

UIL COMPUTER SCIENCE WRITTEN TEST

2021 REGION

APRIL 2021

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 – 2021 REGION

Note: Correct responses are based on **Java SE Development Kit 14 (JDK 14)** 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 14 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 represents the largest positive value?

- A) 363_8 B) 242_{10} C) $F4_{16}$ D) 11110001_2 E) 22220_3

Question 2.

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

- A) 20.0 B) 13 C) 15.5 D) 3.5 E) 17

```
out.print(15 / 6.0 * 5 + 3);
```

Question 3.

What is the output of the code segment to the right? *Spaces are shown as asterisks.*

- A) *****-3.14
B) *****3.14
C) -3.14*****
D) 3.14159265
E) 3.14*****

```
out.printf("%-10.2f", 3.1415926535);
```

Question 4.

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

- A) s B) 115 C) y
D) S E) There is no output due to an error.

```
String str = "java.lang.System.out";  
out.print(str.charAt(str.length() - 8));
```

Question 5.

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

- A) true
B) false

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

Question 6.

Consider the code segment shown on the right. Which of the following must replace **/*missing code*/** to ensure that the output will be 6.0?

- A) `Math.floor(pi + e)`
B) `Math.round(pi + e)`
C) `Math.abs(pi + e)`
D) `Math.ceil(pi + e)`
E) `Math.sqrt(pi + e)`

```
double pi = 3.14, e = 2.72;  
out.print(/*missing code*/);
```

Question 7.

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

- A) 1.955 B) 10.0955 C) 100.955 D) 101
E) There is no output due to an error.

```
int i = 100;  
double d = 95.5;  
char c = 100;  
out.print(i + d / c);
```

Question 8.

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

- A) 5 9 -5 0
- B) -4 7 -3 0
- C) 2 7 0 10
- D) 5 2 0 8
- E) 5 7 -5 8

```
int w = 5, x = 7;
int y = 0, z = 0;
if(w % x < w)
    if(x * y > z)
    {
        z = 10;
        w = 2;
    }
else
    {
        y = -5;
        x = 9;
    }
else
    if(y > x - w)
    {
        z = 8;
        x = 2;
    }
else
    {
        y = -3;
        w = -4;
    }
out.print(w + " " + x + " " + y + " " + z);
```

Question 9.

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

- A) #
- B) ##
- C) #####
- D) #####
- E) There is no output and no error.

```
int x = 11;
do {
    out.print("#");
    x--;
}while(x > 10);
```

Question 10.

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

- A) [2, 3, 0, 4, 1]
- B) [0, 4, 1, 2, 3]
- C) [0, 1, 2, 3, 4]
- D) [1, 2, 3, 3, 0]
- E) [3, 0, 4, 1, 2]

```
int []nums = {4,1,2,3,0};
for(int i = 1; i <= 3; i++)
{
    int m = nums[0];
    for(int j = 0; j < nums.length - 1; j++)
        nums[j] = nums[j + 1];
    nums[nums.length - 1] = m;
}
out.print(Arrays.toString(nums));
```

```

public class Q11
{
    public static void main(String[] args) throws IOException
    {
        File file = new File("data.dat");
        Scanner scr = new Scanner(file);
        while(scr.hasNext())
            out.print(scr.next());
        scr.close();
    }
}

```

Question 11.

For the class shown above, which of the following import statements is NOT required?

- A) import static java.lang.System.out;
- B) import java.util.Scanner;
- C) import java.io.FileNotFoundException;
- D) import java.io.File;
- E) import java.io.IOException;

Question 12.

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

- A) 1 0
- B) 49 5
- C) 72 5
- D) 57 25
- E) 49 30

```

int a = 0, b = 50;
while(b > 0 && a < 50)
{
    b -= 5;
    a = a + b / 3;
}
out.print(a + " " + b);

```

Question 13.

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

- A) 11
- B) 6
- C) 587
- D) 9
- E) 0

```

out.print(72 >> 3 | 11);

```

Question 14.

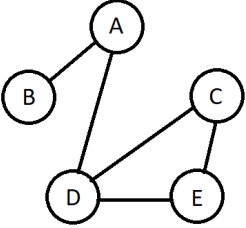
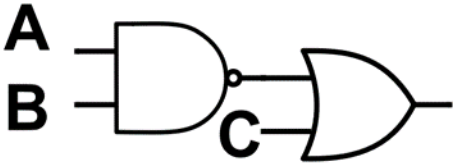
Which of the following represents the output of the code segment shown on the right?

- A) 130
- B) 127
- C) -128
- D) -126
- E) There is no output due to an error.

```

byte b = 120;
for(short m = 1; m <= 10; m++)
    b++;
out.print(b);

```

<p>Question 15.</p> <p>What is the output of the code segment to the right?</p> <p>A) [back, go, race, slide, stop, turn] B) [back, turn, slide, go, stop, race] C) [back, go, stop, race, turn, slide] D) [back, turn, go, stop, race] E) [turn, slide, back, go, stop, race]</p>	<pre>ArrayList<String> list = new ArrayList<String>(); list.add("back");list.add("go"); list.add("stop");list.add("race"); ArrayList<String> more = new ArrayList<String>(); more.add("turn");more.add("slide"); list.addAll(1,more); out.print(list);</pre>
<p>Question 16.</p> <p>What is the output of the code segment to the right?</p> <p>A) 3 e B) 4 a C) 4 k D) 4 e E) 21 e</p>	<pre>char[][] mat = {"turkey".toCharArray(), "deer".toCharArray(), "donkey".toCharArray(), "snake".toCharArray()}; out.print(mat.length + " "); out.print(mat[2][3]);</pre>
<p>Question 17.</p> <p>How many zeroes are in the adjacency matrix for the graph shown on the right?</p> <p>A) 7 B) 12 C) 20 D) 10 E) 15</p>	
<p>Question 18.</p> <p>What is the output of the code segment shown on the right?</p> <p>A) [abc, zm, o] B) [abc, tzm, no] C) [abcxytzmxyno] D) [xyt, xyn] E) [abcxy, zmxy, o]</p>	<pre>String r = "xy."; String s = "abcxytzmxyno"; Pattern p = Pattern.compile(r); String i[] = p.split(s); out.println(Arrays.toString(i));</pre>
<p>Question 19.</p> <p>What is the output of the code segment shown on the right?</p> <p>A) 14 5 12 B) 14 1 9 C) 14 0 9 D) 14 4 12 E) 13 1 9</p>	<pre>int m = 13,n = 5,o = 9; if(m++ < n + o && o * 2 >= m + n--) o += 3; else n -= 4; out.print(m + " " + n + " " + o);</pre>
<p>Question 20.</p> <p>What is the value of the Boolean expression shown in the diagram on the right if A, B and C are all true?</p> <p>A) true B) false</p>	

//Use classes June, May and April to answer questions 21 - 25.

```
public class June {
    private String s1;
    public String s2;

    public June() {out.print("Wednesday");}

    public June(String s) {s1 = s;}

    public June(String s1,String s2) {
        this.s1 = s1;
        this.s2 = s2;
    }

    public String getS1() {return s1;}
    public String getS2() {return s2;}

    public void setS1(String s1) {this.s1 = s1;}
    public void setS2(String s2) {this.s2 = s2;}

    public String concat() {return s1 + s2;}
    public String toString() {return s2 + s1;}
}
public class May extends June{
    public May() {out.print("Tuesday");}

    public May(String s) {super(s);}

    public String concat() {return s2 + super.getS1();}

    public String toString() {return <code1>getS1() + s2;}
}
public class April extends May{
    public static void main(String[] args) {
        //missing code
    }

    public April() {out.print("Monday");}

    public April(String str1,String str2) {
        super(str1);
        <code2>s2 = str2;
    }

    public String concat(String s) {return s + super.concat();}
}
```


Use the classes April, May and June listed on the previous page to answer questions 21 through 25.

Question 21.

Which of the following may replace **<code1>** in the class May?

- I. `this.` II. `super.` III. no additional code

- A) I only
- B) II only
- C) I or II
- D) II or III
- E) I, II or III

Question 22.

Which of the following is required to replace **<code2>** in the class April?

- A) `this.`
- B) `super.`
- C) `May.`
- D) `return`
- E) No additional code is required.

Question 23.

Assuming that **<code1>** and **<code2>** have both been filled in correctly, what is the output of the main method in the April class if the code segment on the right replaces **//missing code**?

- A) MondayTuesdayWednesday
- B) WednesdayTuesday
- C) TuesdayWednesday
- D) WednesdayTuesdayMondayTuesdayWednesday
- E) MondayTuesdayWednesdayTuesdayWednesday

```
April a = new April();  
a.setS1("Tuesday"); a.setS2("Wednesday");  
out.println(a);
```

Question 24.

Assuming that **<code1>** and **<code2>** have both been filled in correctly, what is the output of the main method in the April class if the code segment on the right replaces **//missing code**?

- A) SundayFridayThursdayFridayThursday
- B) ThursdayFridaySundayFridayThursday
- C) ThursdayFridaySunday
- D) SundayFridayThursdayThursdayFriday
- E) There is no output due to an error.

```
April w = new April("Thursday", "Friday");  
out.print(w.concat("Sunday"));  
out.print(w.concat());
```

Question 25.

Assuming that **<code1>** and **<code2>** have both been filled in correctly, what is the output of the main method in the April class if the code segment on the right replaces **//missing code**?

- A) `class April`
- B) `class May`
- C) `class June`
- D) `class June May April`
- E) `class April May June`

```
June x = new April("Monday", "Thursday");  
out.println(x.getClass());
```

<p>Question 26.</p> <p>Which of the following represents the output of the code segment shown on the right?</p> <p>A) Cat Eli Moe Tom Zak B) Cat Cat Eli Moe Zak C) Cat Cat Eli Moe Tom Zak D) Cat Cat Eli Moe Zak E) Cat Eli Moe Tom Zak Cat</p>	<pre>Queue<String> names = new PriorityQueue<String>(Arrays.asList("Cat", "Zak", "Moe", "Tom")); names.add("Eli"); names.offer("Tom"); names.add(names.peek()); names.remove("Tom"); while(!names.isEmpty()) out.print(names.poll() + " ");</pre>
<p>Question 27.</p> <p>Within the client code shown on the right which of the following must replace <code> to ensure that the code will compile and execute as intended?</p> <p>A) list. B) x. C) c. D) Consumer. E) <Customer></p>	<pre>public class Customer { private String name; private double amount; public Customer(String name, double amount) { this.name = name; this.amount = amount; } public String getName() {return name;} public double getAmount() {return amount;} public String toString() { return name + " " + amount + " "; } } //client code ArrayList<Customer> list = new ArrayList<Customer>(); list.add(new Customer("Bob",25.25)); list.add(new Customer("Sue",30.30)); list.add(new Customer("Zak",19.10)); list.add(new Customer("Eli",10.50)); list.add(new Customer("Avi",25.25)); Consumer<Customer> x = c -> {if(c.getAmount()>20) out.print(c);}; for(Customer cust:list) <code>accept(cust);</pre>
<p>Question 28.</p> <p>When <code> has been filled in correctly, what is the output of the client code to the right?</p> <p>A) Bob 25.25 Sue 30.3 Zak 19.1 Eli 10.5 Avi 25.25 B) Bob 25.25 Avi 25.25 Sue 30.3 C) Avi 25.25 Bob 25.25 Sue 30.3 D) Zak 19.1 Eli 10.5 E) Bob 25.25 Sue 30.3 Avi 25.25</p>	<pre>public static void main(String[] args) { int num = method(178,3); out.print(Integer.toBinaryString(num)); } public static int method(int x, int count) { for(int i = 1; i <= count; i++) x >>= 1; return x; }</pre>
<p>Question 29.</p> <p>Which of the following represents the output of the main method shown on the right?</p> <p>A) 101100 B) 10110010000 C) 10111000 D) 10110 E) 10000</p>	<pre>public static void main(String[] args) { int num = method(178,3); out.print(Integer.toBinaryString(num)); } public static int method(int x, int count) { for(int i = 1; i <= count; i++) x >>= 1; return x; }</pre>

Question 30.

Which of the following methods correctly implements an ascending Quicksort?

A.

```
public static void sort(int[] list, int first,
int last)
{
if(first >= last) return;
int left = first;
int right = last;
int pivotValue = list[(left + right)/2];
while(left < right)
{
while(list[left] < pivotValue) left++;
while(list[right] > pivotValue) right--;
if(left <= right)
{
int temp = list[left];
list[left] = list[right];
list[right] = temp;
left++;
right--;
}
}
sort(list, first, right);
sort(list, left, last);
}
```

B.

```
public static void sort(int[] list, int first,
int last)
{
if(first >= last) return;
int left = first;
int right = last;
int pivotValue = list[(left + right)/2];
while(left < right)
{
while(list[left] > pivotValue) left++;
while(list[right] < pivotValue) right--;
if(left <= right)
{
int temp = list[left];
list[left] = list[right];
list[right] = temp;
left++;
right--;
}
}
sort(list, first, right);
sort(list, left, last);
}
```

C.

```
public static void sort(int[] list, int first,
int last)
{
if(first >= last) return;
int left = first;
int right = last;
int pivotValue = list[(left + right)/2];
while(left < right)
{
while(list[left] < pivotValue) left++;
while(list[right] > pivotValue) right--;
if(left <= right)
{
int temp = list[left];
list[left] = list[right];
list[right] = temp;
}
}
sort(list, first, right);
sort(list, left, last);
}
```

D.

```
public static void sort(int[] list, int first,
int last)
{
if(first >= last) return;
int left = first;
int right = last;
int pivotValue = list[(left + right)/2];
while(left < right)
{
while(list[left] < pivotValue) left--;
while(list[right] > pivotValue) right++;
if(left <= right)
{
int temp = list[left];
list[left] = list[right];
list[right] = temp;
left++;
right--;
}
}
sort(list, first, right);
sort(list, left, last);
}
```

E. More than one of the above.

Question 31.

When creating a partition within a Quicksort, the pivot value can be the _____ element in the partition.

I. first **II.** last **III.** middle

- A)** I only
- B)** II only
- C)** III only
- D)** I or III
- E)** I, II or III

Question 32.

Which of the following represents the value stored in count after the code segment shown on the right has completed execution?

- A) n^2
- B) $\log_2(n)$
- C) $n \cdot \log_2(n)$
- D) $\frac{1}{2} \cdot n^2$
- E) $2 \cdot n^2$

```
for(int x = 1; x <= n; x++)
    for(int y = 1; y < n; y*=2)
        count++;
```

Question 33.

Which of the following is the prefix equivalent of $14 / 3 * 2 + 9$.

- A) + * / 14 3 2 9
- B) / 14 3 + * 2 9
- C) + * / 9 2 3 14
- D) / * + 14 3 2 9
- E) 14 3 / 2 * 8 +

Question 34.

Which of the following methods will always return Heads or Tails?

A.

```
public static String flip() {
    int f = (int)(Math.random()) + 2;
    switch(f) {
        case 1: return "Heads";
        case 2: return "Tails";
    }
    return "";
}
```

B.

```
public static String flip() {
    int f = (int)(Math.random() * 2) + 1;
    switch(f) {
        case 1: return "Heads";
        case 2: return "Tails";
    }
    return "";
}
```

C.

```
public static String flip() {
    int f = (int)(Math.random() * 2);
    switch(f) {
        case 1: return "Heads";
        case 2: return "Tails";
        default: return "";
    }
}
```

D.

```
public static String flip() {
    int f = (int)(Math.random() + 2);
    switch(f) {
        case 1: return "Heads";
        case 2: return "Tails";
    }
    return "";
}
```

E.

```
public static String flip() {
    int f = (int)(Math.random() * 2) + 2;
    switch(f) {
        case 1: return "Heads";
        case 2: return "Tails";
        default: return "";
    }
}
```

Question 35.

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

- A) wombats
- B) bearcats
- C) wombats bearcats
- D) wombats wobbegongs
- E) bearcats wobbegongs

```
String number = "125";
try {
    int x = Integer.parseInt(number);
    out.print("wombats ");
}
catch(NumberFormatException e){
    out.print("bearcats ");
}
finally {
    out.println("wobbegongs ");
}
```

Question 36.

Which of the following represents the output of the code segment shown here?

```
String s = "abcdefgh";
for(int i = 0; i < s.length(); i+=2)
{
    if(i > 0)
        s = s.substring(0, i) + s.substring(i + 1, i + 2) + s.substring(i, i + 1) + s.substring(i + 2);
    else
        s = s.substring(i + 1, i + 2) + s.substring(i, i + 1) + s.substring(i + 2);
}
out.print(s);
```

- A) hgfedcba
- B) aacceegg
- C) efghabcd
- D) badcfegh
- E) abcdefgh

Question 37.

The `TreeNode` and `BST` classes shown on the right are an incomplete implementation of a binary search tree data structure. Which of the following best describes the function and purpose of method `(E e)`?

- A) Deletes `e` from the tree if present and returns `true` or returns `false` if the element is not present.
- B) Inserts `e` into the tree and returns `true` or returns `false` if the element is already present.
- C) Returns `true` if the tree is a complete binary tree and `false` if not.
- D) Returns `true` if a preorder traversal can be completed or `false` if not.
- E) Returns `true` if `e` is found in the tree and `false` if not.

```
public class TreeNode<E> {
    protected E element;
    protected TreeNode<E> left;
    protected TreeNode<E> right;

    public TreeNode(E e) {
        element = e;
    }
}

public class BST <E extends Comparable<E>>{
    protected TreeNode<E> root;
    protected int size = 0;
    //Constructors are present but not shown.

    public boolean method(E e) {
        TreeNode<E> current = root;
        while(current != null) {
            if(e.compareTo(current.element) < 0)
                current = current.left;
            else if(e.compareTo(current.element) > 0)
                current = current.right;
            else
                return true;
        }
        return false;
    }
}
```

Question 38.

Which of the following must replace the **/*missing code*/** in the main method shown on the right to ensure that the output will be 8 5 10?

- A) `for(Things t:Things.values())
 out.print(t.getNum() + " ");`
- B) `for(Things t:Things.getNum())
 out.print(t.values() + " ");`
- C) `for(Things t = THING1; t <= THING3; t++)
 out.print(t.getNum()+ " ");`
- D) `Things t = THING1;
while(!t.equals(THING3))
{
 out.print(t.getNum() + " ");
 t.next();
}`
- E) More than one of the above.

```
public enum Things {
    THING1(8),
    THING2(5),
    THING3(10);

    private int num;

    Things(int n){
        num = n;
    }

    public int getNum() {
        return num;
    }

    public static void main(String[] args) {
        /*missing code*/
    }
}
```

Question 39.

Write the output of the main method shown on the right in the blank provided on the answer document.

```
public static void main(String[] args)
{
    out.print(method("dornawxk"));
}

public static String method(String s)
{
    if(s.length()==1)
        return s;
    else
    {
        String u=s.substring(s.length()-1, s.length());
        return method(s.substring(0,s.length()-1)) + u;
    }
}
```

Question 40.

What is the worst case time complexity for access to an element within an array. Write your answer using Big-O notation in the blank provided on the answer document.

★ ANSWER KEY – CONFIDENTIAL ★

UIL COMPUTER SCIENCE – 2021 REGION

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

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

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

Note: Correct responses are based on **Java SE Development Kit 14 (JDK 14)** 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 14 Standard Packages have been imported. Ignore any typographical errors and assume any undefined variables are defined as used.

Explanations:

1.	C	$363_8 = 243_{10}$ $F4_{16} = 244_{10}$ $11110001_2 = 241_{10}$ $22220_3 = 240_{10}$																																				
2.	C	$15/6.0*5+3 = 2.5*5+3 = 12.5+3 = 15.5$																																				
3.	E	-10.2f is the format specifier to left justify a decimal value rounded to two decimal places in 10 spaces.																																				
4.	A	The length of the string is 20. $20 - 8 = 12$. The character at index value 12 is s.																																				
5.	B	$F \parallel T \wedge T \ \&\& \ F =$ $F \parallel F \ \&\& \ F =$ $F \parallel F =$ F																																				
6.	D	$3.14+2.72 = 5.86$ $\text{Math.ceil}(5.86)$ returns 6.0.																																				
7.	C	$100+95.5/100 = 100+0.955 = 100.955$ char data is evaluated as an integer.																																				
8.	B	$w\%x < w = 5\%7 < 5 = \text{false}$. So move to the else statement. $y > x - w = 0 > 7 - 5 = \text{false}$. So move to the next else statement. y gets -3 and w gets -4. x and z are unchanged.																																				
9.	A	The loop executes one time when x is 11.																																				
10.	E	The code segment moves the first element to a temporary variable, shifts all remaining elements to the left one place and then places the first element at the end of the array. There are three shifts.																																				
11.	C	java.lang.System.out is required to print without referencing System. java.util.Scanner is required to instantiate a Scanner object. java.io.File is required to instantiate a File object. java.io.IOException is required because the main method throws an IOException.																																				
12.	D	<table border="1" style="display: inline-table; vertical-align: top;"> <tr> <td>a</td> <td>b</td> <td>$b > 0 \ \&\& \ a < 50$</td> </tr> <tr> <td>0</td> <td>50</td> <td>T</td> </tr> <tr> <td>15</td> <td>45</td> <td>T</td> </tr> <tr> <td>28</td> <td>40</td> <td>T</td> </tr> <tr> <td>39</td> <td>35</td> <td>T</td> </tr> <tr> <td>49</td> <td>30</td> <td>T</td> </tr> <tr> <td>57</td> <td>25</td> <td>F</td> </tr> </table>	a	b	$b > 0 \ \&\& \ a < 50$	0	50	T	15	45	T	28	40	T	39	35	T	49	30	T	57	25	F															
a	b	$b > 0 \ \&\& \ a < 50$																																				
0	50	T																																				
15	45	T																																				
28	40	T																																				
39	35	T																																				
49	30	T																																				
57	25	F																																				
13.	A	$72 \gg 3 = 72/8 = 9 = 1001_2$ $11_{10} = 1011_2$ 1001 <u>OR 1011</u> $1011 = 11_{10}$																																				
14.	D	When a variable is incremented beyond the range of its data type the values will wrap around to the largest negative value in the range and continue until incrementation stops.																																				
15.	B	The call to list.addAll(1,more) inserts all of the elements in more into list starting at index value 1.																																				
16.	C	<table border="1" style="display: inline-table; vertical-align: top;"> <tr> <td></td> <td>0</td> <td>1</td> <td>2</td> <td>3</td> <td>4</td> <td>5</td> </tr> <tr> <td>0</td> <td>t</td> <td>u</td> <td>r</td> <td>k</td> <td>e</td> <td>y</td> </tr> <tr> <td>1</td> <td>d</td> <td>e</td> <td>e</td> <td>r</td> <td></td> <td></td> </tr> <tr> <td>2</td> <td>d</td> <td>o</td> <td>n</td> <td>k</td> <td>e</td> <td>y</td> </tr> <tr> <td>3</td> <td>s</td> <td>n</td> <td>a</td> <td>k</td> <td>e</td> <td></td> </tr> </table> mat.length returns the number of rows in the 2D array.		0	1	2	3	4	5	0	t	u	r	k	e	y	1	d	e	e	r			2	d	o	n	k	e	y	3	s	n	a	k	e		
	0	1	2	3	4	5																																
0	t	u	r	k	e	y																																
1	d	e	e	r																																		
2	d	o	n	k	e	y																																
3	s	n	a	k	e																																	
17.	E	<table border="1" style="display: inline-table; vertical-align: top;"> <tr> <td></td> <td>A</td> <td>B</td> <td>C</td> <td>D</td> <td>E</td> </tr> <tr> <td>A</td> <td>0</td> <td>1</td> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>B</td> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>C</td> <td>0</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>D</td> <td>1</td> <td>0</td> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>E</td> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>0</td> </tr> </table>		A	B	C	D	E	A	0	1	0	1	0	B	1	0	0	0	0	C	0	0	0	1	1	D	1	0	1	0	1	E	0	0	1	1	0
	A	B	C	D	E																																	
A	0	1	0	1	0																																	
B	1	0	0	0	0																																	
C	0	0	0	1	1																																	
D	1	0	1	0	1																																	
E	0	0	1	1	0																																	
18.	A	"xy." is a regular expression representing x and y followed by any character. The split method uses p as its delimiter. The delimiter is not included in the resulting array.																																				
19.	C	The Boolean expression for the if statement evaluates to false because the right hand operand is false. Both m and n are incremented and decremented since the right hand operand was in fact evaluated. The else statement subtracts 4 from n.																																				
20.	A	$\neg D$ is $\neg(A \ \&\& \ B)$ $\neg D$ is $A \parallel B$																																				

21.	E	getS1 has been inherited from June so no additional code is required. However, both this and super can be used as well without error.
22.	E	s2 is public and therefore has been inherited from June. It may be referenced directly.
23.	D	The call to the default constructor causes a call to each default constructor in the inheritance hierarchy (April->May->June). Printing comes down the constructor chain (June->May->April). April has inherited the toString method from May which prints s1 then s2.
24.	A	The call to concat("Sunday") calls the method in the April class which prints "Sunday" and then calls the no argument concat() method in the May class which then prints "Friday" and "Thursday". The second print statement calls the no args concat() method from the May class and prints "Friday" and "Thursday" again.
25.	A	x is an April object.
26.	C	add and offer both insert elements into the priority queue. Duplicates are allowed. peek returns but does not remove the next element in the queue which is "Cat" and then adds "Cat" back to the queue resulting in two "Cat" elements. One "Tom" element is removed. Elements are printed in alphabetical order.
27.	B	x is a Customer object. Customer is a functional interface with one method named accept. x must call the accept method to invoke the lambda expression.
28.	E	The Consumer interface has a single abstract method named accept. x is a Consumer object and can call the accept method which in turn applies the lambda expression to a Customer object. The lambda expression ensures that only Customer objects whose amounts are greater than 20 are printed in the order they were placed in the list.
29.	D	$178_{10} = 10110010_2$ Each iteration of the loop shifts the bits one place to the right leaving 00010110. toBinaryString does not print leading 0's.
30.	A	B. Sorts in descending order. C. Causes a stack overflow because left is not incremented and right is not decremented after the swap. D. Throws an ArrayIndexOutOfBoundsException because left should be incremented and right should be decremented within the two nested while loops.
31.	E	The pivot value can be any value within a partition.
32.	C	The outer loop executes n times. The inner loop only executes $\log_2 n$ times because it is incremented using $y*=2$.
33.	A	Prefix notation places the operators before the operands.
34.	B	A. Always returns Tails. C. Always returns Heads or an empty string. D. Always returns Tails. E. Always returns Tails or an empty string.
35.	D	The string "125" will parse as an integer so wombats is printed. The finally clause always executes so wobbegongs is also printed.
36.	D	The segment swaps each pair of characters in the string.
37.	E	The method does not insert or delete any elements. It simply returns true if the element is found.
38.	A	The loop iterates through each of the constants within the enumerated data type Things. There must be a call to getNum() to print the value associated with each constant.
39.	dornawxk	<pre>String u=s.substring(s.length()-1, s.length()); return method(s.substring(0,s.length()-1)) + u;</pre> just appends the last letter of each substring back onto the substring. Here is the call stack: dornawxk dornawx dornaw dorna dorn dor do d
40.	O(1)	Access by array index occurs in constant time in all cases.