University Interscholastic League

Computer Science Competition

Number 114 (Invitational B - 2009)

General Directions (Please read carefully!):

1) DO NOT OPEN EXAM UNTIL TOLD TO DO SO.

2) **NO CALCULATORS 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.

QUESTION 1

What is the sum of DAD_{16} and 715_{16}?

A. $13C3_{16}$ B. $1FCF_{16}$ C. $14C2_{16}$ D. $4B2_{16}$ E. 2128_{16}

QUESTION 2

What is output by the code to the right?

A. 15 B. 9 C. 12

D. 11 E. 125

```
int x = 3;
int y = 2;
int z = x + y * 3;
System.out.println( z );
```

QUESTION 3

What is output by the code to the right?

A. 9 B. 0 C. 10

D. -11 E. 11

```
int tot = 0;
for(int i = 10; i > 0; i--){
  tot++;
}
System.out.print( tot );
```

QUESTION 4

What is output by the code to the right?

A. lsual B. sisua C. usualc

D. lsualc E. usual

```
String la = "visualc++";
String next = la.substring(2, 6);
next = la.charAt( 3 ) + next;
System.out.print( next );
```

QUESTION 5

What is output by the code to the right?

A. null B. 0 C. 1

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

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

```
int[] vals = new int[10];
System.out.print( vals[0] );
```

QUESTION 6

What is output by the code to the right?

A. 0 B. 28 C. 6

D. 4 E. 0.21428571428571427

```
int r = 6;
int s = 28;
System.out.println( r % s );
```

QUESTION 7

Which answer is logically equivalent to the following `boolean` expression, where `p` and `q` are `boolean` variables?

```
!p && !q
```

A. `!(q || p)` B. `!(p && q)` C. `!q && p` D. `!p || !q` E. `!!p && !!q`

QUESTION 8

What is output by the code to the right?

A. 23 B. 1 C. 123

D. 2 E. 13

```
String n = "kuipers";
if( Character.isLetter(n.charAt(1) ) )
  System.out.print( 1 );
else if ( n.length() > 4 )
  System.out.print( 2 );
if( n != null )
  System.out.print( 3 );
```

QUESTION 9

What replaces **<*1>** in the code to the right so that other classes do not have access to the instance variables `freq` and `letter`?

A. `public`

B. `private`

C. `public static`

D. `private static`

E. `private static final`

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

```
public class Count{
  <*1> int freq;
  <*1> char letter;

  public Count(char let){
    letter = let;
    freq = 0;
  }

  public String toString(){
    return letter + ":" + freq;
  }
}

////////////////////////////////////////
// client code
Count c1 = new Count('A');
System.out.print( c1 );
```

QUESTION 10

What is output by the client code to the right?

A. A:0 B. 0:A C. A0

D. 65:0 E. The output cannot be determined until runtime.

QUESTION 11

What is output by the code to the right?

A. 31 B. 15 C. 16

D. 1 E. 47

```
int m = 29;
int n = 18;
System.out.print( m & n );
```

QUESTION 12

What are the possible values `res` will store after the code to the right is executed?

A. `-2`

B. `-2, -1, 0, 1, 2`

C. `-2, -1, 0, 1`

D. `-3, -2, -1, 0 , 1, 2`

E. `-3, -2, -1, 0`

```
double init = Math.random();
int res = (int)(init * 4) - 2;
```

QUESTION 13

How many lines of output does the code to the right produce?

A. 5 B. 4 C. 6

D. 1 E. 2

```
System.out.println("ABBA\nStar");
System.out.println("Roll\nBe");
```

QUESTION 14

What is output by the code to the right?

A. `true` B. `false` C. `0`

D. `1` E. `12`

```
System.out.printf("%b", 12);
```

QUESTION 15

What is returned by the method call `myst(4, 2)`?

A. `3` B. `2` C. `-1`

D. `5` E. `4`

```
public int myst(int x, int y){
  x++;
  y--;
  x -= y;
  return x;
}
```

QUESTION 16

What is output by the code to the right?

A. `2a` B. `1A` C. `1a`

D. `2A` E. `297`

```
char ch = 'A';
if( Character.isDigit( ch ) )
  System.out.print( 1 );
else
  System.out.print( 2 );
System.out.print(Character.toUpperCase(ch));
```

QUESTION 17

What is output by the code to the right?

A. `6` B. `8.0`

C. `8` D. `7`

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

```
double a = 2.5;
a *= 3;
int x = (int)a;
System.out.print( x );
```

QUESTION 18

What is output by the code to the right?

A. `true` B. `false` C. `null`

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

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

```
ArrayList<String> list1, list2;
list1 = new ArrayList<String>();
list2 = new ArrayList<String>();
list1.add("Glenn");
list2.add( list1.get(0) );
System.out.print( list1 == list2 );
```

QUESTION 19

What is output by the client code to the right?

A. `3` B. `-13` C. `-6`

D. `-2` E. `0`

QUESTION 20

If a section of client code does not meet the precondition of method `comp`, but is otherwise syntactically correct, which of the following is true?

A. The client code will not compile.

B. `comp` will **always** return 0.

C. `comp` will **never** generate a runtime error.

D. `comp` will **sometimes** generate a runtime error.

E. `comp` will **always** generate a runtime error.

```
// pre: dt1.length == dt2.length
public int comp(int[] dt1, int[] dt2){
  int total = dt1[0] - dt2[0];
  int index = 1;
  while( total > 0 && index < dt1.length){
    total += (dt1[index] - dt2[index]);
    index++;
  }
  return total;
}

//client code
int[] arr1 = {10, 4, 8, 3, 12};
int[] arr2 = {4, 2, 5, 16, 7};
System.out.println( comp(arr1, arr2) );
```

QUESTION 21

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

A. 3 B. 4 C. 7

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

E. There is no output due to an infinite loop that occurs when method `show` is called.

```
// all three methods are part of
// the same class.

public int red(int x, int y){
  return red(y) + red(x);
}

public int red(int a){
  return a / 3;
}

public void show(){
  int y = 7;
  System.out.print( red(y, y) );
}
```

QUESTION 22

Which of the following best describes the programming language feature demonstrated by the two methods named `red`?

A. inheritance

B. recursion

C. method overriding

D. polymorphism

E. method overloading

QUESTION 23

If the parameter `s1` contains the values `[1, 2, 3]` and the parameter `s2` contains the values `[1, 2, 4]`, what values are in the `Set` returned by method `demo`?

A. `[1, 1, 2, 2, 3, 4]`

B. `[1, 2, 3]`

C. `[1, 2, 4]`

D. `[1, 2]`

E. `[1, 2, 3, 4]`

```
public Set<Integer> demo(Set<Integer> s1,
                         Set<Integer> s2){
  Set<Integer> result;
  result = new HashSet<Integer>();
  result.addAll( s1 );
  result.addAll( s2 );
  return result;
}
```

QUESTION 24

What is output by the code to the right?

A. `0.0 0.0 0.0 2.0`

B. `-0.7 0.0 0.7 2.5`

C. `0.0 0.0 0.0 0.0`

D. `2.5 0.7 0.0 -0.7`

E. `0.7 -0.7 2.5 0.0`

```
double[] nums = {.7, -.7, 2.5, 0.0};
Arrays.sort( nums );
for( double d : nums )
  System.out.print( d + " " );
```

QUESTION 25

What is output by the code to the right?

A. `[M, G, B]` B. `[G, M, B]`

C. `[B, M, G]` D. `[B, G, M]`

E. `[G, B, M]`

```
LinkedList<String> sample;
sample = new LinkedList<String>();
sample.addFirst("M");
sample.add(0, "B");
sample.addFirst("G");
System.out.print( sample.toString() );
```

QUESTION 26

What is output by the code to the right?

A. 6.0 B. 5.5

C. 5.0 D. 5

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

```
double p = 2.5;
int m = 3;
p += m;
System.out.print( p );
```

QUESTION 27

What is output by the code to the right?

A. 12.7 B. 9.4

C. 7.0 D. 90.0

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

```
String start = "12.7 9.4 90";
String[] elems = start.split("\\s+");
double d;
d = Double.parseDouble( elems[1] );
System.out.print( d );
```

QUESTION 28

Methods `search` and `helper` attempt to implement the binary search algorithm, but there is a logic error in method `helper` that causes the method to return an incorrect value in some situations. Which of the following best describes how to correct the logic error?

A. Replace the line
```
if( s <= e ){
```
with the following
```
if( s < e ){
```

B. Replace the line
```
int m = (s + e) / 2;
```
with the following
```
int m = (s + e) * 2;
```

C. Replace the line
```
else if( data[m] > t )
```
with the following
```
else if( data[m] >= t )
```

D. Replace the line
```
return helper(data, t, 0, m - 1);
```
with the following
```
return
    helper(data, t, s, m - 1);
```

E. Replace the line
```
return helper(data, t, m + 1, e);
```
with the following
```
return helper(data, m + 1, t, e);
```

```
// pre: the elements of data
// are sorted in ascending order
// post: return an index in data that
// contains tgt. return -1 if tgt is
// not present
public int search(int[] data, int tgt){
  int e = data.length - 1;
  return helper(data, tgt, 0, e);
}

private int helper(int[] data, int t,
                   int s, int e){
  if( s <= e ){
    int m = (s + e) / 2;
    if( data[m] == t )
      return m;
    else if( data[m] > t )
      return helper(data, t, 0, m - 1);
    else
      return helper(data, t, m + 1, e);
  }
  else
    return -1;
}
```

QUESTION 29

Assume the logic error in method `search` in question 28 has been corrected. Which of the following best describes what kind of method `helper` is?

A. A class method B. An iterative method C. A constant method

D. An accessor method E. A recursive method

QUESTION 30

What is output by the code to the right?

A. `trivial simple concat add`

B. `concat add trivial simple`

C. `trivial easy simple concat add`

D. `concat add trivial easy simp`

E. `add concat easy simple trivial`

```
TreeMap<String, String> translate;
translate = new TreeMap<String, String>();
translate.put("trivial", "easy");
translate.put("concat", "add");
translate.put("trivial", "simple");
for(Map.Entry<String, String> ent :
                    translate.entrySet() ){
    System.out.print( ent.getKey() + " ");
    String temp = ent.getValue();
    System.out.print( temp + " " );
}
```

QUESTION 31

What is returned by the method call `progress(mat, 2, 1)` where `mat` is the 2D array below?

10	2	8	10	9	5
9	4	3	2	9	1
6	2	0	6	0	0
4	7	3	2	5	12
7	7	4	2	1	4
11	4	12	1	7	3
8	4	0	8	1	3

A. 24 B. 26 C. 27

D. 17 E. 8

```
public int progress(int[][] mat,
                    int r, int c){
  int total = 0;
  int rowLim = mat.length;
  int colLim = mat[0].length;
  while( r < rowLim && c < colLim ){
    total += mat[r][c];
    if( mat[r][c] % 2 == 0 )
      r++;
    else
      c++;
  }
  return total;
}
```

QUESTION 32

Which sorting algorithm do methods `swap` and `sort` implement?

A. `quicksort` B. `insertion sort`

C. `bubble sort` D. `merge sort`

E. `selection sort`

QUESTION 33

Assume in the initial call to method `sort` the parameter `list` contains N unique elements already sorted in ascending order, where N = `list.length`. What is the Big O of method `sort` in that case? Choose the most restrictive correct answer.

A. O(NlogN) B. $O(N^{3/2})$ C. O(1)

D. $O(N^2)$ E. O(N)

```
public void swap(int[] list, int i, int j){
  int temp = list[i];
  list[i] = list[j];
  list[j] = temp;
}

public void sort(int[] list,
                 int st, int end){
  if( st >= end )
    return;
  int p = (st + end) / 2;
  swap(list, p, st);
  int j = st;
  for(int i = st + 1; i <= end; i++){
    if( list[i] <= list[st] ){
      j++;
      swap(list, i, j);
    }
  }

  swap(list, st, j);
  sort(list, st, j - 1);
  sort(list, j + 1, end);
}
```

QUESTION 34

What is output by the following client code?

```
Structure s1 = new Structure();
System.out.print( s1.isEmpty() );
```

A. `false`

B. `true`

C. 0

D. 1

E. The output cannot be determined until runtime.

QUESTION 35

What is output by the following client code?

```
Structure s2 = new Structure();
s2.add(2);
s2.add(7);
s2.add(5);
while( !s2.isEmpty() )
  System.out.print( s2.remove() + " " );
```

A. 2 5 7

B. 7 5 2

C. 2 7 5

D. 5 7 2

E. 7 2 5

QUESTION 36

What type of data structure does the `Structure` class implement?

A. A list

B. A queue

C. A stack

D. A max heap

E. A priority queue

```
public class Structure{
  public static final int CAP = 10;

  private Object[] con;
  private int f;
  private int b;
  private int size;

  public Structure(){
    con = new Object[CAP];
    b = -1;
  }

  public void add(Object obj){
    size++;
    if( size == con.length  )
      resize();
    b = (b + 1) % con.length;
    con[b] = obj;
  }

  public Object get(){
    return con[f];
  }

  public Object remove(){
    size--;
    Object result = con[f];
    f = (f + 1) % con.length;
    return result;
  }

  public boolean isEmpty(){
    return size == 0;
  }

  private void resize(){
    Object[] temp = new Object[size * 2];
    int org = f;
    for(int i = 0; i < size; i++){
      temp[i] = con[org];
      org = (org + 1) % con.length;
    }
    f = 0;
    b = size - 1;
    con = temp;
  }
}
```

QUESTION 37

Assume the method `sample(int[] data)` is $O(N^2)$ where N = `data.length`. When the method `sample` is passed an array with `length` = 2,000 it takes 1 second for method `sample` to complete. If method `sample` is then passed an array with `length` = 8,000 what is the expected time it will take method `sample` to complete?

A. 1 second B. 2 seconds C. 4 seconds D. 8 seconds E. 16 seconds

QUESTION 38

What replaces **<*1>** and **<*2>** in the code to the right so that it compiles with no syntax errors?

	<*1>	<*2>
A.	ListIterator	iterator
B.	Iterator	iterator
C.	ListIterator	listIterator
D.	Iterator	listIterator

E. None of these are correct.

```
public void check(ArrayList<String> arr){
  <*1><String> it;
  it = arr.<*2>() ;
  String temp;
  while( it.hasNext() ){
    temp = it.next();
    if( temp.length() > 5 )
      it.set( temp.toUpperCase() );
  }
}
```

QUESTION 39

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

A. 1 8 4 B. 1 8 3

C. 0 8 4 D. 0 16 4

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

```
public void trace(){
  int x = 10;
  int y = 1;
  for(int i = 0; i < 3; i++){
    x /= 2;
    y *= 2;
  }
  System.out.print( x + " " + y + " " + i);
}
```

QUESTION 40

How many * are output by the code to the right?

A. 0 B. 1 C. 3

D. 150 E. 165

```
for(int i = 1; i <= 10; i++)
  for(int j = 0; j < i; j++)
    for(int k = 0; k < 3; k++)
      System.out.print('*');
```

No material on this page.

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)

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()

class java.util.ArrayList<E> implements List<E>
Methods in addition to the List methods:
- 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.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)

Computer Science Answer Key
UIL Invitational B 2009

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

14. Explanation of the conversion flag `b`: "If the argument *arg* [the second argument] is `null`, then the result is `"false"`. If *arg* is a `boolean` or `Boolean`, then the result is the string returned by `String.valueOf()`. Otherwise, the result is `"true"`. "

20. If the arrays are different sizes an `ArrayIndexOutOfBoundsException` may occur, but not always. In some cases `comp` will run without failure and return a negative number. Consider if `dt1` is `{5, 5}` and `dt2` is `{10}`. `comp` would return `-5` without suffering a runtime error.

33. Method `sort` avoids the worst case for quicksort given values already sorted by picking the middle element of the unsorted portion as the pivot, instead of the first or last element.

39. A syntax error occurs because the last `println` statement attempts to reference the variable `i` which is no longer in scope.