UIL COMPUTER SCIENCE WRITTEN TEST

2022 STATE

MAY 2022

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 – 2022 STATE

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 is equal to $24_{16} * 2A_{16}$?

A) 101111010000_2 **B)** 864_{16} **C)** 10111101000_2 **D)** $5E8_{16}$ **E)** More than one of these.

Question 2.

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

A) 0

B) 20

C) 28

D) 42

E) There is no output due to a runtime error

```
out.print(11 + 17 % 7 * 31 / 10);
```

Question 3.

How many lines are printed by the code to the right?

A) 1

B) 2

C) 3

D) 4

E) 5

```
out.println("\\n");
out.println("\n");
out.printf("%.2f",2.718);
out.println("E");
```

Question 4.

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

A) 48

B) 50

C) 53

D) 55

E) 56

```
int it = 3;
String base = "banana";
for(int i = 0;i<it;i++)
{
  base = base.replace("a","ba");
  base = base.replace("b","bb");
}
out.print(base.length());
```

Question 5.

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

A) true

B) false

```
boolean t = true;
boolean f = false;
boolean x = t ^ f;
out.print(!((t ^ x ^ t)||(!f && t ^ !x)));
```

Question 6.

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

A) 9.0

B) 81.0

C) 108.0

D) 172.0

E) 729.0

```
out.print(Math.sqrt(81*81*81));
```

Question 7.

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

A) 3.00

B) 3.50

C) 3.78

D) 4.28

E) There is no output due to an error

```
int x = 7;
double y = 9.0;
out.printf("%.2f",x / 2 + x / y);
```

Question 8.

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

A) hello world 1

B) hello world 2

C) hello 1

D) hello 2

E) There is no output due to a compile error

```
int what = switch("Java13"){
  case "Java13" -> {
    out.print("hello ");
    yield 1;
  }
  default -> {
    out.print("world");
    yield 2;
  }
};
out.print(what);
```

Question 9.

How many asterisks are printed by the code shown to the right?

A) 42

B) 790

C) 882

D) 903

E) 912

```
for(int x = 1; x <= 42;x++)
{
  for(int y = 0;y<x;y++){
    out.print("*");
  }
}
```

Question 10.

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

A) [8, 6, 1, 8]

B) [9, 6, 1, 8]

C) [8, 7, 1, 8]

D) [2, 7, 1, 8]

E) There is no output due to an error.

```
int[] ints = new int[]{2,7,1,8};
ints[0] += (ints[1] -= ints[2]);
out.print(Arrays.toString(ints));
```

```
public static void main(String[] args) throws IOException
{
      Scanner f = new Scanner(new File("data.dat"));
      int x = 0;
      for(int i = 3;i<=5;i++)
      {
        x += f.nextInt(i);
      }
      out.println(x);
}
```

Question 11.

Consider the main method shown above. Suppose `data.dat` contains: "**`121 33 404`**". What is the output of the code?

A) 0

B) 135

C) 148

D) 558

E) There is no output due to an error

Question 12.

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

A) 4

B) 5

C) 6

D) 7

E) 8

```
int x = 1;
int y = 3;
do{
  x *= y;
  x += --y;
}while(7 % y != 1);
out.print(x+y);
```

Question 13.

What is the correct order of operations for the operators listed on the right?

A) I II III

B) II III I

C) III II I

D) II I III

E) III I II

I. +=

II. <<

III. <=

Question 14.

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

A) -113

B) -112

C) -111

D) 144

E) There is no output due to an error

```
byte x = (byte)(12);
out.print(x *= x);
```

Question 15.

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

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

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

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

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

E) There is no output due to an error.

```
ArrayList<Integer> list;
list = new ArrayList<Integer>();
ListIterator<Integer> it;
it = list.listIterator();
it.add(1);
it.add(2);
it.previous();
it.add(3);
it.next();
it.add(4);
out.println(list);
```

Question 16.

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

A) true **B)** false

C) There is no output due to an error

```
HashSet<Object> a = new HashSet<>();
HashSet<Object> b = new HashSet<>();
TreeSet<Object> c = new TreeSet<>();
HashSet<Object> d = new HashSet<>();
a.add(c);
b.add(d);
out.print(a.equals(b));
```

Question 17.

What is the worst case Big O Complexity of `java.util.PriorityQueue remove()` method? Assume comparison is an O(1) operation.

A) `O(1)`

B) `O(log(N))`

C) `O(N)`

D) `O(Nlog(N))`

E) $O(N^2)$

Question 18.

Which of the following is the *best estimate* of the output of the code segment shown on the right?

A) `0.785`

B) `1.00`

C) `1.57`

D) `2.00`

E) `3.14`

```
Random r = new Random();
double ct = 0;
double it = 10000;
for(int i=1;i<=it;i++)
{
   double x = r.nextDouble();
   double y = r.nextDouble();
   if(x*x + y*y <= 1)
      ct++;
}
out.print((ct / it) * 4);
```

Question 19.

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

A) `0`

B) `1`

C) `17`

D) `32`

E) `33`

```
int it = 33;
int x = 0;
for(int i = 0;i<=it;i++)
{
  x ^= i;
}
out.print(x);
```

Question 20.

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

A) `0`

B) `38`

C) `152`

D) `608`

E) There is no output due to an error

```
int x = 19;
out.print(x << -61);
```

Question 21.

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

A) `0.707 (0.707)`

B) `0.707 (-0.707)`

C) `-0.707 -0.707`

D) `0.707 -0.707`

E) There is no output due to an error

```
double deg = 135.0;
double s = Math.sin(Math.toRadians(deg));
double c = Math.cos(Math.toRadians(deg));
out.printf("%.3f %(.3f", s, c);
```

Question 22.

The method `run()` takes 54 milliseconds to complete on an array of 180 elements. Which of the following is the best estimate of how long it would take to complete on an array of 120 elements?

A) `8 milliseconds`

B) `16 milliseconds`

C) `24 milliseconds`

D) `36 milliseconds`

E) `81 milliseconds`

```
static int run(int[] ints){
  int ret = 0;
  int L = ints.length;
  for(int i = L - 5;i<ints.length;i++)
     for(int j: ints)
        for(int k: ints)
           ret += i+j+k;
  return ret;
}
```

Question 23.

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

A) `banana bano`

B) `bano bo`

C) `bo bo`

D) `banano bo`

E) no characters are printed

```
String str = "banano";
out.print(str.replaceFirst("(an)+?", ""));
out.print(" "+str.replaceFirst("(an)++",""));
```

Question 24.

What is the output of client code shown below?

```
int[] ints = new int[]{1,4,5,7,9,14};
out.print(sh(ints, 8));
```

A) −1

B) 3

C) 4

D) 6

E) 9

Question 25.

What is the output of client code shown below?

```
int[] ints = new int[]{1,4,5,7,9,14};
out.print(sh(ints, 14));
```

A) −1

B) 3

C) 4

D) 6

E) 9

Question 26.

Assume method `sh()` is given a sorted array of 15 integers for `ints` and some integer value for `s`. What is the maximum number of times that the line labeled **`/*LINE:*/`** could be executed?

A) 2

B) 3

C) 4

D) 5

E) None of the above

```
static int sh(int[] ints, int s)
{
   int L = -1;
   int R = ints.length;
   int M = (L + R) / 2;
   while(R - L > 1)
   {
     /*LINE:*/
     M = (L + R) / 2;
     int c = ints[M];
     if(c > s)
       R = M;
     else
       L = M;
   }
   return R;
}
```

Question 27.

How many instance variables does a member of class `RedPanda` have?

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

Question 28.

What is the output of client code shown below?

```
Panda p1 = new RedPanda("Tom","red");
out.println(p1.name);
```

A) Tom

B) Jerry

C) red

D) Tom red

E) There is no output due to an error.

Question 29.

What is the output of client code shown below?

```
Panda p1 = new RedPanda("Tom","red");
out.println(p1);
```

A) Tom

B) Tom red

C) Jerry

D) Jerry red

E) There is no output due to an error.

```
class Panda{
   public String name;

   public Panda()
   {
     this.name = "Jerry";
   }

   public String toString()
   {
     return name;
   }
 }

class RedPanda extends Panda{
   public String name;
   public String color;
   public RedPanda(String name, String color)
   {
     this.name = name;
     this.color = color;
   }

   public String toString(){
      return name+" "+color;
   }
 }
```

Question 30.

Which of the following is logically equivalent to the digital electronics diagram to the right?

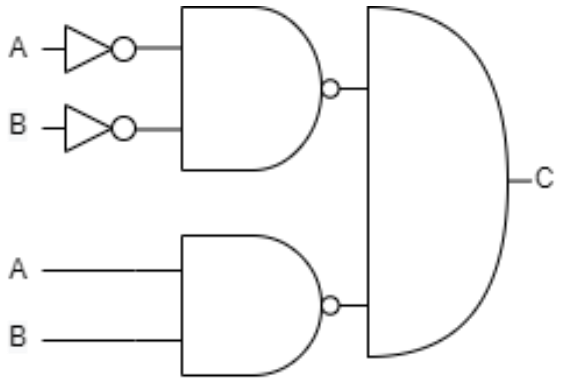
A) A ⊕ B

B) `(A * B) + (!A * !B)`

C) `(A * B)`

D) `(A + B)`

E) None of the above



Question 31.

What is the output of the client code below?

```
FI lambda = (i, j) -> i.split(" ",j);
String[] res = lambda.go("a b c d", 3);
out.print(res.length);
```

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

E) There is no output due to an error

```
public class Lambdas{
  private interface FI
  {
    public String[] go(String i, int j);
  }
}
```

This page intentionally left blank

For Questions 32-38, refer to the following `Structure` class:

```
class Structure {

  ArrayList<Integer> items;

  public Structure(){ items = new ArrayList<>();}

  public void add(int x){items.add(x);}

  public int sz(){return items.size();}

  public int L(int i){return i*2 + 1;}

  public int R(int i){return i * 2 + 2;}

  public void t1(int i)
  {
    if(i >= items.size())
      return;
    t1(L(i));
    t1(R(i));
    out.print(items.get(i));
  }

  public void t2()
  {
    Queue<Integer> que = new LinkedList<Integer>();
    que.add(0);
    while(!que.isEmpty())
    {
      int x = que.poll();
      if(x >= items.size())
        continue;
      out.print(items.get(x));
      que.add(L(x));
      que.add(R(x));
    }
  }

  public int subSum1(int i)
  {
    if(i >= items.size())
      return 0;
    return subSum1(L(i)) + subSum1(R(i)) + items.get(i);
  }

  public int subSum2(int i, Integer[] memo)
  {
    if(i >= items.size())
      return 0;
    if(memo[i] != null)
      return memo[i];
    memo[i] = subSum2(L(i), memo) + subSum2(R(i), memo) + items.get(i);
    return memo[i];
  }
}
```

Question 32.

What is the output of client code marked ***LINE 1*** shown to the right?

A) 12345678

B) 12485367

C) 84526731

D) 84251637

E) 76358421

Question 33.

What is the output of client code marked ***LINE 2*** shown to the right?

A) 12345678

B) 12485367

C) 84526731

D) 84251637

E) 76358421

Question 34.

What is the output of client code marked ***LINE 3*** shown to the right?

A) 0

B) 15

C) 19

D) 28

E) 36

Question 35.

What is the output of client code marked ***LINE 4*** shown to the right?

A) 0

B) 15

C) 19

D) 28

E) 36

```
Structure st = new Structure();
for(int i = 0;i<8;i++)
{
  st.add(i+1);
}

/*LINE 1*/
st.t1(0);

out.println();

/*LINE 2*/
st.t2();

out.println();

/*LINE 3*/
out.println(st.subSum1(0));

Integer[] memo = new Integer[st.sz()];

/*LINE 4*/
out.println(st.subSum2(1, memo));
```

Question 36.

The `Structure` class is an example of a _____.

A) Tree

B) Binary Search Tree

C) PriorityQueue

D) Queue

E) more than one of these

Question 37.

Assume `Structure st` contains `N` elements. Select the most restrictive upper bound on the asymptotic complexity of `allSum1(st)`?

A) $O(1)$

B) $O(N)$

C) $O(N\log(N))$

D) $O(N^2)$

E) $O(2^N)$

```
ArrayList<Integer> allSum1(Structure st)
{
  ArrayList<Integer> sums;
  sums = new ArrayList<Integer>();
  for(int i = 0;i<st.sz();i++)
  {
    sums.add(st.subSum1(i));
  }
  return sums;
}
```

Question 38.

Assume `Structure st` contains `N` elements. Select the most restrictive upper bound on the asymptotic complexity of `allSum2(st)`?

A) $O(1)$

B) $O(N)$

C) $O(N\log(N))$

D) $O(N^2)$

E) $O(2^N)$

```
ArrayList<Integer> allSum2(Structure st)
{
  ArrayList<Integer> sums;
  sums = new ArrayList<Integer>();
  Integer[] memo = new Integer[st.sz()];
  for(int i = 0;i<st.sz();i++)
  {
    sums.add(st.subSum2(i, memo));
  }
  return sums;
}
```

Question 39.

G is an undirected graph consisting of 46 vertices. For every pair of vertices in G, there exists exactly one path between the two. How many edges are in G? Write your answer in the blank provided on the answer document. If it is not possible to deduce the number of edges in G from the given information, write -1 instead.

Question 40.

What is a String that could replace ***CODE*** for the code below to output `true`? Write your answer in the blank provided on the answer document.

```
String A = "A{1,5}B{1,5}C{1,5}";
String B = ".*ABC*";
String C = "....B...";
String S = /*CODE*/;
boolean x = S.matches(A);
boolean y = S.matches(B);
boolean z = S.matches(C);
out.println(x && y && z);
```

★ ANSWER KEY – CONFIDENTIAL ★

UIL COMPUTER SCIENCE – 2022 STATE

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

1) E	11) B	21) A	31) C
2) B	12) D	22) C	32) C
3) D	13) B	23) B	33) A
4) D	14) B	24) C	34) E
5) B	15) A	25) D	35) C
6) E	16) A	26) C	36) A
7) C	17) B	27) D	37) D
8) C	18) E	28) B	38) B
9) D	19) B	29) B	*39) 45
10) A	20) C	30) A	*40) AAAABCCC

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

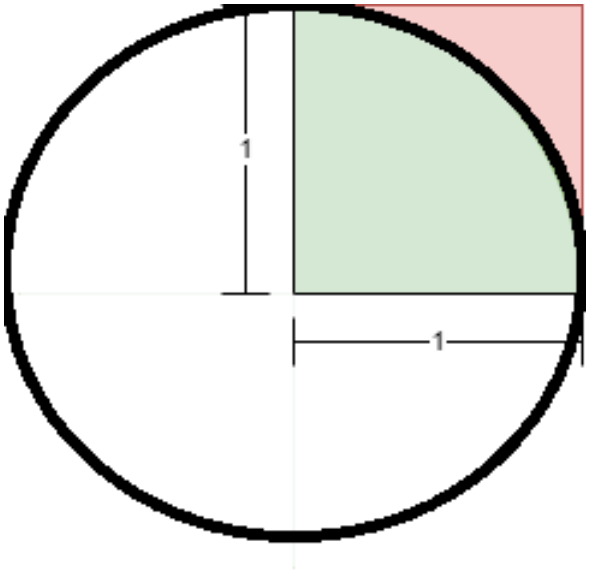
Note: Correct responses are based on **Java SE Development Kit 17 (JDK 17)** from Oracle, Inc. All provided code segments are intended to be syntactically correct, unless otherwise stated (e.g., "error" is an answer choice) and any necessary Java SE 17 Standard Packages have been imported. Ignore any typographical errors and assume any undefined variables are defined as used.

Explanations:

1.	E	24_{16} = 36, $2A_{16}$ = 42, 36 * 42 = 1512, equal to 10111101000_2 and $5E8_{16}$
2.	B	11 + (17 % 7 * 31 / 10) = 11 + (3 * 31 / 10) = 11 + 9 = 20
3.	D	`out.println("\\n");` -> prints one line `out.println("\n");` -> prints 2 lines `out.printf("%.2f",2.718);` -> no carriage return, shares line with E `out.println("E");` -> prints one line
4.	D	(iteration, #a's, #b's) (0, 3, 1) (1, 3, 8) (2, 3, 22) (3, 3, 50) 3 a's, 50 b's, and 2 n's equals 55 characters
5.	B	x = t ^ f = true t ^ x ^ t = t ^ t ^ t = true !(true || (anything)) = false
6.	E	81*81*81 = (9*9)*(9*9)*(9*9) = (9*9*9)(9*9*9) = 729^2
7.	C	x / 2 = 3, x / y = .777777, x/2 + x/y = 3.777, formats to 3.78
8.	C	Yield is a java 13 feature that evaluates a switch case to a given value. This immediately breaks from the switch case.
9.	D	The number of asterisks printed is 1 + 2 + … + 41 + 42. By sum of an arithmetic sequence, this is equal to 21 * 43 = 903
10.	A	The -= evaluates first, and then the result replaces ints[0]
11.	B	Scanner's nextInt(x) parses the next integer in base x. $121_3 + 33_4 + 404_5 = 135_{10}$
12.	D	(iteration, x, y) (0, 1, 3) (1, 2, 5) [at this point, the while condition fails]
13.	B	<< comes before +=, and += comes before <=
14.	B	12 * 12 = 144, which overflows the byte range (max value 127, min value -128) and wraps back around to -112

15.	A	`                    ^`
		`it.add(1);  1^`
		`it.add(2);  12^`
		`it.previous();  1^2`
		`it.add(3);  13^2`
		`it.next();  132^`
		`it.add(4);  1324^`
		`out.println(list);  [1, 3, 2, 4]`

16.	A	Both a and b are sets containing empty sets. Regardless of type, two sets with equal content are considered equal. Hence, c = d and therefore a = b
17.	B	PriorityQueue remove must down heapify after removal. This is a logarithmic time operation.

18.	E	A geometric intuition: x and y are selected uniformly from the range [0, 1]. The probability of a random point from this distribution landing at a location that will increment ct is equal to (acceptance region area)/(total area). The increment criteria is the circle equation – a visualization:  Acceptance region area = ¼ of a unit circle area Total area = 1 Hence the probability of a random point incrementing ct is pi/4. In the long run, ct/it will tend to pi/4, so 4*ct/it tends to pi.
19.	B	Xor is a bitwise operation. Hence we can solve this chain of xor's independently for each bit. 1 bit: half of the 1 bits from 0..31 are ones. Because there are an even number, these cancel each other out, leaving only the 1 bit from 33. Hence the end result for the 1's place is a 1 bit 2, 4, 8, 16 bit: exactly half of all of these from 0..31 are ones. Because there are an even number, these cancel each other out. 32 and 33 have zeros in their 2,4,8, and 16 bit positions, so the end result for all of these bits are zeros. 32 bit: 0..31 are all zeros, and then 32 and 33 are ones, which cancel each other out. Hence the end result is a 0 bit. Overall, binary representation of the result is 000001, or 1.
20.	C	Before shifting, java applies a %32, a +32, and another %32 to the shift amount to ensure it is both positive and in the range [0,31]. -61 -> -29 -> 3. 19 << 3 = 19*8 = 152
21.	A	135 degrees is in the top left quadrant of the unit circle. Hence sin will be positive and cosine will be negative. The "(" format encapsulates a number in parenthesis if it is negative.
22.	C	The complexity of run() is O(n^2) as the i loop does not actually depend on the length of ints. Hence a reduction to 2/3 of the array size will lead to a $(2/3)^2$ expected reduction to runtime. 54 * 4 / 9 = 24 ms.

23.	B	? is a reluctant quantifier – regex will try to match as little as possible with this + is a possessive quantifier – regex will try to match as much as possible with this
24.	C	The binary search code in 24-26 finds the index of the strong ceiling of the search value. (The location of the smallest value strictly larger than s). Hence this call will return index 4.
25.	D	Because no element is strictly greater than 14, the search code returns ints.length, 6.
26.	C	The number of candidates will be cut in half at each iteration of the while loop. Hence there will be a candidate check for a list of 15 elements, 7 elements, 3 elements, and 1 element.
27.	D	RedPanda has its name variable, color, and the name variable inherited from the Panda superclass.
28.	B	The compiler only looks at the left side of variable types, and hence p1.name refers to the superclass name variable, since the left side of p1's type is Panda. Hence p1.name refers to "Jerry" from the implicit constructor.
29.	B	RedPanda overrides toString, referencing it's own name and color variables.
30.	A	This translates to (not neither) and (not both), otherwise known as xor
31.	C	str.split(delimiter, x) stops splitting after the result is of length x
32.	C	T1() explores left child, then right child, then self. (post order traversal)
33.	A	T2() performs a depth first traversal of the tree by utilizing a queue (also known as level-order traversal)
34.	E	subSum1(i) recursively calculates node value sum of the subtree rooted at index i
35.	C	subSum2(i) performs the same as subSum1(i), but also uses Dynamic Programming/Memoization to avoid solving duplicate subproblems. The first time a problem is solved, it will be recorded in memo[]
36.	A	Structure is a tree, but not a binary search tree because no comparison is performed.
37.	D	subSum1(i) is an O(n) operation since it explores an entire subtree. Because allSum1 calls this on every one of the n subtrees, this is $O(N^2)$ overall.
38.	B	allSum2() uses subSum2(i) instead, which memoizes results and only solves each subproblem once. There are O(N) subtrees, and each one references a constant (2) number of subtree queries. Hence allSum2() is O(N) overall.
39.	45	Because there is exactly one path between every pair of nodes and the graph is undirected, we can conclude that no cycles exist. A fully connected graph without cycles can only be a tree, and we know that a tree of N nodes has N-1 edges.
40.	AAAABCCC	Pattern A: only matches strings of the form (some # of A)(some # of B)(some # of C) Pattern B: only matches strings containing "ABC" Pattern C: only matches string of length 8 where the fifth character is a B By combining requirements from patterns A and B, we can deduce that there can only be a single B. Furthermore, everything before the B must be an A, and everything after must be a C. From pattern C we can deduce how many A's and C's are required before and after the B.