UIL Computer Science Written Test

2019 Region

April 2019

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 – 2019 REGION

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 01010011_2 and 10000011_2?

A) $D5_{16}$ **B)** 11010010_2 **C)** $E6_{16}$ **D)** 326_8 **E)** 201_{10}

Question 2.

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

A) -1 **B)** 123 **C)** 120 **D)** -123 **E)** 1

```
out.print(6*(122-81)/5%-3);
```

Question 3.

What is the output of the code segment to the right? # indicates a blank space.

A) `###-58,213`

B) `##(58,213)`

C) `-58,213###`

D) `(,####-58213`

E) `(58,213.0)`

```
out.printf("%(,10d",-58213);
```

Question 4.

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

A) `-59` **B)** `2` **C)** `-107` **D)** `84` **E)** `-73`

```
String s1="25";
String s2="mnop";
out.print(s1.compareTo(s2));
```

Question 5.

Which of the lines shown on the right will print `false`?

A) line #1

B) line #2

C) line #3

D) line #4

E) More than one of the above.

```
boolean a=true,b=true,c=true;
out.println(a&&b||c);//line #1
out.println(a||b&&c);//line #2
out.println(a^b&&c);//line #3
out.println(a&&!b^c);//line #4
```

Question 6.

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

A) `12` **B)** `64` **C)** `81` **D)** `81.0` **E)** `64.0`

```
int m=4,n=3;
out.print(Math.pow(m, n));
```

Question 7.

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

A) `12.228` **B)** `12.0` **C)** `12` **D)** `12.4` **E)** `12.2`

```
int e=(int)(14.48+12.5);
double f=e+(int)35.14;
int g=5;
out.print(f/g);
```

Question 8.

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

A) `b`

B) `a d`

C) `c d`

D) `b d`

E) `d`

```
int o=8,p=-3;
if(o*p<-24)out.print("a ");
if(p-o==-11)out.print("b ");
if(p>0)out.print("c ");
else out.print("d ");
```

Question 9.

Which of the following code segments will NOT print 10 asterisks?

A.

```
int h=10;
do {
     h--;
     out.print("*");
}while(h>1);
```

B.

```
for(int i=0;i<=9;i++)
     out.print("*");
```

C.

```
int j=9;
while(j>=0) {
     out.print("*");
     j--;
}
```

D.

```
for(int k=23;k>13;k--)
     out.print("*");
```

E.

```
int l=1000;
do {
     out.print("*");
     l/=2;
}while(l>0);
```

Question 10.

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

A) `[dog, mouse, frog]`

B) `[mouse, cat, frog]`

C) `[dog, mouse, cat, bird, frog]`

D) `[dog, cat, bird, mouse, frog]`

E) Error. Throws an ArrayIndexOutOfBoundsException.

```
String list[] = {"dog","cat","bird"};
list[1]="mouse";
list[3]="frog";
out.print(Arrays.toString(list));
```

Question 11.

The code segment shown on the right appears in a main method and it is intended to print all of the contents of the file *`datafile.dat`*. Which of the following must replace **<code>** to ensure that the segment will compile and execute as intended? Assume that all necessary classes have been imported and that the main method throws an IOException.

A) `"datafile.dat"`

B) `scanner`

C) `file`

D) `new File()`

E) No additional code is required.

```
File file=new File("datafile.dat");
Scanner scanner=new Scanner(<code>);
while(scanner.hasNext())
     out.print(scanner.next());
scanner.close();
```

Question 12.

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

A) `120`

B) `70`

C) `125`

D) `95`

E) `85`

```
int s=0;
for(int q=1;q<11;q+=2)
     for(int r=0;r<q;r++)
          s=s+r;
out.print(s);
```

Question 13.

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

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

```
int s=-8;
out.print(~++s);
```

Question 14.

Which of the following values cannot be stored in a variable that is of type byte?

A) -127 **B)** 0 **C)** -128 **D)** 128 **E)** 127

Question 15.

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

A) `[0, 6, -3, 4, 4, 5, -3]`

B) `[0, 6, -3, -1, 4, 5, -3]`

C) `[0, 6, 4, 5, -1]`

D) `[0, 6, -3, -1, 4, -3]`

E) `[0, 6, 4, 5, -1, -3]`

```
ArrayList<Integer> list=new
ArrayList<Integer>();
list.add(0);list.add(6);list.add(-3);
list.add(4);list.add(-1);list.add(5);
list.add(list.get(2));
list.set(4, list.remove(3));
out.print(list);
```

Question 16.

What is the output of the main method shown on the right?

A) `4 4`

B) `5 4`

C) `4 5`

D) `5 5`

E) There is no output due to an error.

```
public static void main(String[] args) {
      Cls i=new Cls(5);
      int j=4;
      mtd(i,j);
      out.print(i.num+" "+j);
      }
public static void mtd(Cls i,int j) {
      int t=i.num;
      i.num=j;
      j=t;
      }
public static class Cls{
      public int num;
      public Cls(int i) {num=i;}
      }
```

Question 17.

Which of these methods will correctly return the decimal equivalent of a hexadecimal character? Assume that the hexadecimal character passed to the method is always valid and is an uppercase letter.

A.

```
public static int hexToDecimal(char hex){
   if(hex>=65)
           return hex-55;
       else
           return hex;
   }
```

B.

```
public static int hexToDecimal(char hex){
   if(hex>='A')
           return 10+hex-'A';
       else
           return hex-'0';
   }
```

C.

```
public static int hexToDecimal(char hex){
   if(hex<='F')
           return 10+hex-'A';
       else
           return hex-'0';
   }
```

D.

```
public static int hexToDecimal(char hex){
   if(hex>='A')
           return hex-10+'A';
       else
           return hex+'0';
   }
```

E. More than one of the above.

Question 18.

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

A) `3`

B) `18`

C) `0`

D) `15`

E) `4`

```
int[][] mat= {{1,0,6,9,2},
              {8,5,2},
              {3,4,0,3,1,4},
              {2,7,5,4}};
int q=0,z=0;
for(int i=0;i<mat.length;i++) {
  int p=0;
  for(int j=0;j<mat[i].length;j++)
    p+=mat[i][j];
  if(p>z) {q=i;z=p;}
  }
out.print(q);
```

Question 19.

Which of the following must replace **<code>** in the code segment shown on the right?

A) `m.size()`

B) `m.containsKey()`

C) `m.getKey()`

D) `m.entrySet()`

E) `m.keySet()`

Question 20.

Once **<code>** has been replaced, what is the output of the code segment to the right?

A) `14 9 15 3`

B) `yrt jsv gfh yrt`

C) `14 3 9 15`

D) `14 15 9 3 14`

E) `yrt mbc jsv yrt aaa`

```
Map<String,Integer> m=new
TreeMap<String,Integer>();
m.put("yrt", 14);m.put("mbc", 8);
m.put("qfh", 15);m.put("jsv", 9);
m.put("yrt", 3);m.put("aaa", 14);
m.replace("mbc", 15);
m.remove("qfh");
Set<String> x=<code>;
for(String s:x)
     out.print(m.get(s)+" ");
```

Question 21.

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

A) `1`

B) `4`

C) `5`

D) `6`

E) There is no output due to an error.

```
Pattern p=Pattern.compile("[aeiou]");
String[] items=p.split("greatbigbears");
out.print(items.length);
```

Question 22.

How many class variables does the class `Circle` contain?

A) none

B) 1

C) 2

D) 3

E) 4

Question 23.

What is the output of client code **line #1** ?

A) `[3, 13, 28, 50]`

B) `[3, 13, 28, 50, 79, null]`

C) `[3, 13, 28, 28, 50, null]`

D) `[3, 13, 28, 28, 50]`

E) `[3, 13, 28, 28, 50, 3]`

Question 24.

What is the output of client code **line #2**?

A) 2

B) 3

C) 4

D) 5

E) 6

```
public class Circle {
private double radius;
private double area;
public static int var;

public Circle() {
      radius=1;
      area=Math.PI*Math.pow(radius, 2);
      var++;
      }

public Circle(double r) {
      radius=r;
      area=Math.PI*Math.pow(radius, 2);
      var++;
      }

public String toString() {
      return ""+Math.round(area);
      }
}
//client code
Circle c1=new Circle();
Circle c2=new Circle(2);
Circle c3=new Circle(3);
Circle[] a=new Circle[6];
a[0]=c1;a[1]=c2;a[2]=c3;
for(int i=Circle.var;i<a.length-1;i++)
      a[i]=new Circle(i);
out.print(Arrays.toString(a));//line #1
out.print(Circle.var);//line #2
```

Question 25.

What is the output of this line of client code given the method shown on the right?

```
        mtd("abcd");
```

A) `dcdbcdabcd`

B) `abcdbcdcdd`

C) `aababcabcd`

D) `dcbacbabaa`

E) `abcdddcbaa`

```
public static void mtd(String s) {
if(s.length()<=1) {
      out.print(s);
      return;}
else {
      mtd(s.substring(1));
      out.print(s);}
}
```

Question 26.

What is the output of this line of code?

```
        out.print(Integer.toBinaryString(0b10110001|0b10000001));
```

A) `10110100`

B) `10110001`

C) `10000001`

D) `00110000`

E) Error. Will not compile.

Question 27.

Which of the following must replace **<code 1>** in the `mtd1` method shown on the right?

A) `++count`

B) `count`

C) `count--`

D) `data.length-1`

E) `count++`

Question 28.

Which of the following must replace **<code 2>** in the `mtd2` method shown on the right?

A) `data[j]=data[j+1]`

B) `data[j+1]=data[j]`

C) `data[j]=data[i]`

D) `data[i]=data[j+1]`

E) `data[j]=data[count]`

Question 29.

If **<code 1>** and **<code 2>** have been filled in correctly, what is the output of the client code shown here?

```
AClass ac=new AClass();
ac.mtd1("moon");ac.mtd1("stars");
ac.mtd1("sun");ac.mtd1("planet");
ac.mtd2("stars");ac.mtd1("sun");
String[] list=ac.getData();
for(int i=0;i<ac.getCount();i++)
      out.print(list[i]+" ");
out.print(list.length);
```

A) `moon sun planet sun 4`

B) `moon sun planet 3`

C) `moon sun planet 8`

D) `moon stars stars sun sun planet 6`

E) `moon planet sun 3`

Question 30.

The class `AClass` implements a ________________.

A) PriorityQueue

B) Set

C) List

D) Map

E) LinkedList

```
public class AClass {

private String[] data;
private int count;

public AClass() {}

public AClass(String[] s) {
      data=s;
      count=s.length;
}

public void mtd1(String s) {
      if(data==null)
            data=new String[1];
      if(ok(s)) {
            if(count==data.length-1)
                  data=Arrays.copyOf(data,
                       data.length*2);
            data[<code 1>]=s;
      }
}

public String mtd2(String s) {
      String temp=null;
      for(int i=0;i<count;i++)
            if(data[i].equals(s)) {
                  temp=data[i];
                  for(int j=i;j<count;j++)
                        <code 2>;
                  count--;
                  }
      return temp;
}

private boolean ok(String s) {
      boolean temp=true;
      for(int i=0;i<count;i++)
            if(data[i].equals(s))
                  temp=false;
      return temp;
}

public String[] getData() {
      return data;
}

public int getCount() {
      return count;
}
}
```

Question 31.

If a particular implementation of the Quicksort algorithm uses the middle element in a list as the pivot value and is sorting in ascending order, what will be the state of the partition shown on the right just before it is divided and the Quicksort method is called again?

```
3   8   10   5   7   0   9   6   11
```

A) 11 6 0 5 7 10 9 8 3

B) 0 3 5 6 7 8 9 10 11

C) 11 10 9 8 7 6 5 3 0

D) 3 6 0 5 7 10 9 8 11

E) 3 6 10 5 7 0 9 8 11

Question 32.

Which of the following run time efficiencies is the fastest for very large values of n?

A) O(n log n) **B)** O(log n) **C)** O(n) **D)** O(n^2) **E)** O(2n)

Question 33.

How many instance variables does the object `obj2` encapsulate?

A) 0

B) 1

C) 2

D) 3

E) 4

Question 34.

What is the output of the client code segment up to and including **line #1**?

A) `string 3 string object 7 object  ing`

B) `string 3 object 7 ing`

C) `string 3 object 7 ect`

D) `string 3 string object 7 object  string`

E) `string object ing`

Question 35.

What is the output of the client code segment after **line #1**?

A) `true false false true`

B) `false true true false`

C) `true true false true`

D) `true false true true`

E) `true true true true`

```
//Use the following to answer questions 33,
//34 and 35.
public class A {

      public int x;
      public String s;

      public A(int x,String s) {
            this.x=s.length();
            this.s=s;
            System.out.print(s+" ");
      }

      public String mtd() {
            return s.substring(x/2);
      }
}
public class B extends A {

      public B(int i,String s) {
            super(i,s);
            System.out.print(i+" "+s+" ");
      }
      public String mtd() {
            return s;
      }
}
//client code
A obj1=new B(3,"string");
B obj2=new B(7,"object");
out.print(" "+obj1.mtd());//line #1
out.print((obj1 instanceof A)+" ");
out.print((obj2 instanceof A)+" ");
out.print((obj1 instanceof B)+" ");
out.print((obj2 instanceof B)+" ");
```

Question 36.

Which of the following equations correctly expresses DeMorgan's Law?

- **A)** $\overline{A + B} = \bar{A} * \bar{B}$
- **B)** $\overline{A \oplus B} = A * B + \bar{A} * \bar{B}$
- **C)** $A + B * C = (A + B) * (A + C)$
- **D)** $\overline{A * B} = \bar{A} + \bar{B}$
- **E)** More than one of the above.

Question 37.

Consider the unimplemented method shown below. What must replace ***missing code*** in the method signature to allow the method to properly sort an array of objects of unknown type?

```
public static <E missing code> void sort(E[] list) {
     //code to implement a sorting algorithm
     }
```

- **A)** `extends Comparable<E>`
- **B)** `Comparable<E>`
- **C)** `implements Comparable<E>`
- **D)** `new Comparator(E)`
- **E)** No additional code is required.

Question 38.

If the values shown on the right are placed into a binary search tree in the order shown, which value will be the root node?

- **A)** 0
- **B)** 4
- **C)** 5
- **D)** 9
- **E)** 1

```
4   8   7   1   0   9   2   5
```

Question 39.

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

```
46 4 9 * —
```

Question 40.

What is the least negative value (furthest left from zero on the number line) that can be expressed using signed 8-bit two's complement notation? Write your answer in the blank provided using 8-bit two's complement notation.