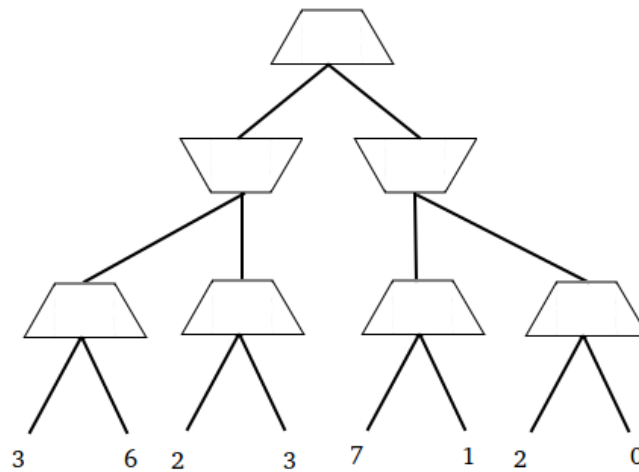


Q1. Games with Magic

(a) Standard Minimax

- (i) Fill in the values of each of the nodes in the following Minimax tree. The upward pointing trapezoids correspond to maximizer nodes (layer 1 and 3), and the downward pointing trapezoids correspond to minimizer nodes (layer 2). Each node has two actions available, Left and Right.
- (ii) Mark the sequence of actions that correspond to Minimax play.



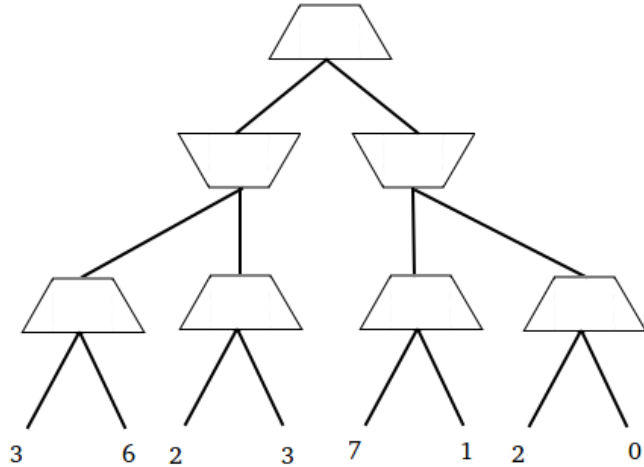
(b) Dark Magic

Pacman (= maximizer) has mastered some dark magic. With his dark magic skills Pacman can take control over his opponent's muscles while they execute their move — and in doing so be fully in charge of the opponent's move. But the magic comes at a price: every time Pacman uses his magic, he pays a price of c —which is measured in the same units as the values at the bottom of the tree.

Note: For each of his opponent's actions, Pacman has the *choice* to either let his opponent act (optimally according to minimax), or to take control over his opponent's move at a cost of c .

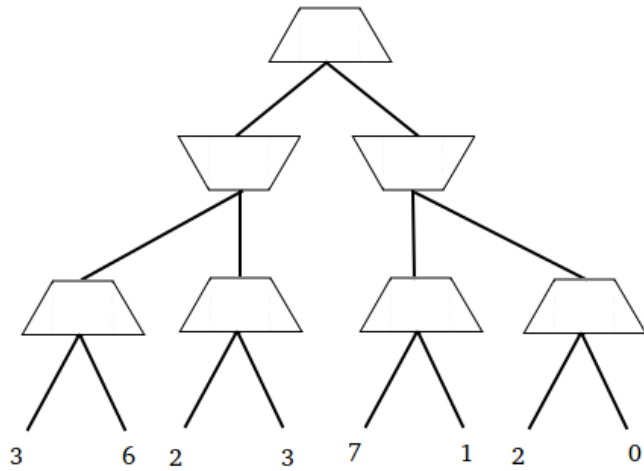
(i) Dark Magic at Cost $c = 2$

Consider the same game as before but now Pacman has access to his magic at cost $c = 2$. Is it optimal for Pacman to use his dark magic? If so, mark in the tree below where he will use it. Either way, mark what the outcome of the game will be and the sequence of actions that lead to that outcome.



(ii) Dark Magic at Cost $c = 5$

Consider the same game as before but now Pacman has access to his magic at cost $c = 5$. Is it optimal for Pacman to use his dark magic? If so, mark in the tree below where he will use it. Either way, mark what the outcome of the game will be and the sequence of actions that lead to that outcome.



(iii) Dark Magic Minimax Algorithm

Now let's study the general case. Assume that the minimizer player has no idea that Pacman has the ability to use dark magic at a cost of c . I.e., the minimizer chooses their actions according to standard minimax. You get to write the pseudo-code that Pacman uses to compute their strategy. As a starting point / reminder we give you below the pseudo-code for a standard minimax agent. Modify the pseudo-code such that it returns the optimal value for Pacman. Your pseudo-code should be sufficiently general that it works for arbitrary depth games.

```

function MAX-VALUE(state)
  if state is leaf then
    return UTILITY(state)
  end if
   $v \leftarrow -\infty$ 
  for successor in SUCCESSORS(state)
do
     $v \leftarrow$ 
max( $v$ , MIN-VALUE(successor))
  end for
  return  $v$ 
end function

```

```

function MIN-VALUE(state)
  if state is leaf then
    return UTILITY(state)
  end if
   $v \leftarrow \infty$ 
  for successor in SUCCESSORS(state)
do
     $v \leftarrow$ 
min( $v$ , MAX-VALUE(successor))
  end for
  return  $v$ 
end function

```

(iv) Dark Magic Becomes Predictable

The minimizer has come to the realization that Pacman has the ability to apply magic at cost c . Hence the minimizer now doesn't play according the regular minimax strategy anymore, but accounts for Pacman's magic capabilities when making decisions. Pacman in turn, is also aware of the minimizer's new way of making decisions.

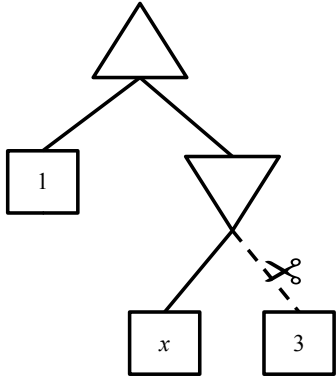
You again get to write the pseudo-code that Pacman uses to compute his strategy. As a starting point / reminder we give you below the pseudo-code for a standard minimax agent. Modify the pseudocode such that it returns the optimal value for Pacman.

```
function MAX-VALUE(state)
  if state is leaf then
    return UTILITY(state)
  end if
   $v \leftarrow -\infty$ 
  for successor in SUCCESSORS(state)
do
     $v \leftarrow$ 
    max( $v$ , MIN-VALUE(successor))
  end for
  return  $v$ 
end function
```

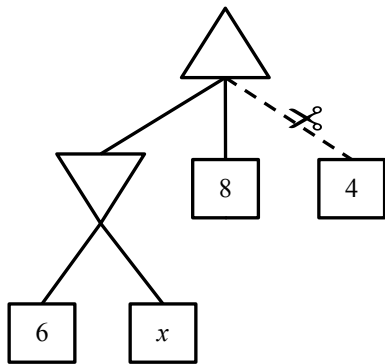
```
function MIN-VALUE(state)
  if state is leaf then
    return UTILITY(state)
  end if
   $v \leftarrow \infty$ 
  for successor in SUCCESSORS(state)
do
     $v \leftarrow$ 
    min( $v$ , MAX-VALUE(successor))
  end for
  return  $v$ 
end function
```

Q2. Games: Alpha-Beta Pruning

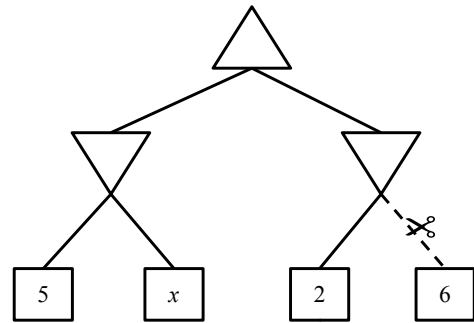
For each of the game-trees shown below, state for which values of x the dashed branch with the scissors will be pruned. If the pruning will not happen for any value of x write “none”. If pruning will happen for all values of x write “all”.



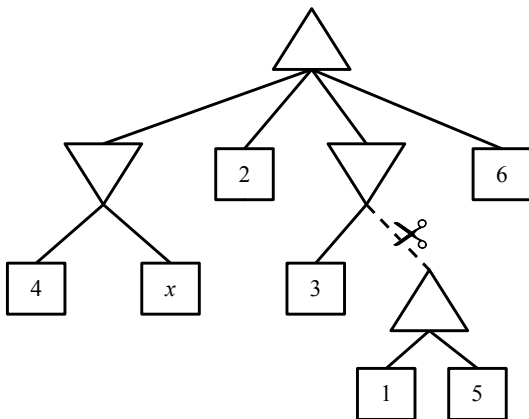
(a) Example Tree. Answer: $x \leq 1$.



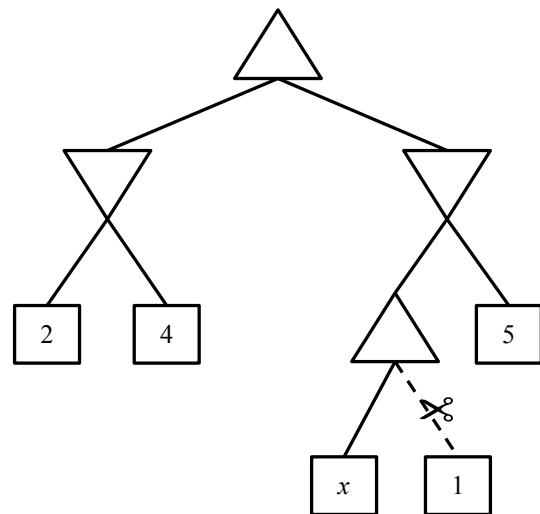
(b) Tree 1. Answer: _____



(c) Tree 2. Answer: _____



(d) Tree 3. Answer: _____



(e) Tree 4. Answer: _____