1. Mechanical Sorting

Show the steps taken by each sort on the following unordered list:

0, 4, 2, 7, 6, 1, 3, 5

(a) Insertion sort

(b) Selection sort

(c) Merge sort

(d) Use heapsort to sort the following array (hint: draw out the heap). Draw out the array at each step:

0, 6, 2, 7, 4
2 Abstract Data Types

Recall the following ADTs when answering this question:

```
List
  add(element); // adds element to the end of the list
  add(index, element); // adds element at the given index
  get(index); // returns element at the given index
  size(); // returns the number of elements in the list

Set
  add(element); // adds element to the collection
  contains(object); // checks if set contains object
  size(); // returns the number of elements in the set
  remove(object); // removes specified object from set

Map
  put(key, value); // adds key-value pair to the map
  get(key); // returns value for the corresponding key
  containsKey(key); // checks if map contains the specified key
  keySet(); // returns set of all keys in map

Stack
  peek(); // returns front element of stack
  pop(); // removes and returns front element of stack
  push(element); // adds element to front of stack

Queue
  peek(); // returns front element of queue without removing it
  poll(); // removes and returns front element of queue
  offer(element); // adds element to front of queue

PriorityQueue
  add(element); // adds element to the PQ
  peek(); // returns front element of PQ without removing it
  poll(); // removes and returns the highest priority element in the PQ
```

(a) For each problem, which of the ADTs given in the previous section might you use to solve each problem? Which ones will make for a better or more efficient implementation?

1. Given a news article, find the frequency of each word used in the article.

2. Given an unsorted array of integers, return the array sorted from least to greatest.
3. Implement the forward and back buttons for a web browser.

(b) Define a Queue class that implements the offer and poll methods of a queue ADT using only a Stack class which implements the stack ADT.

*Hint:* Consider using two stacks.