Practice Test 1: CompSci 100

Michael Hewner

2/12/2012

Name: ____________________________

Netid (please print clearly): ________

Honor code acknowledgement (signature): ____________________________

<table>
<thead>
<tr>
<th>Question</th>
<th>Value</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 1: Algorithms 1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Question 2: Algorithms 2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Question 3: Algorithms 3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Question 4: Classes, Equals, HashCode, Comparable 1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Question 5: Classes, Equals, HashCode, Comparable 2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Question 6: Classes, Equals, HashCode, Comparable 3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Question 7: Big O</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Question 8: Recursion 1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Question 9: Recursion 2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td>40</td>
<td></td>
</tr>
</tbody>
</table>
Question 1: Algorithms 1 [5 Points]

Write a function getNumberDuplicated that takes an array of Strings (which may contain duplicates) and returns the number of (unique) strings that occur more than once.

Examples:

{"dog","cat","dog","dog","bird","bird"} returns 2 because dog and bird are duplicated
{"cat","cat"} returns 1
{} returns 0

```java
public int getNumberDuplicated(String[] inputs) {
    HashMap<String, Integer> found = new HashMap<String, Integer>();
    int numDuplicates = 0;
    for (String word : inputs) {
        if (!found.containsKey(word)) {
            found.put(word, 0);
        }
        int total = found.get(word);
        found.put(word, total + 1);
        if (found.get(word) > 1) {
            numDuplicates += 1;
        }
    }
    return numDuplicates;
}
```
Question 2: Algorithms 2 [5 Points]

Write a function `splitOnZeros` that takes a non-empty `ArrayList` of integers and returns a `ArrayList` of `ArrayLists` where the original list is broken into several smaller parts. Each place the original list contains a 0, the list should be split. The zero itself is removed - it does not become part of either side of the "split" list.

Examples:

```
[1,0,3,4] returns [[1],[3,4]]
[1,0,2,0,3,0,4] returns [[1],[2],[3],[4]]
[0,1,2] returns [[],[1,2]]
[4,5,6] returns [[4,5,6]]
```

```java
public ArrayList<ArrayList<Integer>> splitOnZeros(ArrayList<Integer> input) {
    ArrayList<Integer> list = new ArrayList<Integer>(input);
    int index = list.indexOf(0);
    ArrayList<ArrayList<Integer>> result = new ArrayList<ArrayList<Integer>>(0);
    while (index > -1) {
        ArrayList<Integer> left = new ArrayList<Integer>(0);
        for (int i = 0; i < index; i++)
            left.add(list.remove(0));
        result.add(left);
        list.remove(0); // To remove the 0 itself
        index = list.indexOf(0);
    }
    result.add(list);
    return result;
}
```
Question 3: Algorithms 3 [5 Points]

Imagine you have a HashMap that maps each student’s name to their advisor. Now you want to “reverse” the map. That is, you want the keys of the map to be advisors, and the values to be students. Of course, more than one student may have the same advisor, so each advisor will have a list of students. Write a function that takes in a HashMap of student to advisor and returns a HashMap of advisor to student.

Example:

myStudentToAdvisor = {"Sue": "Dr. Astrachan", "Lisa": "Dr. Rodger", "Steve": "Dr. Rodger"}

the result of reverseAdvisorMap(myStudentToAdvisor):
["Dr. Astrachan" = ["Sue"],
"Dr. Rodger" = ["Lisa", "Steve"]}

```java
public HashMap<String, ArrayList<String>> reverseAdvisorMap(HashMap<String, String> stuToAdvisor) {
    HashMap<String, ArrayList<String>> reversed = new HashMap<String, ArrayList<String>>() {
        for (String student : stuToAdvisor.keySet()) {
            String advisor = stuToAdvisor.get(student);
            if (!reversed.containsKey(advisor)) {
                reversed.put(advisor, new ArrayList<String>());
            }
            ArrayList<String> students = reversed.get(advisor);
            students.add(student);
        }
        return reversed;
    }
```
Question 4: Classes, Equals,HashCode, Comparable 1 [3 Points]

public class Foo { }

public interface Qqqq { }

public class Bar extends Foo implements Qqqq { }

public class Bang extends Bar { }

Circle whether the following lines of code would compile:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Bar var = new Foo();</td>
<td>This code would compile</td>
</tr>
<tr>
<td>(b)</td>
<td>Foo var = new Bar();</td>
<td>This code would compile</td>
</tr>
<tr>
<td>(c)</td>
<td>Qqqq var = new Bar();</td>
<td>This code would compile</td>
</tr>
<tr>
<td>(d)</td>
<td>Bar var = new Qqqq();</td>
<td>This code would compile</td>
</tr>
<tr>
<td>(e)</td>
<td>Foo var = new Bang();</td>
<td>This code would not compile</td>
</tr>
<tr>
<td>(f)</td>
<td>Foo var = new Foo(); System.out.println(var.hashCode());</td>
<td>This code would compile</td>
</tr>
</tbody>
</table>

Notes:

(a) Bar is a subclass of Foo, not the other way around.
(b) As previously mentioned, Bar is a subclass of Foo (i.e. Bar extends Foo).
(c) Bar implements the interface Qqqq.
(d) As stated above, Bar implements Qqqq. An interface cannot extend a class.
(e) Bang extends Bar, which in turn extends Foo.
(f) Every Object has a hashCode method by default.
public class IntComplexNumber() {
    private int myR, myI;
    public IntComplexNumber(int r, int i) {
        myR = r;
        myI = i;
    }
    public int hashCode() {
        return myR + 3*myI;
    }
    public boolean equals(Object o) {
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        IntComplexNumber other = (IntComplexNumber) obj;
        if (other.myR == myR && other.myI == myI) return true;
        return false
    }
}

Mike has used this IntComplexNumber class several times and he's always been satisfied with its performance. Then one day he notices that the performance of mikesMagicHashset is poor. mikesMagicLashset is initialized like this:

```
HashSet<IntComplexNumber> mikesMagicHashSet = new HashSet<IntComplexNumber>();
for(int i = 0; i < 10000; i++) {
    mikesMagicHashSet.add(new IntComplexNumber( 3*i, i));
}
```

What's causing Mike's poor performance?

The `hashCode` method is poorly written. The complex numbers Mike is creating take the form (0, 0), (-3, 1), (-6, 2), and so on. For all of these values, the `hashCode` is 0.
Question 6: Classes, Equals, HashCode, Comparable [5 Points]

Imagine you're writing a class to represent information about people, including their birthdays (day month). We want these people to be sorted by birthday, and if they were born on the same day to be sorted by last name (don't worry about firstname). Here's some example code that uses the PersonWithBirthday class you'll be building:

Arraylist<PersonWithBirthday> list = new Arraylist<PersonWithBirthday>();
list.add(new PersonWithBirthday("Barack", "Obama", 8, 4)); //born August 4
list.add(new PersonWithBirthday("Rogers", "Clemens", 8, 4)); //also born August 4
list.add(new PersonWithBirthday("Alan", "Turing", 6, 23)); //born June 23
Collections.sort(list);
//should be ordered by birthday so Alan Turing should be first,
// followed by Roger Clemens, followed by Barack Obama

Write the class PersonWithBirthday so the above code works:

```java
public class PersonWithBirthday implements Comparable<PersonWithBirthday> {
    private String myFirst, myLast;
    private int myBirthMonth, myBirthDay;

    public PersonWithBirthday(String first, String last, int month, int day) {
        myFirst = first;
        myLast = last;
        myBirthMonth = month;
        myBirthDay = day;
    }

    @Override
    public int compareTo(PersonWithBirthday other) {
        if (myBirthMonth != other.myBirthMonth) {
            return myBirthMonth - other.myBirthMonth;
        }
        if (myBirthDay != other.myBirthDay) {
            return myBirthDay - other.myBirthDay;
        }
        return myLast.compareTo(other.myLast);
    }
}
```
Question 7: Big O

For each of the following functions, given a specific definition for $n$, determine the function's runtime as a function of $n$. Express your answer using Big O notation as we've discussed in class, omitting coefficients and extra terms.

1. 
   public boolean example1(int[] nums) 
   { 
       for(int num : nums) { 
           if(num == 77) 
               return false; 
       } 
       return true; 
   }

   Runtime of example1, where $n$ is the number of elements in nums: $O(n)$

2. 
   public void example2(int input) 
   { 
       long sum = 0; 
       for(int i = 0; i < input; i++) { 
           for(int j = 0; j < input - i; j++) 
               //mysteriousOtherFunction is $O(1)$ 
               mysteriousOtherFunction(i,j); 
       } 
   }

   Runtime of example2, where $n$ is the size of input: $O(n^2)$

3. 
   public TreeSet<Integer> example3(int num) 
   { 
       TreeSet<Integer> result = new TreeSet<Integer>(); 
       for(int i = 0; i < input; i++) 
           result.add(i); 
   }

   return result;

   Runtime of example3, where $n$ is the size of num: $O(n \log n)$
4.

```java
public String[] example4(ArrayList<String> data) {
    String[] output = new String[3*data.size()];
    int current = 0;
    while(!data.isEmpty()) {
        // removes last element
        String element = data.remove(data.size() - 1);
        for(int i = 0; i < 3; i++) {
            output[current] = element;
            current++;
        }
    }
    return output;
}
```

Runtime of example4, where n is the number of elements in data: \( \Theta(n) \)

5.

```java
public void example5(int input) {
    int p = input * input * input;
    for(int i = 0; i < p; i++) {
        System.out.println("Hello!");
    }
    for(int i = 0; i < p; i++) {
        System.out.println("Goodbye!");
    }
}
```

Runtime of example5, where n is the size of input: \( \Theta(n^3) \)
Question 8: Recursion 1

You must solve this question using recursion. Your function should contain no loops, and use no variables outside this function.

Write the function numberQuestionMarks that takes 2 parameters, a string to replace question marks in and a number to start with. Every question mark in the original string should be replaced with a number, starting with numberToStartWith and adding 1 for each subsequence replacement.

Examples:

numberQuestionMarks("hello ? my name is ?", 1) returns "hello 1 my name is 2"
numberQuestionMarks("???", 3) returns "345"
numberQuestionMarks("mike", 3) returns "mike"

```java
public String numberQuestionMarks(String stringToReplace, int startWith) {
    if (stringToReplace == null || stringToReplace.length() == 0) {
        return stringToReplace;
    }
    char first = stringToReplace.charAt(0);
    String rest = stringToReplace.substring(1);
    if (first == '?') {
        return startWith + numberQuestionMarks(rest, startWith + 1);
    }
    return first + numberQuestionMarks(rest, startWith);
}
```
Question 9: Recursion 2

You must solve this question using recursion. Your function should contain no loops, and use no variables outside this function.

Write the function pow that two numbers and raises the given number to the given power. You can assume that neither of the numbers are negative. Solve this with basic multiplication and addition - do not use the functions in Java’s Math library.

Examples:

double pow(double base, int exponent) {
    if (exponent == 0) return 1;
    return base * pow(base, exponent - 1);