CODE ANALYSIS

(Replacement for Ping Pong Parachute)

1. **DESCRIPTION:** Given a set of computer programs or parts of programs, teams will determine either the output or the error that each would produce.
    a. **A TEAM OF UP TO:** 2 **APPROX TIME:** 50 minutes
2. **EVENT PARAMETERS:**
    a. Each team is permitted either one 8.5 by 11 inch, double-sided reference sheet or two 8.5 by 11 inch, single-sided reference sheets.
    b. Competitors may NOT bring electronic devices into the testing room, including computers, tablets, cell phones, smart watches, or calculators.
    c. All computer code shall be given in the Java programming language. Problems should focus on general programming principles, rather than the peculiarities of the Java language.
    d. Code may be given as a sequence of statements to be executed in order, as a method or methods to be called with a given set of arguments, or as a complete program with a `main()` method.
    e. All input and data values shall be specified in the given code or as a given set of arguments to a method call. No other form of input is permitted, including command-line arguments, `System.in`, external files, or graphical user interfaces.
    f. Each problem will take one of the following two forms. Competitors shall:
        i. Determine the output of the code, either the return value of a method call or the text created by `System.out.print()`, `System.out.println()`, and/or `System.out.format()`. Space shall be provided for competitors to write their answer. If a problem's score depends on the number of output lines or the number or spacing of characters, then the answer space shall include a grid.
        ii. Describe the run-time error that results when the given code runs. Competitors are not expected to reproduce the error message or exception type, but rather to describe concisely and specifically what goes wrong, why the code does not run properly, and how to correct it.
3. **PERMITTED TOPICS:**
    a. Integers, unary operator `-`, and binary operators `+`, `-`, `*`, `/`, and `%`. No more than 10% of the score may involve bitwise operators `~`, `&`, `|`, `^`, `<<`, `>>`, or `>>>` or binary, octal, or hexadecimal constants.
    b. Characters must be in the range `' '` (space) to `'~'` (tilde), plus escape sequences `'\\'` (backslash), `'\''` (single quote), and `'\"'` (double quote). Addition and subtraction of characters are permitted as long as the result is in one of the ranges `'0'` to `'9'`, `'A'` to `'Z'`, or `'a'` to `'z'`.
    c. `boolean`, `true`, `false`, unary operators `!` and type casts, binary operators `==`, `!=`, `<`, `<=`, `>`, `>=`, `&&`, `||`, and `^`, ternary operator `?:`, and operator precedence and associativity.
    d. Arrays of fixed size, initializer lists, setting and getting elements using `[]`, two-dimensional arrays, and allocation of fixed-size arrays using the syntax "`new ElementType[length]`".
    e. Strings: String literals made from the permitted character literals and methods `length`, `charAt`, `getChars`, `equals`, `equalsIgnoreCase`, `startsWith`, `endsWith`, `indexOf`, `lastIndexOf`, `substring`, `concat`, and concatenation via the `+` operator.
    f. Variable declarations. No more than 10% of the score may involve issues of variable scope.
    g. Assignments (for example `=` or `+=`), pre- or post-increment (`++`), and pre- or post-decrement (`--`) may be used, but may not use the return values of the operators (for example, `x++ – y`).
    h. `{}` blocks, `if`, `if…else`, `while`, `for`. Any style of comments. Code shall be properly indented, except in problems amounting to no more than 10% of the score.
    i. Method definitions, arguments, return values, and recursion.
    j. `System.out.format()` is limited to fixed text and the format specifiers `%%`, `%n`, `%c`, `%d`, and `%s`.
4. **FORBIDDEN TOPICS:**
    a. `import` statements and `class` definitions may be included to make the code a complete program, but competitors are not expected to interpret their significance.
    b. `assert`, `break`, `continue`, `do…while`, `switch`, `try…catch`.
    c. Classes, interfaces, inheritance, polymorphism, `new` (except as detailed in rule 3d above), constructors, lambdas, method overloading, instance variables, `this`, `finalize`, or `instanceof`.
    d. `enum`s, `static` (except as used to declare a `main()` method), variable-length argument lists, regular expressions, collections, or dynamic memory management.
    e. Methods of the `Array` and `Arrays` classes. File I/O, networking, threads, web programming.
    f. Any feature of Java that is non-standard, or which varies among versions or platforms.
5. **SCORING:**
    a. Each problem shall specify its number of points. Partial credit should be given where appropriate.
    b. No single problem may be worth more than 20% of the total score.