Introduction to **Java**

Tim Magoun and Aravind Koneru

compiled on Friday 4th November, 2016 09:43

This test will evaluate the familiarity of basic programming concepts as well as the knowledge of the Java programming language, which is used as the programming language of numerous FIRST®robotics competitions.

The following topics will be on this test:

- Primitive Types and Operations (`int`, `byte`, `boolean`, `etc`.)
- Modifiers (`final`, `public`, `static`, `etc`.)*
- Comparison Operators (`==`, `!=`, `>=`,`etc`.)
- Assignment operators (`+=`, `*=`, `=`, `etc`)
- Flow Control (`if`, `for`, `break`, `etc`)
- Methods and Parameters*
- Single- and Multi-Dimensional Arrays
- Object Oriented Programming*
- Inheritance and Polymorphism*
- Programming Habits and Conventions

* Starred items are extremely important in programming a robot

DO NOT BEGIN UNTIL INSTRUCTED TO DO SO

PART ONE: Multiple Choice

Instructions: Choose the correct solution to the problem, there is only one correct answer for each problem.

1. The size of a `boolean` variable is
   (a) 1 byte
   (b) 4 bytes
   (c) 1 bit
   (d) 16 bits
2. When adding an `int` to a `double`, the resulting variable will be
   (a) an `int` with lower precision
   (b) an `int` with the same precision
   (c) a `double` with lower precision
   (d) a `double` with same precision
3. When the modifier `private` is used, where could one could access the member?
   (a) Inside the same `class`
   (b) Inside the same `package`
   (c) Inside the same superclass
   (d) Only the processor could access the member
4. When should one use the modifier `static`?
   (a) When the member should not be modified
   (b) When the member needs to be shared across all instances of the class
   (c) When the member should not be accessed by the end-user
   (d) When the member changes in value frequently
5. What data type does a conditional statement return?
   (a) `int`
   (b) `boolean`
   (c) `boolean* pointer`
   (d) conditional statements do not return any data type

6. What is the outcome when one executes the following code?

```
public static void main(String[] args) {
        int x = 3;
        int []y= {3,4};
        if((x > (int) Math.PI) && (y[2] <= 3))
                System.out.print("True");
        System.out.print("False");
}
```

   (a) True
   (b) True False
   (c) False
   (d) Runtime Error: ArrayIndexOutOfBoundsException

7. What is the outcome when one executes the following code?

```
public static void main(String[] args) {
        int x = 3;
        int []y= {3,4};
        if((x > (int) Math.PI) & (y[2] <= 3))
                System.out.print("True");
        System.out.print("False");
}
```

   (a) True
   (b) True False
   (c) False
   (d) Runtime Error: ArrayIndexOutOfBoundsException

8. Which of the following is an equivalent statement for (x || y) && !x

   (a) y && x
   (b) x || y
   (c) !y
   (d) y && (y || x)

9. The statement y || (3 * x) > 24 evaluates

   (a) type `int`

   (b) type `double`

   (c) type `String`

   (d) type `boolean`

10. The output of the following annoying program is

```
public class LoopTracing {
        public static void main(String[] args) {
                int input = 10, output = 0;
                while (input > 1) {
                        output += input;
                        if (input % 2 == 0)
                                input /= 2;
                        else
                                input = (input + 1) / 2;
                }

                System.out.println(output);
        }
}
```

   (a) NullPointerException

   (b) 0

   (c) 20

   (d) The program is an infinite loop

11. What happens if you run the following program?

```
public class SwitchTracing {
        public static void main(String[] args) {
                int input = 9;
                for (; input > 0; input--) {
                        switch (input) {
                        case 1:
                                System.out.print("One ");
                        case 4:
                                System.out.print("Four ");
                        case 9:
                                System.out.print("Nine ");
                        }
                }
        }
}
```

    (a) Nine Four Nine One Four Nine
    (b) Nine Four One Nine Four One Nine Four One
    (c) Nine Four One
    (d) Would not compile because of incomplete for-loop

12. Example Question One

    (a) Answer One
    (b) Answer Two
    (c) Answer Three
    (d) Answer Four

13. Example Question One

    (a) Answer One
    (b) Answer Two
    (c) Answer Three
    (d) Answer Four

14. Example Question One

    (a) Answer One
    (b) Answer Two

(c) Answer Three
(d) Answer Four

15. Example Question One

(a) Answer One
(b) Answer Two
(c) Answer Three
(d) Answer Four

16. Example Question One

(a) Answer One
(b) Answer Two
(c) Answer Three
(d) Answer Four

17. Example Question One

(a) Answer One
(b) Answer Two
(c) Answer Three
(d) Answer Four

18. Example Question One

(a) Answer One
(b) Answer Two
(c) Answer Three
(d) Answer Four

19. Example Question One

(a) Answer One
(b) Answer Two
(c) Answer Three
(d) Answer Four

20. Example Question One

(a) Answer One

(b) Answer Two

(c) Answer Three

(d) Answer Four

CONTINUE TO THE NEXT PAGE

PART TWO: Open Ended Response

*Instructions: Write the most efficient solution to the following methods. You will **not** be given any extra paper.*

1. Write a method that will return an array of n length, filled with the decimal approximations of the sequence $\left[\frac{1}{1}, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \cdots, \frac{1}{n}\right]$ where n is the integer parameter of the method.

```
public static int[] fractionGenerator(int n){
```

DO NOT CONTINUE UNTIL INSTRUCTED TO DO SO

2. Write a method that will recursively determine if a word *str* is a palindrome, where *str* is a string parameter of the method.

```
public static boolean palindromeChecker(String str){
```

3. Given the following super class:

```
        public Counter() {

                value = 10;
                maxValue = 100;
                minValue = 0;

        }

        public Counter(int maxValue, int minValue, int value) {

                this.value = value;
                this.maxValue = maxValue;
                this.minValue = minValue;

        }

        public boolean countUp() {
                value++;
                return checkBounds();
        }

        public boolean countDown() {
                value--;
                return checkBounds();
        }

        public boolean checkBounds() {
                return (value >= minValue || value <= maxValue);
        }
}
```

Write a subclass named *IntervalCounter* that is a subclass of *Counter* and has an additional integer instance field called interval.

```
public class IntervalCounter extends Counter {
        private int interval;
        //Create a default constructor with the initial interval of 2




        //Create an overloaded constructor with all of the parameters




        //Override the countUp and countDown methods so that
        //the value is changed by the interval




        //Create a method named correctValues that will limit the
        //value to the minimum or the maximum values stated
```

END OF EXAM