UIL Computer Science Written Test

2023 Invitational A

January/February 2023

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)
```

Package java.util.function

```
Interface BiConsumer<T,U>
  void accept(T t, U u)

Interface BiFunction<T,U,R>
  R apply(T t, U u)

Interface BiPredicate<T,U>
  boolean test(T t, U u)

Interface Consumer<T>
  void accept(T t)

Interface Function<T,R>
  R apply(T t)

Interface Predicate<T>
  boolean test(T t)

Interface Supplier<T>
  T get()
```

UIL COMPUTER SCIENCE WRITTEN TEST – 2023 INVITATIONAL A

Note: Correct responses are based on **Java SE Development Kit 17 (JDK 17)** 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 17 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 decimal numbers has the largest base 10 value?

A) 100101_2 **B)** 56_8 **C)** 26_{16} **D)** 46_7 **E)** $1A_{12}$

Question 2

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

A) `1` **B)** `4` **C)** `6` **D)** `17` **E)** `17.25`

```
out.print(15 + 5 / 4 + 1);
```

Question 3

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

A)
```
OneTwo
ThreeFour
Five
```

B)
```
One
TwoThree
FourFive
```

C)
```
One
TwoThreeFour
Five
```

D)
```
OneTwo
ThreeFourFive
```

E)
```
OneTwo
Three
FourFive
```

```
out.print("One");
out.println("Two");
out.print("Three");
out.println("Four");
out.print("Five");
```

Question 4

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

A) `niv` **B)** `i` **C)** `iv` **D)** `ive` **E)** `v`

```
String str = "University";
out.print(str.substring(2,3));
```

Question 5

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

A) `true`

B) `false`

```
boolean M = true;
boolean N = false;
out.print(M || true && N);
```

Question 6

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

A) `4.0` **B)** `6` **C)** `5.0` **D)** `5` **E)** `6`

```
out.print((int)Math.floor(5.85));
```

Question 7

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

A) `78` **B)** `67.5` **C)** `67` **D)** `92` **E)** `70`

```
int x = 7;
int y = 8;
double a = 2.0;
out.print(x / a + y * y);
```

Question 8

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

A) MRVVRM

B) VRM

C) RVRM

D) VVRM

E) V

```
int R = 7;
int V = 9;
int W = V - R;

if(R > V)
  out.print("M");
if(2 + R < V)
  out.print("R");
else
  out.print("V");
if(W + R == V)
  out.print("VRM");
```

Question 9

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

A) 1 2 3 4 5 6 7 8 9

B) 1 3 5 7 9

C) 1 4 9 16 25 36 49 64 81

D) 1 4 9

E) 1 9 25 49 81

```
for(int x = 1; x < 10; x=x+2)
  out.print(x*x + " ");
```

Question 10

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

A) 26 **B)** 4 **C)** 221 **D)** 33 **E)** 85

```
int[] stuff = {2,17,3,13,5,11,7};
out.print(stuff[1]*stuff[4]);
```

Question 11

What is output by the code segment to the right?

A) MICH

B) MII

C) MICHI

D) CHGAN

E) MICHIGAN

```
Scanner t = new Scanner("MI CH I GAN");
t.next();
String st = t.next();
t.next();
st += t.next();
out.print(st);
```

Question 12

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

A) 100 **B)** 400 **C)** 210 **D)** 110 **E)** 81

```
int h = 0;
for(int i = 1; i <= 20; i = i + 2)
    h += i;
out.print(h);
```

Question 13

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

A) 40 **B)** 46 **C)** 20 **D)** 10 **E)** 80

```
int a = 10, b = 4, c = 4;
out.print(a << 2 + b >> 1 + ++c);
```

Question 14

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

A) `8` **B)** `16` **C)** `32` **D)** `4` **E)** `64`

```
out.println(Integer.SIZE);
```

Question 15

What is output by the code segment to the right?

A) `[11, 22, 33, 44, 55, 66]`

B) `[44, 55, 66]`

C) `[11, 55, 66]`

D) `[22, 55, 66]`

E) `[22, 44, 66]`

```
ArrayList<Integer> list;
list = new ArrayList<Integer>();
list.add(11);
list.add(22);
list.remove(1);
list.add(33);
list.add(44);
list.remove(1);
list.add(55);
list.add(66);
list.remove(1);
out.println(list);
```

Question 16

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

A) `F` **B)** `G` **C)** `H` **D)** `I` **E)** `J`

```
String car = "FGHIJKLMNOPQRST";
int L = car.indexOf("KL");
out.println(car.charAt(L-1));
```

Question 17

In the code segment to the right, which of the following numbers could NOT be printed?

A) `22` **B)** `24` **C)** `26` **D)** `28` **E)** `30`

```
int T = (int)(Math.random()*7) + 22;
System.out.print(T);
```

Question 18

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

A) `15` **B)** `12` **C)** `4` **D)** `20` **E)** `7`

```
out.print(12 & 7 + 8 ^ 11);
```

Question 19

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

A) `1` **B)** `8` **C)** `7` **D)** `5` **E)** `0`

```
int[][] w = {{5,1,2},{8,0,6},{7,1,3}};
out.print(w[2][1]);
```

Question 20

In the code segment to the right, in line #1, if **<???>** was replaced by 2, what would the output be?

A) 8 **B)** 7 **C)** 5 **D)** 3 **E)** 9

Question 21

In the code segment to the right, in line #1, if **<???>** was replaced by 6, what would the output be?

A) 8 **B)** 7 **C)** 5 **D)** 3 **E)** 9

Question 22

In the code segment to the right, in line #1, if **<???>** was replaced by L-1, what would the code do to the list?

A) It would set all values of the list to 8

B) It would set all values of the list to 9

C) It would sort the list

D) It would delete all odd numbers from the list

E) It would reverse the order of the numbers

```
int[]jenny = {8,6,7,5,3,0,9};
int box;
int L = jenny.length;
int N = <???>;    // line #1

for(int x=1; x<=N; x++)
  for(int y=0; y<=L-2; y++)
    if (jenny[y] > jenny[y+1])
      {
        box = jenny[y];
        jenny[y] = jenny[y+1];
        jenny[y+1] = box;
      }
out.print(jenny[2]);
```

Question 23

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

A) 2 **B)** B **C)** 10 **D)** D **E)** 34

```
int x = 2 << 5;
x++;
++x;
System.out.print((char) x);
```

Question 24

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

A) -33

B) -15

C) 16

D) 17

E) -16

```
int A = 5;
 for(int x = 0; x < 10; x++)
   switch(x)
   {
      case 0: A++;  break;
      case 1: A += 11;
      case 2: A = -A; break;
      case 3: A++; A++;  break;
      case 4: A/=2;
      case 5: A*=2; break;
      case 6: A = -A; break;
      case 7: A++;
      case 8: A++; break;
    }
 out.print(A);
```

Question 25

What is returned by the method call `Go(2)`

A) 1 **B)** 2 **C)** 3 **D)** 4 **E)** 5

Question 26

What is returned by the method call `Go(3)`

A) 9 **B)** 12 **C)** 30 **D)** 15 **E)** 18

Question 27

What is returned by the method call `Go(33)`

A) 165 **B)** 163 **C)** 161 **D)** 159 **E)** 157

```
public static int Go(int x)
{
  if (x==0)
    return 10;
  if (x < 3)
    return x * 2;
  else
    return Go(x-1) + 5;
}
```

Question 28

In the code to the right, what is output on line #1?

A) 12 **B)** 24 **C)** 36 **D)** 48 **E)** null

Question 29

In the code to the right, what is output on line #2?

A) [12, 24, 48, 72]

B) [12, 24, 72]

C) [12, 24]

D) [12, 36]

E) [12, 36, 60]

Question 30

IIn the code to the right, what is output on line #3?

A) [36, 60, 72]

B) [36, 72]

C) [36]

D) [36, 48, 60]

E) [12, 36, 60, 72]

```
Stack<Integer> tall;
tall = new Stack<Integer>();
Stack<Integer> shorter;
shorter = new Stack<Integer>();

tall.push(12);
tall.push(24);
shorter.push(36);
tall.push(48);
out.println(shorter.peek());//line 1

tall.push(60);
shorter.push(tall.pop());
tall.push(72);
shorter.push(tall.peek());
tall.pop();
tall.pop();
out.println(tall);          // line 2
out.println(shorter);       // line 3
```

Question 31

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

A) 8 **B)** 9 **C)** 10 **D)** 11 **E)** 12

```
int x = 8;
for(x = 15; x>=12; x++)
  x = x - 3;
out.print(x);
```

Question 32

In the code to the right, how many class variables does the Dog class contain?

A) 2 **B)** 3 **C)** 4 **D)** 1 **E)** 0

Question 33

In the code to the right, what is the resulting output caused by line #1?

A) 12

B) 22

C) 34

D) 46

E) 80

Question 34

In the code to the right, what is the resulting output caused by line #2?

A) 9

B) 11

C) 13

D) 15

E) 17

```
public class Dog
{
  private int A;
  private int B;

  public Dog()
   {
     A = 11;
     B = A * 2;
   }

  public Dog(int C)
   {
     B = C;
     A = B - 4;;
   }

  public void display()
   {
     A++;
     B +=A;
     out.println(A + B);
   }

 }

//client code
Dog R = new Dog();
R.display();          // line 1
Dog S = new Dog(7);
S.display();          // line 2
```

Question 35

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

A) 770 **B)** 78 **C)** 66 **D)** 846 **E)** 902

```
int T = 0;
for(char x = 'A'; x <= 'L'; x++)
  T += x;
out.print(T);
```

Question 36

If the letters to the right were inserted into an initially empty binary search tree in the order shown, how many leaves would the resulting tree contain?

A) 9 **B)** 10 **C)** 12 **D)** 1 **E)** 19

```
A B C D E F G H I J J I H G F E D C B A
```

Question 37

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

A) 120 **B)** 24 **C)** 72 **D)** 504 **E)** 3024

```
int N = 123456789;
int C = 1;
do
{
   C *= N % 10;
   N /= 10;
}
while (N > 1000000);
out.println(C);
```

Question 38

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

A) 5 **B)** 7 **C)** 9 **D)** 0 **E)** 1021

```
int A = 5;
int B = 7;
int C = 9;
int D = 0;
for (int x = 1; x <= 1000; x++)
{
  D = A;
  A = B;
  B = C;
  C = D;
}
out.print(A);
```

Question 39

After the code to the right is completed, what letter will be at the front of the queue?

```
add A
add B
add C
remove
remove
add D
add E
remove
add F
remove
add G
add H
add I
remove
remove
add J
```

Question 40

Of the 8 possible ordered triplets (example 000), how many will make the expression at the right true?

$$\overline{A * B} \ * \ (A + C)$$

★ANSWER KEY – CONFIDENTIAL★

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

1) B

2) D

3) A

4) B

5) A

6) D

7) B

8) D

9) E

10) E

11) D

12) A

13) D

14) C

15) C

16) E

17) E

18) E

19) A

20) D

21) C

22) C

23) B

24) A

25) D

26) A

27) D

28) C

29) C

30) A

31) D

32) E

33) D

34) D

35) D

36) B

37) D

38) B

*39) G

*40) 4

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

Note: Correct responses are based on **Java SE Development Kit 17 (JDK 17)** 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 12 Standard Packages have been imported. Ignore any typographical errors and assume any undefined variables are defined as used.

Explanations:

1.	B	Convert all to Base 10 and then compare. $100101_2 = 37_{10}$ $56_8 = 46_{10}$ $26_{16} = 38_{10}$ $46_7 = 34_{10}$ $1A_{12} = 22_{10}$
2.	D	Use order of operations. Perform integer division first, then add left to right. 15 + 5 / 4 + 1 15 + 1 + 1 = 17
3.	A	A new line is invoked after each println statement There will be a new line after "Two" and after "Four"
4.	B	str.substring(A,B) This will return a string of characters beginning at position A They continue up to, but not including position B. str.substring(2,3) will therefore only contain "i"
5.	A	M || true && N First, evaluate true && N which is true && false = false Now evaluate M || false which is true || false = true
6.	D	Math.floor(5.85) returns the value 5.0 (int) type casts that value as an integer 5
7.	B	In the expression x / a + y * y since a is a double, x/a will yield a double value 3.5 3.5 + 64 = 67.5
8.	D	R = 7 and V =9, so W will have the value of 2 The first if has a false condition and will cause no output The second also has a false condition, but its else will print a V The third if has a true condition and will print VRM
9.	E	The loop would print 1 3 5 7 9 if it was out.print(x + " ") Since it prints x*x, we get the square of each of those numbers.
10.	E	The first element in an array is at position 0. We are multiplying the elements in positions 1 and 4 17 * 5 = 85
11.	D	Each of the four t.next() statements access a different String within the Scanner String. The second one, "CH", is the initial value of st. The fourth one, "GAN", is added to the end of String st.
12.	A	This loop finds the sum of all odd numbers in the range 1 to 20 1+3+5+7+9+11+13+15+17+19 = 100 Fun fact: The sum of all consecutive odd numbers starting with 1 is always a perfect square.
13.	D	a << 2 + b >> 1 + ++c a << 2 + b >> 1 + **++c** a << 2 + b >> 1 + 5 a << **2 + b** >> 1 + 5 a << 6 >> 1 + 5 a << 6 >> **1 + 5** a << 6 >> 6 **a << 6** >> 6 640 >> 6 10
14.	C	SIZE represents the number of bits used to store a particular data type. Integer.SIZE is 32 Know as many of these as you can.

15.	C	Here is the progression of list. [ ] [11] [11, 22] [11] [11, 33] [11, 33, 44] [11, 44] [11, 44, 55] [11, 44, 55, 66] [11, 55, 66]
16.	E	The index of "KL" in car is 5 The problem then outputs the character in position 4
17.	E	(int)(Math.random()*7) + 22 - This generates numbers included in the following set: {22,23,24,25,26,27,28} - a list that starts with 22 and has 7 elements. Therefore 30 cannot be generated.
18.	E	Order of precedence tells us to add 7+8 first Now we have 12 & 15 ^ 11 Convert all to binary. 1100 & 1111 ^ 1011 AND has priority over XOR 1100 ^ 1011 This gives us 0111 = 7
19.	A	w[2][1] is accessing the element in list #2, item #1. Remember that lists and items are numbered starting with 0 So, 1 is the answer.
20.	D	This is the code for a version of the bubble sort. 8 6 7 5 3 0 9 - original list 6 7 5 3 0 8 9 - after 1st pass through the list 6 5 3 0 7 8 9 - after 2nd pass through the list Item #2 is 3
21.	C	This is the code for a version of the bubble sort. 8 6 7 5 3 0 9 - original list 6 7 5 3 0 8 9 - after 1st pass through the list 6 5 3 0 7 8 9 - after 2nd pass through the list 5 3 0 6 7 8 9 - after 3rd pass through the list 3 0 5 6 7 8 9 - after 4th pass through the list 0 3 5 6 7 8 9 - after 5th pass through the list 0 3 5 6 7 8 9 - after 6th pass through the list Item #2 is 35
22.	C	This is a version of the bubble sort.
23.	B	2 << 5 performs a bitwise left shift 5 This sets x = 64 Each of the next two lines add one to x giving us 66 The output typecasts 66 as a character = B

24.	A	A=5
		Loop iterations
		x=0 A = 6
		x=1 A = 17 then A = -17
		x=2 A = 17
		x=3 A = 18 then A = 19
		x=4 A = 9 then A = 18
		x=5 A = 36
		x=6 A = -36
		x=7 A = -35 then A = -34
		x=8 A=-33
		x=9 No Change

25.	D	Go(2) does not recurse. The second if returns us a value of 4

26.	A	Go(3) recurses Go(3) = Go(2) + 5 Go(2) = 4 So, Go(3) is 9
27.	D	Go(33) recurses Go(33) = Go(32) + 5 Go(32) = Go(31) + 5 Go(31) = Go(30) + 5 continues Go(3) = Go(2) + 5 Go(2) = 4 5 is added with each call. There are 31 calls from 33 to 3 So, 4 + 31(5) = 159
28.	C	At this point, shorter has only one value When we print shorter.peek() it prints 36
29.	C	Here is the evolution of tall [ ] [12] [12, 24] [12, 24, 48] [12, 24, 48, 60] [12, 24, 48] [12, 24, 48, 72] [12, 24, 48] [12, 24]
30.	A	Here is the evolution of shorter [ ] [36] [36, 60] [36, 60, 72]

31.	D	Here is the step-by-step evolution of x
		x = 8
		x = 15
		Is (x>=12)? Yes
		x = 12
		x = 13
		Is (x>=12)? Yes
		x = 10
		x = 11
		Is (x>=12)? No
		Print 11

32.	E	The Dog class has no class variables, both A and B are instance variables. The key word to look for on class variables is "static"
33.	D	Doing a little algebra, one can see that the display method will output 2A + B + 2 For R, A=11 and B = 22 2(11) + 22 + 2 = 46
34.	D	Doing a little algebra, one can see that the display method will output 2A + B + 2 For S, A=3 and B = 7 2(3) + 7 + 2 = 15
35.	D	The loop goes through all the letters A - J and takes a sum of the ASCII values. Thus, it add the numbers 65 through 76 getting a sum of 846.
36.	B	As the tree is created, each new node is a leaf that becomes the right child of the bottom-most node which loses its "leaf status". So after the first 10 nodes, there is only one leaf. Then as the next ten nodes are added, the first is added to the left of the J leaf, but the next nine are added to the left of nodes that are not leafs. Thus, we will have 10 leafs.
37.	D	With each iteration of the loop, C is multiplied by the ones digit. N is then divided by 10, removing the ones digit. This will stop after three iterations. 9 * 8 * 7 = 504

38.	B	With each pass through the loop, the values of A, B, and C rotate amongst themselves with D serving as a helper.
		A=5 B=7 C=9 Original List
		A=7 B=9 C=5 After Pass #1
		A=9 B=5 C=7 After Pass #2
		A=5 B=7 C=9 After Pass #3
		After every 3 passes, the numbers are back in the original order.
		After 999 passes, they are in the original order.
		After one more pass, A=7 B=9 C=5

39.	G	Here is the evolution of the queue: [A] [A, B] [A, B, C] [B, C] [C] [C, D] [C, D, E] [D, E] [D, E, F] [E, F] [E, F, G] [E, F, G, H] [E, F, G, H, I] [F, G, H, I] [G, H, I] [G, H, I, J]
40.	4	Using DeMorgan's Law on the first part of the expression, then finding the "product" of the binomials is a good route to take. But, we can always just inspect the two terms. Since there is an AND statement, both parts must be true. $\overline{A * B}$ - A and B cannot both be true (This eliminates 110 and 111) Either A or C has to be true. (This eliminates 000 and 010) Four combinations work: 000, 010, 110, and 111