Alexander Kim

0. This loop in remove is supposed to move all the entries after theDirectory[index] back one, but when you test on a "full" array (size==theDirectory.length), it crashes. Why? How can you fix it?

```
        for (int i = index; i < size; i++)
            theDirectory[i] = theDirectory[i+1];
```

10

Graded by Brendon Hendricks

- Array out of bounds

for (int i = index+1 ; i < size ; i++) {

the Directory [i-1] = the Directory [i] ;

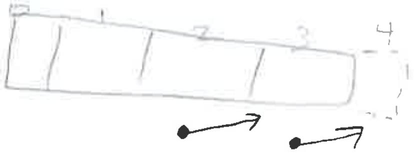
}

size -- ;

1. This loop from addOrChangeEntry is supposed to move entries forward to open up a space for a new entry at theDirectory[index]. Fill in the missing parts of the for loop. size has not been incremented yet.

for (int i = size - 1 ; i >=index ; i --)

```
     theDirectory[i+1] = theDirectory[i];
```



2. What is the O() of the following function?

```
       7 * log (n) + 2 * n - 11
               2
```

O(n)

3. What is the worst case O() running time of ArrayBasedPD.removeEntry? Is your answer for n=size or n=theDirectory.length?

O(n) n = size

4. What is the WORST case O() running time of ArrayBasedPD.removeEntry? What index?

O(n) index = size -1

5. What is the BEST case O() running time of SortedPD.removeEntry? What index?

O(log n)

index = size -1

6. Suppose a method has O(log(n)) running time. It takes 60ms (milliseconds) for n=1000. What is the constant? Indicate which log you are using.

$$60\text{ms} = C \cdot \log_{10}(1000)$$

$$60\text{ms} = 3C$$

$$C = 20$$

7. What is the estimated running time of the method in #6 for n=10000?

$$= 20 \cdot \log_{10} 10{,}000$$

$$= 20 \cdot 4$$

$$80\text{ms} =$$

8. Finish writing the averageTime method (below) for an implementation of Fib.

```
public abstract class Fib {
  /** The Fibonacci number generator 0, 1, 1, 2, 3, 5, ...
      @param n index
      @return nth Fibonacci number */
  public abstract double fib (int n);

  /** The order O() of the implementation.
      @param n index
      @return the function of n inside the O() */
  public abstract double O (int n);
}

  /** Determine the average time in MICROseconds it takes to calculate the
      n'th Fibonacci number.
      @param fib an object that extends the Fib class
      @param n the index of the Fibonacci number to calculate
      @param ncalls the number of times to do the calculation
      @return the average time in microseconds */
  public static double averageTime (Fib fib, int n, int ncalls) {
    // Get the current time in NANOseconds
    long start = System.nanoTime();

        for (int i = 0; i < ncalls; i++){
            fib.Fib(n);
        }
    long end = System.nanoTime();
    return(((end - start)/ncalls)/1000);
```

9. A method takes about 400 microseconds. How many times can you run it in 1 second?

run $\frac{1{,}000{,}0\cancel{0}\cancel{0}\text{ ms}}{4\cancel{0}\cancel{0}\text{ ms}}$ = 2500 times

$$\begin{array}{r} 2500 \\ 4\overline{)10000} \\ 8 \\ \hline 20 \\ 20 \\ \hline 00 \end{array}$$