Exam 1

Due Feb 16, 2017 at 9:15pm **Points** 30 **Questions** 30
Available after Feb 16, 2017 at 8pm **Time Limit** 75 Minutes

This quiz is no longer available as the course has been concluded.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	34 minutes	28 out of 30

Score for this quiz: **28** out of 30
Submitted Feb 16, 2017 at 8:33pm
This attempt took 34 minutes.

Question 1

1 / 1 pts

Suppose you have an array full of instances of a class named `Person`. Each person has a name and a date of birth. The following is an example of such an array.

```
[Joe 02-21-97, Amy 06-10-95, Jan 04-07-96, Bob 11-30-95, Pat
02-21-97, Lee 04-07-96]
```

To arrange the elements in this array in ascending date of birth order, and where multiple persons have the same date of birth they are listed in ascending order of name, which is the correct sequence of sorts to perform? (An example of the desired order is shown below.)

```
[Joe 02-21-97, Pat 02-21-97, Jan 04-07-96, Lee 04-07-96, Amy
06-10-95, Bob 11-30-95]
```

A. Use merge sort to sort the array in ascending order of name, then use merge sort to sort the array in ascending order of date of birth.

B. Use merge sort to sort the array in ascending order of date of birth, then use merge sort to sort the array in ascending order of name.

C. Use quicksort to sort the array in ascending order of date of birth, then use quicksort to sort the array in ascending order of name.

D. Use merge sort to sort the array in ascending order of name, then use quicksort to sort the array in ascending order of date of birth.

Correct!

(•) A

B

C

D

Question 2

1 / 1 pts

Which big-oh expression best characterizes the worst case time complexity of the following code?

```
int count = 0;
for (int i = N; i > 1; i = i / 2) {
   count = count + 1;
}
```

A. $O(1)$

B. $O(\log N)$

C. $O(N)$

D. $O(N^2)$

A

Correct!

B

C

D

Question 3

1 / 1 pts

Which big-oh expression best characterizes the worst case time complexity of the following implementation of `kmin`?

```
/**
 * Returns the kth smallest element in an
 *  array of unique values.
 */
public Comparable kmin(Comparable[] a, int k) {
   java.util.Arrays.sort(a);
   return a[k - 1];
}
```

A. $O(1)$

B. $O(N)$

C. $O(N \log N)$

D. $O(N^2)$

- A
- B
- C

Correct!

- D

Question 4

1 / 1 pts

Which method preserves type safety and ensures that `coll` can only contain instances of some class that implements the `Comparable` interface for other objects of the *same* class?

A.
```
public <T extends Comparable<T>> boolean search(Collection<T> coll, T target) {
  for (T element : coll) {
    if (element.compareTo(target) == 0) {
      return true;
    }
  }
  return false;
}
```

B.
```
public <T extends Comparable> boolean search(Collection<T> coll, T target) {
  for (T element : coll) {
    if (element.compareTo(target) == 0) {
      return true;
    }
  }
  return false;
}
```

C.
```
public boolean search(Collection<Comparable> coll, Comparable target) {
  for (Comparable element : coll) {
    if (element.compareTo(target) == 0) {
      return true;
    }
  }
  return false;
}
```

D.
```
public <T extends Comparable> boolean search(Collection<Comparable> coll, T target) {
  for (Comparable element : coll) {
    if (element.compareTo(target) == 0) {
      return true;
    }
  }
  return false;
}
```

Correct!

- (•) A
- ( ) B
- ( ) C
- ( ) D

Question 5

1 / 1 pts

Given the array a = [66, 67, 20, 86, 55, 74, 11, 91, 43, 47] which sorting algorithm would perform the following sequence of array modifications?

```
[66, 67, 20, 86, 55, 74, 11, 91, 43, 47]
[20, 66, 67, 86, 55, 74, 11, 91, 43, 47]
[20, 66, 67, 55, 86, 74, 11, 91, 43, 47]
[20, 55, 66, 67, 86, 74, 11, 91, 43, 47]
[20, 55, 66, 67, 86, 11, 74, 91, 43, 47]
[20, 55, 66, 67, 86, 11, 74, 91, 43, 47]
[20, 55, 66, 67, 86, 11, 74, 91, 43, 47]
[20, 55, 66, 67, 86, 11, 43, 47, 74, 91]
[11, 20, 43, 47, 55, 66, 67, 74, 86, 91]
```

A. selection sort

B. insertion sort

C. merge sort

D. quicksort

- ○ A
- ○ B
- ◉ C
- ○ D

Correct!

Question 6

0 / 1 pts

Which big-oh expression best characterizes the worst case time complexity of the following implementation of `kmin`?

```
/**
 * Returns the kth smallest element in an array.
 */
public static int kmin(int[] a, int k) {
   for (int i = 0; i < k; i++) {
      int min = i;
      for (int j = i + 1; j < a.length; j++) {
         if (a[j] < a[min]) {
            min = j;
         }
      }
      swap(a, i, min);
   }
   return a[k];
}
```

A. $O(1)$

B. $O(N)$

C. $O(N \log N)$

D. $O(N^2)$

'ou Answered

- (•) A

- ( ) B

- ( ) C

orrect Answer

- ( ) D

Question 7

1 / 1 pts

Which big-oh expression best characterizes the worst case time complexity of the following code?

```
int count = 0;
for (int j = 0; j < N; j++) {
   int i = N;
   while (i > 1) {
      count = count + 1;
      i = i / 2;
   }
}
```

A. $O(N^3)$

B. $O(N^2)$

C. $O(N^2 \log N)$

D. $O(N \log N)$

- A
- B
- C

Correct!

- D

Question 8

1 / 1 pts

What is the *maximum* number of elements that would have to be examined on *any* binary search of the array a = [15, 20, 20, 26, 41, 48, 53, 58, 67, 77, 81, 95]?

A. 2

B. 4

C. 6

D. 12

- A

Correct!

- B
- C

D

Question 9

1 / 1 pts

The `min` method below is intended to return the smallest of its three `int` parameters. Which call below would expose the logic error in this method?

```
public static int min(int a, int b, int c) {
   if ((a < b) && (a < c)) {
      return a;
   }
   if ((b < a) && (b < c)) {
      return b;
   }
   return c;
}
```

A. `min(0, 0, 0)`

B. `min(1, 0, -1)`

C. `min(2, 2, 4)`

D. `min(4, 2, 2)`

A

B

Correct!

C

D

Question 10

1 / 1 pts

Suppose you have collected data from several timing experiments and you have characterized a sorting method's time complexity profile as follows.

- Data in ascending order: $O(N)$
- Data in descending order: $O(N^2)$
- Data in random order: $O(N^2)$

Which sorting algorithm is implemented in this method?

A. selection sort

B. insertion sort

C. merge sort

D. quicksort (without randomization)

A

Correct!

B

C

D

Question 11

1 / 1 pts

Which big-oh expression best characterizes the worst case time complexity of the following code?

```
int count = 0;
int i = 1;
while (i < N) {
   i = i + 1;
   for (int j = 0; j < N; j = j + 5) {
      int k = N;
      while (k > 1) {
         count = count + 1;
         k = k - 1;
      }
   }
}
```

A. $O(\log N)$

B. $O(N^2 \log N)$

C. $O(N^2)$

D. $O(N^3)$

A

B

C

Correct!

D

Question 12

1 / 1 pts

Which big-oh expression best characterizes the worst case time complexity of the following code?

```
int count = 0;
int i = 1;
while (i < 10) {
   i = i + 1;
   for (int j = 0; j < 100; j = j + 5) {
      int k = 25;
      while (k > 1) {
         count = count + 1;
         k = k - 1;
      }
   }
}
```

A. $O(1)$

B. $O(N)$

C. $O(N^2)$

D. $O(N^3)$

Correct!

A

B

C

D

Question 13

1 / 1 pts

The `max` method below is intended to return the largest value in the parameter `coll`. Which test case parameter would expose the logic error in this method?

```
public static <T extends Comparable<T>> T max(Collection<T> coll)
   Iterator<T> itr = coll.iterator();
   T max = itr.next();
   while (itr.hasNext()) {
      if (max.compareTo(itr.next()) < 0) {
         max = itr.next();
      }
   }
   return max;
}
```

A. `coll = [1, 3, 5, 7, 9]`

B. `coll = [9, 7, 5, 3, 1]`

C. `coll = [3, 7, 9, 1, 5]`

D. `coll = [1, 9, 3, 7, 5]`

- A
- B
- C
- D

Correct!

Question 14

1 / 1 pts

Which big-oh expression best characterizes the worst case time complexity of the following code?

```
int count = 0;
int j = 0;
while (j < 100) {
   j = j + 1;
   for (int k = 0; k < N; k++) {
      count = count + 1;
   }
}
```

A. $O(N \log N)$

B. $O(N)$

C. $O(\log N)$

D. $O(N^2)$

- A

Correct!

- (•) B
- ( ) C
- ( ) D

Question 15

1 / 1 pts

Suppose you have collected data from several timing experiments and you have characterized a sorting method's time complexity profile as follows.

- Data in ascending order: $O(N^2)$
- Data in descending order: $O(N^2)$
- Data in random order: $O(N^2)$

Which sorting algorithm is implemented in this method?

A. selection sort

B. insertion sort

C. merge sort

D. quicksort (without randomization)

Correct!

- (•) A
- ( ) B
- ( ) C
- ( ) D

Question 16

1 / 1 pts

Suppose you have collected data from several timing experiments and you have characterized a sorting method's time complexity profile as follows.

- Data in ascending order: $O(N \log N)$
- Data in descending order: $O(N \log N)$
- Data in random order: $O(N \log N)$

Which sorting algorithm is implemented in this method?

A. selection sort

B. insertion sort

C. merge sort

D. quicksort (without randomization)

- A
- B
- C

Correct!

- D

Question 17

1 / 1 pts

Suppose `x`, `y`, and `z` are mutually comparable objects such that `x` < `y` < `z`. That is, `x` precedes `y` in total order, and `y` precedes `z` in total order. Assume that `comp` is a `Comparator` that defines this total order such that `comp.compare(x, y)` evaluates to -6 and `comp.compare(y, z)` evaluates to -3. According to the specification of `compare` in the `Comparator` interface, which one of the following must be true?

A. `x.compareTo(z)` < 0

B. `comp.compare(x, z)` = -9

C. `comp.compare(z, x)` < 0

D. `comp.compare(x, z)` < 0

- A
- B
- C

Correct!

- (•) D

Question 18

1 / 1 pts

If the size (N) of the search space doubles (say, for example, from five million to 10 million elements), by how much does the amount of work required by binary search **increase** in the *worst case*?

A. 1

B. $\log_2 N$

C. N

D. by a factor of two

Correct!

- (•) A
- ( ) B
- ( ) C
- ( ) D

Question 19

1 / 1 pts

Given the array a = [66, 67, 20, 86, 55, 74, 11, 91, 43, 47] which sorting algorithm would perform the following sequence of array modifications?

```
[11, 67, 20, 86, 55, 74, 66, 91, 43, 47]
[11, 20, 67, 86, 55, 74, 66, 91, 43, 47]
[11, 20, 43, 86, 55, 74, 66, 91, 67, 47]
[11, 20, 43, 47, 55, 74, 66, 91, 67, 86]
[11, 20, 43, 47, 55, 74, 66, 91, 67, 86]
[11, 20, 43, 47, 55, 66, 74, 91, 67, 86]
[11, 20, 43, 47, 55, 66, 67, 91, 74, 86]
[11, 20, 43, 47, 55, 66, 67, 74, 91, 86]
[11, 20, 43, 47, 55, 66, 67, 74, 86, 91]
[11, 20, 43, 47, 55, 66, 67, 74, 86, 91]
```

A. selection sort

B. insertion sort

C. merge sort

D. quicksort

Correct!

- (•) A
- ( ) B
- ( ) C
- ( ) D

Question 20

1 / 1 pts

Which of the arrays below would be the final result of *partitioning* the following portion of an array using 59 as the pivot in the quicksort partition implementation presented in lecture? Only the partitioning operation is happening.

```
[97, 20, 84, 24, 25, 59, 93, 13, 94]
```

A. `[20, 93, 13, 97, 59, 24, 25, 94, 84]`

B. `[94, 93, 97, 84, 59, 20, 24, 25, 13]`

C. `[20, 24, 25, 13, 59, 94, 93, 97, 84]`

D. `[20, 24, 84, 97, 59, 13, 25, 93, 94]`

- ( ) A
- ( ) B

Correct!

- C
- D

Question 21

1 / 1 pts

Suppose you have collected data from several timing experiments and you have characterized a sorting method's time complexity profile as follows.

- Data in ascending order: $O(N^2)$
- Data in descending order: $O(N^2)$
- Data in random order: $O(N \log N)$

Which sorting algorithm is implemented in this method?

A. selection sort

B. insertion sort

C. merge sort

D. quicksort (without randomization)

- A
- B
- C

Correct!

- D

Question 22

1 / 1 pts

Suppose `x` and `y` are mutually comparable objects and `x.compareTo(y)` evaluates to 1. According to the specification of `compareTo` in the `Comparable` interface, which one of the following must be true?

A. `y.compareTo(x) == 1`

B. `y.compareTo(x) == -1`

C. `y.compareTo(x) != 0`

D. `x.equals(y) == false`

- A
- B

Correct!

- C
- D

Question 23

1 / 1 pts

Given the array a = [66, 67, 20, 86, 55, 74, 11, 91, 43, 47] which sorting algorithm would perform the following sequence of array modifications?

```
[66, 67, 20, 86, 55, 74, 11, 91, 43, 47]
[20, 66, 67, 86, 55, 74, 11, 91, 43, 47]
[20, 66, 67, 86, 55, 74, 11, 91, 43, 47]
[20, 55, 66, 67, 86, 74, 11, 91, 43, 47]
[20, 55, 66, 67, 74, 86, 11, 91, 43, 47]
[11, 20, 55, 66, 67, 74, 86, 91, 43, 47]
[11, 20, 55, 66, 67, 74, 86, 91, 43, 47]
[11, 20, 43, 55, 66, 67, 74, 86, 91, 47]
[11, 20, 43, 47, 55, 66, 67, 74, 86, 91]
```

A. selection sort

B. insertion sort

C. merge sort

D. quicksort

- A

Correct!

- B
- C
- D

Question 24

1 / 1 pts

The `kmin` method below is intended to return the k^{th} smallest value in the parameter `list`. Which test case parameters would expose the logic error in this method?

```
public static Integer kmin(List<Integer> list, int k) {
   Collections.<Integer>sort(list);
   return list.get(k - 1);
}
```

A. `list = [7, 3, 5, 1, 7], k = 3`

B. `list = [3, 1, 3, 5, 7], k = 3`

C. `list = [1, 3, 5, 7, 9], k = 3`

D. `list = [9, 7, 5, 3, 1], k = 3`

- ○ A
- Correct! ◉ B
- ○ C
- ○ D

Question 25

1 / 1 pts

Consider a binary search for 33 in the following array:

a = [9, 18, 28, 32, 34, 39, 39, 45, 62, 72, 76, 98]

After two elements have been examined (i.e., compared to 33), what slice of the array remains to be searched?

A. [32, 34]

B. [9, 18, 28, 32, 34]

C. [32, 34, 39, 39, 45, 62]

D. [28, 32, 34, 39, 39, 45, 62, 72, 76, 98]

- Correct! ◉ A
- ○ B

C

D

Question 26

0 / 1 pts

Which big-oh expression best characterizes the worst case time complexity of the following code?

```
int i = 1;
while (i < N) {
   i = i * 2;
}
int count = 0;
for (int j = 0; j < N; j = j + 2) {
   for (int k = 0; k < N; k = k + 1) {
      count = count + 1;
   }
}
```

A. $O(\log N)$

B. $O(N^2 \log N)$

C. $O(N^2)$

D. $O(N^3)$

A

You Answered

B

Correct Answer

C

D

Question 27

1 / 1 pts

Suppose you have implemented an algorithm in a method named `foo`, which takes an array of N floating point numbers as data. Suppose also that this algorithm has $O(N^3)$ best, average, and worst case time complexity and that a timing analysis of `foo` showed that approximately 2 seconds were required to process an array of size $N = 256$.
What is the largest array (N) that `foo` could process in less than one hour?

A. 512

B. 1024

C. 2048

D. 4096

- A
- B

Correct!

- C
- D

Question 28

1 / 1 pts

Given the following method signature, which is a true statement regarding the parameter `c`?

```
public static <T> int search(java.util.Collection<T> c, T target)
```

A. `c` is an instance of the `Collection` class.

B. `c` is an instance of some class that implements the `Collection` interface.

C. `c` is an instance of some unknown generic type `T`.

D. `c` is an instance of the `ArrayList` class that has been cast as `Collection` for the method call.

- A

Correct!

- B
- C
- D

Question 29

1 / 1 pts

Suppose you attempted to empirically discover the big-oh running time of a program, and you were able to generate the following timing data.

N	Time	Ratio
128	0.622	—
256	8.886	14.286
512	141.048	15.873
1024	2249.704	15.950
2048	35958.996	15.984

In the table above, the N column records the size of the input for each run, the Time column records the elapsed time in seconds for each run, and Ratio is the elapsed time for the current run divided by the elapsed time for the previous run (i.e., $Time_{\text{i}}/Time_{\text{i-1}}$).
Based on the timing data presented in this table, what is the most reasonable conclusion regarding the underlying big-oh time complexity of the program being timed?

A. $O(15)$

B. $O(\log_{15} N)$

C. $O(N)$

D. $O(N^4)$

- A
- B
- C
- D

Correct!

Question 30

1 / 1 pts

Which of the following algorithms was not an *in place* sort?

A. selection sort

B. insertion sort

C. merge sort

D. quicksort

- A
- B
- C

Correct!

- D

Quiz Score: **28** out of 30