Computer Science Contest #1213-02 Key

October 20, 2012

1) D

2) A

3) D

4) A

5) E

6) D

7) D

8) C

9) B

10) A

11) A

12) B

13) A

14) B

15) C

16) C

17) A

18) A

19) E

20) E

21) B

22) B

23) B

24) C

25) E

26) A

27) C

28) A

29) B

30) E

31) B

32) D

33) B

34) A

35) E

36) A

37) C

38) A

39) C

40) D

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.

Brief Explanations:

1. 10010010+1101 = 10011111. Since there are 8 bits, the first four bits can be represent by a hexadecimal number and so could the last four bits, 1001 is hexadecimal 9 and 1111 is hexadecimal F.
2./ must occur first before the other operations take place. Integer addition does not round.
3. you must evaluate everything to the right of the = before adding it to the variable on the left.
4. x will be calculated for i = 4, 3, 2, 1, and 0. x = 10+6+7+8+9+10 = 50.
5. substring(x,y) means to start at the x index and to stop at the y-1 index.
6. work your way from the inside out: index = n[1], then find k[index].
7. Boolean algebra. c = (a+b)ab+!ab(!a+!b) = aab+abb+!a!ab+!ab!b = ab+ab+!ab+0 = ab+!ab = b(a+!a) = b(1) = b. Hence, c = b.
8. Java 7 allows Strings in the switch statement. Hallelujah!
9. The only numbers possible are 0,1,2,3,4,and 5 because the (int) truncates and does not round.
10. to call a class method, use object.method(); Make sure you adhere to the parameter list. In this case, it is a null parameter list.
11. the sqrt function returns a double.
12. %4.2f means “set aside at least 4 spaces and print to the second decimal” while %05d means “set aside at least 5 spaces and print leading 0s”
13. ‘A’+’B’ is the first operation to be done. A char is considered an integer and is subject to integer addition. ‘A’ is integer value 65. ‘E’ is being added to a String and is treated as a String.
14. It is a 100 element 2D array with the [0][0] starting at 5 and increasing by one. The 5th row, 8th col is the 48th value generated. Remember we started at 5, so the 48th value is 47.
15. you are subtracting the summation of 1 to 25 = 25*24/2 = 300 from the number 50 which yields -250. But you are adding 5+10+15+20 = 50 to that number, which yields -200.
16. indexing starts at 0 so the number 2 is at index 4. Since the value is of type int, you must call Integer.parseInt() to change it from a String to an int.
17. 181 = 10110101

```
     64 = 01000000
      | = 11110101 = 245
```

18. Boolean algebra again :(ab+ac)abc = ababc + acabc = abc + abc = abc. Hence, all must be true. Learn Boolean algebra, it is worth it.
19. the array adds all the odd numbers from 1 to 39: 20 numbers total. It only permenantly removes 2 of them for a total of 18.
20. 2410%50 = 10. 10/3 = 3. This is integer division, so there is no decimals.
21. The loop ends when the value of x is greater than 99. So, any value over 99 will break the loop.
22. “Doctor Who”: s = ‘D’, e = ‘o’, m = ‘r’. So the word becomes “octor Who”. That would return ‘o’.
23. You would think that since no character is the same it should return ‘+’, but no word will ever go below a size of 2. That’s because e = m when the size is two and s will be returned.
24. You cannot user Warrior because constructors are not inherited. You must use super and have the proper parameter list that the parent class expects.
25. Conan’s attack is 15*2 = 30. Ryoko’s defense is 4*3 = 12. The damage is 18
26. Since Ryoko is instantiated as a Ninja, it overloads the attack method of the Warrior class. Hence when ryoko.attack is called, we use the Ninja attack method. Hence Ryoko’s attack is 56 and Conan’s defense is 15. The damage is 41
27. The size of the array is the number of elements in the array (not to be confused by the index values of the array).

28. An array cannot resize, that is an ArrayList, so the length stays the same.
29. This is a String building recursion in which the base case starts the String. So you must dig down to the base case and then add to the string as you climb up.
30. Did you notice that this is the decimal to binary recursive method? Good way of converting to binary?
31. We are working with a size 1/5 of the original size. The method is quadratic, so we must square the size reduction to 1/25. So the time will be 1/25th of the original time. 20/25 = 0.8 seconds.
32. A post-order traversal visits the left child, then the right child, then prints out the value. So the branches are printed before the parent node is printed.
33. A TreeSet holds its values according to its Comparator, in this case, alphabetically.
34. Iterator has an unusual instantiation in that it is actually a method call from the Collection class, hence object.setIterator();
35. You cannot add to the TreeSet and keep using the iterator. Once the Set is changed, a new iterator must be established. This is also true with an enhanced for loop. You can remove from the Set, because it is using the iterator to do it.
36. In a PriorityQueue, you always remove the smallest number. (11)
37. A PriorityQueue is a min heap, which means it is held as a balanced binary tree, with each parent smaller than its children.
38. x << y means to take x and divide it by 2^y.
39. split("g") creates an array that cuts the String up for by every "g", the "g" is removed.
40. the only time split will put a "" value at zero, is when the String starts with the value it is splitting on.