Computer Science 205
Review Sheet for Exam #1
Monday, September 26th

Exam #1 will cover our class notes and handouts since the start of the semester, our first four labs, and our first programming assignment. The main topics to be covered are class methods & instance methods, multidimensional arrays, BitSets, StringBuffers, sorting, and run-time.

Be sure you understand all of the code presented in the class handouts and that you are comfortable writing the code for the labs and your first program. Also, be sure you can do all of the questions on your sorting and searching practice worksheet and the run-time worksheet.

Review of OOP

- Differences between class methods, instance methods, class variables, instance variables
- Differences between `public` and `private` methods; When can they be accessed? What happens if this keyword is removed?
- Defining default (no-args) and explicit constructors; Working with `this`
- Be comfortable with the `CalendarDate` and `Clock` classes we discussed; Can you write a tester program using both?; Are they mutable? Why?

Arrays & Strings

- Definitions of data structures and algorithms; What is the Java API? What is an ADT?
- Using the `String` ADT; Be comfortable using the `indexOf`, `substring`, `compareTo`, and `equals` operations with string objects
- Advantages and disadvantages of arrays and arraylists; What is the difference between the size and capacity of an arraylist?
- Using the `ArrayList` ADT; What do `get` and `set` methods do?
- Declaring an array variable of any dimension with or without initial assignment
- Filling up, printing out, and summing up a multi-dimensional array
- Writing the code to determine the maximum or minimum value in an array
- Passing multi-dimensional arrays as parameters
- Be able to write and call a method which utilizes an array as a parameter
- How multi-dimensional arrays are actually stored in main memory; How are column-major order and row-major order different?
- Understand the code used for resizing an array that has reached its defined capacity
- Be able to use the `Object` data type; Why is it "our first taste of inheritance"?

BitSet and StringBuffer ADTs

- Be able to give an example of when you could use a bitset or string buffer
- Be able to declare a BitSet object and use the `set`, `clear`, and `get` methods; How would you print out a BitSet?; Why are BitSets used? What happens when you access a BitSet element at a position larger than the current capacity? (Ans: It doesn't crash; it automatically expands to that capacity.)

Sorting and Searching Arrays

- Be familiar with each of the algorithms for sorting and searching Be able to work out by hand how each one sorts a group of numbers. Don't worry about having to write the code for these algorithms.
  - Bubble Sort
  - Selection Sort
  - Linear (Sequential) Search
  - Binary Search
- Be able to show what an unsorted array looks like after each pass through the outer loop of a selection sort.
- Be able to determine the number of checks it takes to find a particular item using a linear or binary search. That is, the number of slots whose contents are accessed in looking for an item.

Run-Time and the *Big-O* Notation

- Be able to give the run-times using *Big-O* Notation for the four sorting and searching algorithms listed above
- Know how the different run-times compare to each other in terms of what happens when the number of items being processed increases; How would the graphs of linear, quadratic, cubic, logarithmic, and exponential run-times compare to one another?
- Given a loop be able to express the run-time of that loop using the *Big-O* Notation

Computer Science 205 Exam #1

Answer Sheet

Fall 2016

Monday, September 26th

100 points

Name : Yu-Ching Ho

46/100

1. Circle your answer.

(a)	valid	invalid
(b)	valid	invalid
(c)	valid	invalid
(d)	valid	invalid
(e)	valid	invalid
(f)	valid	invalid
(g)	valid	invalid
(h)	valid	invalid
(i)	valid	invalid
(j)	valid	invalid

-5

2. (a)

~~StringBuffer addString = new StringBuffer(set.home());~~

```
StringBuffer addString = new StringBuffer(names[1].set("home"));
```

+=

-1

(b)

~~System.out.println(ssNum.indexOf~~

```
String ssNum = "344-86-7609";
~~System.out.println(ssNum.substring(5));~~
String s;
if(ssNum.indexOf("8", "6") != null){
    String s = ssNum.substring(1);
}
System.out.println(s);
```

86-7609 +1

-2

-8

3.

```
int[][] myCard = new int[5][5]
int odd = 0;

while (myCard != myCard[5][5])
    if (myCard[i][j] % 2) == 0 {
        myCard.set[i][j] = null;
        if (myCard[i][j] != null) {
            odd++;
        }
    }
    else
        break;
}
System.out.println("Number of odds: " + odd);
```

-1

4. (a)

0 00 00
0003

-2 3 1

-2 (b) reset(), increment(), total(), toString()

-1 (c) Mutable why?

5. ArrayLists

(a)

```
int[] ArrayList <Integer> = new int [9];
int a = 55;
ArrayList.add(a);
```

-1

(b)

```
int last = ArrayList[9];
int first = ArrayList[0];
```

```
int sum = 0;
int last = ArrayList.get[9];
int first = ArrayList.get[0];
sum = last + first;
System.out.println(sum);
```

(c) New size = 40
New capacity = 21

-6

6. private static key {

-8

7. (a) int A[] = {6, 0, -5, 2};
int A[] = {0, -5, 2, 6};
-5 0 6 2

-4

middle = $\frac{-5+6}{2}$ = 1

middle = $\frac{-5+0}{2}$ = 0

-5, 0, 2, 6 ← sorted array

(b) 2 checks show f, l, m

-2

8. Run Time Short Answer

-2 (a) O(n) n 2

(b) checks = $\log_2 1000$ checks = $\frac{1000}{2}$ = 500

(c) checks = $\frac{1000}{\log_{1000}}$ = 10

9. *Big-O* Notation

(a) ~~$n^2 \times n^2 \times n^2 = n^6 \Rightarrow O(n^3)$~~ $2n \times 2n \times 2n = 8n^3 \Rightarrow O(8n^3)$ ✓

(b) ~~$O(2n)$~~ $2n + 2n = 4n \Rightarrow O(4n)$ ✓

(c) $O(n \log_2)$ ✓

-16

10. (a)

-20

(b) Example of a valid call from the main method :

-4

-2