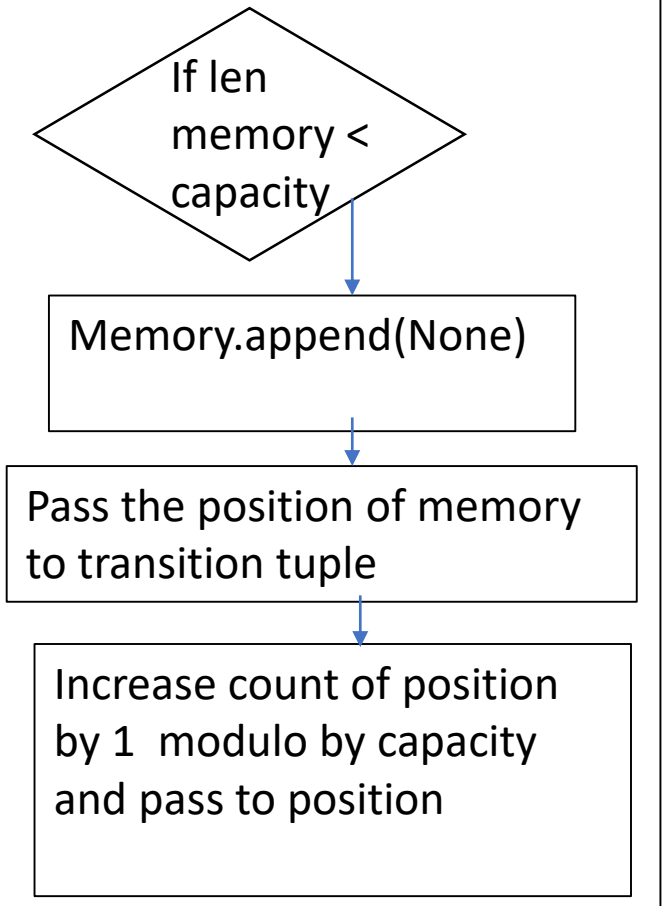
Class replay memory(object)

Function push -saving transition



Func init
Define capacity, memory array and position

Func length
Returns the len memory

Func sample
parameter: batch size
Func has .sample method and returns random batch of transitions for training

Class dqn(nn.module)

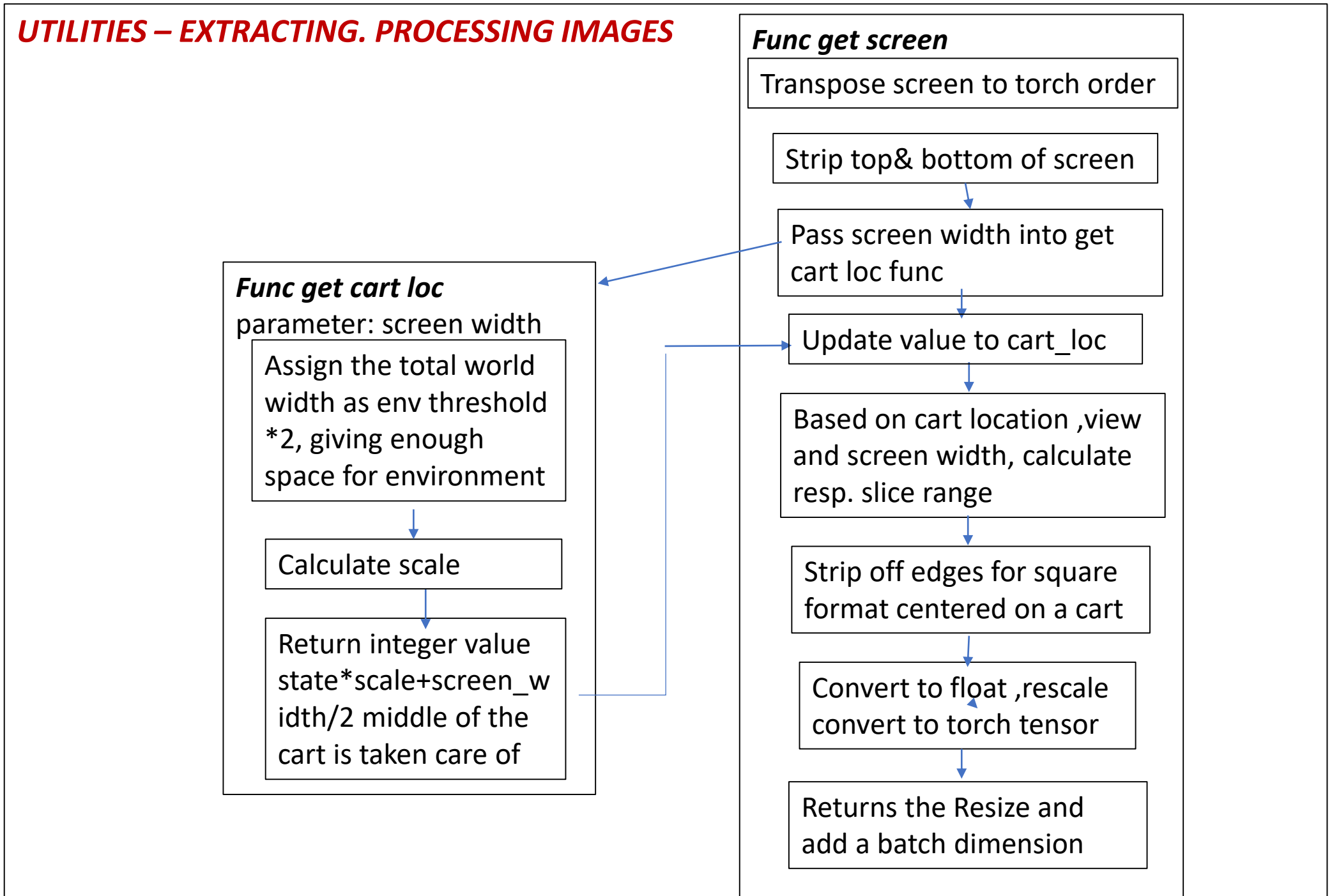
Func init
Parameter: height , width , output

Define 3 layers of conv network with input, output ,kernel size and stride
batchnorm after every layer

Fun conv2d_sizeout
Parameters: size, kernel size, stride
Compute output of conv layers
Calculate width, height, and input size

Func forward
Parameter: x
forward pass the 3 conv layers
Returns a tensor

UTILITIES – EXTRACTING. PROCESSING IMAGES



TRAINING

Call dqn class, optimize model() and assign random batch of transitions for training in replay memory

Declare training parameters like Batch size, gamma, eps_start,end,decay, tgtupdate
Get screen size, num of actions from gym action space.

To class dqn,
To class replaymemory
To optimize_model()

Func select action

- parameter: state

Random sample

Calculate eps_threshold

Increment steps_done

sample > eps thresh

else

If true

Return action with larger expected reward

Return tensor output of action using randrange

Func plot durations

Plot figure

Find tensor durations

Training plot

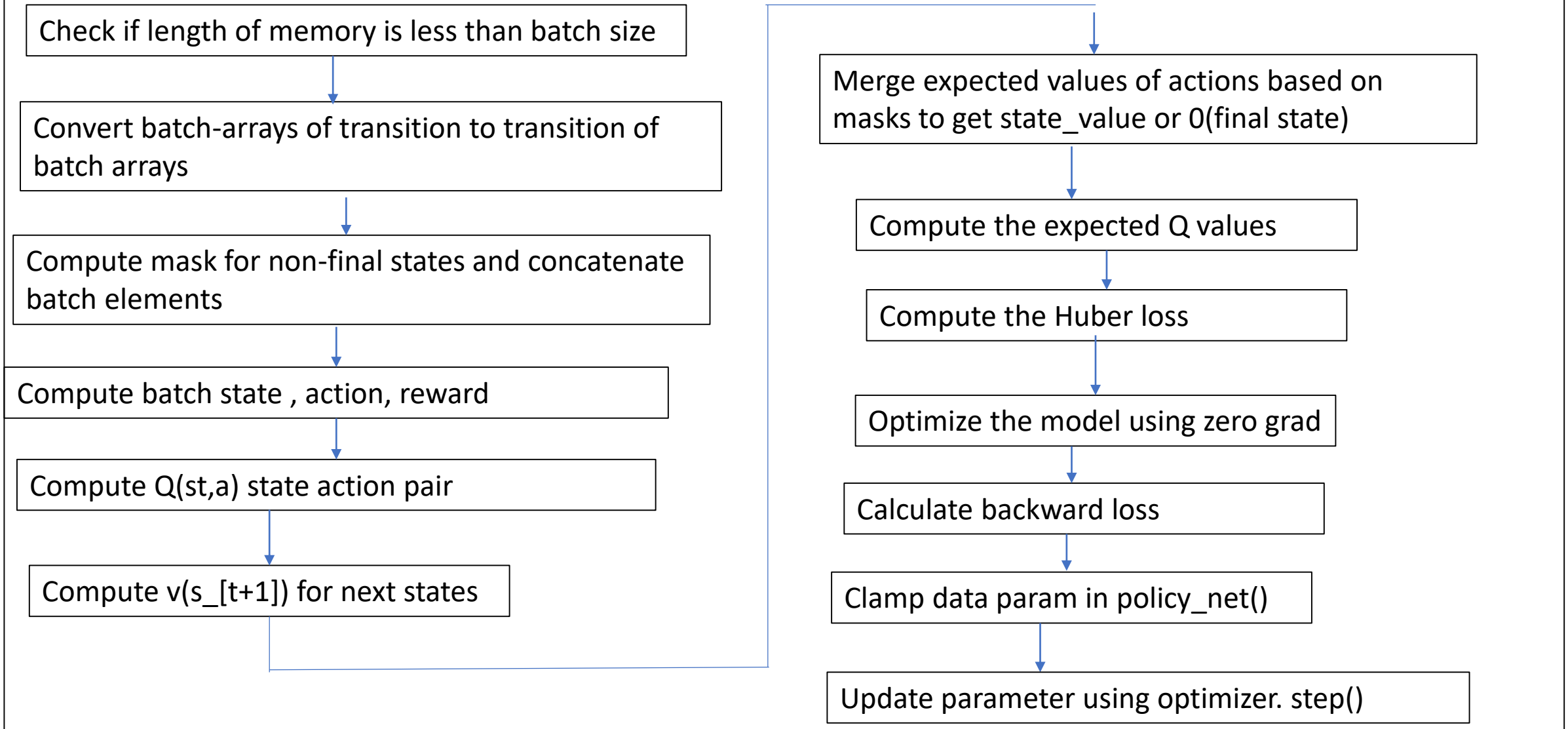
if dur len >=100

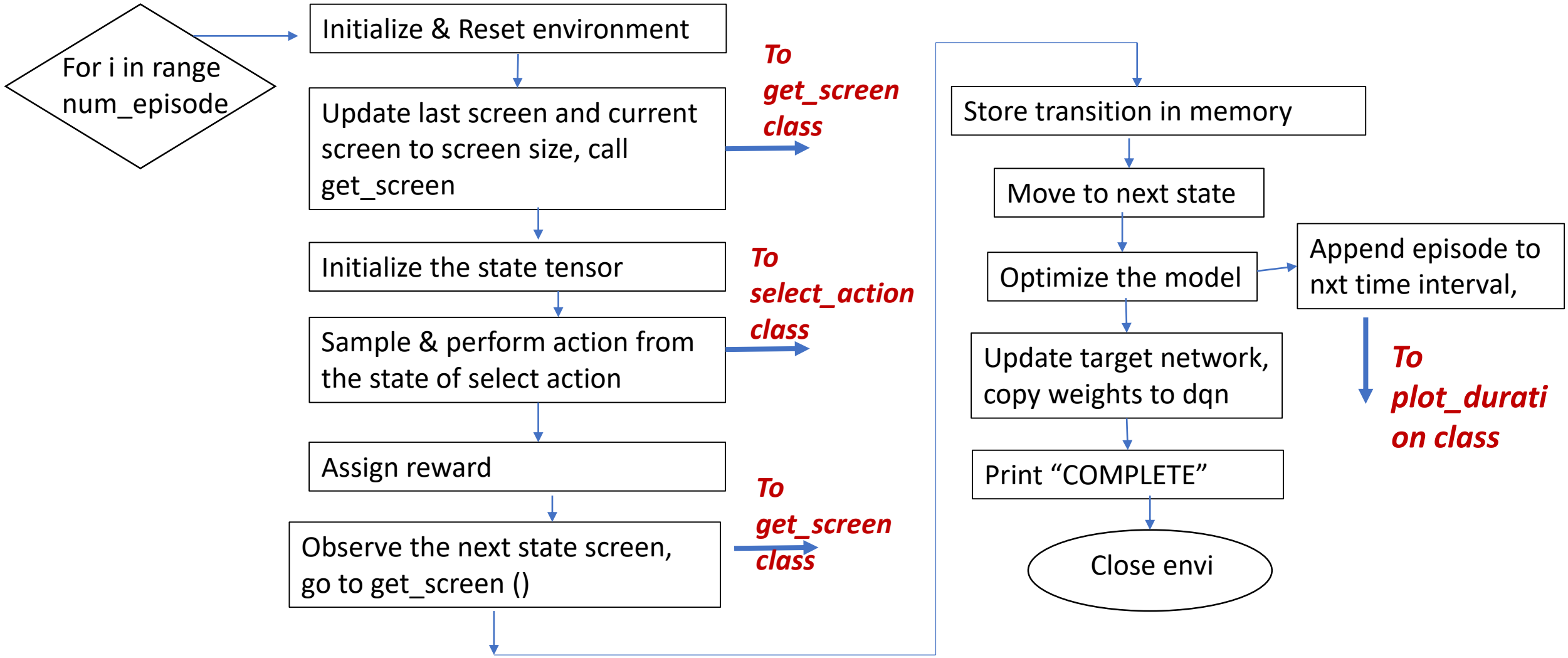
Calculate the mean value of durations

Plot across 100 episode averages

clear output, display current fig

Func optimize model




For i in range num_episode
Initialize & Reset environment
Update last screen and current screen to screen size, call get_screen
To get_screen class
Initialize the state tensor
To select_action class
Sample & perform action from the state of select action
Assign reward
Observe the next state screen, go to get_screen ()
To get_screen class
Store transition in memory
Move to next state
Optimize the model
Append episode to nxt time interval,
To plot_duration class
Update target network, copy weights to dqn
Print “COMPLETE”
Close envi