1 Fill in the Blanks

Fill in the following blanks related to min-heaps:

1. `removeMin` has a best case runtime of ____________ and a worst case runtime of ____________.

2. `insert` has a best case runtime of ____________ and a worst case runtime of ____________.

3. A ____________ or ____________ traversal on a min-heap can output the elements in sorted order.

4. The fourth smallest element in a min-heap with 1000 distinct elements can appear in ____________ places in the heap.

5. Given a min-heap with $2^n - 1$ distinct elements, for an element
   - to be on the second level it must be less than ____________ element(s) and greater than ____________ element(s).
   - to be on the bottommost level it must be less than ____________ element(s) and greater than ____________ element(s).
2 Heap Mystery

We are given the following array representing a min-heap where each letter represents a unique number. Assume the root of the min-heap is at index zero, i.e. A is the root.

Array: [A, B, C, D, E, F, G]

Four unknown operations are then executed on the min-heap. An operation is either a removeMin or an insert. The resulting state of the min-heap is shown below.

Array: [A, E, B, D, X, F, G]

(a) Determine the operations executed and their appropriate order. The first operation has already been filled in for you!

1. removeMin()

2. 

3. 

4. 

(b) Fill in the following comparisons with either >, <, or ? if unknown. Note that this question does not assume a specific ordering of operations from the previous part, i.e. we don’t know which of the two possible

1. X _____ D

2. X _____ C

3. B _____ C

4. G _____ X
3 A Wordsearch

Given an N by N wordsearch and N words, devise an algorithm to solve the wordsearch in $O(N^3)$. Each word is at most N letters. For simplicity, no word is contained within another, i.e. if the word "bear" existed, "be" could not exist as well. See below for an example wordsearch:

```
H G E O R G E
M E N U J R A
U T N E H I S
H A A R I N S
O D R I Y A A
S N A A S H R
M I S L A T A
G L I C C E H
```

**Hint:** Add the words to a Trie, and you may find the `longestPrefixOf` operation helpful. Recall that `longestPrefixOf` accepts a `String` key and returns the longest prefix of the given string that exists in the Trie, or `null` if no prefix exists.