Problem Statement : Trinary Heap Analysis
Briefly analyze the running time for your heap's insert() and removeMin() operation. Write up your
Analysis.

Answer :
Trinary heap is a heap with 3 children as successors for every node starting from root.
An array can be represented as trinary heap with the following properties.
Assuming array index starts from 0 the parent of any child is given as ⌊(child-1)/3⌋
And 3 children are indexed as 3*parent + 1 to 3*parent + 3.

The height of the trinary hep is given as $log_3 n$

Now let us analyze insert() and removeMin() operations.

insert() : Inserting in heap is a 2 step process
1) Insert element at last index of array
2) Move the inserted value up comparing with it's parent value until parent value is less than current inserted value.

This functionality is implemented in heap3.java
Runtime Analysis for insert()
1) Insert value in array at last index which takes constant or O(1) time
2) Find parent index O(1) time
3) Loop from parent till root until we find parent value < current value,

In worst case this operation will compare heap height size elements and swaps, which is O($log_3 n$)

Hence in the worst case insert() operation takes O($log_3 n$) time.

removeMin() : remove in heap is a 3 step process
1) Remove the root value which is the smallest value in min-heap
2) Remove last element from array and put it as root of heap
3) Move the new parent value down until it finds the child value bigger than current value

This functionality is implemented in heap3.java
Runtime Analysis for removeMin()
1) Remove root value O(1) time or constant time
2) Remove last index from array and make it new root O(1) time
3) Compare root with all it's children value and swap it with the smallest child until we find the all child values greater than current value. Here heap will compare 3 values in the worst case multiplied by root height. Representing it as $O(3log_3 n)$.

(a) Is it asymptotically faster or slower than the same operation in a binary heap?
For insert operation the only difference between trinary heap and binary heap is the height of the tree which $log_3 n$ for trinary heap compared to log n of binary heap.
Hence insert operation is faster by some constant for trinary heap compared to that of binary heap.

For removeMin() operation trinary heap will make one extra comparison as it has 3 childs compared to 2 of binary heap but height of trinary heap is smaller than binary heap so the complexity differs just by a small constant.

(b) Would you expect it to have a larger or a smaller constant factor than the same operation with
a binary heap?
Trinary heap height is smaller by some constant as compared to binary heap and as all the operation in heap is bound by its height, trinary heap will have a smaller constant factor as compared to that of binary heap.