Test 2 Practice Maps/Big-Oh: Compsci 201

Owen Astrachan

October 27, 2018

Name: ________________________________

NetID/Login: __________

Community standard acknowledgment (signature) ________________________________

<table>
<thead>
<tr>
<th>Problem</th>
<th>Value</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>NetID</td>
<td>1 pt.</td>
<td></td>
</tr>
<tr>
<td>Problem 1</td>
<td>12 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 2</td>
<td>20 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 3</td>
<td>12 pts.</td>
<td></td>
</tr>
<tr>
<td>Problem 4</td>
<td>12 pts.</td>
<td></td>
</tr>
<tr>
<td>TOTAL:</td>
<td>58 pts.</td>
<td></td>
</tr>
</tbody>
</table>

This test has 10 pages, be sure your test has them all. Write your NetID clearly on each page of this test (worth 1 point).

In writing code you do not need to worry about specifying the proper import statements. Don’t worry about getting function or method names exactly right. Assume that all libraries and packages we’ve discussed are imported in any code you write. You can write any helper methods you would like in solving the problems. You should show your work on any analysis questions.

You may consult your six (6) note sheets and no other resources. You may not use any computers, calculators, cell phones, or other human beings. Any note sheets must be turned in with your test.
PROBLEM 1 : (Origami, Original, Ostentatious (12 points))

Part A (4 points) Consider method foo below. Using big-Oh, what is the runtime of the call foo(n). Your answer should be in terms of n and you should justify/explain the answer accounting for all the code in method foo.

```java
public int foo(int n) {
    int sum = 0;
    for(int k=0; k < n; k += 2) {
        for(int j=1; j <= n; j++) {
            sum += j;
        }
        for(int j=0; j < k; j++) {
            sum += 2*j;
        }
    }
    return sum;
}
```

Part B (2 points)
Consider method stub below.

```java
public int stub(int n) {
    int sum = 0;
    for(int k=1; k <= n; k *= 2){
        sum += k;
    }
    return sum;
}
```

What is the exact value returned by stub(16), and by stub(742)
Part C (4 points) What is the big-Oh complexity of the runtime of the call $\text{stub}(n)$ in terms of $n$. Explain your answer.

Part D (2 points) What value is returned by the call $\text{stub}(n)$. Explain your answer which should be in terms of big-Oh and $n$, not an exact answer but an answer using big-Oh. (it may help to assume that $n$ is a power of 2, that’s fine).
As an example of how to think about some of the questions in this section, consider the method `stuff` below. The runtime complexity of this method is $O(n)$ and the value returned by the function is $O(n^2)$ for parameter $n$. As a concrete example, note than when $n = 100$ the loop executes 100 times doing an $O(1)$ operation each time. The value returned is $100 \times 100 = 100^2$. Note that even if the return statement was `return sum*2` that the value returned would still be $O(n^2)$.

```java
public int stuff(int n){
    int sum = 0;
    for(int k=0; k < n; k++){
        sum += n;
    }
    return sum;
}
```

In all these problems $n$ is a positive number. In each problem you should provide two big-Oh expressions: one for runtime and one for value returned. Briefly justify each answer you provide.

**Part A (4 points)**

What is the runtime complexity and the value returned by method `evaluate` below in terms of $n$? Use big-Oh and justify your answer briefly. Label the run-time and the value returned. Justify each.

```java
public int evaluate(int n){
    int sum = 0;
    for(int k=0; k < n; k++){
        sum += 1;
    }
    for(int k=0; k < n; k++){
        sum += 2;
    }
    for(int k=0; k < n; k++){
        sum += 3;
    }
    return sum;
}
```
Part B (4 points)
What is the runtime complexity and the value returned by method `calculate` below in terms of \( n \)? Use big-Oh and justify your answer briefly. For this Part B the runtime and the value returned have the same big-Oh expression.

```java
public int calculate(int n){
    int sum = 0;
    for(int k=0; k < n; k += 5){
        for(int j=0; j < n; j += 10){
            sum += 1;
        }
    }
    return sum;
}
```

Part C (4 points)
What is the runtime complexity and the value returned by method `value` below in terms of \( n \)? Use big-Oh and justify your answer briefly. You should have two answers, each justified.

```java
public int value(int n){
    int result = 1;
    for(int k=0; k < n; k++){
        result *= 2;
    }
    return result;
}
```
Part D (4 points)
What is the runtime complexity and the value returned by method `mathize` below in terms of \( n \)? Use big-Oh and justify your answer briefly. You should have two answers, each justified. If it helps, assume \( n \) is a power of two.

```java
public int mathize(int n){
    int sum = 0;
    int amount = n;
    while (n > 0){
        sum += amount;
        n = n/2;
    }
    return sum;
}
```

Part E (4 points)
Consider the method `stuff` from the beginning of this problem, reproduced below. Recall that the runtime complexity is \( O(n) \) and the value returned is \( O(n^2) \).

```java
public int stuff(int n){
    int sum = 0;
    for(int k=0; k < n; k++){
        sum += n;
    }
    return sum;
}
```

Give big-Oh expressions for both the runtime complexity and the value returned for each of the expressions below. Justify your answer briefly.

```java
int x = stuff(stuff(n)); // big-Oh for runtime and value returned
int y = stuff(stuff(n/4)); // big-Oh for runtime and value returned
```
PROBLEM 3: (Big Courses (12 points))

In this problem you’ll write code to find information in data about enrollment in courses at Duke. The format of the data is as follows where each String represents one student (shown by first name) and the courses that student is taking — separated from the name by a colon ‘:’ and with each course separated by a comma ‘,’ from other courses.

```java
String[] classes = {
    "owen:math216,compsci201,econ101,soc111",
    "fred:compsci201,econ101,evanth101,eos101",
    "mary:eos101,compsci92,music145,evanth101",
    "nancy:math216,phy141,chem201,compsci92",
    "fran:chem201,evanth101,soc111,psych101",
    "george:psych101,evanth101,chem201,pubpol165"
};
```

As an example of how to get information from this data, the method `enrollment` below returns the number of students enrolled in a specific course specified by parameter `course`. This output is generated by the code that follows using array `classes` as above.

```java
import java.util.*;
public class ACES {
    public int enrollment(String[] list, String course){
        int count = 0;
        for(String s : list) {
            String[] first = s.split(":");
            String[] courses = first[1].split",");
            if (Arrays.asList(courses).indexOf(course) >= 0){
                count++;
            }
        }
        return count;
    }
    public static void main(String[] args){
        String[] classes = /* not shown, see above */
        ACES m = new ACES();
        for(String s : new String[]{"compsci201","evanth101", "eos101", "chem201","art57"}){
            System.out.printf("%d\t%s\n",m.enrollment(classes, s),s);
        }
    }
}
```

Here’s the code generating this output. You’ll be asked questions about this code and you’ll be asked to write more code manipulating this data.
Part A (4 points)
Briefly explain in words the purpose of both calls to `split` and the purpose of the call to `indexOf` in the implementation of `enrollment` that’s shown.

Part B (8 points)
Write the method `classMap` whose parameter is a list of student data and which returns a `Map<String,List<String>>` in which keys are course names and the value associated with each key is a list of students enrolled in the course. For example, for the key `compsci201` the value would be the list `{"owen", "fred"}` and for the key `chem201` the key would be the list `{"nancy", "fran", "george"}`. You should return a `HashMap` in which the values are `ArrayList<String>` objects.

```java
import java.util.*;
public class ACES {

    public Map<String,List<String>> classMap(String[] list){

    }
}
```
In the Markov assignment the class EfficientMarkov required the use of a map instance variable, e.g.,

```java
Map<String, ArrayList<String>> myMap = new HashMap<>();
```

For an order-K Markov model each key in the map was a string of K-characters, called a K-gram. Each value was an `ArrayList` of single-character strings that followed the K-gram in the training text.

**Part A (4 points)**

Explain in at most three sentences why the use of the map in methods `setTraining` and `getFollows` made EfficientMarkov much faster than MarkovModel when the size $N$ of the training text is large and a large number $T$ of random characters are generated.

**Part B (8 points)**

Complete method `markovReverse` whose parameter is a map for an efficient order-K Markov model and that returns a reverse map in which keys are single-characters strings from the values of the parameter. The value corresponding to the key is a `Set<String>` of K-character strings that map to the key.

For example, consider the training text "abcabdabc" and an order-2 Markov model. The map created in EfficientMarkov contains five keys mapped as shown:

```
"ab" -> {"c", "d", "c"}
"bc" -> {"a", PSEUDO_EOS}
"bd" -> {"a"}
"da" -> {"b"}
"ca" -> {"b"}
```

If this map is stored in `myMap` then the call `markovReverse` should return a map that contains five keys mapped as shown. Note that the order of the Strings in each Set that's a value doesn't matter.

Do NOT include PSEUDO_EOS as a key in the map returned, see return below.

```
"b" -> {"da", "ca"}
"a" -> {"bd", "bc"}
"c" -> {"ab"}
"d" -> {"ab"}
```
Complete the method below. Assume that the constant `PSEUDO_EOS` is accessible in writing `markovReverse`, e.g., the method is in `MarkovModel`.

```java
public Map<String, Set<String>> markovReverse(Map<String, ArrayList<String>> map) {
```