CS 1331 Exam 1 Practice

ANSWER KEY

- Signing signifies you are aware of and in accordance with the **Academic Honor Code of Georgia Tech**.
- Calculators and cell phones are NOT allowed.
- This is an object-oriented programming test. Java is the required language. Java is case-sensitive. DO NOT WRITE IN ALL CAPS. A Java program in all caps will not compile. Good variable names and style are required. Comments are not required.

Question	Points per Page	Points Lost	Points Earned	Graded By
Page 1	9	-	=	
Page 2	10	-	=	
Page 3	10	-	=	
Page 4	10	-	=	
Page 5	20	-	=	
Page 6	10	-	=	
TOTAL	69	-	=	

1. **True or False**

   In each of the blanks below, write "T" if the statement beside the blank is true, "F" otherwise.

[1] (a) **F** A public top-level Java class may be defined in a source file with any base name as long as the file extension is `.java`.

[1] (b) **F** Java identifiers can contain letters, digits, and the underscore symbol and may start with a digit.

[1] (c) **F** The statement `int x = 3f/4f;` will compile, but the result will be truncated so that `x` gets the value `0`.

[1] (d) **F** In a for loop header, `for (`*initializer*`;` *condition*`;` *update*`)`, the Java compiler requires *initializer* to initialize a loop variable and *update* to update it.

[1] (e) **T** The declarations `double scores[]` and `double[] scores` are equivalent.

[1] (f) **F** Java arrays have a variable number of elements of mixed types.

[1] (g) **F** Given an array named `scores`, the statement `scores[scores.length + 1]` will not compile.

[1] (h) **F** Instance methods of a class can be called without first instaniating an object of the class.

[1] (i) **F** Every Java class has a default no-arg constructor in addition to the constructors you write yourself.

2. **Expression Evaluation**

For each expression below, write the value and then the Java data type of the evaluated legal expression in the space provided. Be exact. The type you give must be the **exact spelling of a Java primitive type including uppercase vs lowercase as it would appear in your program.**

Expression: `7 / 2`

[1] (a) Calculated value: 3

[1] (b) Java primitive type: `int`

Expression: `64 - 16 * 2`

[1] (c) Calculated value: 32

[1] (d) Java primitive type: `int`

Expression: `2.5f + 3.0 - 1.5f`

[1] (e) Calculated value: 4.0

[1] (f) Java primitive type: `double`

Expression: `new Double(1) / 2`

[1] (g) Calculated value: 0.5

[1] (h) Java primitive type: `double`

Expression: `true && "Foo".equals("foo")`

[1] (i) Calculated value: `false`

[1] (j) Java primitive type: `boolean`

Points available: 10 - points lost: ______ = points earned: ______. Graded by: ______

3. **Multiple Choice** Circle the letter of the correct choice.

   Given:

```
    public class Kitten {

        private String name = "";

        public Kitten(String name) {
            name = name;
        }

        public String toString() {
            return "Kitten: " + name;
        }

        public boolean equals(Object other) {
            if (this == other) return true;
            if (null == other) return false;
            if (!(other instanceof Kitten)) return false;
            Kitten that = (Kitten) other;
            return this.name.equals(that.name);
        }
    }
```

   Assume the following statements have been executed:

```
        Kitten maggie = new Kitten("Maggie");
        Kitten fiona = new Kitten("Fiona");
        Kitten fiona2 = new Kitten("Fiona");
        Kitten mittens = fiona;
```

[2] (a) What is the value of `maggie`?

- **A. the address of a `Kitten` object**
- B. null
- C. automatically set to 0
- D. undefined

[2] (b) What is printed on the console after the following statement is executed?

`System.out.println(maggie.toString());`

- **A. Kitten:**
- B. Kitten: null
- C. Kitten: Maggie

[2] (c) What is the value of the expression `fiona == mittens)`?

- **A. true**
- B. false

[2] (d) What is the value of the expression `fiona.equals(fiona2) && fiona == mittens`?

- **A. true**
- B. false

[2] (e) After executing `Kitten[] kittens = new Kitten[5];` , what is the value of `kittens[0]` ?

- **A. null**
- B. the address of a `Kitten` object
- C. automatically set to 0
- D. undefined

Points available: 10 - points lost: ______ = points earned: ______. Graded by: ______

4. **Tracing**

Consider the following code:

```
public class StrangeLogic {
    private static int counter = 0;

    private static boolean incrementCounter() {
        counter++;
        return true;
    }
    public static void main(String args[]) {
        boolean a = true, b = false;
        if (b || incrementCounter()) {
            System.out.print("Boo");
        }
        if ((a || b) && incrementCounter()) {
            System.out.print(" ya!");
        }
        System.out.println(counter);
    }
}
```

[5] (a) What is printed when main is executed?

Solution: Boo ya! 2 (1 pt for Boo, 2 for ya!, 2 for 2)

Consider the following code:

```
public class AlGore {

    public static void main(String[] args) {
        String mystery = "mnerigpaba";
        String solved = "";
        int len = mystery.length();
        for (int i = 0, j = len - 1; i < len/2; ++i, --j) {
            solved += mystery.charAt(i) + mystery.charAt(j);
        }
        System.out.println(solved);
    }
}
```

[5] (b) What is printed when main is executed?

Solution: manbearpig

Points available: 10 - points lost: ______ = points earned: ______. Graded by: ______

5. **Short Answer**

[2] (a) Assume you have a Java class named `Foo` that you want to be able to run from the command line. Write the header for the method you need to define in `Foo` to make it executable from the command line.

Solution: `public static void main(String[] args)`

[2] (b) Assume you are at the command line in the directory of the file that contains the definition for a Java class named `Foo`. Write the command that you would execute on the command line to compile `Foo`.

Solution: `javac Foo.java`

[2] (c) If the command above executes successfully, what file will be produced?

Solution: `Foo.class`

[2] (d) Write the command that will execute the `Foo` class you compiled above.

Solution: `java Foo`

[3] (e) What will the following code print?

```
for (int i = 10; i > 0; i--);
   System.out.print(‘‘Meow! ’’);
```

Solution: Meow!

[4] (f) Assume you have the following start of a Student class and Student constructor. Finish the constructor. Do not change the code that is given.

```
public class Student {
    private String name;
    private String email;
    public Student(String name, String email) {
        // Your code goes here
      this.name = name;  // 2 point
      this.email = email; // 2 point
}
```

[5] (g) Convert the following `for` loop to an equivalent `while` loop.

```
for (int i = 10; i > 0; i--) {
   System.out.println(i);
}
```

Solution:

```
int i = 10;                 // 1 pts
while (i > 0) {             // 1 pt
   System.out.println(i);   // 1 pt
   i--;                     // 1 pt
}
```

6. **Complete the Method**

[5] (a) Fill in the code for the following method that takes an array of numbers and returns the number of even numbers in the array argument. Your code should use a `for` loop.

```
public int evens(int[] numbers) {
    // Your code goes here
```

Solution:

```
    int count = 0;                   // 1 point
    for (int number: numbers) {      // 1 point
        if ((number % 2) == 0) {     // 1 point
            count++;                 // 1 point
        }
    }
    return count;                    // 1 point
```

```
}
```

[5] (b) Fill in the code for the following method that takes an array of numbers and a number and returns `true` if the array contains the number, `false` otherwise. You will need a loop, and your loop must not execute more iterations than necessary, and you cannot use `break` or `continue`.

```
public boolean contains(int[] numbers, int n) {
    // Your code goes here
```

Solution:

```
boolean found = false;                      // 1 point for initializations
int i = 0;                                  // 1 point
do {
    if (numbers[i] == n) {                 // 1 point for conditional setting
        found = true;                       //   of found to true
    }
    i++;
} while (!found && (i < numbers.length))    // 1 point for termination condition
return found;                               // 1 point for return
```

```
}
```