UIL COMPUTER SCIENCE WRITTEN TEST

2018 DISTRICT

MARCH 2018

General Directions (Please read carefully!)

1. DO NOT OPEN THE EXAM UNTIL TOLD TO DO SO.
2. There are 40 questions on this contest exam. You will have 45 minutes to complete this contest.
3. All answers must be legibly written on the answer sheet provided. Indicate your answers in the appropriate blanks provided on the answer sheet. Clean erasures are necessary for accurate grading.
4. You may write on the test packet or any additional scratch paper provided by the contest director, but NOT on the answer sheet, which is reserved for answers only.
5. All questions have ONE and only ONE correct answer. There is a 2-point penalty for all incorrect answers.
6. Tests 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 test until told to do otherwise. You may use this time to check your answers.
7. If you are in the process of actually writing an answer when the signal to stop is given, you may finish writing that answer.
8. All provided code segments are intended to be syntactically correct, unless otherwise stated. You may also assume that any undefined variables are defined as used.
9. A reference to many commonly used Java classes is provided with the test, and you may use this reference sheet during the contest. AFTER THE CONTEST BEGINS, you may detach the reference sheet from the test booklet if you wish.
10. Assume that any necessary import statements for standard Java SE packages and classes (e.g., `java.util`, `System`, etc.) are included in any programs or code segments that refer to methods from these classes and packages.
11. NO CALCULATORS of any kind may be used during this contest.

Scoring

1. Correct answers will receive **6 points**.
2. Incorrect answers will lose **2 points**.
3. Unanswered questions will neither receive nor lose any points.
4. In the event of a tie, the student with the highest percentage of attempted questions correct shall win the tie.

STANDARD CLASSES AND INTERFACES – SUPPLEMENTAL REFERENCE

package java.lang

```
class Object
  boolean equals(Object anotherObject)
  String toString()
  int hashCode()

interface Comparable<T>
  int compareTo(T anotherObject)
    Returns a value < 0 if this is less than anotherObject.
    Returns a value = 0 if this is equal to anotherObject.
    Returns a value > 0 if this is greater than anotherObject.

class Integer implements Comparable<Integer>
  Integer(int value)
  int intValue()
  boolean equals(Object anotherObject)
  String toString()
  String toString(int i, int radix)
  int compareTo(Integer anotherInteger)
  static int parseInt(String s)

class Double implements Comparable<Double>
  Double(double value)
  double doubleValue()
  boolean equals(Object anotherObject)
  String toString()
  int compareTo(Double anotherDouble)
  static double parseDouble(String s)

class String implements Comparable<String>
  int compareTo(String anotherString)
  boolean equals(Object anotherObject)
  int length()
  String substring(int begin)
    Returns substring(begin, length()).
  String substring(int begin, int end)
    Returns the substring from index begin through index (end - 1).
  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 fromIndex. Returns -1 if str is not found.
  int indexOf(int ch)
  int indexOf(int ch, int fromIndex)
  char charAt(int index)
  String toLowerCase()
  String toUpperCase()
  String[] split(String regex)
  boolean matches(String regex)
  String replaceAll(String regex, String str)

class 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 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, int b)
  static int max(int a, int b)
  static long round(double a)
  static double random()
    Returns a double greater than or equal to 0.0 and less than 1.0.
```

package java.util

```
interface List<E>
class ArrayList<E> implements List<E>
  boolean add(E item)
  int size()
  Iterator<E> iterator()
  ListIterator<E> listIterator()
  E get(int index)
  E set(int index, E item)
  void add(int index, E item)
  E remove(int index)

class LinkedList<E> implements List<E>, Queue<E>
  void addFirst(E item)
  void addLast(E item)
  E getFirst()
  E getLast()
  E removeFirst()
  E removeLast()

class Stack<E>
  boolean isEmpty()
  E peek()
  E pop()
  E push(E item)

interface Queue<E>
class PriorityQueue<E>
  boolean add(E item)
  boolean isEmpty()
  E peek()
  E remove()

interface Set<E>
class HashSet<E> implements Set<E>
class TreeSet<E> implements Set<E>
  boolean add(E item)
  boolean contains(Object item)
  boolean remove(Object item)
  int size()
  Iterator<E> iterator()
  boolean addAll(Collection<? extends E> c)
  boolean removeAll(Collection<?> c)
  boolean retainAll(Collection<?> c)

interface Map<K,V>
class HashMap<K,V> implements Map<K,V>
class TreeMap<K,V> implements 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()

interface Iterator<E>
  boolean hasNext()
  E next()
  void remove()

interface ListIterator<E> extends Iterator<E>
  void add(E item)
  void set(E item)

class Scanner
  Scanner(InputStream source)
  Scanner(String str)
  boolean hasNext()
  boolean hasNextInt()
  boolean hasNextDouble()
  String next()
  int nextInt()
  double nextDouble()
  String nextLine()
  Scanner useDelimiter(String regex)
```

UIL COMPUTER SCIENCE WRITTEN TEST – 2018 DISTRICT

Note: Correct responses are based on **Java SE Development Kit 8 (JDK 8)** from Oracle, Inc. All provided code segments are intended to be syntactically correct, unless otherwise stated (e.g., "error" is an answer choice) and any necessary Java SE 8 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 using: `import static java.lang.System.*;`**

Question 1.

Which of the following is the sum of 11111111_2 and 11110011_2?

A) $1F3_{16}$ **B)** 497_{10} **C)** 762_8 **D)** 111010010_2 **E)** None of the above.

Question 2.

What is the output of the code segment to the right?

A) 18 **B)** 18.0 **C)** 10.5 **D)** 10 **E)** 11

```
out.println(10+5-3*6.0/4);
```

Question 3.

What is the output of the code segment to the right?

A)
```
Go

Spurs
Go!
```

B)
```
Go
Spurs
Go!
```

C)
```
Go
Spurs Go!
```

D)
```
Go Spurs
Go!
```

E)
```
Go Spurs Go!
```

```
out.println("Go\n");
out.print("Spurs\nGo!");
```

Question 4.

What is the output of the code segment to the right?

A) laoon **B)** ploon **C)** lmoon **D)** plmoon **E)** loon

```
String s="planet";
String t="moon";
String u=s.substring(1, 2)+t.substring(1);
out.print(u);
```

Question 5.

What is the output of the code segment to the right?

A) true **B)** false

```
out.print(true^false&&true||false);
```

Question 6.

What is the output of the code segment to the right?

A) 4 **B)** 4.0 **C)** 8 **D)** 8.0 **E)** 2

```
int x=64;
out.print(Math.cbrt(x));
```

Question 7.

What is the output of the code segment to the right?

A) 68.0 **B)** D **C)** 68 **D)** 68.35 **E)** Will not compile. Type mismatch error.

```
char c='A';
int i=8;
double d=4.65;
out.print(c+i-d);
```

Question 8.

What is the output of the code segment to the right?

A) `yes`

B) `no`

C) `yes and no`

D) `maybe`

E) `yes yes and no`

```
boolean yes=false,no=true,maybe=true;
if(yes)
  out.print("no ");
else if(no)
  out.print("yes ");
else if(maybe)
  out.print("yes and no ");
else
  out.print("maybe");
```

Question 9.

Which of the following must replace **<code>** in the loop shown on the right to ensure that the code segment will print exactly 6 X's?

A) `int i=1;i<10;i+=2`

B) `int i=0;i<=10;i+=2`

C) `int i=1;i<6;i++`

D) `int i=0;i<=6;i++`

E) `int i=1;i<=6;i+=2`

```
for(<code>)
     out.print("X");
```

Question 10.

What is output by the code segment listed to the right?

A) `[0, 8, 12, 10, 0]`

B) `[8, 12, 10, 12]`

C) `[0, 8, 12, 10, 12]`

D) `[8, 12, 10, 12, 0]`

E) Error. Throws an ArrayIndexOutOfBoundsException.

```
int[] list= new int[5];
list[1]=8;
list[2]=12;
list[3]=10;
list[4]=list[list[2]-list[3]];
out.print(Arrays.toString(list));
```

Question 11.

What is printed by the code segment shown on the right if the following values are contained in `datafile.dat`? Assume that all necessary classes have been imported and that the main method throws an IOException.

```
5 9 1 7 -3 4 6 2 3 8
```

A) `16 -3`

B) `16 4`

C) `22 -3`

D) `22 4`

E) Error. Throws a NoSuchElementException .

```
File f=new File("datafile.dat");
Scanner s=new Scanner(f);
int a=0;
while(s.nextInt()>0)
     a+=s.nextInt();
out.print(a+" "+s.nextInt());
```

Question 12.

What is the output of the code segment to the right?

A) `44.0 1`

B) `55.0 1`

C) `45.0 1`

D) `55.0 0`

E) `45.0 0`

```
double d=0;
int i=10;
do {
     d+=--i;
}while(i>0);
out.print(d+" "+i);
```

Question 13.

In any given expression, which of the following operators would be applied last?

A) `&&` **B)** `*` **C)** `<=` **D)** `^` **E)** `||`

Question 14.

Which of the following statements will not compile?

A) `long l=Short.MAX_VALUE;`

B) `int i=Byte.BYTES;`

C) `int j=Byte.SIZE;`

D) `byte b=Integer.MIN_VALUE;`

E) `short s=Byte.MAX_VALUE;`

Question 15.

What is the output of the code segment to the right?

A) `[6, 0, 4, 5]`

B) `[6, 4, 5]`

C) `[6]`

D) `[0, 4, 5]`

E) `[5]`

```
ArrayList<Integer> a=new
ArrayList<Integer>();
a.add(4);
a.set(0, 0);
a.add(5);
a.set(0, 6);
a.remove(1);
out.print(a);
```

Question 16.

What is printed by the code segment shown on the right?

A) `four three two two one`

B) `four three two one`

C) `one two three four`

D) `one two two three four`

E) `four three one`

```
Stack<String> s=new Stack<String>();
s.push("one");
s.push("two");
s.push("two");
s.pop();
s.push("three");
s.push("four");
while(!s.isEmpty())
      out.print(s.pop()+" ");
```

Question 17.

What is the output of the client code shown on the right?

A) `PecosPecoPecPe`

B) `PPePecPecoPecos`

C) `PecosPecoPecPeP`

D) `PePcePocePsoceP`

E) `PPPPP`

```
public static String rec(String s,int i) {
if(s.length()==1)
      return s;
else
      return s+rec(s.substring(0,i),i-1);
}
//client code
String s="Pecos";
out.print(rec(s,s.length()-1));
```

Question 18.

Which of the following should replace **<code 1>** in the class shown on the right?

A) `double`

B) `int`

C) `static`

D) `this`

E) `super`

Question 19.

Which of the following should replace **<code 2>** in the class shown on the right?

A) `l,w,h`

B) `double length,double width,double height`

C) `length,width,height`

D) `double l,double w,double h`

E) No additional code is required

Question 20.

What is the output if this client code that is implemented in a different class than `Box`.

```
Box b1=new Box(10,10,10);
out.print("Height="+b1.height+" ");
out.print(b1.surfaceArea()+" ");
out.print(b1.volume);
```

A) `10.0 600.0 1000.0`

B) `Height=10.0 300.0 1000.0`

C) `Height=10.0 600.0 1000.0`

D) `Height=10.0 1000.0 600.0`

E) There is no output due to an error.

```
//Use the following code to answer questions
//18, 19 and 20.

public class Box {

public <code 1> surfaceArea() {
  return 2*(height*width+length*
            height+length*width);
  }

public Box(<code 2>) {
  length=l;
  width=w;
  height=h;
  volume=length*width*height;
}

private double length,width,height;
public double volume;
}
```

Question 21.

What is the output of the code segment shown on the right?

A) `[4, 5, 6, 7]`

B) `[2, 3, 4, 5]`

C) `[2, 4, 6, 8]`

D) `[1, 3, 5, 7]`

E) `[6, 7, 8, 9]`

```
int[][] mat= new int[4][4];
for(int x=0;x<4;x++)
      for(int y=0;y<4;y++)
            mat[y][x]=x+2*y;
out.println(Arrays.toString(mat[2]));
```

Question 22.

What is the output of the code segment shown on the right?

A) `true true false`

B) `true false true`

C) `false true true`

D) `false false true`

E) `false true false`

```
out.print("123ABC".matches("\\D{3}\\W{3}")+" ");
out.print("555-5555".matches(".+")+" ");
out.print("Alphabet".matches("A[a-z]?"));
```

Question 23.

Which of the following represents the correct signature of a method named `tip` that has an amount for a meal and the desired tip percent as its parameters and returns the appropriate tip amount?

A) `public static void tip(double amount, int percent)`

B) `public static tip(double amount, int percent)`

C) `public static double tip(amount, percent)`

D) `public static double tip(double amount, int percent)`

E) `tip(double amount, int percent)`

Question 24.

Which of the following methods will return N! (N factorial) ?

A)

```
public static long fac(long n) {
long f=1;
for(long x=n;x>=1;x--)
     f*=x;
return f;
}
```

B)

```
public static long fac(long n) {
long f=1,x=2;
while(x<=n) {
     f=f*x;
     x++;
}
return f;
}
```

C)

```
public static long fac(long n) {
if(n==1)
     return 1;
else
     return fac(n-1);
}
```

D) A and B

E) A, B and C

Question 25.

Which of the following Java expressions will correctly round `n` to `r` decimal places if `n` is a `double` and `r` is an `int`?

A) `(int)(r*Math.pow(10, n)+0.5)/Math.pow(10, n)`

B) `(n*Math.pow(10, r)+0.5)/Math.pow(10, r)`

C) `(int)(n*Math.pow(10, r)+0.5)/Math.pow(10, r)`

D) `(int)(n*10+0.5)/10`

E) `(int)(n/Math.pow(10, r)+0.5)*Math.pow(10, r)`

Question 26.

What is the smallest possible value that the code shown on the right will produce?

A) 6

B) 11

C) 66

D) 1

E) 0

```
Random r=new Random();
System.out.print(r.nextInt(6)*11);
```

Question 27.

Which of the following must replace **<code>** in the method shown on the right to ensure the method will sort `a` in ascending order?

A) `k>=0&&a[k]<ce`

B) `k>=0&&a[k]>ce`

C) `k>=0||a[k]>ce`

D) `k>=i&&a[k]>ce`

E) `k>=ce&&a[i]>ce`

Question 28.

Once **<code>** has been filled in correctly, which of the following sorting algorithms is implemented by the `uilSort` method?

A) heap sort

B) quick sort

C) insertion sort

D) selection sort

E) merge sort

Question 29.

What is the least restrictive worst case time efficiency (Big O value) for the `uilSort` method?

A) O(1)

B) O(n)

C) $O(n^2)$

D) O(log n)

E) O(n log n)

Question 30.

Which of the following shows the order of the elements in array `a` when code execution reaches the comment statement and `i` equals 2 given the following client code?

```
int[] a= {5,3,1,0,2,4};
uilSort(a);
```

A) `[0, 1, 3, 5, 2, 4]`

B) `[1, 2, 3, 0, 5, 4]`

C) `[5, 3, 1, 4, 2, 0]`

D) `[1, 3, 5, 0, 2, 4]`

E) `[0, 1, 2, 5, 3, 4]`

```
//Use the following method to answer
//questions 27, 28, 29 and 30.
public static void uilSort(int[] a) {
  int i=1;
  do {
    int ce=a[i];
    int k=i-1;
    while(<code>) {
      a[k+1]=a[k];
      k--;
      }
    a[k+1]=ce;
    //comment
    i++;
  }while(i<a.length);
}
```

Question 31.

What is the output of the code segment shown here given the method implementation on the right?

```
int g,h=0;
for(g=1;g<=3;g++)
  out.print(doSomething(g,h)+" ");
out.print(g+" "+h);
```

A) 13 14 15 3 0

B) 13 14 15 4 0

C) 22 23 24 4 0

D) 22 23 24 9 6

E) 13 14 15 9 6

```
public static int doSomething(int g,int h) {
while(h<=5) {
      g=h+++g;
      h++;
      }
return g+h;
}
```

Question 32.

What is printed by the line of code shown on the right?

A) 14

B) 0

C) 30

D) 15

E) 16

```
out.print(14|15&16);
```

Question 33.

What is printed by the code segment shown on the right?

A) Go

B) Fight

C) Win

D) Error. Will not compile.

E) Error. Throws a run time exception.

```
Double d1=new Double(18.99);
Double d2=19.00;
if(d1.compareTo(d2)==0)
     out.print("Go");
else if(d1.compareTo(d2)>0)
     out.print("Fight");
else
     out.print("Win");
```

Question 34.

Which of the following lines of code will not compile correctly?

A) `int i=2147483647;`

B) `double d=250.84d;`

C) `int h=0xABC;`

D) `char c=0b11111111;`

E) None of the above. All of the lines shown above will compile correctly.

Question 35.

What is the output of the code segment shown on the right?

A) #@&*%@@%&&

B) #@&*@@%&&

C) #@@*%@@%@&

D) #@@*@@%@&

E) #&*%%&&

```
String s="March2018",t="";
for(int i=0;i<s.length();i++) {
      switch(s.substring(i, i+1)) {
      case "M":t+="#";break;
      case "c":t+="*";
      case "0":t+="%";break;
      case "r":
      case "1":
      case "8":t+="&";break;
      default:t+="@";
      }
}
out.print(t);
```

Question 36.

Which pair of the Boolean expressions listed on the right are equivalent?

A) I and II

B) II and III

C) III and IV

D) I an IV

E) II and IV

I. $\bar{A} * \bar{B}$

II. $\overline{A * B}$

III. $\bar{A} + \bar{B}$

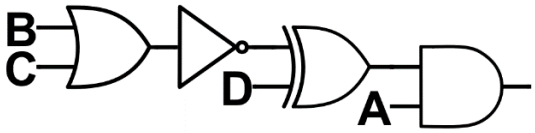
IV. $\overline{A \oplus B}$

Question 37.

What is the value of the Boolean expression shown in the diagram on the right if A is true, B is false, C is true and D is false?

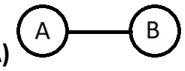
A) true

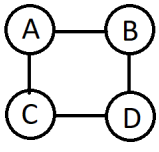
B) false

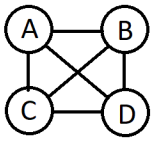


Question 38.

Which of the graphs illustrated here is a complete graph?

A)


B)


C)


D) A and C

E) A, B and C

Question 39.

Evaluate the prefix expression shown on the right and write your answer in the blank provided?

* − + 8 5 3 2

Question 40.

What is the decimal equivalent of this signed binary 8-bit two's complement value?

10101010

★ANSWER KEY – CONFIDENTIAL★

UIL COMPUTER SCIENCE – 2018 DISTRICT

Questions (+6 points for each correct answer, -2 points for each incorrect answer)

1) C	11) B	21) A	31) C
2) C	12) E	22) E	32) D
3) A	13) E	23) D	33) C
4) E	14) D	24) D	34) E
5) A	15) C	25) C	35) A
6) B	16) A	26) E	36) B
7) D	17) D	27) B	37) B
8) A	18) E	28) A	38) D
9) B	19) B	29) B	*39) 20
10) C	20) C	30) C	*40) -86

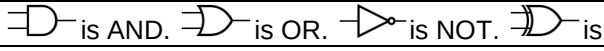
** See "Explanation" section below for alternate, acceptable answers.*

Note: Correct responses are based on **Java SE Development Kit 8 (JDK 8)** from Sun Microsystems, Inc. All provided code segments are intended to be syntactically correct, unless otherwise stated (e.g., "error" is an answer choice) and any necessary Java SE 8 Standard Packages have been imported. Ignore any typographical errors and assume any undefined variables are defined as used.

Explanations:

<table>
<tr><td>1.</td><td>C</td><td>$11111111_2+11110011_2=111110010_2$ (eliminates D). $111110010_2 = 498_{10}$ (eliminates B). $1F3_{16} = 499_{10}$ (eliminates A). $762_8 = 498_{10}$.</td></tr>
<tr><td>2.</td><td>C</td><td>10+5-3*6.0/4=
10+5-18.0/4=
10+5-4.5=
15-4.5=
10.5</td></tr>
<tr><td>3.</td><td>A</td><td><code>println</code> method inserts a new line after the string is printed. The <code>\n</code> escape sequence inserts a new line where ever it has been inserted in the string.</td></tr>
<tr><td>4.</td><td>E</td><td>
<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td></td><td>0</td><td>1</td><td>2</td><td>3</td></tr>
<tr><td>p</td><td>l</td><td>a</td><td>n</td><td>e</td><td>t</td><td></td><td>m</td><td>o</td><td>o</td><td>n</td></tr>
</table>
The two argument substring method starts at the index number of the first argument and goes to the second argument minus one. So, the first substring is from one to one (just the l). The one argument substring method starts at the index number of the argument and continues to the end of the string. In this case 1 to 3 (oon).
</td></tr>
<tr><td>5.</td><td>A</td><td>T^F&&T||F=
T&&T||F=
T||F=
T</td></tr>
<tr><td>6.</td><td>B</td><td>The <code>cbrt</code> method returns the cube root of its argument as a double. 4X4X4=64.</td></tr>
<tr><td>7.</td><td>D</td><td>ASCII value of ‘A’ is 65. 65+8-4.65=68.35</td></tr>
<tr><td>8.</td><td>A</td><td>Once a <code>true</code> value is encountered, in this case the boolean variable <code>no</code> is true, the code for that <code>if</code> statement is executed and the remaining <code>else</code> statements are skipped.</td></tr>
<tr><td>9.</td><td>B</td><td><code>i</code> takes the values 0, 2, 4, 6, 8, and 10. Once I becomes 12, the loop stops. That makes 6 six iterations of the loop.</td></tr>
<tr><td>10.</td><td>C</td><td>
<table>
<tr><th>int[] list=new int[5]</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th></tr>
<tr><td></td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>list[1]=8</td><td>0</td><td>8</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>list[2]=12</td><td>0</td><td>8</td><td>12</td><td>0</td><td>0</td></tr>
<tr><td>list[3]=10</td><td>0</td><td>8</td><td>12</td><td>10</td><td>0</td></tr>
<tr><td>list[4]=list[list[2]-list[3]];</td><td>0</td><td>8</td><td>12</td><td>10</td><td>12</td></tr>
</table>
</td></tr>
<tr><td>11.</td><td>B</td><td><code>nextInt</code> returns the next value beyond (in front of) the cursor (pointer) in the datafile and then advances the cursor to the next position, even in the condition for a while loop. 5, 1 and -3 are used in the while loop condition statement. 9 and 7 are added to <code>a</code>. 4 is returned by the final call to <code>nextInt</code> because even though the loop terminates when -3 is read, the cursor advances to the next position.</td></tr>
<tr><td>12.</td><td>E</td><td><code>i</code> is decremented before it is added to <code>d</code> with each iteration of the loop.
<pre>
   d i
 9.0 9
17.0 8
24.0 7
30.0 6
35.0 5
39.0 4
42.0 3
44.0 2
45.0 1
45.0 0
</pre>
</td></tr>
<tr><td>13.</td><td>E</td><td>Precedence from first to last for the operators shown is: <code>* <= ^ && ||</code></td></tr>
<tr><td>14.</td><td>D</td><td>The <code>MIN_VALUE</code> for <code>Integer</code> is -2147483648. The largest negative value that can be stored in a variable of type <code>byte</code> is -128.</td></tr>
</table>

<table>
<tr><td>15.</td><td>C</td><td>
<table>
<tr><td><code>a.add(4);</code></td><td>[4]</td></tr>
<tr><td><code>a.set(0,0);</code></td><td>[0]</td></tr>
<tr><td><code>a.add(5);</code></td><td>[0, 5]</td></tr>
<tr><td><code>a.set(0,6);</code></td><td>[6, 5]</td></tr>
<tr><td><code>a.remove(1);</code></td><td>[6]</td></tr>
</table>
</td></tr>
<tr><td>16.</td><td>B</td><td>
<table>
<tr><td colspan="2">Stacks use a first in, last out protocol for accessing data.</td></tr>
<tr><td><code>s.push("one");</code></td><td>one</td></tr>
<tr><td><code>s.push("two");</code></td><td>one two</td></tr>
<tr><td><code>s.push("two");</code></td><td>one two two</td></tr>
<tr><td><code>s.pop();</code></td><td>one two</td></tr>
<tr><td><code>s.push("three");</code></td><td>one two three</td></tr>
<tr><td><code>s.push("four");</code></td><td>one two three four</td></tr>
<tr><td colspan="2">Elements are popped out from right to left.</td></tr>
</table>
</td></tr>
<tr><td>17.</td><td>C</td><td>
<table>
<tr><td>i</td><td>s</td></tr>
<tr><td>5</td><td>Pecos</td></tr>
<tr><td>4</td><td>Peco</td></tr>
<tr><td>3</td><td>Pec</td></tr>
<tr><td>2</td><td>Pe</td></tr>
<tr><td>1</td><td>P</td></tr>
</table>
</td></tr>
<tr><td>18.</td><td>A</td><td>Since <code>height</code>, <code>width</code> and <code>length</code> are all doubles, <code>surfaceArea</code> must return a double value.</td></tr>
<tr><td>19.</td><td>D</td><td><code>l</code>, <code>w</code>, and <code>h</code> have not been declared locally so they must be passed as parameters. The parameter list must show the type and name of each parameter.</td></tr>
<tr><td>20.</td><td>E</td><td>The field <code>height</code> has been declared private, therefore, it cannot be directly accessed from client code that is in another class than <code>Box</code>.</td></tr>
<tr><td>21.</td><td>A</td><td>The matrix looks like this after the loops are done:
[0, 1, 2, 3]
[2, 3, 4, 5]
[4, 5, 6, 7]
[6, 7, 8, 9]
mat[2] is the third row down.</td></tr>
<tr><td>22.</td><td>E</td><td>\D{3} matches exactly 3 non-digits and \W{3} matches exactly 3 non-word characters.
.+ matches any character one or more times.
A[a-z]? matches a capital A followed by any lower case letter once or not at all.</td></tr>
<tr><td>23.</td><td>D</td><td>A method signature must contain a return type, name and parameter list if necessary. All parameters must have a type and name.</td></tr>
<tr><td>24.</td><td>D</td><td>A and B are correct. For answer choice C to be correct the last line should be:
<code>return n*fac(n-1).</code></td></tr>
<tr><td>25.</td><td>C</td><td>Example where n=4.192837 and r=3.
(int)(4.192837*Math(10,3)+0.5)/Math.pow(10,3)=
(int)(4.192837*1000.0+0.5)/1000.0=
(int)(4192.837+0.5)/1000.0=
(int)4193.337/1000.0=
4193/1000.0=
4.193</td></tr>
<tr><td>26.</td><td>E</td><td><code>nextInt(6)</code> will return a whole number between 0 (inclusive) and 6 (exclusive).
0 * 11 = 0.</td></tr>
</table>

27.	B	This is an insertion sort so we are getting the next element in the unsorted portion of the array then shifting elements to the right until we find the proper place for the unsorted element or when we reach the front of the array. Then the unsorted element is placed (inserted) into the proper location.
28.	C	See #29.
29.	C	Best Case O(n), Average Case $O(n^2)$, Worst Case $O(n^2)$
30.	D	i=1 [3, 5, 1, 0, 2, 4] i=2 [1, 3, 5, 0, 2, 4] i=3 [0, 1, 3, 5, 2, 4] i=4 [0, 1, 2, 3, 5, 4] i=5 [0, 1, 2, 3, 4, 5]
31.	B	The variables `g` and `h` in the client code are unchanged by the calls to the `doSomething` method so their final values are 4 and 0. Within the method, for this expression, `h+++g`, the increment operator is applied to the variable h like this: (h++)+g. Here is a trace of the variable values when the code has been run. g=1 h=2 g=3 h=4 g=7 h=6 13 g=2 h=2 g=4 h=4 g=8 h=6 14 g=3 h=2 g=5 h=4 g=9 h=6 15
32.	A	$14 = 1110_2$ $15 = 1111_2$ $16 = 10000_2$ 01111 & 10000 = 00000 00000 | 1110 = 1110 $1110_2 = 14$
33.	C	`compareTo` returns 0 if d1 and d2 are equal, a value less than 0 if d1 is less than d2, and a value greater than 0 if this d1 is greater than d2. `Double d2=19.00;` is allowed due to autoboxing.
34.	E	2147483647 is within the range of values for the int data type. The letter d following 250.84 designates the value as a double. It is optional. 0x designates a value as hexadecimal. Hexadecimals can be assigned to int type variables. 0b designates a binary number. 11111111 = 255. 255 is a valid ASCII value.
35.	A	When there is no break statement present execution of the code goes to the next case selector. When there is no code present after a case selector, execution goes to the next case selector.
36.	B	DeMorgan's Law states $\overline{A * B} = \bar{A} + \bar{B}$
37.	B	 is AND. is OR. is NOT. is XOR.
38.	D	Every pair of vertices are connected by an edge in a complete graph. A and D and D and B are not connected in answer choice B.
39.	20	* - + 8 5 3 2 = * - 13 3 2 = * 10 2 = 20
40.	-86	Take the complement of 10101010 to get 01010101 then add 1 to get 01010110 which is 86. We know the answer is negative since the sign bit was one so the final answer is -86.