Computer Science Answer Key
UIL District 2 2014

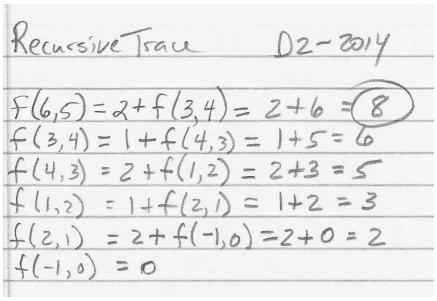
1) B	11) B	21) C	31) A
2) A	12) A	22) A	32) C
3) E	13) D	23) E	33) E
4) D	14) B	24) C	34) B
5) D	15) D	25) A	35) C
6) A	16) E	26) A	36) E
7) D	17) B	27) C	37) D
8) E	18) D	28) A	38) D
9) E	19) C	29) B	39) B
10) B	20) A	30) B	40) C

Note to Graders:

- All provided code segments are intended to be syntactically correct, unless otherwise stated (e.g. error is an answer). **Ignore any typographical errors.**
- Any necessary Standard Java 2 Packages are assumed to have been imported as needed.
- Assume any undefined (undeclared) variables have been defined as used.

Explanations:

1. $100010_2 + 100000_2 = 34_{10} + 32_{10} = 66_{10} = 102_8 = 42_{16} = 100010_2$
2. This is simple arithmetic. Just remember the data types for the output.
3. The first true result is obvious since both x and y reference the same object. For the **y=5** reassignment, there is a common memory section in Java for Strings and for smaller value integers that objects share when they are instantiated simply with the equals sign. Therefore, even though it looks like a separate object is created, it simply references the 5 that is in common memory, and therefore it is still pointing to the same memory location. However, when the **new** operator is used, a separate memory location is used, which results in **false** for the == operator.
4. Since the ++ is a post-increment operator, the value is output first, then the variable is incremented, with the result shown.
5. The **lastIndexOf** method is straight forward…the last index of the letter 'a' is in position 8 of the string.
6. Remembering that Java lists use zero based indexing (first element is in position zero), the elements in position 1 and 3 are 3 and 2, whose sum is 5.
7. The only way for the OR (||) operator to be false is when both Boolean values are false.
8. Both output statements are executed here, the first one because the ***if*** statement is true, and the second one regardless of the if statement since it is not attached to it, despite the indentation. The resulting output is simple math.
9. Right shift 2 is essentially dividing by 4 (2^2), and left shift 2 is multiplying by 4, with obvious results.
10. The floor function returns the nearest lower whole number value of the decimal, in this case, -6.0.
11. The traditional modifier method of classes starts with the word **set**, and in this case **setNumStrings** is the method to use, giving it the desired value as a parameter.
12. Similarly, the word **get** is the traditional prefix for accessor methods of instance variables, therefore **getType** is the one to use in this situation.
13. The toString method in this class definition lists the type first, followed by a colon, then the number of strings, and the word "string".
14. This is simple arithmetic. Nuff said.
15. This **showGrid** method outputs the entire matrix from bottom row to top, in right to left column order.
16. The value 260.0 divides into 130.0, 65.0, 32.5, 16.3, 8.13, 4.1, 2.03, 1.02, and finally 0.51, with 9 divisions. 250 requires only 8 divisions, and 600 requires 10.
17. The contents of the array at the start are: 0 5 2 0 0 0. After each loop iteration the contents are: 0 5 2 3 0 0, 0 5 2 3 -1 0, and 0 5 2 3 -1 4. Position 4 contains -1 at the conclusion of the method call.
18. The greatest value at the end is 5.
19. Since the length of the string is 20, the substring calls with 15, 5 and 10, and 7 and 12 all will return a string of length 5.
20. The expression **p xor q and p** simplifies to **p and not q**, which means the only true result is when p is true and q is false, indicated by 101 in the output. Using Boolean identities, the simplification sequence is as follows: **p^q&&p = p&&!(q&&p) || !p&&q&&p = p&&(!q||!p) = p&&!q**. You can also use the truth table process to evaluate this expression.
21. 42.0 % 13 results in 3.0, which is then incremented to become 4.0.
22. Decimal 10 in binary is 1010.
23. The short data type is stored in 16 bits of memory.
24. The natural log of E (**2.718281828459045**), the base of the natural logs, is 1.00.
25. The recursive trace for this question is shown on the right.
26. The binary representation for -1 is a string of 32 1s, which when right shifted 32 places circles back to the same 32 1s.
27. The split for this problem results in the following: `[IL, veA, a, ade]`, with a length of 4 and "`ade`" in position 3.
28. The base 5 equivalent of 34 is 114.
29. The replaceAll method does not change the existing String (Strings are immutable), but instead returns a new String with the modifications indicated. The original String w is not changed, however a new String **s** is created changing all 'n's to 'm's.



30. The A and B signals go into a NOR gate, which goes into an XOR gate with C, resulting in NOT(A OR B) XOR C.
31. This code is fine as is. Unlike the interface, the abstract class does not require the word `public` preceding the method name.
32. The call to methods A1 and A2 simply result in the output, "`I made a 240`".
33. The contents of the queue after each command are: [3], [3, 5], [3, 5, 9], [5, 9], [5, 9, 6], [9, 6], [6], [6, 2], [6, 2, 7], with 6 at the front.
34. The least efficient of these O(N) ratings is $O(N^2)$, which is typically characterized by some nested loop process, such as an insertion sort or bubble sort.
35. Although there are 16 words in this sentence, only 14 are unique, which is what this code does (sets have no duplicates).
36. The expression A AND B AND A OR 0 simplifies to just A AND B, since the repeated A dissolves into just one A, and the OR 0 is the identity rule and effectively disappears.
37. Since 97 is the ASCII value for lower case 'a', 100 represents 'd', which is where this diagonal of characters starts, producing the series "defgh".
38. The contents of this list after each command is as follows: [], [4], [4, 5], [4, 5, 6], [4, 5, 6, 5], [4, 5, 6, 5, 7], [5, 6, 5, 7].
39. To find out the number of 1s in this matrix, simply count the number of arrows, which is 6. Since it is a 4X4 matrix, which means 16 elements, the remaining 10 elements are zeroes.
40. This one is tricky. The first two statements in the p method actually effect the actual parameters, the lists x and y, since arrays are passed by reference, but the third statement (a=b) does not. Even though **a** is reassigned to reference the **b** list in the method, this does not make the original x reference change, therefore it still points to its original list. Here is the state of each list after each command.
    - x[0] = 10 y[0] = 5
    - a[0] = 15 b[0] = 5
    - a[0] = 10 b[0] = 10
    - x[0] = 15 y[0] = 10
    - a[0] = 25 b[0] = 15
    - a[0] = 10 b[0] = 10
    - x[0] = 10 y[0] = 25