CS 206 MIDTERM EXAM

Fall 2019

NAME:

---

1. You have 80 minutes to complete this exam. All answers should be written on this exam (you may use the back of the pages).
2. This is a closed book exam. Use nothing other than pen/pencil and paper.
3. The exam contains 9 pages including this instruction sheet. Make sure you have all the pages. Each question's point value is next to its number.
4. In order to be eligible for as much partial credit as possible, show all of your work for each problem, and clearly indicate your answers. Credit cannot be given for illegible answers.
5. Java code provided in test questions is correct; it has no syntax errors.
6. Java code that you write should be as close to correct (runnable) as possible. Small syntax errors will not cost you points, but code that is unclear will.
7. Be sure to appropriately comment any code that you write and choose good variable names.
8. If you finish early, wait patiently for me to appear. If you are waiting, do not turn on any electronics, open books, etc. Please be as quiet as possible.
9. I will poke in periodically during the exam. If you have a question, move on and ask me when I appear. If you cannot move on, I will be in my office (Park 204).
10. Be sure to read the entire exam before answering any questions, so you have time to think about the potentially tricky problems.
11. If you need extra paper there is some at the front of the room.
12. If you used extra paper or took apart you exam, use the stapler at the front of the room
13. Sign, in ink, on the bottom of this page to indicate that you will abide by the Honor Code in taking this examination. Also write your name on the top of this page

Problem 1: Reading Java — 16 points (4 parts, 4 points each)

Write the output of the main function of each of the programs

Part A:

```
public class PartA {
    public static void main(String[] args)
    {
     Integer[] arr1= new Integer[2];
     arr1[0]=5;
     arr1[1]=10;
     Integer[] arr2 = arr1;
     arr2[1]=200;
     System.out.println(arr1[1] + " " + arr2[1]);
    }}
```

Part B:

```
public class Overloaded {
    public Overloaded(int i) {
     System.out.println(i);
    }
    public void oloader() {
     System.out.println(1);
    }
    public void oloader(int v) {
     System.out.println(10);
    }
    public void oloader(int v, int w) {
     System.out.println(100);
    }
    public static void main(String[] args)
    {
     Overloaded oo = new Overloaded(0);
     oo.oloader(100000);
     (new Overloaded(2)).oloader();
    }}
```

Part C:

```
public class SNums {
    private int[] nums;
    public SNums(int nNums) {
        nums = new int[nNums];
        for (int i = 0; i < nNums; i++) nums[i] = nNums*i;
    }
    public String toString() {
        return "(" + nums[0] + ", " + nums[nums.length/2] + ", "
+ nums[nums.length-1] + ")";
    }
    public static void main(String[] args) {
        SNums nums = new SNums(10);
        System.out.println(nums);
    }}
```

Part D:

```
import java.util.ArrayList;
public class ALFunc {
    private ArrayList<Integer> nums;
    public ALFunc () {
        this.nums = new ArrayList<Integer>();
        this.nums.add(0); this.nums.add(100);
    }
    public void f() {
        for (int i = 0; i < this.nums.size(); i++) {
            this.nums.set(i, this.nums.get(i) + 1); }
    }
    public String toString() {
        String s= "";
        for (int i=0;i<nums.size();i++)
            s = s + nums.get(i) + " ";
     return s;
    }
    public static void main(String[] args) {
     ALFunc fun = new ALFunc();
     fun.f();
     System.out.println(fun);
    } }
```

Problem 2: Arrays and ArrayLists — 20 points

Write a method that takes as input two arrays of Strings and copies only those strings that occur at the same index in both arrays to a new ArrayList, which is returned.

Problem 3: Complexity — 16 points (4 parts, 4 points each)

For each method below indicate its big-O time complexity in terms of n, where n is equal to the size the input array ii

```
int part1(int[] ii) {
    int j=1;
    for (int i=0; i<500; i++)
       j = j+ i;
    return j;
}


int part2(int[] ii) {
    int j=1;
    for (int i=0; i<ii.length; i++)
        j *= i;
    return j;
}


int part3(int[] ii) {
    int j=1;
    for (int i=ii.length; i<=1; i=i/2)
        j=j+i;
    return j;
}


int part4(int[] ii) {
    int j=1;
    for (int k=0; k<ii.length; k++)
        for (int kk=0; kk<ii.length; kk++)
            for (int kkk=0; kkk<ii.length; kkk++)
                j += 1;
    return j;
}
```