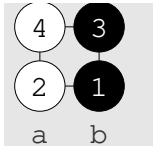


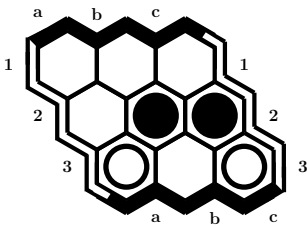
1. [4 marks] A go game starts with move sequence shown: self-capture is not allowed. For move continuations (a) and (b), give (i) first illegal move or **all ok** if all moves legal, (ii) why illegal: occupied, no liberties, superko or **all ok**, and after last legal move (iii) Black score (stones + territory) and (iv) White score.



- (a) 5.B[b1] 6.W[b2] 7.B[b1] 8.W[a2] (i) **7.B[b1]** (ii) **superko** (iii) **0** (iv) **4**
 (b) 5.B[b2] 6.W[b1] 7.B[b2] 8.W[a1] (i) **all ok** (ii) **all ok** (iii) **1** (iv) **1**

ROUGH WORK HERE

2. [2 marks] Here are `parent[x]` values for a hex position. Next a black stone is played at `b3`. Below, after each union, show the changes to `parent[x]`. `Union(a,b)` sets parent of root of `a`'s component to be root of `b`'s. `Tp`, `Bm`, `Lf`, `Rt` are top, bottom, left, right sides respectively.



<code>x</code>	<code>Tp Bm Lf Rt a1 b1 c1 a2 b2 c2 a3 b3 c3</code>
current <code>parent[x]</code>	<code>Tp Bm Lf Rt a1 b1 c1 a2 b2 b2 Lf b3 Rt</code>
after union <code>b3 b2</code>	<code>Tp Bm Lf Rt a1 b1 c1 a2 b2 b2 Lf b2 Rt</code>
then after union <code>b3 Bm</code>	<code>Tp Bm Lf Rt a1 b1 c1 a2 bm b2 Lf b2 Rt</code>

3. [3 marks] This line is from the hex board initialization in `hexgo/stone_board.py`:

```
self.nbr_offset = ((-1,0),(-1,1),(0,1),(1,0),(1,-1),(0,-1))
```

(a) Give the corresponding line for go board initialization for neighbour order above, right, below, left.

```
self.nbr_offset = ((-1,0),(0,1),(1,0),(0,-1))
```

(b) Carefully explain the purpose of line 4 below:

Line 4 ensures that neighbours that are outside the boundaries of the board are not checked.

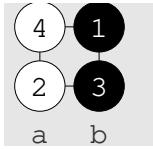
```
1 for r in range(self.r):
2     for c in range(self.c):
3         for (y,x) in self.nbr_offset:
4             if r+y in r_range and c+x in c_range:
5                 ...
```

4. [3 marks] Here is the start of the while loop in the go scoring function from class. Explain carefully: what does line `b_nbr |= (self.brd[x]==BLACK)` do?

```
while (len(empty_points) > 0):
    q = empty_points.pop()
    for j in self.nbr_offsets:
        x = j + q
        b_nbr |= (self.brd[x]==BLACK)
```

This line sets `b_nbr` to true if any neighbour is black.

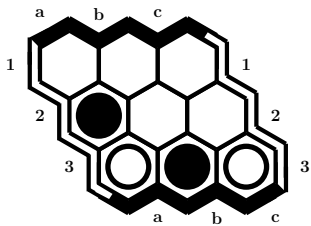
1. [4 marks] A go game starts with move sequence shown: self-capture is not allowed. For move continuations (a) and (b), give (i) first illegal move or **all ok** if all moves legal, (ii) why illegal: occupied, no liberties, superko or **all ok**, and after last legal move (iii) Black score (stones + territory) and (iv) White score.



- (a) 5.B[b1] 6.W[b2] 7.B[b1] 8.W[a2] (i) **all ok** (ii) **all ok** (iii) **1** (iv) **1**
 (b) 5.B[b2] 6.W[b1] 7.B[b2] 8.W[a1] (i) **7.B[b2]** (ii) **superko** (iii) **0** (iv) **4**

ROUGH WORK HERE

2. [2 marks] Here are `parent[x]` values for a hex position. Next a black stone is played at `b2`. Below, after each union, show the changes to `parent[x]`. `Union(a,b)` sets parent of root of `a`'s component to be root of `b`'s. `Tp`, `Bm`, `Lf`, `Rt` are top, bottom, left, right sides respectively.



<code>x</code>	<code>Tp Bm Lf Rt a1 b1 c1 a2 b2 c2 a3 b3 c3</code>
current <code>parent[x]</code>	<code>Tp Bm Lf Rt a1 b1 c1 a2 b2 c2 Lf Bm Rt</code>
after union <code>b2 Bm</code>	<code>Tp Bm Lf Rt a1 b1 c1 a2 bm c2 Lf Bm Rt</code>
then after union <code>b2 a2</code>	<code>Tp a2 Lf Rt a1 b1 c1 a2 bm c2 Lf Bm Rt</code>

3. [3 marks] This line is from the hex board initialization in `hexgo/stone_board.py`:

```
self.nbr_offset = ((-1,0),(-1,1),(0,1),(1,0),(1,-1),(0,-1))
```

- (a) Give the corresponding line for go board initialization for neighbour order left, below, right, above.

```
self.nbr_offset = ((-1,0),(0,1),(1,0),(0,-1))
```

- (b) Carefully explain the purpose of line 4 below:

Line 4 ensures that neighbours that are outside the boundaries of the board are not checked.

```
1 for r in range(self.r):
2     for c in range(self.c):
3         for (y,x) in self.nbr_offset:
4             if r+y in r_range and c+x in c_range:
5                 ...
```

4. [3 marks] Here is the start of the while loop in the go scoring function from class. Explain carefully: what does line `b_nbr |= (self.brd[x]==BLACK)` do?

```
while (len(empty_points) > 0):
    q = empty_points.pop()
    for j in self.nbr_offsets:
        x = j + q
        b_nbr |= (self.brd[x]==BLACK)
```

This line sets `b_nbr` to true if any neighbour is black.
