DSA Homework 2 COC 2720

1) To shuffle cards in List L, the first step is to break and assign variables into two lists with slicing.

```
length = len(L)
list1=L[0: length//2]
list2 = L[length//2 : length]
```

Next, a new list should be created to store the results

```
final_list = []
```

Next, a for loop can be used to append alternating values while looping

```
for x in range (length//2):
    final_list.append(list1[x])
    final_list.append(list2[x])
```

Finally, the newly shuffled list can be returned

```
return final_list
```

2) If S needs to be traversed using a Queue, the best way is to complete the operation then move the value to the Stack. The first step would be to create an empty list for queue. (Note: this code uses the Queue and Stack code taught in class)

```
def find_target(S, x):
    Q = Queue()
    var_found = False
```

The next step is to transfer values from S to Q while checking

```
while not S.is_empty():
    value = S.pop()
    if value == x:
        is_found = True
    Q.enqueue(value)
```

Then, all of Q needs to be put back to S so we can reverse.

```
while not Q.is_empty():
    S.push(Q.dequeue())  # takes the head value from Q and makes it
                         S's top
```

Now that Q is in S, the list needs to be reversed by enquing S back to Q, with S's top as Q's tail

```
while not S.is_empty():
    Q.enqueue(S.pop())
```

Next, the Q's values need to pushed to S so that it's in the original order

```
while not Q.is_empty():
    S.push(Q.dequeue()) # pushes to top Q's head value
```

Finally the nothing truth can be returned

```
return is_found
```

3) To solve this problem, a queue can be used to move the cows. First step is to initialize a queue and counter.

```
def cows():
    Q = Queue()
    time = 0
```

Next, Marie and Daisy cross by pushing them to the Queue

```
q.enqueue('Marie') # q's head
q.enqueue('Daisy')
time += 4
```

Then, Marie needs to be sent back with the yoke by dequeuing

```
q.dequeue() # dequeues Marie, sending her back
time += 2
```

Next, to optimize time, Crazy and Lazy can be sent together for 20 minutes by enqueuing them both

```
q.enqueue('Crazy')
q.enqueue('Lazy')
time += 20
```

Then, Daisy needs to be sent back with the yoke by dequeuing

```
q.dequeue()
time += 4
```

Finally, Marie and Daisy need to cross the bridge together by enqueuing

```
q.enqueue('Daisy')
q.enqueue('Marie')
time += 4
```

Now, the total time elapsed is $4+2+20+4+4=34$ and all four cows have crossed the bridge successfully.