★ANSWER KEY – CONFIDENTIAL★

UIL COMPUTER SCIENCE – 2018 DISTRICT

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

1) C	11) B	21) A	31) C
2) C	12) E	22) E	32) D
3) A	13) E	23) D	33) C
4) E	14) D	24) D	34) E
5) A	15) C	25) C	35) A
6) B	16) A	26) E	36) B
7) D	17) D	27) B	37) B
8) A	18) E	28) A	38) D
9) B	19) B	29) B	*39) 20
10) C	20) C	30) C	*40) -86

KEY

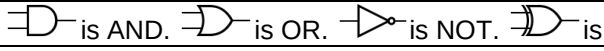
** See "Explanation" section below for alternate, acceptable answers.*

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.

Explanations:

<table>
<tr><td>1.</td><td>C</td><td>$11111111_2+11110011_2=111110010_2$ (eliminates D). $111110010_2 = 498_{10}$ (eliminates B). $1F3_{16} = 499_{10}$ (eliminates A). $762_8 = 498_{10}$.</td></tr>
<tr><td>2.</td><td>C</td><td>10+5-3*6.0/4=
10+5-18.0/4=
10+5-4.5=
15-4.5=
10.5</td></tr>
<tr><td>3.</td><td>A</td><td><code>println</code> method inserts a new line after the string is printed. The <code>\n</code> escape sequence inserts a new line where ever it has been inserted in the string.</td></tr>
<tr><td>4.</td><td>E</td><td>
<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td></td><td>0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td>p</td><td>l</td><td>a</td><td>n</td><td>e</td><td>t</td><td></td><td>m</td><td>o</td><td>o</td><td>n</td></tr>
</table>
The two argument substring method starts at the index number of the first argument and goes to the second argument minus one. So, the first substring is from one to one (just the l). The one argument substring method starts at the index number of the argument and continues to the end of the string. In this case 1 to 3 (oon).
</td></tr>
<tr><td>5.</td><td>A</td><td>T^F&&T||F=
T&&T||F=
T||F=
T</td></tr>
<tr><td>6.</td><td>B</td><td>The <code>cbrt</code> method returns the cube root of its argument as a double. 4X4X4=64.</td></tr>
<tr><td>7.</td><td>D</td><td>ASCII value of ‘A’ is 65. 65+8-4.65=68.35</td></tr>
<tr><td>8.</td><td>A</td><td>Once a <code>true</code> value is encountered, in this case the boolean variable <code>no</code> is true, the code for that <code>if</code> statement is executed and the remaining <code>else</code> statements are skipped.</td></tr>
<tr><td>9.</td><td>B</td><td><code>i</code> takes the values 0, 2, 4, 6, 8, and 10. Once I becomes 12, the loop stops. That makes 6 six iterations of the loop.</td></tr>
<tr><td>10.</td><td>C</td><td>
<table>
<tr><th>int[] list=new int[5]</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th></tr>
<tr><td></td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>list[1]=8</td><td>0</td><td>8</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>list[2]=12</td><td>0</td><td>8</td><td>12</td><td>0</td><td>0</td></tr>
<tr><td>list[3]=10</td><td>0</td><td>8</td><td>12</td><td>10</td><td>0</td></tr>
<tr><td>list[4]=list[list[2]-list[3]];</td><td>0</td><td>8</td><td>12</td><td>10</td><td>12</td></tr>
</table>
</td></tr>
<tr><td>11.</td><td>B</td><td><code>nextInt</code> returns the next value beyond (in front of) the cursor (pointer) in the datafile and then advances the cursor to the next position, even in the condition for a while loop. 5, 1 and -3 are used in the while loop condition statement. 9 and 7 are added to <code>a</code>. 4 is returned by the final call to <code>nextInt</code> because even though the loop terminates when -3 is read, the cursor advances to the next position.</td></tr>
<tr><td>12.</td><td>E</td><td><code>i</code> is decremented before it is added to <code>d</code> with each iteration of the loop.
<pre>   d i
 9.0 9
17.0 8
24.0 7
30.0 6
35.0 5
39.0 4
42.0 3
44.0 2
45.0 1
45.0 0</pre></td></tr>
<tr><td>13.</td><td>E</td><td>Precedence from first to last for the operators shown is: <code>* <= ^ && ||</code></td></tr>
<tr><td>14.</td><td>D</td><td>The <code>MIN_VALUE</code> for <code>Integer</code> is -2147483648. The largest negative value that can be stored in a variable of type <code>byte</code> is -128.</td></tr>
</table>

<table>
<tr><td>15.</td><td>C</td><td>
<table>
<tr><td><code>a.add(4);</code></td><td>[4]</td></tr>
<tr><td><code>a.set(0,0);</code></td><td>[0]</td></tr>
<tr><td><code>a.add(5);</code></td><td>[0, 5]</td></tr>
<tr><td><code>a.set(0,6);</code></td><td>[6, 5]</td></tr>
<tr><td><code>a.remove(1);</code></td><td>[6]</td></tr>
</table>
</td></tr>
<tr><td>16.</td><td>B</td><td>
<table>
<tr><td colspan="2">Stacks use a first in, last out protocol for accessing data.</td></tr>
<tr><td><code>s.push("one");</code></td><td>one</td></tr>
<tr><td><code>s.push("two");</code></td><td>one two</td></tr>
<tr><td><code>s.push("two");</code></td><td>one two two</td></tr>
<tr><td><code>s.pop();</code></td><td>one two</td></tr>
<tr><td><code>s.push("three");</code></td><td>one two three</td></tr>
<tr><td><code>s.push("four");</code></td><td>one two three four</td></tr>
<tr><td colspan="2">Elements are popped out from right to left.</td></tr>
</table>
</td></tr>
<tr><td>17.</td><td>C</td><td>
<table>
<tr><td>i</td><td>s</td></tr>
<tr><td>5</td><td>Pecos</td></tr>
<tr><td>4</td><td>Peco</td></tr>
<tr><td>3</td><td>Pec</td></tr>
<tr><td>2</td><td>Pe</td></tr>
<tr><td>1</td><td>P</td></tr>
</table>
</td></tr>
<tr><td>18.</td><td>A</td><td>Since <code>height</code>, <code>width</code> and <code>length</code> are all doubles, <code>surfaceArea</code> must return a double value.</td></tr>
<tr><td>19.</td><td>D</td><td><code>l</code>, <code>w</code>, and <code>h</code> have not been declared locally so they must be passed as parameters. The parameter list must show the type and name of each parameter.</td></tr>
<tr><td>20.</td><td>E</td><td>The field <code>height</code> has been declared private, therefore, it cannot be directly accessed from client code that is in another class than <code>Box</code>.</td></tr>
<tr><td>21.</td><td>A</td><td>The matrix looks like this after the loops are done:
[0, 1, 2, 3]
[2, 3, 4, 5]
[4, 5, 6, 7]
[6, 7, 8, 9]
mat[2] is the third row down.</td></tr>
<tr><td>22.</td><td>E</td><td>\D{3} matches exactly 3 non-digits and \W{3} matches exactly 3 non-word characters.
.+ matches any character one or more times.
A[a-z]? matches a capital A followed by any lower case letter once or not at all.</td></tr>
<tr><td>23.</td><td>D</td><td>A method signature must contain a return type, name and parameter list if necessary. All parameters must have a type and name.</td></tr>
<tr><td>24.</td><td>D</td><td>A and B are correct. For answer choice C to be correct the last line should be:
<code>return n*fac(n-1).</code></td></tr>
<tr><td>25.</td><td>C</td><td>Example where n=4.192837 and r=3.
(int)(4.192837*Math(10,3)+0.5)/Math.pow(10,3)=
(int)(4.192837*1000.0+0.5)/1000.0=
(int)(4192.837+0.5)/1000.0=
(int)4193.337/1000.0=
4193/1000.0=
4.193</td></tr>
<tr><td>26.</td><td>E</td><td><code>nextInt(6)</code> will return a whole number between 0 (inclusive) and 6 (exclusive).
0 * 11 = 0.</td></tr>
</table>

27.	B	This is an insertion sort so we are getting the next element in the unsorted portion of the array then shifting elements to the right until we find the proper place for the unsorted element or when we reach the front of the array. Then the unsorted element is placed (inserted) into the proper location.
28.	C	See #29.
29.	C	Best Case O(n), Average Case $O(n^2)$, Worst Case $O(n^2)$
30.	D	i=1 [3, 5, 1, 0, 2, 4] i=2 [1, 3, 5, 0, 2, 4] i=3 [0, 1, 3, 5, 2, 4] i=4 [0, 1, 2, 3, 5, 4] i=5 [0, 1, 2, 3, 4, 5]
31.	B	The variables `g` and `h` in the client code are unchanged by the calls to the `doSomething` method so their final values are 4 and 0. Within the method, for this expression, `h+++g`, the increment operator is applied to the variable h like this: (h++)+g. Here is a trace of the variable values when the code has been run. g=1 h=2 g=3 h=4 g=7 h=6 13 g=2 h=2 g=4 h=4 g=8 h=6 14 g=3 h=2 g=5 h=4 g=9 h=6 15
32.	A	$14 = 1110_2$ $15 = 1111_2$ $16 = 10000_2$ 01111 & 10000 = 00000 00000 | 1110 = 1110 $1110_2 = 14$
33.	C	`compareTo` returns 0 if d1 and d2 are equal, a value less than 0 if d1 is less than d2, and a value greater than 0 if this d1 is greater than d2. `Double d2=19.00;` is allowed due to autoboxing.
34.	E	2147483647 is within the range of values for the int data type. The letter d following 250.84 designates the value as a double. It is optional. 0x designates a value as hexadecimal. Hexadecimals can be assigned to int type variables. 0b designates a binary number. 11111111 = 255. 255 is a valid ASCII value.
35.	A	When there is no break statement present execution of the code goes to the next case selector. When there is no code present after a case selector, execution goes to the next case selector.
36.	B	DeMorgan's Law states $\overline{A * B} = \bar{A} + \bar{B}$
37.	B	 is AND. is OR. is NOT. is XOR.
38.	D	Every pair of vertices are connected by an edge in a complete graph. A and D and D and B are not connected in answer choice B.
39.	20	* - + 8 5 3 2 = * - 13 3 2 = * 10 2 = 20
40.	-86	Take the complement of 10101010 to get 01010101 then add 1 to get 01010110 which is 86. We know the answer is negative since the sign bit was one so the final answer is -86.