University Interscholastic League

Computer Science Competition

Number 128 (District 2 - 2011)

General Directions (Please read carefully!):

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. **All provided code segments are intended to be syntactically correct, unless otherwise stated. Ignore any typographical errors and assume any undefined variables are defined as used.**

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.

10) Assume that any necessary import statements for standard Java packages and classes (e.g. `.util`, `ArrayList`, etc.) are included in any programs or code segments that refer to methods from these classes and packages.

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.

What is the sum of $A56_{16}$ and $2D6_{16}$?

  A. $D32_{16}$ B. $C3B_{16}$ C. $D2C_{16}$ D. $B2B_{16}$ E. $1A12_{16}$

QUESTION 2

What is output by the code to the right?

  A.   330   B.   660   C.   1

  D.   0.66   E.   0

```
double a = 0.0033;
double b = a * 10 * (10 * 2);
System.out.print( (int) b );
```

QUESTION 3

What is output by the code to the right?

  A.   -12   B.   -1   C.   0

  D.   12   E.   13

```
int val = 0;
for(int i = 1; i < 13; i++)
   val--;
System.out.print(val);
```

QUESTION 4

What is output by the code to the right?

  A.   C%E&&R-F   B.   c%e&&r-f

  C.   c5e77r-f   D.   cerf

  E.   C%e&&r-F

```
String ts = "C%e&&r-F";
System.out.print(ts.toLowerCase());
```

QUESTION 5

What is output by the code to the right?

  A.   4A   B.   5null   C.   4C

  D.   4n   E.   4null

```
String[] lets = {"A", "C", null, "A", "D"};
System.out.print(lets.length);
System.out.print(lets[2]);
```

QUESTION 6

What is output by the code to the right?

  A.   0   B.   4   C.   6

  D.   20   E.   22

```
int x2 = 2;
int y2 = 3;
int z2 = x2 + y2 * 10 * 100 / 3 / 100 * 2;
System.out.print(z2);
```

QUESTION 7

Which answer is logically equivalent to the following `boolean` expression, where `w`, `x`, `y`, and `z` are `int` variables?

```
!((x != y) || (w >= z))
```

A. `(x == y) && (w < z)`   B. `(x != y) && (w >= z)`   C. `(x == y) || (w < z)`

D. `(x == y) || (w <= z)`   E. `!(x != y) || !(w == z)`

What is output by the code to the right?

A. 1

B. 14

C. 24

D. 34

E. 4

```
int x3 = 14;
int y3 = 14;
int z3 = x3 / y3 * x3;
if( x3 > y3 && x3 > z3 )
  System.out.print(1);
else if( y3 > z3 && y3 > x3 )
  System.out.print(2);
else if( z3 > y3 && z3 > x3 )
  System.out.print(3);
System.out.print(4);
```

What replaces **<*1>** in the code to the right so that the instance variable named `dir` in the `Critter` object being instantiated is set to the value stored in the parameter named `dir`?

A. `Critter.dir = dir`

B. `this.dir = dir`

C. `int dir = dir`

D. `dir = dir`

E. `static int dir = dir`

Assume **<*1>** is filled in correctly.

```
public class Critter{
  public static final int NORTH = 0;
  public static final int EAST = 1;
  public static final int SOUTH = 2;
  public static final int WEST = 3;

  private int dir;

  public Critter(int dir){
    <*1>;
  }

  public int move(){
    if(dir == EAST)
      dir = SOUTH;
    else
      dir = EAST;
   return dir;
  }
}

// client code
Critter c1 = new Critter();
c1.move();
c1.move();
System.out.print(c1.dir);
```

What is output by the client code to the right?

A. -1

B. 0

C. 3

D. There is no output due to syntax errors in the client code.

E. There is no output due to a runtime error caused by the client code.

What is output by the code to the right?

A. 0   B. 1   C. 49

D. 73   E. 122

```
int m = 73;
int n = 49;
System.out.print(m & n);
```

What is output by the code to the right?

A. 3   B. 3.3   C. 3.33

D. -3   E. -3.33

```
int m2 = 10 / -3;
System.out.print(Math.abs(-m2));
```

What is output by the code to the right?

A. CC++   B.   C
    PHP   C++
                        PHP

C. CC++PHP   D.   C
                        C++PHP

E. CBPHP

```
System.out.print("C");
System.out.print("C++");
System.out.print("\nPHP");
```

What is output by the code to the right?

A. +1638   B. +16384   C. 20000

D. +16380   E. 1638

```
System.out.printf("%+4d", 16384);
```

What is returned by the method call run("ABA")?

A. 3   B. 6   C. 9

D. 12   E. 15

```
public int run(String s){
  int x = s.length();
  s = s + s;
  int y = s.length();
  return x + y;
}
```

What is output by the code to the right?

A. 1   B. 11   C. 18

D. 28   E. 40

```
String stars = "";
for(int i = 0; i < 7; i++)
  for(int j = 0; j < 4; j++)
    stars = stars + "*";
System.out.print(stars.length());
```

What is output by the code to the right?

A. 4   B. 5   C. 6

D. 7   E. 13

```
String raw = "%xx*xx$a++u^^";
String[] dats = raw.split("\\W+");
System.out.print(dats.length);
```

Which of the following can replace **<*1>** in the code to the right so that the output is 5?

I.   name.indexOf("R") == 1
II. name.length() == 14
III. name != null

A. I only   B. II only   C. III only

D. I and II only   E. II and III only

```
String name = "Richard_M._Karp";
int x = (<*1>) ? 5 : 0;
System.out.print(x);
```

What is output by the code to the right?

A. 22   B. 21   C. 20

D. 1   E. 0

```java
int vv = 0;
for(int i = 0; i >= 20; i++)
  vv++;
System.out.print(vv);
```

Method `_AWFUL` to the right will not compile due to a single syntax error. Which of the following best describes the syntax error in the method?

A. `_AWFUL` is not a valid method name.

B. `x * 5` is not a valid length for an array.

C. `for(;;)` is not a valid for loop.

D. `short` is not a valid variable name.

E. The `if` statement requires braces.

```java
public boolean[] _AWFUL(int x, String s) {
  boolean flags[] = new boolean[x * 5];
  int short = 0;
  for(;;) {
     flags[short] = s.charAt(short++) > 50;
     if(short >= s.length())
       break;
  }
  return flags;
}
```

What is output by the code to the right?

A. 1   B. 1234   C. 123455

D. There is no output due to a syntax error.

E. There is no output due to a runtime error.

```java
String k = "1234.55";
Scanner sc = new Scanner(k);
System.out.print(sc.nextInt());
```

Which of the following is not a class in the Java Class Library?

A. Structure   B. String   C. List   D. Double   E. Map

What is the largest value that can be output by method `showval` to the right?

A. 127   B. 128   C. 255

D. 256   E. 2147483647

```java
public void showVal(byte b) {
  System.out.print(b);
}
```

Consider method `logic` to the right. When will the following conditions be true at the point marked `// question 24`?

	x == 20	y >= 0
A.	never	never
B.	never	sometimes
C.	sometimes	sometimes
D.	sometimes	always
E.	always	always

```java
public int logic(int x, int y) {
  if(x != 5 || x != 7)
    x = 20;
  else
    x *= 2;
  if(-100 < y && y < 100)
    y = y * y;
  else
    y = 1234;

  // question 24

  return x * y;
}
```

Method `search` to the right is a flawed implementation of the binary search algorithm. The method does not always work as intended. Which of the following changes must be made so that the method correctly implements the binary search algorithm?

A. change `while( r < 0 && s < e)`
    to `while( r < 0 && s <= e)`

B. change `int m = (s + e) / 2`
    to `int m = s + e / 2`

C. change `int e = data.length - 1`
    to `int e = data.length + 1`

D. change `return r`
    to `return s`

E. change `if( data[m] == t )`
    to `if( data[m] == r )`

Assume method `search` has been corrected.

Which of the following is required as a pre condition so that method `search` works as intended and does not cause a runtime error? Pre conditions are the things that must be true before a method is called for the method to work.

A. `data != null`

B. `data != null and the elements in
    data are ascending order`

C. `data != null and data.length > 0`

D. `data != null and t != 0`

E. `data != null and t != -1`

```
/* pre: question 26
   post: return an index of data such that
   data[return value] == t or -1 if t is
   not present in data.
*/
public int search(int[] data, int t) {
  int s = 0;
  int e = data.length - 1;
  int r = -1;
  while(r < 0 && s < e) {
    int m = (s + e) / 2;
    if( data[m] == t )
      r = m;
    else if(t > data[m])
      s = m + 1;
    else
      e = m - 1;
  }
  return r;
}
```

What is output by the code to the right?

A. r   B. o   C. 114

D. s   E. R

```
String winner = "AlanEmerson";
System.out.print(winner.charAt(8));
```

What is output by the code to the right?

A. 4   B. -4.0   C. -3.0

D. 0   E. 3

```
double neg = -1.7;
neg *= 2;
System.out.print(Math.ceil(neg));
```

What replaces **<*1>** in the code to the right so that the variable `it` refers to an `Iterator` object for the `ArrayList v`?

A.   `new Iterator<Integer>()`

B.   `new Iterator<Integer>(v)`

C.   `v.iterator<Integer>()`

D.   `v.iterator()`

E.   `new Iterator()`

Assume **<*1>** is filled in correctly.

What is output by the code to the right?

A.   `1210`   B.   `57`   C.   `512710`

D.   `107125`   E.   `5710`

```java
int[] data = {5, 12, 7, 10};
List<Integer> v = new ArrayList<Integer>();
for(int x : data)
  v.add(x);

Iterator<Integer> it = <*1>;
while(it.hasNext()) {
  int temp = it.next();
  if(temp < 10) {
    System.out.print(temp);
    it.remove();
  }
}
```

Which of the following can replace **<*1>** in the code to the right so that the value stored in `p` is doubled?

```
I.    p *= 2
II.   p = p * p
III.  p = p * 2
```

A.   I only   B.   II only   C.   III only

D.   I and III only   E.   I, II, and III

Assume **<*1>** is filled in correctly.

What sorting algorithm does method `sort` implement?

A.   heap sort

B.   quick sort

C.   selection sort

D.   radix sort

E.   insertion sort

```java
// pre: all elements in vals >= 0
public void sort(int[] vals) {
  int[] w = new int[vals.length];
  int[] h = new int[vals.length];
  int p = 1;
  for(int i = 0; i < 31; i++) {
    int w_i = 0, h_i = 0;
    for(int j = 0; j < vals.length; j++) {
      int d = vals[j] & p;
      if(d == 0)
        w[w_i++] = vals[j];
      else
        h[h_i++] = vals[j];
    }
    int i_o = 0;
    for(int j = 0; j < w_i; j++)
      vals[i_o++] = w[j];
    for(int j = 0; j < h_i; j++)
      vals[i_o++] = h[j];
    <*1>;
  }
}
```

What is the order of method `addAll` to the right if `s` is the following type of `Set`? `vals` contains N distinct values. Pick the most restrictive, correct set of values.
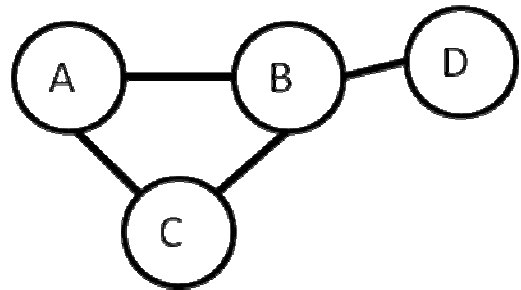
	HashSet	TreeSet
A.	O(N)	O(N)
B.	O(N)	O(NlogN)
C.	O(1)	O(logN)
D.	O(N)	O(N^2)
E.	O(NlogN)	O(N^2)

```
// pre: s.size() == 0
public void addAll(Set<Integer> s,
                              int[] vals) {
  for(int elem : vals)
    s.add(elem);
}
```

What kind of graph does the picture to the right represent?

A.   a directed unweighted graph

B.   a directed weighted graph

C.   an undirected unweighted graph

D.   an undirected weighted graph

E.   a binary search tree

What replaces **<*1>** in method `isVowel` to the right so that the method returns true if the char `ch` is equal to 'a', 'e' 'i', 'o', or 'u'?

A.   "aeiou".indexOf(ch) != -1

B.   ch == 'a' || 'e' || 'i' || 'o' || 'u'

C.   ch == "aeiou"

D.   ch ||= "aeiou"

E.   ch.equals("aeiou")

```
public boolean isVowel(char ch) {
  return <*1>;
}

public void sample() {
  String st = "abracadadra";
  int track = 0;
  for(int i = 0; i < st.length(); i++) {
    if(isVowel(st.charAt(i)))
      continue;
    track += i;
    i += 2;
  }
  System.out.print(track);
}
```

Assume **<*1>** is filled in correctly.

What is output when by the code to the right when method `sample` is called?

A.   0   B.   3   C.   13

D.   25   E.   30

What is output by the code to the right?

A.   2600   B.   2500   C.   260

D.   27   E.   26

```
Set<Character> vals;
vals = new HashSet<Character>();
for(int i = 0; i < 26; i++)
  for(int j = 0; j < 10; j++)
    vals.add( (char) (i + 'a') );
System.out.print(vals.size());
```

QUESTION 38

Given the class `Elem` to the right, what is output by the following client code?

```
Elem e1 = new Elem("X", null);
Elem e2 = new Elem("Y", e1);
Elem e3 = new Elem("Z", e2);
e1.n = e3;
Elem t = e2;
for(int i = 0; i < 32; i++)
  t = t.n;
System.out.print(t.d);
```

A.   e1

B.   e2

C.   X

D.   Y

E.   Z

QUESTION 39

Given the classes `Elem` and `Structure` to the right, what is output by the following client code?

```
Structure st = new Structure();
st.add(37, 0);
st.add(42, 0);
st.add(13, 1);
st.add(17, 0);
System.out.print(st.size() + " " +
  st.get(2));
```

A.   2 42

B.   4 17

C.   4 13

D.   4 37

E.   4 42

QUESTION 40

What type of data structure do the `Structure` and `Elem` classes implement?

A.   an array based list

B.   a heap

C.   a queue

D.   a binary search tree

E.   a linked list

```
public class Elem {
  public Elem n;
  public Object d;

  public Elem(Object dd, Elem nn) {
    n = nn;
    d = dd;
  }
}

public class Structure {

  private Elem f = new Elem(null, null);
  private int s = 0;

  public void add(Object obj, int pos) {
    Elem t = getTo(pos - 1);
    t.n = new Elem(obj, t.n);
    s++;
  }

  public void remove(int pos) {
    Elem t = getTo(pos - 1);
    t.n = t.n.n;
    s--;
  }

  public Object get(int pos) {
    return getTo(pos).d;
  }

  public int size() {
    return s;
  }

  public Elem getTo(int pos) {
    Elem t = f;
    for(int i = 0; i < pos; i++, t = t.n);
    return t;
  }
}
```

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)

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, in 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.lang.Exception

```
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)
```

```
    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)
```

Computer Science Answer Key
UIL District 2 2011

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

Notes:

The clause "Choose the most restrictive correct answer." is necessary because per the formal definition of Big O, an algorithm that is $O(N^2)$ is also $O(N^3)$, $O(N^4)$, and so forth.

10. The syntax errors are: client code (code external to the class) may not access `private` instance variables and the `Critter` class does not have a default constructor. The implicit default constructor is not available if there are any explicit constructors.

24. The boolean expressions `x != 5 || x != 7` is always true so `x` will always be set to `20`.