Name: __________________________________________

NetID: ________

Statement of integrity: I did not, and will not, violate the rules of academic integrity on this exam.

__________________________________________
(Signature)

Circle your lecture time: 9:05 or 11:15

Q1: ______ 15pts ____
Q2: ______ 20pts ____
Q3: ______ 30pts ____
Q4: ______ 35pts ____
Total: ______ 100pts ____

Instructions:

• This is a 90-minute, closed-book exam; no calculators are allowed.
• There are 4 questions worth a total of 100 points ⇒
  A rough time budget: Try not to spend more than 18 minutes on a 20-point question.
• Raise your hand if you have any questions.
• Use the backs of pages or ask for additional sheets of paper as necessary.
• Clarity, conciseness, and style count for credit.
• If you supply multiple answers, we will grade only one.
• Use only Java code. No credit for code written in other programming languages.
• Do not use switch, break, or System.exit statements.

You may find the following methods useful:

Math class: pow, random, floor
String class: equals
System class: print, println
JLiveRead class: readInt, readDouble
Question 1: (15 points)

Part (a): (7 points)
A 1-d array has the values \{19, 11, 3, -2, 1, 30\}. Suppose we will sort the array in ascending order using the selection sort algorithm. The first line below shows the original state of the array. Write on each subsequent line the values in the array after one iteration of the selection sort algorithm (an “iteration” is one pass through the outer loop of the algorithm).

| Original state: | 19 | 11 | 3 | -2 | 1 | 30 |
| After first iteration: |   |   |   |   |   |   |
| After second iteration: |   |   |   |   |   |   |
| After third iteration: |   |   |   |   |   |   |

Part (b): (8 points)

Complete the code fragment below to find the first occurrence of a target integer value in an array of integers. Assume \( x \), a 1-d array of int values, has been created.

```java
int target = JLiveRead.readInt();  // Value to look for in the GIVEN int array x

// Find and print the location of the first occurrence of target in array x.
int k = 0;
while ( ___________________________________________________________ )
    k++;
if ( ______________________________________________________________ )
    System.out.println("Target found at position " + k);
else
    System.out.println("Target not found");
```
Question 2: (20 points)

Complete method `charsToInt` to convert an array of `char` to the integer value that the array represents. The method returns the integer value. For example, the `char` array with the characters ‘0’, ‘2’, ‘0’, ‘4’ represents the integer value 204. You may use pre-defined methods from the `Math` class only. (But you don’t have to use any pre-defined methods at all!)

*Hint:* Remember that we use base 10 numbers. For example, the integer value 2014 can be obtained by the following calculation: 4 + 1*10 + 0*100 + 2*1000.

```java
/** = Integer value represented by char array digits. Assume array digits contains *
* only digit characters (no letters, blanks, positive or negative signs, etc.) *
*/
public static int charsToInt(char[] digits) {
    int value; //the value that array digits represents.
    return value;
}
```
Question 3: (30 points)

Complete classes BadCoin and Q3 below to simulate the tossing of a weighted coin. Toss the coin 1000 times. Count the number of times that tails immediately follows heads. For example, in the following sequence of outcomes (heads or tails), tails immediately follows heads three times:

| heads, heads, heads, tails, heads, heads, tails, tails, heads, heads, heads, tails |

```java
/** A weighted Coin where heads (0) shows up three times as often as tails (1) */
class BadCoin {
    private int face;  //face-up: 0 is heads; 1 is tails
    /** Constructor: face has random value */
    public BadCoin() { roll(); }
    /** Assign a random value to face given the weighted property of Bad coin. */
    public void roll() {
        // Insert roll logic here
    }
    /** = Get the face shown by this BadCoin. */
    public int getFace() { return face; }
}

public class Q3 {
    public static void main(String[] args) {
        //Declare and instantiate a 1-d array toss to store the sequence of outcomes.
        int count = 0;  //no. of times tails immediately follows heads so far
        //Simulate the tossing of a BadCoin. toss[k] has the value 0 for heads or 1
        //for tails. Count the number of times that tails immediately follows heads.
        // Insert simulation logic here
    }
}
```
Question 4: (35 points)

Complete class CDOrganizer below. Class CDOrganizer is a client of class Song, which is given (you don’t write code for class Song). Class CDOrganizer keeps track of the Songs that one wants to write to a blank CD, the length of the Songs, as well as the remaining time left on the CD (so that you don’t try to write more than a CD can hold).

Class Song below shows only the variable declarations and the method headers. Assume class Song is implemented correctly.

/* Class Song */
class Song {
    private String artist;    //artist of a Song
    private String title;     //title of a Song
    private double minutes;   //minutes of a Song

    /** Constructor: assign values to fields artist, title, and minutes */
    public Song(String artist, String title, double minutes) {...}

    /** Getters */
    public String getArtist() { ... }
    public String getTitle() { ... }
    public double getMinutes() { ... }
}

Class CDOrganizer has the following variables and methods:

- Class constant MAXsongs has the value 30, representing the maximum number of songs on the CD
- Instance variable minutes: remaining minutes left on the CD (type double)
- Instance variable songs: 1-d array of the Songs on the CD. Array length is MAXsongs.
- Instance variable time: 2-d array of int with dimension MAXsongs-by-2. Each row stores the time of a Song such that the first cell is the number of minutes and the second cell is the number of seconds. For example, if the first Song on the CD is 3.5 minutes long, then the first row of array time has the values 3, 30. You may round or truncate the value for seconds as you wish.
- A constructor that has one input parameter specifying the available minutes on a blank CD. It also instantiates arrays songs and time.
- Instance method addSong(Song s) adds Song s to the next available spot in the songs array, update array time, and adjusts the amount of free time remaining on the disc. Assume there is no time gap between Songs.

Do not define any other instance (or class) variables or methods.

Read the incomplete class below before you start writing code! Follow the specifications above and in the comments. You must use the variable and parameter names and types as specified above.

public class CDOrganizer {

    public static final int MAXsongs= 30; //Maximum number of songs on the CD
    private double minutes;               //Minutes remaining on the CD
    private _________________ songs;     //1-d array of Songs on the CD
    private _________________ time;      //2-d array of time of the Songs on the CD

    //Class CDOrganizer continues on next page
// Class CDOrganizer, continued (Question 4, continued)

/** Constructor: CD has m free minutes. Instantiate arrays songs and time to * hold the maximum number of Songs allowed. */
public CDOrganizer(double m) {

/** Instance method addSong adds Song s to the next available spot in the songs * array and updates the field minutes and the array time. If another Song of * the same title is already on the CD, if there are already MAXsongs songs on * the CD, or if the CD does not have enough remaining time to include Song s, * then the method displays the message “ERROR!” and does not add Song s. */
public void addSong(Song s) {

}