University Interscholastic League

Computer Science Competition

Number 138 (Invitational B - 2013)

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. **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 output by the code to the right?

A. `0`

B. `6`

C. `0.25`

D. `6.25`

E. `6.0`

```
System.out.println((double) (5/20) + 6);
```

QUESTION 2

What is output by the code to the right?

A. `0`

B. `14`

C. `58`

D. `3`

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

```
System.out.println(29/3%3*2);
```

QUESTION 3

What is the value of $3A9_{16} + 28B_{16}$?

A. 634_{16} B. $69A_{16}$ C. 794_{16} D. 745_{16} E. 640_{16}

QUESTION 4

What are the values of `x` and `y` after the code to the right has been executed?

A. `x = 25`
`y = 40`

B. `x = 1`
`y = 100`

C. `x = 25`
`y = 450`

D. `x = 25`
`y = 30`

E. `x = 1`
`y = 60`

```
int x = 25;
int y = 15;
if(x >= 10) {
  if(x <= 50) {
    y = x*4;
    if(y <= 100) {
      x %= 4;
    }
    else y = 40;
  }
}
else y *= x+5;
```

QUESTION 5

What is output by the code to the right?

A. `\\//Itsybitsy`
`spider\\`

B. `\\/Itsybitsy`
`spider\\`

C. `\\/Itsybitsy spider\\`

D. `\//Itsybitsy spider\`

E. `\//Itsybitsy`
`spider\`

```
System.out.print("\\//Itsybitsy\nspider\\");
```

QUESTION 6

Assume `x, y` and `z` have type `int` and have been initialized. Then the expression

```
(x <= y+z) && !((y > z) || (z > x))
```

will always be true if:

A. `x==y and z>0`

B. `x==y and y==z and z>=0`

C. `x<y and z>=0`

D. `y<=z or z<=x`

E. `x>0 and y<0 and z>0`

QUESTION 7

How many times is the method `System.out.println()` called when the code to the right is executed?

A. 81 B. 0 C. 36

D. 45 E. 64

```
for(int i = 0; i < 9; i++){
  for(int j = 0; j < i; j++) {
    System.out.println(j);
  }
}
```

QUESTION 8

What is output by the code to the right?

A. `10 24` B. `5 33` C. `10 30`

D. `5 27` E. `5 30`

```
int x=15;
int y=0;
while(x>=10) {
  y+=3;
  while(y<x) {
    y = 2*x;
  }
  x-=5;
}
System.out.print(x + " " + y);
```

QUESTION 9

What is the output when the following statement is executed?

```
System.out.print("LateReplacement". replace('e', 'a').substring(4, 10));
```

A. `aRapla` B. `eReple` C. `Replec` D. `Raplac` E. `Lataa`

QUESTION 10

Assume that `max` is of type `int`. Which statement is true?

A. Loop 1 and Loop 2 produce the same output for all values of `max`.

B. Loop 1 and Loop 2 never produce the same output.

C. Loop 1 and Loop 2 only produce the same output when `max` is 1.

D. Loop 1 and Loop 2 produce the same output only when `max` is greater than 1.

E. Loop 1 and Loop 2 produce the same output only when `max` is less than 1.

```
// Loop 1
for(int i = 1; i < max; i++)
  System.out.print(i);

// Loop 2
int i = 1;
while(i < max){
  System.out.print(i);
  i++;
}
```

QUESTION 11

What value is returned by the method call

`myst(2, "a")`?

A. `"222a"` B. `"22a"` C. `"aaa"`

D. `"aaaa"` E. `"222"`

```
public static String myst(int a, String x){
  String y = a+x;
  for(int i = 0; i < a; i++) {
    y = a+y;
  }
  return y;
}
```

QUESTION 12

What is output by the code to the right?

A. `43.5` B. `24.0` C. `48.25`

D. `48` E. `48.0`

```
double y = 3.5; int n = 3; double x = 2.5;
System.out.print(y+n*x + (int)((y+n)*2));
```

QUESTION 13

Assume boolean variables `p`, `q` and `r` have all been initialized. Which of the following expressions are false only when `p`, `q` and `r` are all false?

A. `!(!p && !q) || !r`

B. `(p || !q) && !(q && r)`

C. `(p || q) || !(!r && !p)`

D. `(p && r) || q`

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

QUESTION 14

What is the output when the following statement is executed? The character ␣ represents a blank space in the output.

```
System.out.printf("Hello%5s%n%07.2f!", "Fred", 18.44);
```

A.
```
HelloFred
␣␣18.44!
```

B.
```
Hello␣Fred
0018.44!
```

C.
```
Hello␣Fred␣␣18.44!
```

D.
```
Hello␣Fred
␣␣18.44!
```

E.
```
Hello␣Fred
18.44␣␣!
```

QUESTION 15

What is output by the client code to the right?

A. 0 B. 3

C. 5 D. 12

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

Replace the line labeled line 0 as indicated below before answering Question 16.

QUESTION 16

What is output by the code to the right if the statement labeled line 0 in the client code is replaced with the following statement:

```
int m = myst(t);
```

A. 0 B. 3 C. 5

D. 12 E. There is no output due to a compilation error.

```
public class Fifteen {
  public int myst(String[] r) {
    int n = 0;
    for(int i = 0; i < r.length; i++) {
      n = n + r[i].length();
    }
   return n;
  }
}

// Client code
Fifteen x = new Fifteen();
String[] t = {"Hello", "World", "hi"};
int m = x.myst(t); // line 0
System.out.print(m); // line 1
```

QUESTION 17

What is the output when the code to the right is executed?

A. 35 140 B. 35 75

C. 40 140 D. 30 75

E. 35 105

```
int n = 5;
int sum = 0;
do
{
   sum = sum + n;
   n = n + 5;
} while(n < 35);
System.out.print(n + " " + sum);
```

QUESTION 18

Which of the following expressions is false? Assume `String s` and `int n` have been initialized as follows:

```
int n = 27;
String s = "27";
```

I. `n == (int) (n + 0.7)`

II. `String.valueOf(n).equals(s)`

III. `Integer.parseInt(s) == n`

IV. `'2' == s.charAt(0)`

A. I

B. II

C. III

D. IV

E. All of the expressions I through IV are true.

QUESTION 19

What statement can replace **<*1>** in the code to the right so that the `Account` class compiles correctly and so that line 1 in the client code prints `0.0`?

I. `this(initial)`

II. `balance = initial`

III. `this.balance = initial`

A. I only

B. II only

C. III only

D. I and II

E. II and III

```
public class Account {
  private double balance;
  private String status;

  public Account(double initial, String s){
    status = s;
    <*1>;
  }

  public double getBalance() {
    return balance;
  }
}

// Client code
Account a = new Account(0, "overdrawn");
System.out.print(a.getBalance()); // line 1
```

QUESTION 20

If `n=a.length` in the method `met` to the right, what is the big `O` of the `met` method? Give the most restrictive correct answer.

A. `O(n)` B. `O(nlogn)`

C. $O(n^2)$ D. `O(logn)`

E. `O(1)`

QUESTION 21

What algorithm does method `met` implement?

A. insertion sort B. selection sort

C. binary search D. quick sort

E. radix sort

```
public static int met(int[] a) {
  int max, maxPos;
    for(int i = 0; i < a.length-1; i++) {
      max = a[i];
      maxPos = i;
      for(int j = i+1; j < a.length; j++) {
        if(max < a[j]) {
          max = a[j];
          maxPos = j;
        }
      }
      int temp = a[i];
      a[i] = a[maxPos];
      a[maxPos] = temp;
    }
    return maxPos;
}
```

QUESTION 22

What is the value returned by the call `s(6)`?

A. `0` B. `1` C. `8`

D. `40` E. `80`

```
public int s(int n) {
  if(n == 0) return 10;
  else if(n == 1) return 5;
  return 2*s(n-2);
}
```

QUESTION 23

What is output by the code to the right?

A. `0` B. `8` C. `17`

D. `28` E. `29`

```
int a = 25 ^ (16 >>> 2);
System.out.print(a);
```

QUESTION 24

What is output by the code to the right?

A. 1 B. 2 C. 3

D. 4 E. 5

```
int[] a = {1, 2, 3, 4};
for(int i : a)
  i += 1;
System.out.print(a[a.length-1]);
```

QUESTION 25

What is output by the code to the right?

A. 1 B. 2 C. 3

D. 4 E. 5

```
int[][] b = {{1, 2, 3, 4}, {0, 1}, {1, 2, 3,
4, 5}};
System.out.print(b[1].length);
```

QUESTION 26

What is output by the code to the right?

A. hey

B. hello

C. hi

D. Output will vary from one run to the next.

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

```
List<String> list = new ArrayList<String>();
list.add("hey");
list.add("hello");
list.add("hi");
Iterator<String> it = list.iterator();
String obj = it.next();
it.remove();
it.remove();
System.out.print(it.next());
```

QUESTION 27

Which object in set `s` is printed last?

A. "Chinay" B. "Elvis"

C. "Abba" D. "Beyonce"

E. It varies from one run to the next.

```
Set<String> s = new TreeSet<String>();
s.add("Chinay");
s.add("Elvis");
s.add("Beyonce");
s.add("Abba");
for(String x : s){
  System.out.print(x + " ");
}
```

QUESTION 28

In the code to the right, which of the following code segments could replace **<*1>** to correctly implement the `Two` class constructor?

I.
```
s = st;
x = y;
k = a;
```

II.
```
super(st, y);
k = a;
```

III.
```
k = a;
super(st, y);
```

A. I only B. II only C. III only D. I and III E. II and III

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

QUESTION 29

Which of the following is a correct substitution for **<*2>** so that the `display()` method in the `Two` class returns a String containing the values of all 3 instance fields?

I.
```
return super.display() + " " + k;
```

II.
```
return s + " " + x + " " + k;
```

III.
```
return super.s + " " + super.x + "
      " + this.k;
```

IV.
```
return super.getS() + " " +
      super.getX() +  " " + k;
```

A. I only B. II only C. III only D. IV only E. I and IV

```
public class One {
  private String s;
  private double x;

  public One(String st, double y){
    s = st;
    x = y;
  }
  public String getS() {
    return s;
  }

  public double getX() {
    return x;
  }

  public String display(){
    return s + " " + x;
  }
}

public class Two extends One {
  private int k;

  public Two(String st, double y, int a){
    <*1>
  }

  public String display(){
    <*2>
  }
}
```

QUESTION 30

What is the output by the code to the right?

A. `41Hi32` B. `5Hi5`

C. `5Hi32` D. `41Hi5`

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

```
System.out.print(4+1+"Hi"+3+2);
```

QUESTION 31

What is output by the code to the right?

A. `5 3` B. `6 3` C. `4 4`

D. `5 2` E. `3 3`

```
int[] a = {1, 2, 3, 4};
int[] b = {3, 4, 5, 6};
b = a;
for(int i = 0; i < b.length; i++)
  b[i]++;
System.out.print(b[2] + " " + a[2]);
```

QUESTION 32

Which expression can replace **<*1>** so that the main method compiles without an error?

I. A blank (nothing is needed in place of **<*1>**).

II. `throw Exception`

III. `throws IOException`

IV. `throws Error`

A. I only

B. II only

C. III only

D. IV only

E. III and IV

```
public static void main(String[] args) <*1> {
  File f = new File("appoint.txt");
  if(f.canRead()){
    Scanner in = new Scanner(f);
  }
  else System.out.print("File does not exist
or is unreadable");
}
```

QUESTION 33

Let `n = a.length`. What is the big `O` of the method to the right? Give the most restrictive correct answer.

A. $O(n^2)$ B. $O(n)$ C. $O(n \log n)$

D. $O(1)$ E. $O(n^3)$

```
public void mu(int[] a) {
   for(int i = a.length-1; i>=1; i--){
      for(int j = i; j >= 0; j--){
       System.out.print(i*j+a[i]*a[j]);
      }
   }
}
```

QUESTION 34

Consider the output to the right. Which of the following code segments produce it?

I.

```
String s = "";
for(int i = 1; i <=9; i++) {
  s += i;
  System.out.println(s);
}
```

II.

```
for(int i = 1; i <= 9; i++) {
  for(int j = i; j >=1; j--) {
    System.out.print(j);
  }
  System.out.println();
}
```

III.

```
String s = "";
for(int i = 1; i <=9; i++) {
  s = i + s;
  System.out.println(s);
}
```

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

D. I and III E. II and III

```
1
21
321
4321
54321
654321
7654321
87654321
987654321
```

QUESTION 35

The text file "`stuff.txt`" contains both integer and string tokens. What expression can replace **<*1>** so that the output of the code to the right is the sum of all of the integer tokens?

I. `scan.canRead()`

II. `scan.hasNextInt()`

III. `Integer.parseInt(scan.hasNext())`

A. `I only` B. `II only`

C. `III Only` D. `II and III`

E. `I and III`

```
Scanner scan = new Scanner(new
File("stuff.txt"));
int sum = 0;
while(scan.hasNext()){
  if(<*1>) {
    int num = scan.nextInt();
    sum += num;
  }
  else scan.next();
}
System.out.print(sum);
```

QUESTION 36

In the code to the right, what is the output for the statement labeled line 1?

A. `1` B. `2`

C. `3` D. `4`

E. `6`

```
String s = "What? Game: today";
String lim = ":|a|\\?";
String[] t = s.split(lim);
System.out.println(t.length); //line 1
System.out.print(t[1]); // line 2
```

QUESTION 37

In the code to the right, what is the output for the statement labeled line 2?

A. `t` B. `at?` C. `Wh`

D. `Ga` E. `G`

QUESTION 38

For the class and interface to the right, consider the following client code. Which is correct and produce NO compilation errors? The output from the client code should be "555".

I.
```
Stuff t = new Stuff();
System.out.print(t.two(5));
```
II.
```
Stuff t = new Stuff2();
System.out.print(t.two(5));
```
III.
```
Stuff2 t = new Stuff();
System.out.print(t.two(5));
```

A. I only B. II only

C. III only D. I and II

E. II and III

```
public interface Stuff {

   double one();
   String two(int x);
}

public class Stuff2 implements Stuff {

   public double one() {
      return 3.5;
   }

   public String two(int x) {
    return ""+x+x+x;
  }

   public void three() {
      System.out.print(3);
   }
}
```

QUESTION 39

Which of the following strings from `r` could be the output from this code? Choose the answer that includes all possible outputs.

A. yah

B. hello, hi, bye, yah

C. hello, hi, bye

D. hello, hi, bye, yah, yay

E. hello, hi

```
Random ran = new Random();
String[] r = {"hello", "hi", "bye", "yah",
"yay"};
System.out.print(r[ran.nextInt(3)]);
```

QUESTION 40

Consider the code to the right. The maximum number of `String`s that will be printed is:

A. `0` B. `10`

C. `L.size()` D. `L.size()-1`

D. `s.length()`

```
List<String> L = new LinkedList<String>();
//<Assume elements have been added to L>
for(String s : L){
  if(s.length() > 10)
    System.out.println(s);
}
```

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)