University Interscholastic League

Computer Science Competition

Number 129 (Regional - 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.

QUESTION 1

What does BAD$_{16}$ minus 100101111101$_{2}$ equal?

A. 32F$_{16}$ B. 2FF$_{16}$ C. 230$_{16}$ D. 23F$_{16}$ E. 430$_{16}$

QUESTION 2

What is output by the code to the right?

A. 2.5 B. 3.25 C. 5.0

D. 5.5 E. 5.83333333

```
int x = 10;
int y = 3;
double a = x / y + x / (3.0 + 1.0);
System.out.print(a);
```

QUESTION 3

What is output by the code to the right?

A. 8 B. 12 C. 16

D. 32 E. 64

```
int prod = 1;
for(int i = 1; i < 5; i++)
  prod *= 2;
System.out.print(prod);
```

QUESTION 4

What is output by the code to the right?

A. -1 B. 0 C. 4

D. 5 E. 12

```
String d = "ABBAABABABAAABABA";
System.out.print(d.indexOf("ABAB", 7));
```

QUESTION 5

What is output by the code to the right?

A. -1 B. 0 C. 4

D. 5 E. 14

```
int[] first = {2, 3, 2, 12, 3};
first = new int[4];
first[2] += first[3];
System.out.print(first[2]);
```

QUESTION 6

What is output by the code to the right?

A. -117 B. -53 C. 11

D. 127 E. 137

```
byte b1 = (byte) -128;
b1++;
b1 += 10;
System.out.print(b1);
```

QUESTION 7

How many combinations of values for the variables p, q, and r will result in s being set to true?

A. 13 B. 7 C. 5

D. 3 E. 1

```
boolean p, q, r;
//code to initialize p, q, and r

boolean s = p && (q || !r);
```

QUESTION 8

What is output by the code to the right?

A. `11` B. `12` C. `21`

D. `22` E. `1`

```
boolean flag = false;
String name = "Alice";
if(!flag && name.contains("ice"))
  System.out.print(1);
else
  System.out.print(2);
if(flag || name.length() == -10)
  System.out.print(1);
else
  System.out.print(2);
```

QUESTION 9

What replaces **`<*1>`** in the code to the right to redirect to the other constructor in the `Play` class with arguments `"run"` and `3`?

A. `super("run", 3)`

B. `Play("run", 3)`

C. `Play(run, 3)`

D. `this("run", 3)`

E. `System.Play("run", 3)`

```
public class Play{
  private String type;
  private int yards;

  public Play(){ <*1>; }

  public Play(String t, int y){
    type = t;
    yards = y;
  }

  public boolean gain(){
    return yards > 0;
  }

  public String toString(){
    return type + yards + gain();
  }
}

// client code
Play p1 = new Play();
System.out.print(p1);
```

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

QUESTION 10

What is output by the client code to the right?

A. `null0false`

B. `null0true`

C. `run30`

D. `run31`

E. `run3true`

QUESTION 11

What is output by the code to the right?

A. `6` B. `4` C. `2`

D. `1` E. `0`

```
int m = 101;
int n = 99;
System.out.print(m ^ n);
```

QUESTION 12

What is output by the code to the right?

A. `10` B. `2` C. `5`

D. `0` E. `7`

```
int m2 = Math.max(Math.min(5, 2),
                          Math.min(10, 7));
System.out.print(m2);
```

QUESTION 13

What is output by the code to the right?

A. `ICODE"` B. `I''CODE''"`

C. `I'CODE'""` D. `I'CODE'`

E. There is no output due to a syntax error in the code.

```
System.out.print("I''CODE''\"");
```

QUESTION 14

What is output by the code to the right?

A. `5 5` B. `10 10` C. `10 5`

D. `5.0 10.0` E. `5 10`

```
String f = "%2$d %1$d";
System.out.printf(f, 5, 10);
```

QUESTION 15

What is returned by the method call `manip(3, 2.5)`?

A. `9.0` B. `14.0` C. `15.25`

D. `20.0` E. `22.5`

```
public double manip(double a, double b){
  a++;
  b++;
  return a * b;
}
```

QUESTION 16

What is output by the code to the right?

A. `72` B. `30` C. `10`

D. `5` E. `1`

```
String stars = "";
for(int i = 0; i < 2; i++)
  for(int j = 0; j < 3; j++)
    for(int k = 0; k < 5; k++)
      stars = stars + "*";
System.out.println(stars.length());
```

QUESTION 17

Which of the following replaces **<*1>** in the code to the right so that the output is 128?

A. `x1 = x1 * x1`

B. `x1 = x1 << 3`

C. `x1++`

D. `x1 += 10`

E. None of answers A through D produces output of `128`.

```
int x1 = 1;
while(x1 < 100) {
  <*1>;
}
System.out.println(x1);
```

QUESTION 18

Which `boolean` expression does the truth table to the right represent? `p`, `q`, and `r` are `boolean` variables.

A. `r = p && q` B. `r = !(p && q)`

C. `r = p || q` D. `r = !(p || q)`

E. `r = (p || q) && !(p && q)`

p	q	r
false	false	false
false	true	true
true	false	true
true	true	false

QUESTION 19

Which of the following best explains why the class to the right will not compile?

A. The `GPA` class does not implement a `compareTo` method that returns an `int`.

B. The `GPA` class does not have a default constructor.

C. The instance variable `credits` may not be `public`.

D. The clause `implements Comparable<GPA>` must be changed to `implements Comparable<T>` .

E. The `GPA` class does must state which class it `extends`.

```
public class GPA implements Comparable<GPA>{
  public int credits;
  private int gradePoints;

  public void takeClass(int gp) {
    credits += 3;
    gradePoints += gp;
  }

  public int getValue() {
    return gradePoints / credits;
  }
}
```

QUESTION 20

What is output by the code to the right?

A. `true true true`

B. `false false true`

C. `false true false`

D. `true false false`

E. `false true true`

```
Object[] vals = {new Object(), "S", "I"};
boolean[] res = new boolean[3];
res[0] = vals[0] instanceof String;
res[1] = vals[1] instanceof Comparable<?>;
res[2] = vals[2] instanceof Object;
for(boolean b : res)
  System.out.print(b + " ");
```

QUESTION 21

What is output by method `sample` if the argument `sc` is an `ArrayList<String>` equal to `["A", "ZZ", "A", "BB", "CC"]`?

A. `[ZZ, A, ZZ, ZZ, A, ZZ, BB, CC]`

B. `[A, ZZ, A, BB, CC]`

C. `[A, ZZ, A, BB, CC, ZZ, ZZ]`

D. `[A, ZZ, ZZ, A, ZZ, BB, CC]`

E. There is no output due to an infinite loop.

```
public void sample(ArrayList<String> sc) {
  ListIterator<String> it;
  it = sc.listIterator();
  while(it.hasNext())
    if(it.next().length() < 2)
      it.add(sc.get(1));
  System.out.print(sc);
}
```

QUESTION 22

What is output when the method call `theta(new int[]{-5, -3, -8});` is made?

A. `-5`   B. `-3`   C. `-8`

D. `-5-3`   E. `-3-8`

QUESTION 23

What is output when the method call `theta(new int[]{12, 15});` is made?

A. `-5`   B. `-3`

C. `-8`   D. `-5-3`

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

```
public void theta(int[] v) {
  if(v[0] > v[1] && v[0] > v[2])
    System.out.print(v[0]);
  else if(v[1] > v[0] && v[1] > v[2])
    System.out.print(v[1]);
  else
    System.out.print(v[2]);
}
```

QUESTION 24

What is output by the code to the right?

A. 121

B. 222

C. 212

D. 1212

E. 2121

```
String[] ns = {"mitra", "scroggs",
                       "lehmann", "scott"};

for(int i = 1; i < ns.length; i++)
  if(ns[i - 1].compareTo(ns[i]) < 0)
    System.out.print(1);
  else
    System.out.print(2);
```

QUESTION 25

Which of the following lines of code contains a syntax error?

A. `double z = 3.7 % 1.3;` B. `long x4 = 5647;` C. `double d5 = 1.7 & 5.6;`

D. `int _x = 0;` E. `int x6 = ~(-10);`

QUESTION 26

What is output by the code to the right?

A. [A, 1, 42, A]

B. [42, A, 3.7, B, 42]

C. [A, B, 42, 3.7, A]

D. [A, B, 42, A]

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

```
ArrayList<Object> objs;
objs = new ArrayList<Object>();
objs.add(42);
objs.add(0, "A");
objs.add(3.7);
objs.add(1, "B");
objs.set(objs.size() - 1, objs.get(0));
System.out.print(objs);
```

QUESTION 27

An array with 1,000,000 distinct `ints` in random order is passed to a method that uses the heapsort algorithm, it takes 4 seconds for the method to complete. What is the expected time for the method to complete when sorting an array with 4,000,000 distinct `ints` in random order?

QUESTION THROWN OUT. Correct answer is 17.6 seconds which is not present.

A. 1 second B. 2 seconds C. 3.5 seconds D. 4.4 seconds E. 64 seconds

QUESTION 28

Which of the following is not a valid Java identifier?

A. `small_name` B. `B1234567` C. `b555555555` D. `CardType` E. `big+name`

QUESTION 29

Which of the following lines of code contains a syntax error?

A. `Object var1 = new LinkedList<Integer>();`

B. `Comparable<Integer> var = new Integer(133);`

C. `ArrayList<Object> var3 = new Object();`

D. `String var4 = "KellyJ".substring(3);`

E. `Queue<String> var5 = new LinkedList<String>();`

QUESTION 30

What is returned by the method call `tough(2, 3)`?

A. `1` B. `3` C. `5`

D. `17` E. `19`

```
public int tough(int x, int y){
  if(x <= 0 || y <= 0)
    return 1;
  else
    return 1 + tough(x - 1, y) +
      tough(x, y - 1);
}
```

QUESTION 31

What is output by the code to the right?

A. `[2, 4, 22, 37, 42]`

B. `[42, 22, 4, 2, 37]`

C. `[42, 22, 42, 4, 2, 37, 42]`

D. `[2, 4, 22, 37, 42, 42, 42]`

E. The output will vary from one run of the program to the next.

```
TreeSet<Integer> ts;
ts = new TreeSet<Integer>();
int[] input = {42, 22, 42, 4, 2, 37, 42};
for(int i : input)
  ts.add(i);
System.out.print(ts);
```

QUESTION 32

What is output by method `getR` to the right when the method call `getR(new double[1000])` is made?

A. `9` B. `10001` C. `1023`

D. `2048` E. `1000000`

QUESTION 33

What is the order (Big O) of method `getR` to the right? N = `org.length`. Assume the `Math.random()` method is O(1). Pick the most restrictive correct answer.

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

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

```
public double[] getR(double[] org) {
  int in = 0;
  int c = 0;
  int s1 = org.length;
  double [] r = new double [1];
  while(r.length < org.length) {
    if(in == r.length) {
      int s2 = r.length * 2;
      double[] temp = new double[s2];
      for(int i = 0; i < r.length; i++) {
        temp[i] = r[i];
        c++;
      }
      r = temp;
    }
    r[in++] = (int) (Math.random() * s1);
  }
  System.out.print(c);
  return r;
}
```

GO ON TO THE NEXT PAGE.

Question 34

What is output by the code to the right? Method `toHexString` returns a string representation of the integer argument as an unsigned integer in base 16 with hexadecimal digits `0123456789abcdef`.

A. `0` B. `1` C. `2`

D. `3` E. `4`

```
int[] nums = {3003, 1600, 1117, 546, 21};
int c = 0;
for(int i : nums) {
   String sh = Integer.toHexString(i);
   int j = 0;
   for(; j < sh.length(); j++)
      if(sh.charAt(j) != sh.charAt(0))
         break;
   c += j != sh.length() ? 0 : 1;
   // question 35
}
System.out.println(c);
```

Question 35

Consider the code to the right. When will the following conditions be `true` at the point marked `// question 35`?

	`sh.equals("45d")`	`i > j`
A.	never	never
B.	never	sometimes
C.	sometimes	sometimes
D.	sometimes	always
E.	always	always

Question 36

What is output by the code to the right?

A. `-3`

B. `363`

C. `36-3`

D. `-36333`

E. `-363`

```
Stack<Integer> stk = new Stack<Integer>();
stk.push(3);
stk.push(stk.peek() + stk.peek());
stk.push(-3);
while(!stk.isEmpty())
  System.out.print(stk.pop());
```

GO ON TO THE NEXT PAGE.

Question 37

What is output by the following client code?

```
Structure<String> str1;
String[] strData = {"A","B","C","D"};
str1 = new Structure<String>(strData);
System.out.println(str1.spaceUsed());
```

A. 20
B. 24
C. 26
D. 30
E. 64

Question 38

What is output by the line marked `// q38` when the following client code is executed?

```
Structure<String> str2;
String[] data =
          {"A","B","C","D","E","F","G"};

str2 = new Structure<String>(data);
String[] ls = {"GGGCFFF", "AEFBBDA"};
for(int i = 0; i < ls[0].length(); i++)
  str2.addE(ls[0].charAt(i) + "",
                  ls[1].charAt(i) + "");

Object[] t1 = str2.con("C", "A");
Object[] t2 = str2.con("D", "G");
String q38 = t1[0] + " " + t2[0];
String q39 = t1[1] + " " + t2[1];
System.out.println(q38); // q38
System.out.println(q39); // q39
```

A. 2 3
B. 3 2
C. 0 0
D. 1 2
E. 2 1

Question 39

What is output by the line marked `// q39` when the client code from question 38 is executed?

A. 0 0
B. false false
C. false true
D. true false
E. true true

Question 40

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

A. A binary search tree
B. A heap
C. A weighted, directed graph
D. An unweighted, undirected graph
E. A stack

```
public class Structure<E> {
  private boolean[][] es;
  private List<E> d;

  public Structure(E[] v) {
    es = new boolean[v.length][v.length];
    d = new ArrayList<E>();
    for(E e : v)
      d.add(e);
  }

  public int spaceUsed() {
    int total = d.size();
    total += es.length * es.length;
    return total;
  }

  public void addE(E v1, E v2) {
    int[] vs = {loc(v1), loc(v2)};
    es[vs[0]][vs[1]] = true;
    es[vs[1]][vs[0]] = true;
  }

  private int loc(E v) {
    return d.indexOf(v);
  }

  public Object[] con(E v1, E v2) {
      List<Integer> b;
      b = new ArrayList<Integer>();
      int[] num = {0};
      int[] vs = {loc(v1), loc(v2)};
      boolean result = h(vs, b, num);
      return new Object[]{num[0], result};
  }

  private boolean h(int[] vs,
              List<Integer> b, int[] num) {
    num[0]++;
    if(es[vs[0]][vs[1]]) return true;
    b.add(vs[0]);
    for(int i = 0; i < d.size(); i++) {
      if(es[vs[0]][i] && !b.contains(i))
        if(h(new int[]{i, vs[1]}, b, num))
          return true;
    }
    return false;
  }
}
```

No Test 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()`
- `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)

Computer Science Answer Key
UIL Regional 2011

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

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.

27. The heapsort algorithm is O(NlogN). If the amount of data is doubled, the time will go up a little more than a factor of 4.