The midterm exam for Sections 2,3,4 is Monday October 29.

Topics covered: Chapters 1 through 6 except Advanced Topics.

General instructions: write all your answers in the exam booklet provided. All questions will ask you to write methods, either static (class) methods, or instance (non-static) methods that access instance variables of a class.

None of the methods you write should read from the console or write to the output screen (in particular, there should be no calls to System.out in your solutions). None of the questions will involve applets. Read the questions carefully.

For questions involving strings, you may find the following methods useful:

- length (returns integer): if \texttt{str} is "hello", \texttt{str.length()} is 5
- charAt (returns char): if \texttt{str} is "hello", \texttt{str.charAt(1)} is 'e'
- subString (returns String): if \texttt{str} is "hello", \texttt{str.substring(1,3)} is "el"

You can expect about 4 questions on the exam. The following questions would be typical.

1. The Fibonacci numbers are the numbers 0, 1, 1, 2, 3, 5, 8, 13, 21, etc. (each number is obtained by adding the previous two together). Write a static method

   \begin{verbatim}
   public static int firstFibPast(int n)
   \end{verbatim}

   that returns the first Fibonacci number that is \textit{bigger than or equal to} the parameter \texttt{n}. For example, if \texttt{n} has the value 10, the method would return 13. If \texttt{n} has the value 0, the method would return 0.

2. Write a static method

   \begin{verbatim}
   public static int longestBlank(String \texttt{str})
   \end{verbatim}

   that returns the \textit{maximum number of consecutive blanks} in the String parameter \texttt{str}. For example, if the string is "abc def geh ijk", the method would return 2. If there are no blanks in the string, it returns 0.

Continued on Next Page
3. Write a static method

   public static int toInteger(String str)

   that converts the string given by the parameter str to a corresponding integer. For example, if str is the string "1234", the method returns the integer 1234. If the string contains any non-digit characters, the method returns 0. You may not use the parseInt method of the Integer class, but you may use the length, charAt and substring methods of the String class (see above).

4. Write a static method

   public static String reverseString(String str)

   that returns the reverse of the parameter str. For example, if parameter str is "hello", the method should return "olleh".

5. Write a static method

   public static boolean prime(int n)

   that returns true if the parameter n is a prime number, and false otherwise. (A number n is prime if the only positive integer less than n that divides n exactly (with zero remainder) is 1.)

6. Write a static method

   public static int sumDivisors(int n)

   that returns the sum of the positive integers less than the parameter n that divide into n exactly (with zero remainder). For example, if n has the value 8, the method returns the value 7 (since 7 = 1 + 2 + 4) and if n has the value 9, the method returns the value 4 (since 4 = 1 + 3).

   Continued on Next Page
7. Consider the following class.

```java
class Player {
    // a constructor for constructing a player for the coin toss game
    public Player(int goal) {
        heads = 0;  // initialize instance variable heads
        GOAL = goal;  // set the final instance variable GOAL
    }
    // instance variables
    private int heads;
    private final int GOAL;
}
```

Write a method for the Player class

```java
public void flip()
```
that simulates the tossing of a coin by calling `Math.random()`, incrementing the instance variable `heads` if the random number generated is greater than one half.

8. Write a method for the Player class

```java
public void flipMany(int n)
```
that simulates the tossing of n coins by calling `Math.random()` exactly n times, each time incrementing the instance variable `heads` if the random number generated is greater than one half.

9. Write a method for the Player class that returns true if a player has “won” the game, meaning that its number of “heads” is greater than or equal to `GOAL` (specified above as a final instance variable of the class). Otherwise, it returns false.

10. Write a method for the Player class

```java
public boolean equals(Player other)
```
that checks whether two players have the same `heads` and `GOAL` values. One of the players is the explicit parameter, and the other one is the implicit parameter. This would be called, for example, by the code

```java
jack = new Player(10);  // construct jack object
jill = new Player(10);  // construct jill object
jack.flipMany(20);  // toss jack’s coin 20 times
jill.flipMany(20);  // toss jill’s coin 20 times
if (jack.equals(jill))  // check whether jack and jill have same
    ......  // values for their instance variables
```