

note "State = [active\_readers,active\_writer]" as n1

None : 0,0

W : 0,1

R1 : 1,0

R2 : 2,0

R3 : 3,0

R4 : 4,0

[\*] --> None

None -left-> W : write

W -right-> None : finish

None -right-> R1 : read

R1 -left-> None : finish

R1 -right-> R2 : read

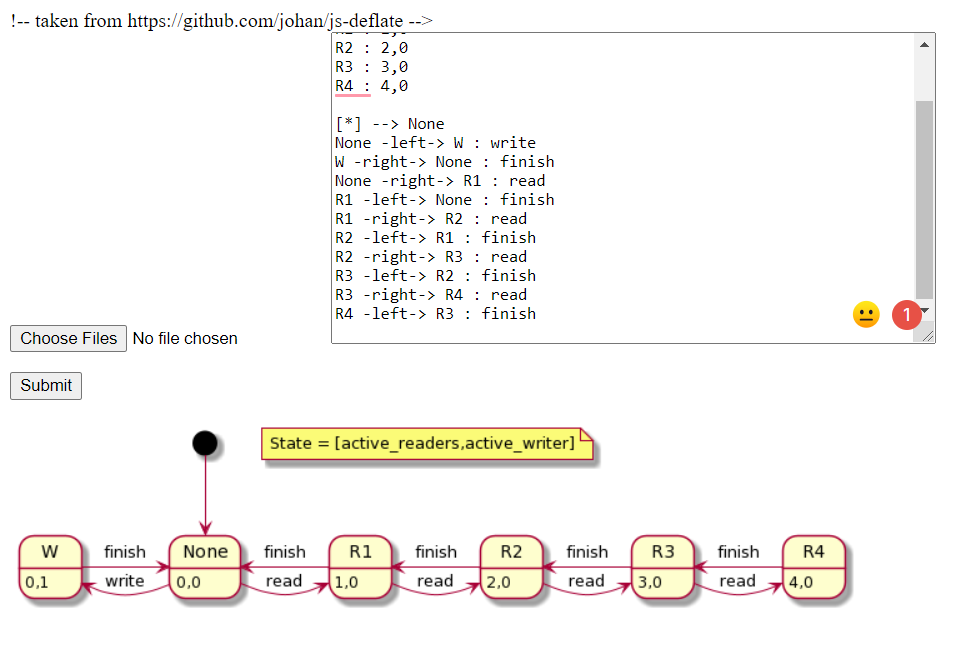
R2 -left-> R1 : finish

R2 -right-> R3 : read

R3 -left-> R2 : finish

R3 -right-> R4 : read

R4 -left-> R3 : finish



**Plantuml text for Sleeping Barber**

note "State = [Barber,Seat,Wait\_Customers]" as n1

None : Sleep,Empty,0

One : Cut,Occupied,0

Two : Cut,Occupied,1

Three : Cut,Occupied,2

Four : Cut,Occupied,3

[\*] --> None

None -right-> One : enter

One -left-> None : exit

One -right-> Two : enter

Two -left-> One : exit

Two -right-> Three : enter

Three -left-> Two : exit

Three -right-> Four : enter

Four -left-> Three : exit

