~\OneDrive - VietNam National University - HCM INTERNATIONAL UNIVERSITY\Desktop\DSA\DSA LAB NEW\Lab 1 OOP Reviews & Arrays\ITITSB22029_DoMinhDuy_Lab1\Problem 2\Problem 2.iii\HighArray.java

```java
import java.util.Random;

// HighArray.java
// demonstrates array class with high-level interface
/////

public class HighArray {
    private long[] a;
    private int nElems;
    private int comparisons;

    public HighArray(int max) {
        a = new long[max];
        nElems = 0;
        comparisons = 0;
    }

    public boolean find(long searchKey) {
        comparisons = 0;
        int j;
        for (j = 0; j < nElems; j++) {
            comparisons++;
            if (a[j] == searchKey) {
                break;
            }
        }
        return j != nElems;
    }

    public void insert(long value) {
        a[nElems] = value;
        nElems++;
    }

    public boolean delete(long value) {
        int j;
        for (j = 0; j < nElems; j++) {
            if (value == a[j]) {
                break;
            }
        }

        if (j == nElems) {
            return false;
        } else {
            for (int k = j; k < nElems - 1; k++) {
                a[k] = a[k + 1];
```

```
48              }
49              nElems--;
50              return true;
51          }
52      }
53
54      public long getMax() {
55          if (nElems == 0) {
56              return -1;
57          }
58          long max = a[0];
59          for (int i = 1; i < nElems; i++) {
60              if (a[i] > max) {
61                  max = a[i];
62              }
63          }
64          return max;
65      }
66
67      public void noDups() {
68          for (int i = 0; i < nElems; i++) {
69              for (int j = i + 1; j < nElems; j++) {
70                  if (a[i] == a[j]) {
71                      a[j] = -1; // Mark duplicate with -1
72                  }
73              }
74          }
75          int newSize = 0;
76          for (int i = 0; i < nElems; i++) {
77              if (a[i] != -1) {
78                  a[newSize++] = a[i];
79              }
80          }
81          nElems = newSize;
82      }
83
84      public int getComparisons() {
85          return comparisons;
86      }
87
88      public static void main(String[] args) {
89          Random rand = new Random();
90          HighArray arr = new HighArray(1000);
91
92          // Insert 100 random items
93          for (int i = 0; i < 100; i++) {
94              arr.insert(rand.nextInt(1000));
95          }
96
97          // Find a random item and print the number of comparisons
```

```
98          long searchKey = arr.a[rand.nextInt(100)];
99          arr.find(searchKey);
100         System.out.println("Comparisons to find " + searchKey + ": " + arr.getComparisons());
101
102         // Compute and print the average number of comparisons over 100 trials
103         int totalComparisons = 0;
104         for (int i = 0; i < 100; i++) {
105             searchKey = arr.a[rand.nextInt(100)];
106             arr.find(searchKey);
107             totalComparisons += arr.getComparisons();
108         }
109         System.out.println("Average comparisons over 100 trials: " + (totalComparisons /
    100.0));
110
111         // Print the average number of comparisons for arrays with sizes 100, 200, ...,
112         // 1000
113         for (int size = 100; size <= 1000; size += 100) {
114             arr = new HighArray(size);
115             for (int i = 0; i < size; i++) {
116                 arr.insert(rand.nextInt(1000));
117             }
118             totalComparisons = 0;
119             for (int i = 0; i < 100; i++) {
120                 searchKey = arr.a[rand.nextInt(size)];
121                 arr.find(searchKey);
122                 totalComparisons += arr.getComparisons();
123             }
124             System.out.println("Average comparisons for size " + size + ": " +
    (totalComparisons / 100.0));
125         }
126     }
127 }
128
```