1. **Which of the following Java operators can be used with boolean variables? (Choose all that apply.)**

    1. ==
    2. +
    3. --
    4. ¡
    5. %
    6. <=
    7. Cast with boolean

R = 1, 4 and 7. Boolean variables can only be compared with logical operators. Option 1 is the comparison operator and option 2 is an operator that is used as a complement to make a distinction between two values.

2. **What data type (or types) Will allow the following code snippet to compile? (choose all that apply.)**

    1. Byte apples = 5;
    2. Short oranges = 10;
    3. ___________ bananas = apples + oranges;

    1. Int
    2. Long
    3. Boolean
    4. Double
    5. Short
    6. Byte

R = 1, 2 and 4. The result of the operation has to be stored in a variable of type int or larger, int, long and boolean have a capacity of equal or larger size.

3. What change, when applied independently, would allow the following code snippet to compile?

```
3: long ear = 10;
4: int hearing = 2 * ear;
```

1. No change; it compiles as is.
2. Caste ear on line 4 to int.
3. Change the data type of ear on line 3 to short.
4. Cast 2 * ear on line 4 to int.

5. Change the data type of hearing on line 4 to short
6. Change the data type of hearing on line 4 to long

R = 2, 3, 4 and 6. Int is of type long, so it cannot be stored in an int variable, option 2 and 4 perform that task, option 3 stores the variable in a lower capacity data type and option 6 stores the result in a higher capacity variable.

4. What is the output of the following program?

```
1: public class CandyCounter {
2: static long addCandy(double fruit, float
vegetables) {
3: return (int)fruit+vegetables;
4: }
5:
6: public static void main(String[] args) {
7: System.out.print(addCandy(1.4, 2.4f) + "-
");
8: System.out.print(addCandy(1.9, (float)4)
+ "-");
9: System.out.print(addCandy((long)(int)
(short)2, (float)4)); } }
```

1. 4-6-6.0
2. 3-5-6
3. 3-6-6
4. 4-5-6
5. The code does not compile because of line 9.
6. None of the above

R = 6. The code will not compile because the cast that is applied to the fruit + vegetables operation must be applied to both variables.

5. What are the unique outputs of the following code snippet? (Choose all that apply.)

```
int a = 2, b = 4, c = 2;
System.out.println(a > 2 ? --c : b++);
System.out.println(b = (a!=c ? a : b++));
System.out.println(a > b ? b < c ? b : 2 : 1);
```

1. 1

2. 2
3. 3
4. 4
5. 5
6. 6
7. The code does not compile.

R = 1, 4 and 5. The result of the first printout will be 4, since the increment operator is after the variable, the result of the second printout will be 5 since the condition is false and the variable b was incremented by 1 in the previous line and the result of the last printout will be 1 since the first condition is not met, the second one will not be executed.

6. Given the following code snippet, what is the value of the variables after it is executed? (Choose all that apply.)

```
int ticketsTaken = 1;
int ticketsSold = 3;
ticketsSold += 1 + ticketsTaken++;
ticketsTaken *= 2;
ticketsSold += (long)1;
```

1. ticketsSold is 8
2. ticketsTaken is 2
3. ticketsSold is 6
4. ticketsTaken is 6
5. ticketsSold is 7
6. ticketsTaken is 4
7. The code does not compile.

R = 3 and 6. In the third line, the value of 1 is added to ticketSold and the value of tickesTaken, which is equal to 1, its value is equal to 5, and in the last line, 1 is added, resulting in 6, TicketsTaken is increased by 1 on the third line and multiplied by 2 on the next line.

7. What is the output of the following code snippet? (Choose all that apply.)

```
3: int temperature = 4;
4: long humidity = -temperature + temperature * 3;
5: if (temperature>=4)
6: if (humidity < 6) System.out.println("Too
Low");
7: else System.out.println("Just Right");
```

```
8: else System.out.println("Too High");
```

1. Too Low
2. Just Right
3. Too High
4. A NullPointerException is thrown at runtime.
5. The code will not compile because of line 7.
6. The code will not compile because of line 8.

R = 2. The humidity value is 12, in line 5 the condition is true since temperature is equal to 4, in line 6 the condition is false since humidity is not less than 6.

8. Which statements, when inserted independently into the following blank, will cause the code to print 2 at runtime? (Choose all that apply.)

```
int count = 0;
BUNNY: for(int row = 1; row <=3; row++)
RABBIT: for(int col = 0; col <3 ; col++) {
if((col + row) % 2 == 0)
_____________________;
count++;
}
System.out.println(count);
```

1. break BUNNY
2. break RABBIT
3. continue BUNNY
4. continue RABBIT
5. break
6. continue
7. None of the above, as the code contains a compiler error

R = 2, 3 and 5. All conditions will cause the second cycle to stop and can continue in the first cycle when the condition is true.

9. What is the output of the following code snippet?

```
2: boolean keepGoing = true;
3: int result = 15, meters = 10;
4: do {
5: meters--;
6: if(meters==8) keepGoing = false;
7: result -= 2;
8: } while keepGoing;
9: System.out.println(result);
```

1. 7
2. 9
3. 10
4. 11
5. 15
6. The code will not compile because of line 6.
7. The code does not compile for a different reason.

R = 7. The code will not compile because in line 8 keepGoing should be enclosed in parentheses

10. What is the output of the following code snippet? (Choose all that apply.)

```
9: int w = 0, r = 1;
10: String name = "";
11: while(w < 2) {
12: name += "A";
13: do {
14: name += "B";
15: if(name.length()>0) name += "C";
16: else break;
17: } while (r <=1);
18: r++; w++; }
19: System.out.println(name);
```

1. ABC
2. ABCABC
3. ABCABCABC
4. Line 15 contains a compilation error.
5. Line 18 contains a compilation error.
6. The code compiles but never terminates at runtime.
7. The code compiles but throws a NullPointerException at runtime.

R = 6. When the doWhile cycle is tried a second time r will be greater than 1 which will cause the cycle to never end.

11. What is output by the following code? (Choose all that apply.)

```
1: public class Fish {
2: public static void main(String[] args) {
3: int numFish = 4;
4: String fishType = "tuna";
5: String anotherFish = numFish + 1;
6: System.out.println(anotherFish + " " +
fishType);
7: System.out.println(numFish + " " + 1);
8: } }
```

1. 4 1
2. 5
3. 5 tuna
4. 5tuna
5. 51tuna
6. The code does not compile.

R = 6. The code will not compile because in line 5 you are trying to store an int variable in a String variable.

12. What is the result of the following code?

```
7: StringBuilder sb = new StringBuilder();
8: sb.append("aaa").insert(1, "bb").insert(4,
"ccc");
9: System.out.println(sb);
```

1. abbaaccc
2. abbaccca
3. bbaaaccc
4. bbaaccca
5. An empty line
6. The code does not compile.

R = 2. The string "aaa" will be added to the string, then "bb" will be inserted in position 1, and finally "ccc" will be inserted in position 4.

13. What is the result of the following code?

```
12: int count = 0;
13: String s1 = "java";
14: String s2 = "java";
15: StringBuilder s3 = new StringBuilder("java");
16: if (s1 == s2) count++;
17: if (s1.equals(s2)) count++;
18: if (s1 == s3) count++;
19: if (s1.equals(s3)) count++;
20: System.out.println(count);
```

1. 0
2. 1
3. 2
4. 3
5. 4
6. An exception is thrown.
7. The code does not compile.

R = 7. The comparison in line 18 cannot be performed with the equality operator.

14. What is the result of the following code?

```
public class Lion {
public void roar(String roar1, StringBuilder
roar2) {
roar1.concat("!!!");
roar2.append("!!!");
}
public static void main(String[] args) {
String roar1 = "roar";
StringBuilder roar2 = new
StringBuilder("roar");
new Lion().roar(roar1, roar2);
System.out.println(roar1 + " " + roar2);
} }
```

1. roar roar
2. roar roar!!!
3. roar!!! roar
4. roar!!! roar!!!
5. An exception is thrown.

6. The code does not compile.

R = 2. The String variable is immutable, a new object must be created to store the concatenation result.

15. Which of the following can replace line 4 to print "avaJ"? (Choose all that apply.)

```
3: var puzzle = new StringBuilder("Java");
4: // INSERT CODE HERE
5: System.out.println(puzzle);
```

1. puzzle.reverse();
2. puzzle.append("vaJ$").substring(0, 4);
3. puzzle.append("vaJ$").delete(0, 3).deleteCharAt(puzzle.length() - 1);
4. puzzle.append("vaJ$").delete(0, 3).deleteCharAt(puzzle.length());
5. None of the above

R = 1 and 3. Option 1 will display the string in reverse order, option 3 will append a string and remove the first 3 characters of the string, then remove the last character.