UIL Computer Science Written Test

2023 District

March 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)
```

STANDARD CLASSES AND INTERFACES – SUPPLEMENTAL REFERENCE

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 DISTRICT

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

What is the product of 7_{10} and 101011_2?

A) 455_8 **B)** 545_8 **C)** 544_8 **D)** 454_8 **E)** 445_8

Question 2

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

A) `8` **B)** `4` **C)** `5` **D)** `6` **E)** `56`

```
out.print(35 % 12 / 4 + 27 * 2 % 4);
```

Question 3

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

A)
```
5.6
Five
SixSeven
Eight
```

B)
```
5.6Five
SixSeven
Eight
```

C)
```
5.7Five
SixSeven
Eight
```

D)
```
5.7
Five
SixSevenEight
```

E)
```
5.7
Five
SixSeven
Eight
```

```
double A = 5.678;
out.printf("%.1f\n",A);
out.print("Five\nSix");
out.println("Seven\nEight");
```

Question 4

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

A) `0` **B)** `2` **C)** `3` **D)** `7` **E)** `-1`

```
String str = "University";
String yes = str.substring(3,7);
out.print(yes.indexOf("i"));
```

Question 5

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

A) `true`

B) `false`

```
boolean A = !true;
boolean B = !A;
boolean C = A && B;
out.print(A ^ B ^ C);
```

Question 6

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

A) `0` **B)** `7` **C)** `-7` **D)** `-7.0` **E)** `7.0`

```
double H = Math.pow(5,2)-Math.pow(2,5);
out.print(Math.abs(H));
```

Question 7

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

A) 1 **B)** 1.0 **C)** 1.5 **D)** 1.75 **E)** 2.0

```
double U = 70 / 20;
double T = (U +  U) / 4;
out.print(T);
```

Question 8

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

A) 3

B) 4

C) 5

D) 19

E) 25

```
int T = 12;
if (T > 10)
   T = T - 3;
if (T == 12)
   T *= 2;
else
   T /= 2;
if (T - 10 < 0)
   T++;
else
   T--;
out.print(T);
```

Question 9

What is the last number printed by the loop to the right?

A) 127

B) 128

C) 199

D) 200

E) 255

```
for(int x = 1; x < 200; x = x * 2 + 1)
   out.println(x);
```

Question 10

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

A) 4 **B)** 7 **C)** 22 **D)** 25 **E)** 28

```
int[] ten = new int[10];
for(int x=0; x<10; x++)
  ten[x] = x * 3 + 4;
out.print(ten[7]);
```

Question 11

What is output by the code segment to the right?

A) ABCDEFGH

B) BCGHJK

C) BCGHJKO

D) ADEFILMN

E) ADEFI

```
String St = "A BC DEF GH I JK LMN O";
Scanner B = new Scanner(St);
for(int x=0; x<=2; x++)
 {
    B.next();
    out.print(B.next());
 }
```

Question 12

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

A) 0 **B)** 4 **C)** 6 **D)** 7 **E)** 55

```
int total = 0;
for(int x = -10; x <= 10; x++ )
  if (x > -3 && x < 3)
     total += x;
out.print(total);
```

Question 13

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

A) 2324696

B) 2324520

C) 2324588

D) 2224696

E) 2224624

```
int W = 22;
out.print(W++);
out.print(++W);
out.print(W>>2);
out.print(W<<2);
```

Question 14

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

A) 8 **B)** 16 **C)** 32 **D)** 64 **E)** 128

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

Question 15

What is output by the code segment to the right?

A) [11, 22, 33, 55]

B) [11, 55, 44, 22]

C) [55, 22, 33, 22]

D) [44, 22, 33, 11]

E) [44, 55, 11, 22]

```
ArrayList<Integer> eagle;
eagle = new ArrayList<Integer>();
eagle.add(11);
eagle.add(0,22);
eagle.add(33);
eagle.add(0,44);
eagle.add(eagle.get(1));
eagle.set(1,55);
eagle.remove(3);
out.print(eagle);
```

Question 16

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

A) 120 **B)** 240 **C)** 250 **D)** 256 **E)** 480

```
int E = 500;
for(int x=1; x<=5; x++)
  E /= 2;
for(int x=1; x<=4; x++)
  E *= 2;
out.print(E);
```

Question 17

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

A) 2400

B) 2800

C) 3200

D) 3500

E) 1000000

```
int A = 0;
for(int x=1; x<=100; x++)
  for(int y=1; y<=100; y*=2)
    for(int z=1; z<=100; z*=3)
       A++;
out.print(A);
```

Question 18

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

A) 25 **B)** 48 **C)** 77 **D)** 98 **E)** 129

```
String St = "ABCDE";
for (int x=1; x<=5; x++)
  St = St + St.substring(1,St.length()-1);
out.print(St.length());
```

Question 19

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

A) 0 **B)** 6 **C)** 10 **D)** 13 **E)** 20

```
boolean[] F = new boolean[20];
F[1] = true;
for(int x=2; x<=19; x++)
  F[x] = F[x-1] ^ F[x-2];
int C = 0;
for (int x=0; x<=19; x++)
   if (F[x])
     C++;
out.print(C);
```

Question 20

In the code segment to the right, what is the output of line 1?

A) 70 **B)** 60 **C)** 50 **D)** 40 **E)** 30

Question 21

In the code segment to the right, what is the output of line 2?

A) 35 **B)** 36 **C)** 38 **D)** 42 **E)** 50

Question 22

In the code segment to the right, what is the output of line 3?

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

```
public class GotIt
{
   private int A;
   private int B;
   private int C;

   public GotIt(int H)
   {
      A = H;
      B = H * 2;
      C = B * 2;
   }

   public GotIt()
   {
      A = 5;
      B = 11;
      C = 20;
   }

  public int SendIt()
  {
     return A + B + C;
  }
}
/////////////////////////////////////
// Client code
GotIt Bob = new GotIt(10);
GotIt Ann = new GotIt();
GotIt Ted = new GotIt(0);
out.print(Bob.SendIt()); // Line 1
out.print(Ann.SendIt()); // Line 2
out.print(Ted.SendIt()); // Line 3
```

Question 23

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

A) 4 **B)** 24 **C)** 26 **D)** 27 **E)** 54

```
out.print(0b11011);
```

Question 24

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

A) 19 **B)** 21 **C)** 30 **D)** 32 **E)** false

```
int A = 32;
int B = 21;
out.print(A < B ? B : A - 2);
```

Question 25

What is returned by the method call Wow("GO")?

A) GO

B) XGOX

C) UIL

D) XUILX

E) XX

Question 26

What is returned by the method call Wow("STOP")?

A) STOP

B) XSTOX

C) STO

D) XTOPX

E) XOX

Question 27

What is returned by the method call Wow("COMPUTER")?

A) XCOMX

B) XOMPX

C) XMPUX

D) XPUTX

E) XUTEX

```
public static String Wow(String W)
  {
    if (W.length()<3)
      return "UIL";
    else if (W.length()==3)
      return "X" + W + "X";
    else if (W.length()>6)
      return Wow(W.substring(1,4));
    else
      return Wow(W.substring(1));
  }
```

Question 28

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

A) 10 **B)** 9 **C)** 8 **D)** 7 **E)** 6

Question 29

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

A) 13 **B)** 11 **C)** 9 **D)** 7 **E)** 6

Question 30

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

A) 20 **B)** 6 **C)** 15 **D)** 7 **E)** 8

```
Stack<Integer>Bob;
Bob = new Stack<Integer>();
for (int x=6; x<=10; x++)
  Bob.push(x);
out.print(Bob.peek());      //Line 1
for (int x=11; x<=13; x++)
  Bob.pop();
out.print(Bob.pop());       //Line 2
for (int x=14; x<=20; x++)
  Bob.push(x);
out.print(Bob.size());      //Line 3
```

Question 31

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

A) 10 seconds

B) 24 seconds

C) 25 seconds

D) 48 seconds

E) 50 seconds

The Big O Notation for a sorting routine is O(n^2). When we sort a list of 3000 numbers, the process takes 2 seconds. How long do we predict the same sort will work on a list of 15,000 numbers?

Question 32

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

A) 2

B) 4

C) 6

D) 8

E) 10

Question 33

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

A) 1

B) 3

C) 5

D) 7

E) 9

Question 34

In the code to the right, what is output by line #3?

A) 0

B) 1

C) 2

D) 3

E) 4

```
ArrayList<String>Words;
Words = new ArrayList<String>();
Words.add("MILK");
Words.add("EGGS");
Words.add("BUTTER");
Words.add("OLEO");
Words.add("APPLE");
Words.add("PIE");
Words.add("STRAWBERRIES");
Words.add("BANANA");
Words.add("YOGURT");
Words.add("HAM");
int A = 0;
int B = 0;
int C = 0;
for(String T:Words)
  if(T.contains("A"))
    A++;
for(String T:Words)
  if(T.indexOf("T")==-1)
    B++;
for(String T:Words)
  if(T.matches("..E.*"))
    C++;

out.print(A);     // Line 1
out.print(B);     // Line 2
out.print(C);     // Line 3
```

Question 35

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

A) 125 **B)** 152 **C)** 215 **D)** 251 **E)** 521

```
int L = 512;
int M = L / 100;
int N = (L - M * 100) / 10;
int R = L % 10;
int U = R*100 + M*10 + N;
out.print(U);
```

Question 36

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

A) 0 **B)** 9 **C)** 16 **D)** 22 **E)** 25

```
int A = 9;
int B = 16;
int C = 25;
out.print(A ^ B ^ C);
```

Question 37

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

A) 9

B) 11

C) 13

D) 15

E) 17

```
String A = new String("HORSE");
String B = new String("PIG");
String C = A;
A = B;
B = C;
int AL = A.length();
int BL = B.length();
int CL = C.length();

out.print(AL+BL+CL);
```

Question 38

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

A) 22A37BA12

B) 22A2710B77

C) 22A2710BA12

D) 22A2710BA12

E) 22A37BAB

```
int A = 10;
int B = 12;
int C = 27;
out.print(A+B+"A"+C+A+"B"+('A'+B));
```

Question 39

Evaluate the postfix expression to the right. Write your answer in the answer blank for #39.

3 4 + 5 * 20 - 5 / 4 ^

Question 40

Create the binary search tree using the letters in the word to the right. After building the tree, determine how many leaf nodes the tree contains.

TRAMPOLINE

★ANSWER KEY – CONFIDENTIAL★

UIL Computer Science – 2023 District

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

1) A

2) B

3) E

4) E

5) A

6) E

7) C

8) C

9) A

10) D

11) B

12) A

13) D

14) C

15) E

16) B

17) D

18) D

19) D

20) A

21) B

22) E

23) D

24) C

25) C

26) D

27) B

28) A

29) D

30) E

31) E

32) B

33) D

34) C

35) D

36) A

37) C

38) B

*39) 81

*40) 2

** 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 17 Standard Packages have been imported. Ignore any typographical errors and assume any undefined variables are defined as used.

Explanations:

1.	A	Convert the base two number to base 10. $101011_2 = 43_{10}$ $43_{10} * 7_{10} = 301_{10}$ $301_{10} = 455_8$ (4 * 64) + (5 * 8) + (5 * 1) = 256 + 40 + 5 = 301 Actually, a cooler solution is to convert the binary number to octal and do the arithmetic there. If you learn this method, there are fewer chances to make conversion errors.
2.	B	Use order of operations. 35 % 12 / 4 + 27 * 2 % 4 Perform integer modulus first. 11 / 4 + 27 * 2 % 4 Next do integer division. 2 + 27 * 2 % 4 Now, multiply. 2 + 54 % 4 Again with the integer modulus. 2 + 2 2 + 2 = 4

3.	E	out.printf("%.1f\n",A);
		This prints A formatted with one decimal place (rounded) followed by a carriage return
		out.print("Five\nSix");
		On the next line, this prints "Five" on the next line, then "Six" on the following line because of the "\n"
		out.println("Seven\nEight");
		This prints "Seven" next to "Six." The \n sends "Eight " to the last line.

4.	E	yes.substring(3,7) is "vers" (starts at position 3 and stops before position 7) So, there is no "i" in yes giving us a value of -1 Cool hint, when you spot a substring with 2 arguments, subtract the numbers. This will tell you how many characters will be returned.
5.	A	A is not true, or false. B is not A, or true A && B is false (two trues are needed) So, C is false. A ^ B ^ C becomes false ^ true ^ false Work left to right keeping in mind that ^ needs a true and a false to be true. **false ^ true** ^ false true ^ false = true
6.	E	Math.pow(5,2) returns 25.0 Math.pow(2,5) returns 32.0 25.0 - 32.0 is -7.0 Math.abs(-7.0) returns the absolute value of -7.0 which is 7.0
7.	C	The key to the problem is that U has the value of 3.0. 70/20 performs integer division, then the value is handed to U as a double.. (3.0 + 3.0) is 6.0 6.0 / 4 is 1.5 as real number division takes place
8.	C	The first if is true so T becomes 9 Since T is 9 now, the second if is false, the else makes T the value of 4. Now T is 4, so the final if is true causing one to be added to T. T finishes the adventure at 5.
9.	A	x = x*/ 2 + 1 The values of x that would be printed are: 1 3 7 15 31 63 127

10.	D	The array called ten begins with the following values: {0,0,0,0,0,0,0,0,0,0}. With each iteration of x in the loop, the xth position takes on the value of 3x + 4. The array becomes: {4,7,10,13,16,19,22,25,28,31} Therefor ten[7] is 25
11.	B	St = "A BC DEF GH I JK LMN O"; The Scanner called B accesses the String St. Inside the loop, the first B.next() skips a value. The print statement prints the next. The loop iterates three times giving us BC,GH, and JK So, we get BCGHJK as the output
12.	A	The loop runs through the values -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 If an x is larger than -3 and smaller than 3, it is added to the total. -2 -1 0 1 and 2 are added to that total. The sum is zero.
13.	D	The first print statement prints 22, then adds one to W. The next, adds one more, then prints 24. The next, uses integer division twice, printing the value 6. W does not change. It is still 24. The last, doubles the value 24 twice, printing 96.
14.	C	In Java, 32 bits (4 bytes) are used to store integer data
15.	E	Here is the progression of eagle. [ ] [11] [22,11] [22,11,33] [44,22,11,33] [44,22,11,33,22] [44,55,11,33,22] [44,55,11,22]
16.	B	E begins at 500. It then undergoes being divided by 2 five consecutive times, 500 - 250 - 125 - 62 - 31 - 15 Then, starting at 15, we double it four times. 15 - 30 - 60 - 120 - 240
17.	D	Be careful on this one. It is easy to "underestimate" the y and the z loops. The x loop iterates 100 times - no problem. The y loop iterates 7 times - 1,2,4,8,16,32,64 The z loop iterates 5 times - 1,3,9,27,81 100*7*5 = 3500
18.	D	The value of St does not matter, only the length. With each iteration, we add St.length()-2 characters to St. 0 - length is 5 1 - length is 8 (5+3) 2 - length is 14 (8 + 6) 3 - length is 26 (14 + 12) 4 - length is 50 (26 + 24) 5 - length is 98 (50 + 48)
19.	D	For purposes of brevity, I will use 0 for false and 1 for true. [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] = original array Then we set F[1] = true. [0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] In the loop, to get the value of F[2], do F[0] ^ F[1] = false ^ true = true [0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] In the loop, to get the value of F[3], do F[1] ^ F[2] = true ^ true = false [0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] In the loop, to get the value of F[4], do F[2] ^ F[3] = true ^ false = true [0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0] Watch the cool pattern emerge as you trace until the end. [0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1,1,0,1] There are 13 trues.

20.	A	Bob uses the one-parameter constructor. 10 is sent in, assigning these values: A=10, B=20, C=40 SendIt returns the sum of the three … 70
21.	B	Ted uses the zero-parameter constructor. It assigns these values: A=5, B=11, C=20 SendIt returns the sum of the three … 36
22.	E	Ann uses the one-parameter constructor. 0 is sent in, assigning these values: A=0, B=0, C=0 SendIt returns the sum of the three … 0
23.	D	This takes the binary number 11011 and prints the base 10 value. That value is 27 (16+8+2+1)
24.	C	A is not less than B, so the statement will print A-2, which is 30.
25.	C	Wow("GO") with a length of 2, goes straight to the first if and returns "UIL"
26.	D	Wow("STOP") with a length of 4, goes to the last return Wow("STOP") = Wow("TOP") Then, with length of 3, the second return is invoked giving us "XTOPX"
27.	B	Wow("COMPUTER") has a length greater than 6. Wow("COMPUTER") = Wow("OMP") Then, with length of 3, the second return is invoked giving us "XOMPX"
28.	A	At the time of Line 1, Bob is [6,7,8,9,10] Bob.peek() returns the value of the top item … 10
29.	D	The second loop pops three items to give us [6,7] Then, on line 2 we print and pop. That item is the 7.
30.	E	Moving onward, we have [6] The last loop pushes 7 more numbers onto the stack giving us [6,14,15,16,17,18,19,20] The final stack size is 8.
31.	E	First, look at the size of the original list and the size of the second list. Here we see that the first list has 3000 items and the second list has 15,000. The second list is 5 times as big. 5 will be our n. The Big O is O(**n^2).** Plugging in 5 for n gives us 25. Theoretically, it should take the process 25 times as long. The original took 2 seconds. 50 seconds is our answer.

32.	B	`Words.add("MILK");`
		`Words.add("EGGS");`
		`Words.add("BUTTER");`
		`Words.add("OLEO");`
		`Words.add("APPLE");`
		`Words.add("PIE");`
		`Words.add("STRAWBERRIES");`
		`Words.add("BANANA");`
		`Words.add("YOGURT");`
		`Words.add("HAM");`
		This counts the words in the list that contain at least one "A"
		4 = APPLE, STRAWBERRIES, BANANA, HAM

33.	D	This counts the words in the list that do not contain at least one "T" 7 = All but BUTTER, STRAWBERRIES, and YOGURT
34.	C	This counts the words in the list that have an E in String position 2. There may be 0 or more characters of any type after the E. 2 = PIE, OLEO

35.	D	I have a feeling the programmer had hopes of reversing the integer, but messed up along the way.
		M = 512 / 100 = 5
		N = (512 - 500) / 10 =1
		R = 512 % 10 = 2
		OK. The digits are separated.
		But, the U formula does something odd.
		R = 2*100 + 5*10 + 1 = 251

36.	A	First, convert each number to binary. A = 01001 B = 10000 C = 11001 Notice that we added a leading zero to A so that all numbers would be 5 bits long. 01001 ^ 10000 ^ 11001 Work left to right. 01001 ^ 10000 = 11001 Hmm. 11001 ^ 11001 = 00000 = 0

37.	C	Step by step
		A = "HORSE"
		B = "PIG"
		C = "HORSE"
		A = "PIG"
		B = "HORSE"
		We end up with 2 HORSEs and 1 PIG.
		The sum of the lengths is 13.

38.	B	`int A = 10;`
		`int B = 12;`
		`int C = 27;`
		`out.print(A+B+"A"+C+A+"B"+('A'+B));`
		Note: 'A' + B will have a value of 77. The ASCII code for 'A" is 65.
		
		We will add A+B at the beginning because it comes before any mention of Strings.
		We will not use arithmetic to add C and A since it is embedded in the Strings.
		
		22 A 27 10 B 77
		22A2710B77 is the output

39.	81	**3 4 +** 5 * 20 - 5 / 4 ^ **7 5 *** 20 - 5 / 4 ^ **35 20 -** 5 / 4 ^ **15 5 /** 4 ^ **3 4 ^** **81**
40.	2	After building the binary search tree, only the E and the N have no "children."