



University Interscholastic League Computer Science Competition

Number 151 (District 1 - 2015)

General Directions:

- 1) DO NOT OPEN EXAM UNTIL TOLD TO DO SO.**
- 2) NO CALCULATOR OF ANY KIND MAY BE USED.**
- 3) There are 40 questions on this contest exam. You have 45 minutes to complete this contest. If you are in the process of actually writing an answer when the signal to stop is given, you may finish writing that answer.
- 4) Papers may not be turned in until 45 minutes have elapsed. If you finish the test before the end of the allotted time, remain at your seat and retain your paper until told to do otherwise. Use this time to check your answers.
- 5) All answers must be written on the answer sheet/Scantron card provided. Indicate your answers in the appropriate blanks provided on the answer sheet or on the Scantron card. Clean erasures are necessary for accurate Scantron grading.
- 6) You may place as many notations as you desire anywhere on the test paper, but not on the answer sheet or Scantron card, which are reserved for answers only.
- 7) You may use additional scratch paper provided by the contest director.
- 8) All questions have ONE and only ONE correct (BEST) answer. There is a penalty for all incorrect answers.
- 9) A reference to commonly used Java classes is provided at the end of the test, and you may use this reference sheet during the contest. You may detach the reference sheets from the test booklet, but **DO NOT DO SO UNTIL THE CONTEST BEGINS.**

Scoring:

- 1) All questions will receive 6 points if answered correctly; no points will be given or subtracted if unanswered; 2 points will be deducted for an incorrect answer.

Note: Correct responses are based on Java, **J2sdk v 1.7.25**, from Sun Microsystems, Inc. All provided code segments are intended to be syntactically correct, unless otherwise stated (i. e. `error` is an answer choice) and any necessary Java 2 Standard Packages have been imported. Ignore any typographical errors and assume any undefined variables are defined as used. **For all output statements, assume that the `System` class has been statically imported... `import static java.lang.System.*`;**

QUESTION 1	
Which of these is NOT equivalent to $202_{10} + 10000101_2$?	
A. 335_{10} B. 517_8 C. $14F_{16}$ D. 101001011_2 E. All are	
QUESTION 2	
What is output by the code segment to the right?	<code>out.println(17 + 7 - 1 / 5);</code>
A. 4 B. 4.6 C. 18.2 D. 23.8 E. 24	
QUESTION 3	
What is output by the code segment to the right?	<code>out.println("A" + 10 + 50);</code>
A. A60 B. 6560 C. A1050 D. 651050 E. 125	
QUESTION 4	
What is output by the code segment to the right?	<code>String s = "abcbcabcbcabcbcabcb"; out.println(s.lastIndexOf("abc",10));</code>
A. 9 B. 10 C. 11 D. 18 E. 19	
QUESTION 5	
What is output by the code segment to the right?	<code>boolean p = true; boolean q = true; out.println(!p&&q);</code>
A. false B. true	
QUESTION 6	
What is output by the code segment to the right?	<code>int x = 64; out.println(Math.cbrt(x));</code>
A. 4 B. 4.0 C. 8 D. 8.0 E. 16	
QUESTION 7	
What is output by the code segment to the right?	<code>long j = -55; double d = -3.5; out.println(j%d);</code>
A. -2.5 B. 2.5 C. 15.7 D. -15.7 E. There is no output due to an error.	
QUESTION 8	
For which of these input values will the output of the code segment to the right be "yes"?	<code>int x = <input value>; if(!(x>20 x<10)) out.println("yes"); else out.println("no");</code>
I 9 II 10 III 11 IV 19 V 21	
A. I only B. V only C. I and V only D. II, III, and IV only E. I, II, III, IV, and V	
QUESTION 9	
How many stars will be output by the code segment to the right?	<code>int x = 4; String s = ""; do{ s+="*"; x+=2; }while(x%7!=0); out.println(s);</code>
A. 3 B. 4 C. 5 D. 6 E. 7	

<p>QUESTION 10</p> <p>What is output by the code segment to the right?</p> <p>A. 10 B. 106 C. 138 D. 202 E. 250</p>	<pre>char[] list=new char[5]; list[1]='1'; list[2]='2'; list[3]=3; list[4]=4; out.println(list[0]+list[1]+list[2]+list[3]+list[4]);</pre>
<p>QUESTION 11</p> <p>In the code segment to the right, which statement below must be placed in <code block> in order for this code segment to work properly?</p> <p>A. import java.io.*; B. import java.util.*; C. import static java.lang.System.*; D. import static java.lang.Math.*; E. More than one of these.</p>	<pre><code block> public class test{ public static void main (String [] args){ Scanner kb = new Scanner(System.in); } }</pre>
<p>QUESTION 12</p> <p>What is output by the code segment to the right?</p> <p>A. 10 55 B. 10 45 C. 11 45 D. 11 55 E. None of these</p>	<pre>int x = 100; int y = 1; for(;y<10;y++) x-=y; out.println(y+" "+x);</pre>
<p>QUESTION 13</p> <p>Here are three lines taken from the Java Order of Precedence chart. Which choice represents the correct order of precedence for these three lines?</p> <p>A. I, II, III B. III, II, I C. II, I, III D. I, III, II E. III, I, II</p>	<p>I. ^ II. III. ? :</p>
<p>QUESTION 14</p> <p>What is output by the code segment to the right?</p> <p>A. 8 B. 16 C. 32 D. 64 E. None of these</p>	<pre>out.println(Character.SIZE);</pre>
<p>QUESTION 15</p> <p>What is output by the code segment to the right?</p> <p>A. false false B. false true C. true false D. true true E. There is no output due to an error.</p>	<pre>int [] pList = {10,20,30,40,50}; ArrayList<Integer> aList = new ArrayList<Integer>(); for(int a:pList) aList.add(a/10); out.print(pList.contains(30)+ " "); out.println(aList.contains(30));</pre>
<p>QUESTION 16</p> <p>How many ordered pairs make this boolean expression false?</p> <p>A. 0 B. 1 C. 2 D. 3 E. 4</p>	$\overline{\overline{A} + B}$
<p>QUESTION 17</p> <p>What is output by the code to the right?</p> <p>A. 7 B. 7.0 C. 17 D. 17.0 E. There is no output due to an error.</p>	<pre>long j = 20; int k = -15; double p = 5; out.println(j+k/p);</pre>
<p>QUESTION 18</p> <p>What is output by the code segment to the right?</p> <p>A. 3 B. 4 C. 5 D. 6 E. 7</p>	<pre>int [][] grid = new int[5][4]; for(int r=1;r<grid.length;r++) for(int c=0;c<grid[r].length;c++) grid[r][c] = r+c; out.println(grid[3][2]);</pre>
<p>QUESTION 19</p> <p>Which of the following choices represents the decimal equivalent of the two's complement binary value 11001010?</p> <p>A. -51 B. -52 C. -53 D. -54 E. -55</p>	

QUESTION 20		
What is the output at the end of the third iteration in the method execution called by the client code to the right?		
A. 8 4 2 7 6 B. 2 4 8 7 6 C. 8 7 4 2 6 D. 2 4 7 8 6 E. 8 7 6 4 2		<pre>1 public static void mystD1(int[] list){ 2 for (int j = 1; j < list.length; j++){ 3 int temp = list[j]; 4 int i = j; 5 while (i > 0 && temp > list[i - 1]){ 6 list[i] = list[i - 1]; 7 i--; 8 } 9 list[i] = temp; 10 for(int x:list) 11 out.print(x+" "); 12 out.println(); 13 }</pre>
QUESTION 21		
What algorithm is represented by the method <code>mystD1</code> ?		
A. Insertion sort B. Selection sort C. Bubble sort D. Merge sort E. Quick sort		
QUESTION 22		
In which line of this method must a change be made so that the sorting order is reversed?		
A. Line 2 B. Line 4 C. Line 5 D. Line 6 E. Line 8		
QUESTION 23		
What is the least restrictive order of magnitude for the average case in the sort shown in the code to the right?		
A. $O(N)$ B. $O(N^2)$ C. $O(\log N)$ D. $O(N \log N)$ E. $O(1)$		<pre><client code> int [] list = {8,2,4,7,6}; mystD1(list);</pre>
QUESTION 24		
What is the output of the code segment to the right?		
A. 43210 6543210 10 3210 543210 B. 3210 543210 0 210 43210	C. EEEEEE GGGGGGG BB DDDD FFFFFFF D. EEEE GGGGGGG B DDD FFFFF E. There is no output due to an error.	<pre>String s = "EGBDF"; char [] list = s.toCharArray(); for(char a:list){ int x = a-64; while(x-->0) out.print(a); out.println(); }</pre>
QUESTION 25		
What is output by the code segment to the right?		
A. 30.0 B. 45.0 C. 60.0 D. 90.0 E. None of these		<pre>double val = 0.5; out.printf("%.1f\n", Math.toDegrees(Math.acos(val)));</pre>
QUESTION 26		
What is output by the code segment to the right?		
A. 1.23.456.095.4 B. 1.2 3.45 6.09 5.4 C. [1.2 3.45 6.09 5.4] D. [1.2, 3.45, 6.09, 5.4] E. There is no output due to an error		<pre>double [] list = {1.2,3.45,6.09,5.4}; out.println(Arrays.toString(list));</pre>

QUESTION 27

Which choice below represents the two terms listed in order to correctly replace <thing1> and <thing2> in the method definition to the right?

- A. int and return B. return and int
C. public and return D. public and int
E. return and public

```
static <thing1> mystD1 (int x){
    if(x%4==0)
        <thing2> x*5;
    <thing2> x;
}
```

QUESTION 28

Assuming <thing1> and <thing2> have been correctly replaced, what is the output of the client code to the right?

- A. 1 0 B. 1 40 C. 45 0 D. 9 8 E. 9 40

```
<client code>
out.print(mystD1(9)+ " ");
out.println(mystD1(8));
```

QUESTION 29

In the Dwelling class definition to the right, how many instance fields are there?

- A. 1 B. 2 C. 3 D. 4 E. 5

QUESTION 30

Using the Dwelling class definition to the right, with the block comment symbols (*/* */*) in place as shown, what is the output of the client code below?

```
Dwelling d = new Dwelling();
out.println(d);
```

- A. null 0 false
B. Dwelling@150bd4d
C. null 0 rooms not sturdy
D. : 0 rooms: not sturdy
E. null: 0 rooms: not sturdy

```
class Dwelling{
    private String type;
    private int numRooms;
    private boolean sturdy;
    /*
    public Dwelling(){
        type = "house";
        numRooms = 3;
        sturdy = true;
    }
    public Dwelling(String ty, int nr,
                    boolean st){
        type = ty;
        numRooms = nr;
        sturdy = st;
    }
    */
    public String toString(){
        return type+": "+numRooms
            +" rooms: " +
            (sturdy?"sturdy":"not sturdy");
    }
}
```

QUESTION 31

If the block comment symbols are removed in the Dwelling class definition to the right, what is the output of the client code below?

```
Dwelling d = new Dwelling();
out.println(d);
d = new Dwelling("tent",2,false);
out.println(d);
```

- A.
house: 3 rooms: sturdy
tent: 2 rooms: not sturdy
B.
Dwelling@150bd4d
Dwelling@1bc4459
C.
house: 3: true
tent: 2: false
D.
house: 3 rooms: true
tent: 2 rooms: false
E.
There is no output due to an error.

<p>QUESTION 32</p> <p>What is output by the code to the right?</p> <p>A. 2 B. 2.5 C. 22 D. 60</p> <p>E. There is no output due to an error</p>	<pre>out.println(Integer.toString(12,5));</pre>
<p>QUESTION 33</p> <p>Using the generic stack pseudocode on the right, what is the sum of all popped items after the push and pop sequence is complete?</p> <p>A. 14 B. 17 C. 18 D. 22 E. 25</p>	<pre>Push 3 Push 9 Push 6 Push 4 Pop x Pop x Push 5 Pop x Push 3 Pop x</pre>
<p>QUESTION 34</p> <p>What is the preorder traversal of the binary tree shown to the right?</p> <p>A. UIDRIICTLST B. RDIICITUSLT C. RIDCTIISTLU</p> <p>D. UILDISTRICKT E. ULTSIITCDIR</p>	
<p>QUESTION 35</p> <p>Which of the following is NOT a simple path in the graph shown to the right?</p> <p>I. ABEC II. EABEC III. DCBEA IV. DCBF</p> <p>A. I only B. II only C. IV only D. I and III only E. II and IV only</p>	
<p>QUESTION 36</p> <p>Which of the following logical statements is represented by the digital electronics diagram on the right?</p> <p>A. $\overline{A \oplus B} * C$ B. $A \oplus B * C$ C. $(A + B) * C$ D. $(A \oplus B) * C$ E. None of these</p>	
<p>QUESTION 37</p> <p>Which of the expressions below is the postfix equivalent to the expression shown?</p> <p>A. $- + I / N ^ T E L$ B. $I N T E ^ / - L +$ C. $I N T E ^ / L + -$ D. $- I + / N ^ T E L$ E. None of these</p>	$I + N / T ^ E - L$

QUESTION 38

Which of the following values is NOT a possible outcome of the code shown to the right?

A. 33 B. 35 C. 36 D. 37 E. 38

```
Random r = new Random();
out.print(r.nextInt(5)+33);
```

QUESTION 39**Free Response Question:**

Simplify the Boolean Algebra expression shown below as much as possible.

$$\overline{A * B + A + B}$$

QUESTION 40**Free Response Question:**

Find $f(5)$ according to the recursive function definition shown below.

$$f(5) =$$

$$f(x) = \begin{array}{ll} f(x-3)+1 & \text{when } x > 0 \\ 3 & \text{when } x = 0 \\ 2 & \text{when } x < 0 \end{array}$$

NO TEST MATERIAL ON THIS PAGE

Standard Classes and Interfaces — Supplemental Reference

class java.lang.Object

- o boolean equals(Object other)
- o String toString()
- o int hashCode()

interface java.lang.Comparable<T>

- o int compareTo(T other)
Return value < 0 if this is less than other.
Return value = 0 if this is equal to other.
Return value > 0 if this is greater than other.

class java.lang.Integer implements Comparable<Integer>

- o Integer(int value)
- o int intValue()
- o boolean equals(Object obj)
- o String toString()
- o int compareTo(Integer anotherInteger)
- o static int parseInt(String s)
- o static int parseInt(String s, int radix)

class java.lang.Double implements Comparable<Double>

- o Double(double value)
- o double doubleValue()
- o boolean equals(Object obj)
- o String toString()
- o int compareTo(Double anotherDouble)
- o static double parseDouble(String s)

class java.lang.String implements Comparable<String>

- o int compareTo(String anotherString)
- o boolean equals(Object obj)
- o int length()
- o String substring(int begin, int end)
Returns the substring starting at index begin and ending at index (end - 1).
- o String substring(int begin)
Returns substring(from, length()).
- o int indexOf(String str)
Returns the index within this string of the first occurrence of str. Returns -1 if str is not found.
- o int indexOf(String str, int fromIndex)
Returns the index within this string of the first occurrence of str, starting the search at the specified index.. Returns -1 if str is not found.
- o charAt(int index)
- o int indexOf(int ch)
- o int indexOf(int ch, int fromIndex)
- o String toLowerCase()
- o String toUpperCase()
- o String[] split(String regex)
- o boolean matches(String regex)

class java.lang.Character

- o static boolean isDigit(char ch)
- o static boolean isLetter(char ch)
- o static boolean isLetterOrDigit(char ch)
- o static boolean isLowerCase(char ch)
- o static boolean isUpperCase(char ch)
- o static char toUpperCase(char ch)
- o static char toLowerCase(char ch)

class java.lang.Math

- o static int abs(int a)
- o static double abs(double a)
- o static double pow(double base, double exponent)
- o static double sqrt(double a)
- o static double ceil(double a)
- o static double floor(double a)
- o static double min(double a, double b)
- o static double max(double a, double b)
- o static int min(int a, int b)
- o static int max(int a, int b)
- o static long round(double a)
- o static double random()
Returns a double value with a positive sign, greater than or equal to 0.0 and less than 1.0.

interface java.util.List<E>

- o boolean add(E e)
- o int size()
- o Iterator<E> iterator()
- o ListIterator<E> listIterator()
- o E get(int index)
- o E set(int index, E e)
Replaces the element at index with the object e.
- o void add(int index, E e)
Inserts the object e at position index, sliding elements at position index and higher to the right (adds 1 to their indices) and adjusts size.
- o E remove(int index)
Removes element from position index, sliding elements at position (index + 1) and higher to the left (subtracts 1 from their indices) and adjusts size.

class java.util.ArrayList<E> implements List<E>

class java.util.LinkedList<E> implements List<E>, Queue<E>

Methods in addition to the List methods:

- o void addFirst(E e)
- o void addLast(E e)
- o E getFirst()
- o E getLast()
- o E removeFirst()
- o E removeLast()

```

class java.util.Stack<E>
    o boolean isEmpty()
    o E peek()
    o E pop()
    o E push(E item)

interface java.util.Queue<E>
    o boolean add(E e)
    o boolean isEmpty()
    o E peek()
    o E remove()

class java.util.PriorityQueue<E>
    o boolean add(E e)
    o boolean isEmpty()
    o E peek()
    o E remove()

interface java.util.Set<E>
    o boolean add(E e)
    o boolean contains(Object obj)
    o boolean remove(Object obj)
    o int size()
    o Iterator<E> iterator()
    o boolean addAll(Collection<? extends E> c)
    o boolean removeAll(Collection<?> c)
    o boolean retainAll(Collection<?> c)

class java.util.HashSet<E> implements Set<E>

class java.util.TreeSet<E> implements Set<E>

interface java.util.Map<K,V>
    o Object put(K key, V value)
    o V get(Object key)
    o boolean containsKey(Object key)
    o int size()
    o Set<K> keySet()
    o Set<Map.Entry<K, V>> entrySet()

class java.util.HashMap<K,V> implements Map<K,V>

class java.util.TreeMap<K,V> implements Map<K,V>

interface java.util.Map.Entry<K,V>
    o K getKey()
    o V getValue()
    o V setValue(V value)

interface java.util.Iterator<E>
    o boolean hasNext()
    o E next()
    o void remove()

interface java.util.ListIterator<E> extends
    java.util.Iterator<E>
    Methods in addition to the Iterator methods:
    o void add(E e)
    o void set(E e)

```

```

class java.lang.Exception

```

- o Exception()
- o Exception(String message)

```

class java.util.Scanner

```

- o Scanner(InputStream source)
- o boolean hasNext()
- o boolean hasNextInt()
- o boolean hasNextDouble()
- o String next()
- o int nextInt()
- o double nextDouble()
- o String nextLine()
- o Scanner useDelimiter(String pattern)