

Points off	1	2	3	4	5	Total off	Net Score

CS 314 – Midterm 2 – Fall 2012

Your Name _____

Your UTEID _____

Circle your TA's name: John Zihao

Instructions:

1. There are 5 questions on this test.
2. You have 2 hours to complete the test.
3. You may not use a calculator or any other electronic devices while taking the test.
4. When writing a method assume the preconditions of the method are met.
5. When writing a method you may add helper methods if you wish.
6. When answering coding questions ensure you follow the restrictions of the question.
7. When you complete the test show the proctor your UTID. Give them the test and any scratch paper. Please leave the room quietly.

1. (2 points each, 30 points total) Short answer. Place your answers on the attached answer sheet.
 - a. If a question contains a syntax error or other compile error, answer "Compile error".
 - b. If a question would result in a runtime error or exception answer "Runtime error".
 - c. If a question results in an infinite loop answer "Infinite loop".
 - d. Recall when asked for Big O your answer should be the most restrictive correct Big O function. For example Selection Sort has an average case Big O of $O(N^2)$, but per the formal definition of Big O it is correct to say Selection Sort also has a Big O of $O(N^3)$ or $O(N^4)$. I want the most restrictive, correct Big O function. (Closest without going under.)

- A. In class we used maps to determine the frequency of words in a file. The words (represented as `Strings`) were the keys and the value for each key was its frequency (represented as an `Integer`).

When we used a `TreeMap` and printed out the keys, describe the order in which the keys appeared.

B. Consider the following method:

```
public static void count(Scanner sc, Map<String, Integer> map) {
    while(sc.hasNext()) {
        String word = sc.next();
        if(!map.containsKey(word))
            map.put(word, 1);
        else
            map.put(word, map.get(word) + 1);
    }
}
```

Assume the parameter `map` is initially empty. Assume `sc` is connected to a source with 100,000,000 total words and 500,000 distinct words. Will method `count` run faster with a `TreeMap` or `HashMap` or will the time be about the same?

C. Consider the following partial class definition:

```
public abstract class Property {
    private String name;

    public Property(String n) {name = n;}

    public abstract int getRent();

    // other methods and constructors not shown
}
```

The following line of code causes a compile error:

```
Property p = new Property("Boardwalk");
```

Objects of an abstract type cannot be instantiated (created). Why then would an abstract class have constructors?

D. What is the order (Big O) of method `d`? Assume `list` is a Java `LinkedList`. $N = list.size()$

```
public List<Integer> d(List<Integer> list) {
    List<Integer> result = new LinkedList<Integer>();
    while(list.size() > 0)
        result.add(list.remove(0));
    return result;
}
```

E. What is the order (Big O) of method `d` if the following changes are made?

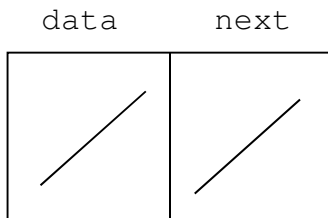
The parameter `list` is a Java `ArrayList` and the line

```
List<Integer> result = new LinkedList<Integer>();
```

is altered to

```
List<Integer> result = new ArrayList<Integer>();
```

F. Draw the variables, references, and objects that exist after the following code executes. Draw node objects as shown below and boxes for variables. (The example has both instance variables set to `null`. The example does not show any of the variables that actually refer to the `Node` object. You must show all variables and their references in your drawing.) Use arrows to show references and a forward slash to indicate variables that store `null`. Assume the `Node` class is the one from our singly linked list examples in class.



```
Node<Object> n1 = new Node<Object>("A", null); // data, next
Node<Object> n2 = new Node<Object>(n1, n1);
n2.getNext().setNext(new Node<Object>(12, n1));
```

G. What is returned by the method call `g(1023102)`?

```
public static int g(int n) {
    if(n == 0)
        return n;
    else
        return n % 10 + g(n / 10);
}
```

H. What is returned by the method call `h(21)`?

```
public static int h(int n) {
    if(n == 0)
        return 3;
    return 1 + h(n - 2);
}
```

I. What is returned by the method call `i("abb")`?

```
public static String i(String s) {
    if(s.length() > 10)
        return s;
    else
        return i(s + s);
}
```

J. What is returned by the method call `j(8)`?

```
public static int j(int n) {
    if(n <= 3)
        return n;
    return 1 + j(n - 2) + j(n - 1);
}
```

K. You have an array with 32,000,000 distinct elements in random order.

You have to search the array a number of times to determine if a given element is present or not.

What number of searches will make it worthwhile (result in overall less work) to sort the array before doing the searches? Assume $\log_2 32,000,000 = 25$.

Show your work.

L. An array with 1,000,000 distinct elements in random order takes 10 seconds to sort using a method that implements the mergesort algorithm. What is the expected time to sort an array with 4,000,000 distinct elements in random order using the same method?

M. Consider the following time data for a method that uses an unknown sorting algorithm:

- time to sort array with 1,000,000 elements all equal to the same value: 1 sec
- time to sort array with 2,000,000 elements all equal to the same value: 2 sec
- time to sort an array with 100,000 distinct elements in random order: 5 seconds
- time to sort an array with 200,000 distinct elements in random order: 20 seconds

Of the sorting algorithms we studied, which does the sorting method most likely use?

N. What does the following expression evaluate to? (Single integer for answer)

$12 \ 5 \ 7 \ - \ 5 \ 10 \ + \ * \ +$

O. What is output by the following code?

```
// Java Queue interface and Java LinkedList
Queue<Integer> queue = new LinkedList<Integer>();
for(int i = 3; i < 8; i++)
    queue.add(i * 2); // enqueue operation

for(int i = 0; i < queue.size(); i++)
    System.out.print(queue.remove()); // dequeue operation
```

Extra Credit (1 point) If the login id for an account is "Student" then what would the password be? (Put you answer on the answer sheet.)

2. Maps and Lists (18 points total) Movies often make references to other movies. For example *The Avengers* references *The Wizard of Oz*. (flying monkeys) The Internet Movie Database (IMDB) lists references for many movies. For this question you are given a map of movies and the movies each references. The keys of the map are movies and the associated values are lists of movies the movie represented by the key references.

Write a method that determines which movies are referenced the most. Return an `ArrayList` of the movies that are referenced by the most other movies.

Movies are represented as `Strings` with the movie title and release year. For example `"The Dark Knight Rises - 2012"` or `"Monty Python and the Holy Grail - 1975"`.

The keys in the `Map` are `Strings` (a movie). The associated values are `ArrayLists` of `Strings`. (a list of movies referenced by the key)

Consider this small example.

Key - Movie	Value - list of movies key references
"Superman - 1978"	["Peter Pan - 1953", "Cinderella - 1950", "Star Wars - 1977"]
"Star Wars - 1977"	["Metropolis - 1927", "The Hidden Fortress - 1958"]
"The Hidden Fortress - 1958"	["Battleship Potemkin - 1925"]
"The Avengers - 2012"	["The Wizard of Oz - 1939", "Point Break - 1991"]
"What Dreams May Come - 1998"	["Mary Poppins - 1964", "Peter Pan - 1953"]
"Bad Santa - 2003"	["Office Space - 1999", "The Wizard of Oz - 1939"]

In this case the method would return an `ArrayList` with the following elements. (Each is referenced by two movies.)

```
["Peter Pan - 1953", "The Wizard of Oz - 1939"]
```

The elements in the `ArrayList` the method returns may be in any order. There shall be no duplicates in the list.

Recall these methods from the `Map` interface:

- `Set<K> keySet ()` - Returns a `Set` view of the keys contained in this map.
- `V get (Object key)` - Returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.
- `boolean containsKey (Object key)` - Returns true if this map contains a mapping for the specified key.
- `V put (K key, V value)` - Associates the specified value with the specified key in this map

Recall this method from the `Set` interface:

- `Iterator<E> iterator ()` Returns an iterator over the elements in this set.

You may use Maps, ArrayLists, Iterators (implicit or explicit), Integers, and Sets to solve this problem, but no other classes from the Java standard library. Your method must be as efficient as possible given the constraints of the question.

Complete the following method:

```
/*   pre: movieRefs != null
      post: return an ArrayList of movies that are referenced the most.
           movieRefs is not altered as a result of this method call.
*/
public ArrayList<String> getMoviesWithMaxReferences
    (Map<String, ArrayList<String>> movieRefs) {
```

3. Stacks and Queue. (17 points) Implement a method, `getMissingGroupingSymbols`, that determines and returns what the missing grouping symbols are given a `String` of grouping symbols. The only characters in the `String` will be `'('`, `)'`, `'['`, and `']'`. The method returns a `String` with the characters necessary to complete the `String` so that all symbols match correctly. `'('` only match `)'` and `'['` only match `']'`. Grouping symbols may not overlap. In other words the grouping symbols must be correctly nested. If it is not possible to complete the grouping symbols or the grouping symbols present are not nested correctly, the method returns `null`.

Examples:

- `"["` returns `"]"`
- `"[()]"` returns `null` -- symbols nested incorrectly
- `"([()]"` returns `"])"`
- `"([])([]"` returns `"])"`
- `"(())[()]"` returns `""`
- `"(["` returns `null` -- not possible to complete
- `""` returns `""`
- `"[()]"` returns `null` -- symbols nested incorrectly
- `")"` returns `null` -- symbols nested incorrectly

You must solve the problem with a single `Stack<Character>`.

Assume `Stack` class has the following instance methods: `push`, `pop`, `top`, and `isEmpty`. `pop` returns the element being removed.

Your method must be as efficient as possible given the constraints of the question.

```
/*
pre: str != null, all elements of str are either '(', ')', '[', or ']'
post: return a String with the symbols necessary to complete str or
null if the symbols in str are not nested correctly or if it is not
possible to complete str.
*/
public String getMissingGroupingSymbols(String str) {
```

Complete this method on the next page.

Complete this method on the next page.

Complete this method on the next page.

Complete this method on the next page.

```
/*
pre: str != null, all elements of str are either '(', ')', '[', or ']'
```



```
post: return a String with the symbols necessary to complete str or
null if the symbols in str are not nested correctly or if it is not
possible to complete str.
*/
public String getMissingGroupingSymbols(String str) {
```

4. Linked Lists (17 points total) Complete a `removeRepeats` method for a `LinkedList` class. The method removes consecutive repeated elements from the calling `LinkedList` object.

- You may not use any other methods in the `LinkedList` class unless you implement them yourself as a part of this question
- Your solution must be $O(1)$ space, meaning no matter how many elements are in the `LinkedList`, your solution always uses the same amount of space. In other words you can't use an auxiliary array or `List`.
- The `LinkedList` class uses singly linked nodes.
- When the list is empty, `first` is set to `null`.
- None of the data in the list equals `null`.
- The list has a reference to the first node in the list and no other instance variables. **In other words the list doesn't track the size of the list or the last node.**
- If the list is not empty the last node has its next reference set to `null`.
- You may use the `Node` class and the `Object` `equals` method. You may not use any other Java classes.
- Your method shall be as efficient as possible give the constraints of the question.

```
public class LinkedList<E> {  
    private Node<E> first; // first node in chain of nodes  
  
}
```

Here are the methods from the `Node` class.

```
public class Node<E> {  
    public Node(E item, Node<E> next)  
    public E getData()  
    public Node<E> getNext()  
    public void setData(E item)  
    public void setNext(Node<E> next)  
}
```

Examples:

```
[] .removeRepeats() -> resulting list []  
[A, B, A, B, A, B].removeRepeats() -> resulting list [A, B, A, B, A, B]  
[A, A, A, B, B, A].removeRepeats() -> resulting list [A, B, A]  
[A, A, A, A].removeRepeats() -> resulting list [A]  
[A, A, B, A, A, A, D].removeRepeats() -> resulting list [A, B, A, D]  
[A, B, C, B, A].removeRepeats() -> resulting list [A, B, C, B, A]  
[A, A, B, B, A, A, C, C, C, A].removeRepeats()  
-> resulting list [A, B, A, C, A]
```

Complete the following method:

```
// pre: none  
// post: repeated consecutive elements removed from this LinkedList  
public void removeRepeats() {
```

5. Recursive Backtracking (18 points). In the game Boggle the board consists of a square grid of letters. The goal of the game is to find words on the board by connecting letters while following certain rules.

C	T	E	A
U	O	A	G
P	M	M	R
T	S	A	W

The rules for forming words are:

- a given cell may only be used once per word
- given the current letter in the grid, the next letter in a word must be in one of the eight surrounding cells
- paths are not allowed to wrap around the edges of the board. (For example it is not allowed to move from the P on the left edge to the R on the right edge)

For example given the board above, the word COMPUTE can be formed starting at (0, 0) as shown below:

C	T	E	A
U	O	A	G
P	M	M	R
T	S	A	W

Complete an instance method in a `BoggleBoard` class that determines if a word can be formed from a given starting location.

Your method shall use recursive backtracking and be as efficient as possible given the constraints of the question.

The `BoggleBoard` class contains a 2d array of `chars` that represents the board. All elements in the 2d array are upper case, English letters, A - Z. The 2d array is not altered when the method is finished.

You may use the `String` class, native arrays, and the `BoggleBoard` `inbounds` method.

Recall you can add helper methods if you wish.

```
public class BoggleBoard {
```

```
    private char[][] theBoard;
```

```
    // returns true if row and col inbounds on the board.
    public boolean inbounds(int row, int col)
```

Complete the following method.

```
/* pre: word != null, word.length() > 0, all elements of word are
upper case English letters, A - Z.
If called from client code, inbounds(row, col)

post: return true if word can be formed on the Boggle board start
at the cell specified by row, col. If it cannot be formed return
false. The board is not altered when this method is finished.

public boolean isPresent(String word, int row, int col) {
```


Question 1 answer Sheet.

Name _____

A. _____

B. _____

C. _____

D. _____

E. _____

F. _____

G. _____

H. _____

I. _____

J. _____

K. _____

L. _____

M. _____

N. _____

O. _____

Extra Credit: _____

