University Interscholastic League
Computer Science Competition

Number 153 (Region - 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 $74_{16} + 1101010_2$?

A. 11011110_2 B. 336_8 C. 220_{10} D. DE_{16} E. All are equivalent

QUESTION 2

What is the result of the expression shown?

A. 43 B. 91 C. 9 D. 25 E. 0

```
out.println(21 / 3 * 5 + 8);
```

QUESTION 3

In an output statement such as the one shown below, which choice below correctly labels each part of the statement?

```
System.out.println("Hello");
```

A. `System`-object, `out`-class, `println`-method, `"Hello"`-parameter
B. `System`-class, `out`-method, `println`-object, `"Hello"`-parameter
C. `System`-class, `out`-object, `println`-parameter, `"Hello"`-method
D. `System`-class, `out`-object, `println`-method, `"Hello"`-parameter
E. `System`-parameter, `out`-object, `println`-method, `"Hello"`-class

QUESTION 4

What is output by the code segment below?

```
String s = "Sara's seven sisters slept soundly in sand.";
out.println(s.replace("s","th"));
```

A. `Sara'th seven sisters slept soundly in sand.`
B. `Thara's theven thithterth thlept thoundly in thand.`
C. `Thara's seven sisters slept soundly in sand.`
D. `Sara'th theven thithterth thlept thoundly in thand.`
E. There is no output due to an error.

QUESTION 5

What is output by the code segment to the right?

A. `false` B. `true`

```
boolean p = false;
boolean q = true;
out.println(p&&q&&!q||q);
```

QUESTION 6

What is output by the code segment to the right?

A. 3 B. 3.0 C. 1.5 D. 2.0
E. There is no output due to an error.

```
int x = 1000;
out.println(Math.log10(x));
```

QUESTION 7

What is output by the code to the right?

A. 66.7 B. 66 C. 34.7 D. B
E. There is no output due to an error.

```
float f = 33.3f;
char a = 'd';
out.println(a - f);
```

QUESTION 8

What is output by the code to the right?

A. 6
B. 18
C. 60
D. 48
E. 160

```
int a = 0;
for(int x=0;x<10;x++){
      switch(x%5) {
            case 0: a++;break;
            case 1: a-=2;
            case 2: a+=5;break;
            case 3: a--;
            default: a=a+a;
      }
}
out.println(a);
```

QUESTION 9

What is output by the code to the right?

A. 10 B. 8 C. 6
D. There no output due to an infinite loop
E. There no output due to an error

```
int a = 100;
while(a>10) a/=1.5;
out.println(a);
```

QUESTION 10

Which of the original elements of the array shown to the right is no longer in the array after execution of the code?

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

```
int [] list = {5,3,1,4,2};
list[3] = list[list[2]];
list[list[2]] = list[4];
```

QUESTION 11

Which of the statements below will NOT work properly when inserted into the code to the right?

I. `k.next().charAt(4);`
II. `k.nextChar(4);`
III. `k.nextLine().charAt(4);`

A. I only B. II only C. III only
D. I and III only E. All will work properly

```
String s = "grandbaby";
Scanner k = new Scanner(s);
char d = <statement>
```

QUESTION 12

What is output by the code segment to the right?

A. 10 4
B. 10 5
C. 11 3
D. 11 4
E. 9 6

```
String t = "";
String s = "state of texas";
do{
 int x = s.length()/2;
 t+=s.substring(x,x+1);
 s=s.substring(0,x)+s.substring(x+1);
}while(t.length()<10);
out.println(t.length()
      + " "+s.length());
```

QUESTION 13

What is output by the code segment to the right?

A. 4 B. 10 C. 20 D. 10true E. 20false

```
int x=2,y=5,z=7;
x+=x*y&z;
out.println(x);
```

QUESTION 14

What is output by the code segment to the right?

A. truetruetruetrue
B. truefalsetruefalse
C. truetruefalsefalse
D. truefalsefalsefalse
E. falsefalsefalsefalse

```
Integer x = 100;
Integer y = 100;
Integer z = new Integer(100);
out.print(x==y);
out.print(x==z);
x+=30;
y+=30;
out.print(x==y);
out.println(x==z);
```

QUESTION 15

What is output by the code segment to the right?

A. [2, 5] B. [3, 5, 7]

C. [5, 7, 2] D. [3, 5, 7, 2]

E. There is no output due to an error.

```
int [] list={9,3,5,7,2,4,1,6};
ArrayList<Integer> aList = new
ArrayList<Integer>();
for(int x:list)   aList.add(x);
out.println(aList.subList(2,5));
```

QUESTION 16

Which of the choices below shows the correct order of values to fill the blanks in the code to the right so that the output value is true?

A. 1 3 6 B. 1 6 3 C. 3 1 6

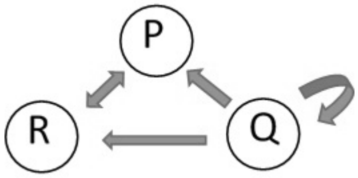
D. 3 6 1 E. 6 1 3

```
String s = "animaniacs";
String t = "amaniacal";
out.println(s.regionMatches
                  (___,t,___,___));
```

QUESTION 17

For the graph shown to the right, how many zeroes are in the adjacency matrix?

A. 2 B. 3 C. 4 D. 5 E. 6



QUESTION 18

What is output by the code to the right?

A. 240 B. 255 C. 00000000

D. 11110000 E. 11111111

```
int x = 15, y = 4;
String s;
S = Integer.toBinaryString(x<<y);
out.println(s);
```

QUESTION 19

How many ordered triples make this boolean expression false?

A. 2 B. 3 C. 4 D. 5 E. 6

$$(A + B) * (\bar{A} + \bar{C})$$

QUESTION 20

Which choice below represents the ordered pair of values that would correctly fill in the blanks in the code to the right to generate a random integer in the range of 27 to 56, inclusive?

A. 27 29 B. 27 56 C. 29 28

D. 29 30 E. 30 27

```
Random r = new Random();
out.println(r.nextInt(___)+___);
```

QUESTION 21

Which of the following choices is NOT true about the Java PriorityQueue class?

A. Based on a heap data structure

B. Allows insertion of null elements

C. Head of the queue is the least object

D. Dynamic, able to grow and shrink as needed

E. Uses natural order of objects based on Comparable interface

QUESTION 22

Which of the following choices will correctly instantiate a new static two-dimensional array of doubles as shown below?

```
1.2 3.4
5.6 7.8 9.0
8.7 6.5
4.3
```

A. `double [] dubs = {1.2 3.4 5.6 7.8 9.0 8.7 6.5 4.3};`

B. `double [] dubs = {1.2, 3.4, 5.6, 7.8, 9.0, 8.7, 6.5, 4.3};`

C. `double [][] dubs = {1.2, 3.4}, {5.6, 7.8, 9.0}, {8.7, 6.5}, {4.3};`

D. `double [][] dubs = {{1.2, 3.4}, {5.6, 7.8, 9.0}, {8.7, 6.5}, {4.3}};`

E. All will work properly

QUESTION 23

Which of these descriptions regarding the least restrictive running times of TreeMap and HashMap methods is NOT correct?

A. HashMap get - constant time B. HashMap put - constant time
C. TreeMap get - log(n) time D. TreeMap put - log(n) time
E. All are correct

QUESTION 24

Which of the expressions below is NOT equivalent to the expression shown on the right?

A. $(\overline{\overline{A} * B}) + (\overline{\overline{A + \overline{B}}})$

B. $(\overline{A} + \overline{B}) + (A + \overline{B})$

C. $A + \overline{B}$

D. $(\overline{\overline{A} * B}) + (A + \overline{B})$

E. $(\overline{\overline{A}} + \overline{B}) + (A + \overline{B})$

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

QUESTION 25

Each of four choices below matches one of the following traversals of the tree shown to the right: inorder, preorder, postorder, reverse order. Which choice does not match any of these traversals?

A. JZNLPAR B. ZAJRLPN C. ZARLPNJ
D. RAPLNZJ E. RPNLAJZ

```
      Z
     / \
    A   J
   / \
  R   L
     / \
    P   N
```

QUESTION 26

What is output by the code segment to the right?

A. 8.1341 B. 8.135
C. 43.1 D. 53.1341
E. 91.1

```
int x = 5;
char a = '1';
String s = "34";
double d = 3.1;
out.println(x+d+s+a);
```

QUESTION 27

Which choice below correctly replaces `<blank>` in the code to the right?

A. String B. int C. char
D. double E. void

QUESTION 28

Assuming the <blank> in the method to the right has been replaced correctly, what is output by the client code to the right?

A. 18 B. 18.0 C. 117
D. 117.0 E. There is no output due to an error.

```
static <blank> mystR15(int x,int y){
  return Math.sqrt(x)+Math.cbrt(y);
}
//client code
out.println(mystR15(121,343));
```

QUESTION 29

Which of the following statements is NOT correct about the Java collections Set, HashSet, and TreeSet?

A. A `Set` is a collection that contains no duplicate elements.
B. A `Set` uses the `equals` and `hashCode` methods to determine element duplication.
C. A `HashSet` uses the `equals` and `hashCode` methods to determine element duplication.
D. A `TreeSet` uses the `equals` and `hashCode` methods to determine element duplication.
E. All of these statements are true.

QUESTION 30

Regarding the code to the right, which statement below best describes the implementation of class Two?

A. Class Two must implement the `alpha()` method.
B. Class Two must override the `beta()` method.
C. Class Two must implement `alpha()` and override `beta()`
D. Class Two is not required to do anything special.
E. This class structure is invalid due to an error in the code.

QUESTION 31

Assuming all is well with this class structure with any possible error having been corrected, and the alpha method having been defined to return the value 9, what is the output for the client code to the right?

A. `1.09B`
B. `1.0966`
C. `10.0B`
D. `76.0`
E. There is no output due to an error.

QUESTION 32

Which choice below is NOT be a valid set of statements for the implementation of Class Two in the code to the right?

A.
```
int alpha(){return 9;}
```

B.
```
String beta(){return "B";}
```

C.
```
int alpha(){return 9;}
String beta(){return "B";}
```

D.
```
int alpha(){return 10;}
String beta(){return "C";}
```

E. More than one of these is invalid.

```
abstract class One{
 double d = 1.0;
 abstract int alpha();
 String beta(){return "B";}
}
class Two extends One{
//    <implementation>
}
//client code
Two fer = new Two();
out.println(fer.d+fer.alpha()
           +fer.beta());
```

QUESTION 33

Which of the following choices best describes the quick sort algorithm?

A. Relies on a partition operation in which a pivot value is selected, and all elements smaller than the pivot are moved before it and all greater elements are moved after it.

B. Continuously divides a list into halves until each half is only one element, and then builds the halves back in sorted order until the entire list is sorted.

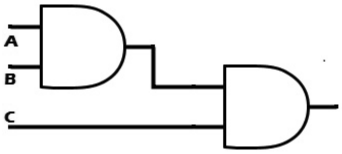
C. Works by taking elements from the list one by one and placing them in their correct position into a new sorted list.

D. Starts at the beginning of the array and finds the best value for each position, swaps it with the value in the current position, and repeats these steps for the remainder of the list.

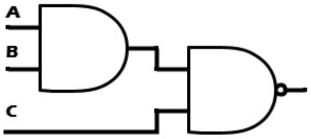
E. Starts at the beginning of the data set and compares the first two elements, and if the first is greater than the second, it swaps them. It continues doing this for each pair of adjacent elements to the end of the data set. It then starts again with the first two elements, repeating until no swaps have occurred on the last pass.

QUESTION 34

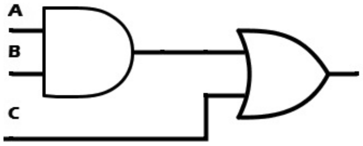
Which diagram below represents the logical statement on the right? A + B + C



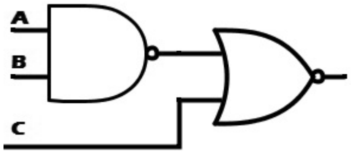
A.



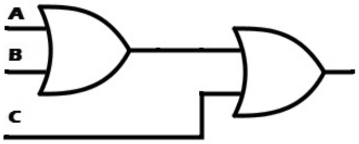
B.



C.



D.



E.

QUESTION 35

Which of the following choices represents the decimal equivalent of the two's complement binary value 10011101?

A. -99 B. -101 C. -103 D. -105 E. -107

QUESTION 36

What is output by the code segment shown?

A. 4 B. 5 C. 6
D. 7 E. 8

```
LinkedList<Integer> list = new
LinkedList<Integer>();
list.add(4);
list.add(5);
list.addFirst(6);
list.addLast(7);
list.add(8);
list.pop();
out.println(list.pop());
```

QUESTION 37

What is output by the code segment to the right?

A. 40 B. 41 C. 63
D. 64 E. There is no output due to an error.

```
String s = "";
for(int x=1;x<40;x*=2)
      for(int y=1;y<=x;y++)
            s+='*';
out.println(s.length());
```

QUESTION 38

What is output by the code to the right?

A. 1 2 3 4 5 7 7 9
B. 1 5 9 7 2 4 3 7
C. 2 3 4 7 7 9 1 5
D. 9 7 1 5 2 4 3 7
E. 9 2 4 5 7 3 1 7

```
int [] list = {9,7,5,2,4,3,1,7};
Arrays.sort(list,1,5);
for(int x:list)
   out.print(x+" ");
```

QUESTION 39

Free Response Question:
Convert the expression below into the equivalent postfix expression.

- + / S A M * ^ S U ^ N G

QUESTION 40

Free Response Question:

Find f(3,5) using the function definition shown below.

```
f(x,y) = x              when y = 1
       = x * f(x,y-1)   when y > 1

f(3,5) = __________
```

Standard Classes and Interfaces — Supplemental Reference

class java.lang.Object

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

interface java.lang.Comparable<T>

- 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>

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

class java.lang.Double implements Comparable<Double>

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

class java.lang.String implements Comparable<String>

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

class java.lang.Character

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

class java.lang.Math

- static int abs(int a)
- static double abs(double a)
- static double pow(double base, double exponent)
- static double sqrt(double a)
- static double ceil(double a)
- static double floor(double a)
- static double min(double a, double b)
- static double max(double a, double b)
- static int min(int a, in b)
- static int max(int a, int b)
- static long round(double a)
- 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>

- boolean add(E e)
- int size()
- Iterator<E> iterator()
- ListIterator<E> listIterator()
- E get(int index)
- E set(int index, E e)
  Replaces the element at index with the object e.
- 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.
- 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:

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

class java.util.Stack<E>

- boolean isEmpty()
- E peek()
- E pop()
- E push(E item)

interface java.util.Queue<E>

- boolean add(E e)
- boolean isEmpty()
- E peek()
- E remove()

class java.util.PriorityQueue<E>

- boolean add(E e)
- boolean isEmpty()
- E peek()
- E remove()

interface java.util.Set<E>

- boolean add(E e)
- boolean contains(Object obj)
- boolean remove(Object obj)
- int size()
- Iterator<E> iterator()
- boolean addAll(Collection<? extends E> c)
- boolean removeAll(Collection<?> c)
- 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>

- Object put(K key, V value)
- V get(Object key)
- boolean containsKey(Object key)
- int size()
- Set<K> keySet()
- 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>

- K getKey()
- V getValue()
- V setValue(V value)

interface java.util.Iterator<E>

- boolean hasNext()
- E next()
- void remove()

interface java.util.ListIterator<E> extends java.util.Iterator<E>

Methods in addition to the Iterator methods:

- void add(E e)
- void set(E e)

class java.lang.Exception

- Exception()
- Exception(String message)

class java.util.Scanner

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