first name **last name** **id#**

1.
```
def makemove(p, cell, position, h): # assume cell empty
  putstone(p, cell, position) # cell in position gets color p
  if in_history(position, h):                 # line A
  if 0 == liberties(cell, position):          # line B
  removecaptured(opponent(p), cell, position) # line C
    return(ILLEGAL) # suicide                 # line D
    return(ILLEGAL) # superko                 # line E
  h.append(position) # add new position to move history
```

Function `makemove()` makes a legal Go move, but lines A-E are out of order. Give the correct order of execution of these five lines: ___ ___ ___ ___ ___

2.
```
def score(self):  # FILL IN THE MISSING LINES OF THIS PYTHON FUNCTION
  b,w = 0,0 # points for black, white
  seen = [False]*Cell.n   # all cells start unseen
  for c in range(Cell.n): # for each cell on the board
    if   self.brd[c] == Cell.b: b += 1 # c is black
    elif self.brd[c] == Cell.w: w += 1 # c is white
    elif not seen[c]:    # c is empty and not yet seen
      reach_b, reach_w, cells = False, False, 0
      seen[c] = True ; L = [c]
      while len(L)>0: # traverse from c
        t = L.pop() ; cells += 1
        for u in Neighbours[t]: # for each cell u adjacent to cell t
          if   self.brd[u] == Cell.b: ________________________________________
          elif self.brd[u] == Cell.w: ________________________________________
          elif not seen[u]: seen[u] = True ; _________________________________
      if   reach_b and not reach_w:   ________________________________________
      elif reach_w and not reach_b:   ________________________________________
  return b-w  # Tromp-Taylor score of input Go position
```

3. Solve this sliding tile puzzle. After each move, show the position (you might not need each blank position). The first move has been done for you.

```
start   5 4 3     4 3   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _
          2 1   5 2 1   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _

_ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   1 2 3  finish
_ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   _ _ _   4 5
```

When solving this starting position with breadth-first search, the number of positions encountered is around **(circle one)** 10 50 100 150 200 250 300 350 700 1400 3000 6000 12000

For sliding tile puzzles, what algorithm(s) always find(s) a shortest solution? **(circle ALL that apply)**

a) breadth-first search

b) A*-search with Manhattan distance heuristic

c) depth-first search

d) A*-search with number-misplaced-tiles heuristic

4.
```
  a b c
3 . . .
2 . . x
1 o . .
```

For this tic-tac-toe position,

a best move for `x` is ________

and the minimax result for `x` is **(circle one)** win lose draw

```
3 x o o
2 x o o
1 x x .
```

For this Go position with move history 1.B[b1] 2.W[b2] 3.B[a2] 4.W[c3] 5.B[a3] 6.W[c2] 7.B[a1] 8.W[b3] and Tromp-Taylor no-suicide positional superko rules,

a best move for Black (`x`) is ________

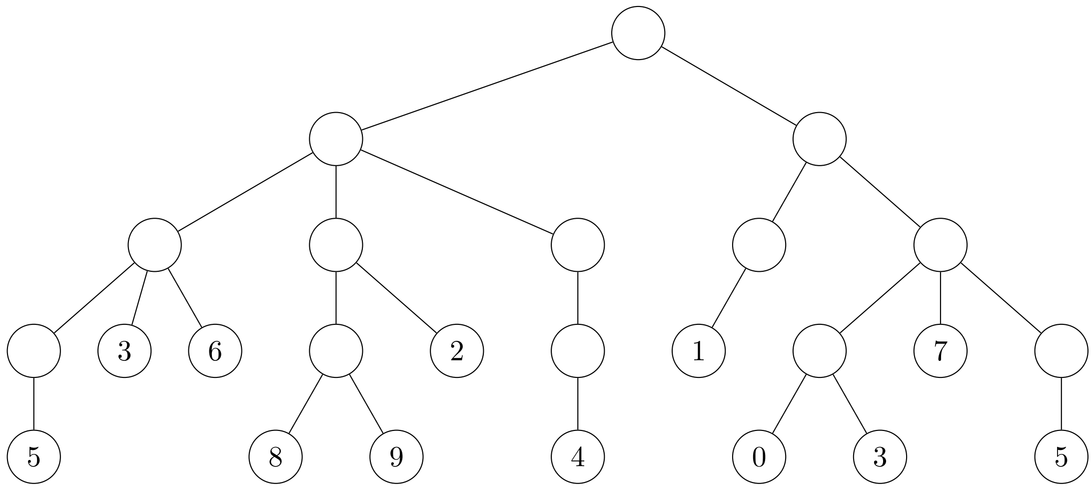
and Black's minimax score is ________ . **Explain briefly below.**

____

____

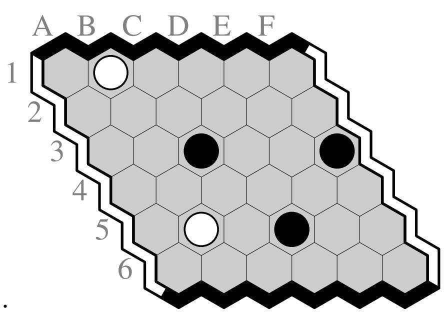
____

5. **Label** each node with its minimax (not negamax) value. The leafs are done for you. Root is a max node.



In this Hex position, what is White's mustplay region? (Hint: if Black plays E2 then E2-top, E2-F3 and F3-bottom are virtually connected using E1, F1, F2, E3, E4, F4, E5, F5, C6, D6, E6, F6; if Black plays C4 then C4-top, C4-D5 and D5-bottom are virtually connected using C1, D1, E1, F1, C2, D2, E2, D3, D4, C5, C6, D6.) **(below, circle all)**

6.



C1 D1 E1 F1 C2 D2 E2 F2 D3 E3 C4 D4 E4 F4 C5 E5 F5 C6 D6 E6 F6

For this position with White to play, give all winning moves: ______________________

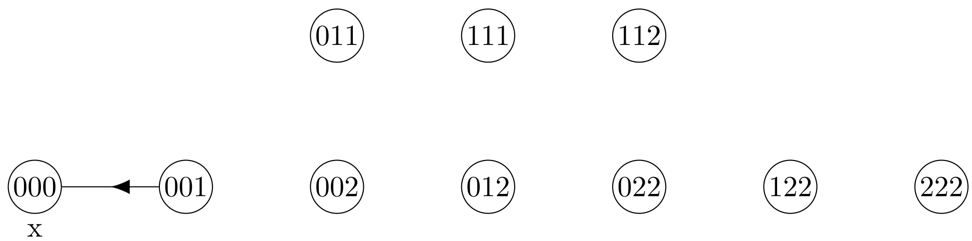
Which features strengthen computer Hex players? **(circle ALL that apply)**

a) filling captured cells b) subtracting the komi c) filling dead cells

d) modeling the board as a resistance network e) computing virtual connections and mustplay

f) in Monte Carlo simulations, replying when an opponent threatens a 2-cell virtual connection

7. Nim(2,2,2) starts with 3 piles, each with 2 stones. Below are nodes from the nim(2,2,2) state space dag (directed acyclic graph) grouped into isomorphism classes (e.g. (2,1,1), (1,2,1), (1,1,2) are in the class (2,1,1)). On the diagram, **add an arrow** to show each possible move (the arrow from (0,0,1) to (0,0,0) is already there) and **put** `x` under each losing node (the `x` under (0,0,0) is already there).



Now assume that we represent the nim(2,2,2) state space with a tree instead of a dag and that nodes are not grouped by isomorphism (so (2,1,1), (1,2,1) and (1,1,2) are different nodes). In the tree, how many children does root (2,2,2) have? ________________

Roughly, how many nodes are in the tree ? ________________ **show your work**

8. Give all winning first moves from nim(26,23,12,5). You might not need each line. **show your work**

Move ________ stones from the pile with ________ .

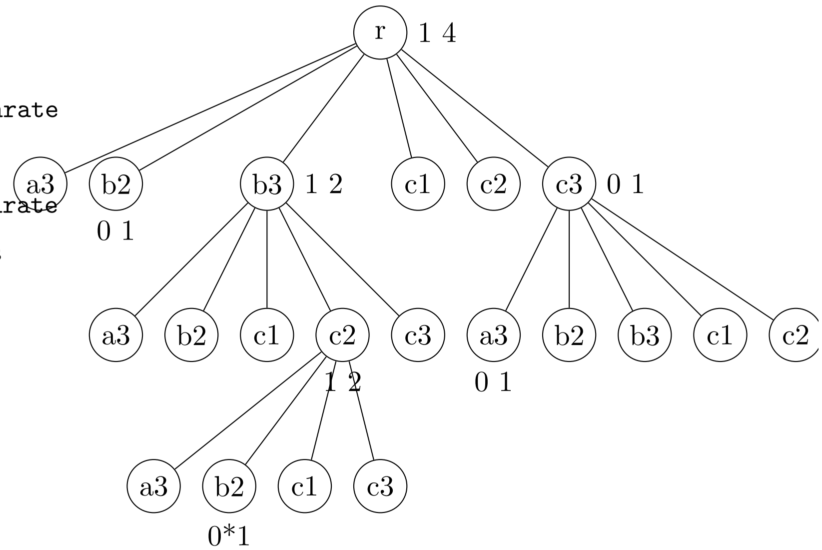
Move ________ stones from the pile with ________ .

Move ________ stones from the pile with ________ .

Move ________ stones from the pile with ________ .

Move ________ stones from the pile with ________ .

9.
```
0  while time remains:
1    node = root
2    while node has at least one child:
3      node = a child of node with largest winrate
5    add all children of node to tree
6    select a child x of node with largest winrate
7    from x, play random moves until game ends
8    if root player won, d = 1, else d = 0
9    node = x
10   repeat:
       node.wins += d
       node.visits += 1
       node = node.parent
12   until node == root
```



In the pseudocode above, the leaf selection phase is lines ________.

10. Consider the MCTS tree above. Unlabelled nodes have wins,visits 0,0. Assume that `winrate(wins,visits)` is defined as `(wins+1)/(visits+3)`, e.g. `winrate(0,0)` is `1/3` and `winrate(1,1)` is `2/4`. In the next MCTS iteration, the path to a leaf could be **(circle one)** r-a3 r-b2 r-b3-a3 r-b3-c2-a3 r-b3-c2-b2 r-c3-a3 r-c3-b2.

Assume instead that the path to the leaf was r-c3-c2 and that from a child of this leaf a simulation resulted in a win. Then after this iteration the labels on these c2,c3,r will be respectively ________ ________ ________.

In exploitation/exploration MCTS, the winrate function is modified by adding a term that rewards **(circle one)** a) high wins b) low wins c) high visits d) low visits e) high wins/visits f) low wins/visits .

The original AlphaGo modified MCTS by **(circle one)**

a) combining simulations with policy net calls b) combining simulations with value net calls c) replacing simulations with policy net calls d) replacing simulations with value net calls .

11. In game 4 of the AlphaGo match, Lee Sedol's move 78 was a surprise. **(circle ALL that apply)**

    a) most strong humans would not play this move b) AG's value net gave this move a low score c) AG's policy net gave this move a low probability d) AG's search gave this move a low probability .

    After the game, Lee Sedol said that **(circle ALL that apply)**

    a) the move felt right b) he saw that this move was winning c) from the previous moves, he felt that AG was in trouble d) based on game 2, he thought that AG would not expect this move .

    Because AG did not expect this move **(circle ALL that apply)**

    a) it turned on its database access code b) it spent more than 10 minutes on the next move c) it had to build a new search tree almost from nothing d) it did not have time to build a large search tree on its move .

12. Recall that Tromp's alpha-beta program solves 2x2 Go (positional superko): when moves are ordered so that *pass* is checked last, the search tree has about 20 000 000 nodes and max depth 58.

    In a smallest proof tree showing that the 2x2 Go 1st-player minimax score is ≥ 1,

    each node where the 1st-player moves next must show **(circle one)** **all** **one** of the children and

    each node where the 2nd-player moves next must show **(circle one)** **all** **one** of the children.

    This tree has only ________ nodes and max depth ________ . **show your work**