1. Lab 1: OOP Reviews & Arrays

1.1. Objectives

i. Review basic OOP and Java skills
ii. Arrays and operations with arrays: find, insert, delete
iii. Ordered arrays, binary search
iv. Implementing arrays is Java
v. Arrays of objects
vi. Efficiency of aray operations, big O notation

1.2. Problem 1: Simple application

i. Write a function to convert array to number. (10pts)

Suppose we have loaded an array with the digits of an integer, where the highest power is kept in position 0, next highest in position 1, and so on.

The ones position is always at position array.Length - 1:

```
int[] digits = { 2, 0, 1, 8 };
```

ii. Write a function to input a list of integer numbers and return the median of that list (10pts).

iii. Find the min-gap (10pts)

Write a method named minGap that accepts an integer array and a number of elements as parameters and returns the minimum 'gap' between adjacent values in the array. The gap between two adjacent values in an array is defined as the second value minus the first value.

For example, suppose a variable called array is an array of integers that stores the following sequence of values:

```
int[] array = {1, 3, 6, 7, 12};
```

The first gap is 2 (3 - 1), the second gap is 3 (6 - 3), the third gap is 1 (7 - 6) and the fourth gap is 5 (12 - 7).

Thus, the call of minGap(array, n) should return 1 because that is the smallest gap in the array. If you are passed an array with fewer than 2 elements, you should return 0.

iv. GasMileage.java (input, the use of Scanner class to read numeric data)

What happens if you enter 20.5 for miles? Fix the problem in two ways.

What happens if enter miles and gallons on the same line?

Modify the program so that it reads two lines of inputs with car, miles, and gallons (separated by blanks) and prints mpg for each car.

v. Student.java, Students.java, students.txt (text files, loops, decision making, access modifiers)

Add public/private to the declaration of student names and print students with System.out.println(st.lname), explain.

Change the while-loop with a for-loop

Use grade to determine the type of the student: excellent (> 89), ok [60,89], and failure (< 60)

Do counting and averaging within each student type (excellent, ok, and failure)

1.3. Problem 2

i. Compile and Run the Example Programs

- Array, LowArray, HighArray
- OrderedArray
- ClassDataArray

ii. Programming Projects 2.2 in Text-Book (lowArray.java)

iii. HighArray.java

Add a method called getMax() that returns the value of the highest key in the array, or –1 if the array is empty. Add some code in main() to exercise this method. You can assume all the keys are positive numbers.

Write a noDups() method for the HighArray class. This method should remove all duplicates from the array. That is, if three items with the key 17 appear in the array, noDups() should remove two of them. Don't worry about maintaining the order of the items. One approach is to first compare every item with all the other items and overwrite any duplicates with a null (or a distinctive value that isn't used for real keys). Then remove all the nulls. Of course, the array size will be reduced.

Insert 100 random items (generated with nextInt()) and find one of them chosen at random. Print the number of comparisons (add a counter of comparisons in class HighArray and set it in the find method).

Compute and print the average number of comparisons to find a random item over 100 trials.

Print the average number of comparisons to find a random item in arrays with 100, 200, 300,...,1000 items. Analyze the trend.

iv. OrderedApp.java

- Insert 100 random items (generated with nextInt()) and find one of them chosen at random. Print the number of comparisons (add a counter of comparisons in class HighArray and set it in the find method).
- Compute and print the average number of comparisons to find a random item over 100 trials.
- Print the average number of comparisons to find a random item in arrays with 100, 200, 300,...,1000 items. Analyze the trend.
- Compare the complexity of linear (HighArrayApp.java) and binary (OrderedApp.java) search.