★ANSWER KEY – CONFIDENTIAL★

UIL COMPUTER SCIENCE WRITTEN TEST – 2016 INVITATIONAL B

Questions (+6 points for each correct answer, -2 points for each incorrect answer)

1) D	11) E	21) A	31) C
2) A	12) C	22) C	32) E
3) C	13) C	23) B	33) D
4) A	14) A	24) A	34) B
5) B	15) D	25) D	35) C
6) E	16) E	26) B	36) A
7) B	17) D	27) E	37) D
8) B	18) D	28) C	38) E
9) D	19) B	29) A	*39) $\overline{P} * (Q+R)$
10) A	20) C	30) D	40) -42

* See "Explanation" section below for alternate, acceptable answers.

> **Note:** Correct responses are based on **Java SE Development Kit 8 (JDK 8)** from Sun Microsystems, Inc. All provided code segments are intended to be syntactically correct, unless otherwise stated (e.g., "error" is an answer choice) and any necessary Java SE 8 Standard Packages have been imported. Ignore any typographical errors and assume any undefined variables are defined as used.

Explanation

1)   D $42_{10} * 3_{10} =$ `1111110`$_2$ $= 1332_4 = 176_8 = 126_{10} = 7e_{16}$

2)   A   `3 * 2.5 = 7.5`

3)   C   `"d%d.o%o.x%x"`: The substrings, `"%d"`, `"%o"`, and `"%x"`, insert the 3 integer parameters into the output formatted as *decimal*, *octal*, and *hexadecimal* integers, respectively.

4)   A   `dna.indexOf("CAA")` returns a value of 1; `dna.indexOf("CA", 1)` returns a value of 1

5)   B

p	q	r	!(p || !(q && !r))
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

6)   E   "Incompatible types" error occurs at compile time. `Math.round()` returns a `long`, which cannot be assigned to `int` without loss of precision.

7)   B   In addition to assigning a value of 5 to `val`, the compound assignment statement `val %= 20` also evaluates to the `int` value assigned to `val` (i.e., 5).

8)   B   28 is a multiple of 4. The first conditional evaluates to `true`. All other conditions are unchecked.

9)   D   Loop iterates through 6 times when `control` is 64, 32, 16, 8, 4, and 2. Each iteration prints a "+" if `control` is still even *after* being halved. The 6th iteration (when `control` is initially 2 and gets halved to 1) does not print anything.

10)   A   Each index position from 1 through `digits.length - 1` is assigned 1 more than its predecessor.

11)   E   Sum requires that input be read as `int` values using `nextInt()`, so loop should be conditioned on the Boolean method `hasNextInt()`.

12)   C   Sum of all integers in file = 11 + 9 + 70 + 3 + –50 + 19 + 12 + 5 + 7 + 1 + 4 + –3 = 88

13)   C   `(5.0 + ((7 / 2) * 3)) = (5.0 + (3 * 3)) = (5.0 + 9) = 14.0`. Note that `7 / 2` uses integer division that truncates result to 3.

14)   A   As a 64-bit, floating-point value the `double` data type can store a much wider range of values than can be stored in a 32-bit `float`.

15)   D   Individual letters are inserted into the `List` in reverse order. Removes every `'b'` that does not immediately follow a `'b'` that was previously removed (i.e., for each pair of adjacent `'b'` characters, only the first is removed from the list). Output `String` consists of the remaining letters in the same reversed order in which they are stored in the `List`.

16)   E   `\\D+`   Equates to "1 or more non-digits".
              `[int f]+`   Equates to "1 or more of the 5 characters: `i`, `n`, `t`, space, or `f`".
              `\\D\\D\\W?`   Equates to "2 non-digits followed by 0 or 1 non-word (non-alphanumeric) characters".
              `(ft )|(in)`   Equates to "the substring `'ft '` (including the space) OR the substring `'in'` (no space)".

17)   D   While the *IEEE 754: Standard for Binary Floating-Point Arithmetic* defines distinct binary representations for negative zero and positive zero, the specification states, "*Comparisons shall ignore the sign of zero (so +0 = − 0).*" Thus, negative zero (`-1.0`)and positive zero (`0.0`) are considered equal values.

18)   D   The `nonNice()` method calculates the arithmetic mean of the array contents. Integer division truncates the mean before the value is returned.

19)   B   sum = (0 + 2 + 4 + 6 + 8) + (1+ 3 + 5 + 7 + 9) + (2 + 4 + 6 + 8) + (3 + 5 + 7+ 9) + (4 + 6 + 8) = 107

20)   C   The `PriorityQueue` class is implemented as a min-heap with its `toString()` method producing a level-by-level traversal of the resulting tree. The structure of the min-heap after each element is added is shown below:
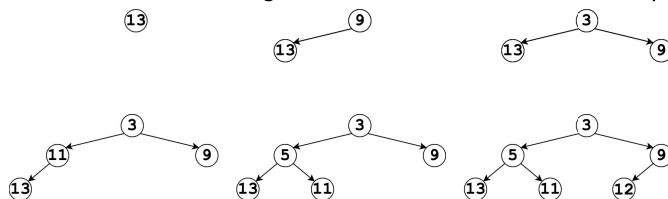
21)   A   Bubble Sort arranges `String` values in descending order (i.e., lowercase letters > uppercase letters > digits).

22)   C   Depth-First Search produces a pre-order traversal of a binary tree.

23)   B   Breadth-First Search produces a level-by-level traversal of a binary tree.

24)   A   `10000111 & 00001111 = 00000111 = 7`

25)   D   The superclass, `Alpha`, has only 1 constructor and it requires a `String` parameter.

26)   B   Private `inty` field in the `Omega` class is static.

27)   E   All references in `Omega` to `inty` refer to the local copy of `inty` declared in the `Omega` class.

28)   C   `Alpha` and `Inty` are abstract and cannot be directly instantiated.

29)   A   The tree and `huff()` method function similarly to a Huffman code. The `int` parameter serves as a bit stream (starting with the least significant bit) containing a Huffman-encoded message. $2658_{10} = 101001100010_2$. Reading right to left, `010 = A, 001 = C, 100 = E, 101 = D`.

30)   D   When `c == 0`, the value of node `r` is returned. When `r == null` (i.e., `r` references an empty tree), a `nullPointerException` is thrown. When `c != 0` and `r` is a leaf, the method results in infinite recursion.

★ANSWER KEY – CONFIDENTIAL★

31)   C   When `c != 0`, the method either traverses down to a child of `n`, reducing the value of `c` (i.e., approaches the base case where `c == 0`), or it restarts the traversal back at `r`. When `r == null` (i.e., `r` references an empty tree), a `nullPointerException` is thrown.

32)   E   `('T' = 84) < ('e' = 101) < ('r' = 114)`

33)   D   `4 + (((8 + 7) * 2) * (3 - 1)) = 64`

34)   B   Tree is complete. Every interior node is greater than its children.

35)   C

X Y Z	$(X + \overline{Y})(X + Z)$	$X\overline{Y} + XZ$	$\overline{X}\,Y + \overline{X}\;\overline{Z}$	$X + \overline{Y}Z$	$X(\overline{Y} + Z)$	$X + (\overline{Y} + Z)$
0 0 0	**0**	0	1	**0**	0	1
0 0 1	**1**	0	0	**1**	0	1
0 1 0	**0**	0	1	**0**	0	0
0 1 1	**0**	0	1	**0**	0	1
1 0 0	**1**	1	0	**1**	0	1
1 0 1	**1**	1	0	**1**	1	1
1 1 0	**1**	0	0	**1**	0	1
1 1 1	**1**	1	0	**1**	0	1

36)   A   Each `Struct` object represents a node in a directed graph. The `Struct` class statically stores a collection of all nodes in the graph.

37)   D   The `Struct` class is backed by a `TreeSet`. The `contains()` method for a `TreeSet` has $O(\log_2 N)$ performance.

38)   E

Before `v3.add(v2)`   After `v3.add(v2)`
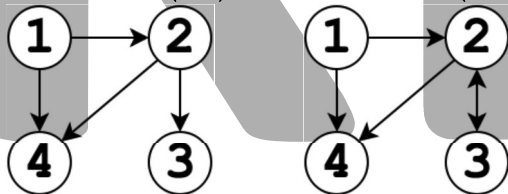
39)   Any answer that equivalently expresses "(Logical-NOT P) Logical-AND (Q Logical-OR R)" is acceptable (e.g., "!P && (Q || R)", "P' (Q + R)", " $\overline{P}$ (Q + R)").

40) $11010110_2 = -42_{10}$. $00101010_2 = +42_{10}$.