STUDY SESSION QUESTIONS, NOT ON THE MIDTERM

Answer whether each of the following errors is a syntax error, run-time error, or logic error:

```
int value = 64.0;
System.out.println("Can't wait for Wednesday!);
int[] array = new int[10]; System.out.println("Last value is " + array[10]);
int[] array = new int[10]; System.out.println("First value is " + array[1]);
```

a> Syntax: double cannot be assigned to float

b> Syntax: missing double-quote

c> Runtime: array index out of bounds

d> logic: first value is array[0]

Which of the following code snippets could cause a runtime, but *not* a compile time error?
More than one might be correct. Assume the necessary import statements are present.

a) `Scanner scan = new Scanner(System.in);  int num = scan.nextLine();`
b) `int num = Integer.parseInt(args[0]);`
c) `String[] arr = new String[4];`
d) `int[] arr = new int[2];  arr[2] = 100;`
e) `ArrayList<string> ls = new ArrayList<>();`

a> compile time: nextLine() returns String, cannot assign to int
b> runtime, not compile time: args[0] might not hold an int
c> no error
d> runtime not compile time: array index out of bounds
e> compile time: "string" should be capitalized

Fill in the body of the following function:

```
int sumArrayElements(int[] theArray) {
      int total = 0;
      for(int i = 0; i < theArray.length; i++) {
            total += theArray[i];
      }
      return total;
}
```

A palindrome is a word that is spelled the same backwards as it is forwards, such as “radar”. The following code is supposed to print TRUE if its input is a palindrome, but the code is buggy. Please fix it.

```
class Palindrome {
  Palindrome(String input) {
    while (index <= input.length) {
      if (input.charAt(index) == input.charAt(input.length) - index) {
        System.out.println("TRUE");
      }
    }
    System.out.println("FALSE");
  }
}
```

```
class Palindrome {
  Palindrome(String input) {
    int index = 0;
    while (index < input.length) {
      // if any character does not match its partner, print FALSE and return
      if (input.charAt(index) != input.charAt(input.length - index - 1)) {
        System.out.println("FALSE");
        return;
      }
      index++;
    }
    // if all chars match their partners, print TRUE
    System.out.println("TRUE");
  }
}
```

Write a class for a `Circle`. A `Circle` is described only by a `radius`, which should be a `double`. The constructor should allow a user to create a `Circle` of any desired size, but only ever with a value above 0. The `Circle` class should only have two methods besides the constructor: `perimeter()`, which should return the value of the perimeter; and `area()`, which should return the value of the area. (Recall that the formula for perimeter is 2*p*r, the formula for area is p*(r*r), and the approximate value of p is 3.14.)

BONUS: how would you throw an exception if the given radius is not ok?

```
class Circle {
	int radius;
	final double PI = 3.14;

	Circle(int d) {
		if (d <= 0) {
			throw new Exception("bad radius value " + d);
		}
		radius = d;
	}
	double perimeter() { return 2*PI*radius; }
	double area() { return PI*radius*radius; }
}
```

Write code for the main() method that reads an array of one or more strings from the command line and prints them out, centered, surrounded by asterisks on the border, e.g.

```
************************
* Hi Mom, how are you? *
*   Happy birthday!    *
************************
```

```
class Banner {
     static public void main(String[] args) {
          // Get longest string. Skip error checking.
          int longest = 0;
          for (int i = 0; i < args.length; i++) {
               if (longest < args[i].length()) {
                    longest = args[i].length();
               }
          }
          final int LineLen = longest + 4; // star and space on each side

          // Print top line
          for(int j = 0; j < LineLen; j++) {
               System.out.print("*");
          }
          System.out.println();

          // Print each string
          for(int k = 0; k < args.length; k++) {
               // How many spaces on EACH SIDE for this string?
               // The "-2" is for the asterisks on each side.
               // The 'extra' is because we may need an odd # of spaces.
               final int padding = (LineLen-args[k].length()-2)/2;
               final int extra = (LineLen-args[k].length()-2) % 2;
```

```
                System.out.print("*");
                for (int m = 0; m < padding; m++) {
                      System.out.print(" ");
                }
                System.out.print(args[k]);
                for (int m = 0; m < padding+extra; m++) {
                      System.out.print(" ");
                }
                System.out.println("*"); // including newline
          }

          // Print bottom line
          for(int j = 0; j < LineLen; j++) {
                System.out.print("*");
          }
          System.out.println();
     }
}
```

Write a program that reads an integer from the command line that represents some number of seconds. (The number will be an integer >= 0.) The program shall print

```
<<number>> hours, <<number>> minutes, <<number>> seconds
```

where the number of seconds and number of minutes is less than 60. The time represented by the printed answer should equal the input. For example, if the input is 3600, the output should be 1 hours, 0 minutes, 0 seconds.

```
class TimeFix {
	static public void main(String[] args) {
		int input = Integer.parseInt(args[0]); // ignore error checking here
		int seconds = input % 60;
		int minutes = (input/60) % 60;
		int hours = input / 3600;
		System.out.println(hours + " hours, " + minutes + " minutes, " + seconds + " seconds");
	}
}
```