completing their task. Deadlock occurs when two or more threads are blocked forever. Starvation occurs when a single thread is perpetually denied access to a shared resource. Livelock is a form of starvation where two or more threads are active but conceptually blocked forever. Finally, race conditions occur when two threads execute at the same time, resulting in an unexpected outcome.

Understand the impact of using parallel streams. The Stream API allows for the easy creation of parallel streams. Using a parallel stream can cause unexpected results, since the order of operations may no longer be predictable. Some operations, such as reduce() and collect(), require special consideration to achieve optimal performance when applied to a parallel stream.

Review Questions

The answers to the chapter review questions can be found in the Appendix.

1. Given the following code snippet, which options correctly create a parallel stream? (Choose all that apply.)

```
var c = new ArrayList<Thread>();
var s = c.stream();
var p = _______________;
```

   A. new ParallelStream(s)

   B. c.parallel()

   C. s.parallelStream()

   D. c.parallelStream()

   E. new ParallelStream(c)

   F. s.parallel()

2. Given that the sum of the numbers from 1 (inclusive) to 10 (exclusive) is 45, what are the possible results of executing the following program? (Choose all that apply.)

```
1:  import java.util.concurrent.locks.*;
2:  import java.util.stream.*;
3:  public class Bank {
4:    private Lock vault = new ReentrantLock();
5:    private int total = 0;
6:    public void deposit(int value) {
7:      try {
8:        vault.tryLock();
9:        total += value;
10:     } finally { vault.unlock(); }
}
public static void main(String[] unused) {
  var bank = new Bank();
  IntStream.range(1, 10).parallel()
    .forEach(s -> bank.deposit(s));
  System.out.println(bank.total);
}}
```

A. 45 is printed.

B. A number less than 45 is printed.

C. A number greater than 45 is printed.

D. An exception is thrown.

E. None of the above, as the code does not compile.

3. Which of the following statements about the Callable call() and Runnable run() methods are correct? (Choose all that apply.)

A. Both methods return void.

B. Both can throw unchecked exceptions.

C. Both can be implemented with lambda expressions.

D. Runnable returns a generic type.

E. Both can throw checked exceptions.

F. Callable returns a generic type.

4. Which lines need to be changed to make the code compile? (Choose all that apply.)

```
ExecutorService service =   // w1
  Executors.newSingleThreadScheduledExecutor();
service.scheduleWithFixedDelay(() -> {
  System.out.println("Open Zoo");
  return null;   // w2
}, 0, 1, TimeUnit.MINUTES);
var result = service.submit(() ->   // w3
  System.out.println("Wake Staff"));
System.out.println(result.get());   // w4
```

A. It compiles and runs without issue.

B. Line w1

C. Line w2

D. Line w3

E. Line w4

F. It compiles but throws an exception at runtime.

5. What statement about the following code is true?

```
var value1 = new AtomicLong(0);
final long[] value2 = {0};
IntStream.iterate(1, i -> 1).limit(100).parallel()
  .forEach(i -> value1.incrementAndGet());
IntStream.iterate(1, i -> 1).limit(100).parallel()
  .forEach(i -> ++value2[0]);
System.out.println(value1+" "+value2[0]);
```

A. It outputs 100 100.

B. It outputs 100 99.

C. The output cannot be determined ahead of time.

D. The code does not compile.

E. It compiles but throws an exception at runtime.

F. It compiles but enters an infinite loop at runtime.

G. None of the above

6. Which statements about the following code are correct? (Choose all that apply.)

```
var data = List.of(2,5,1,9,8);
data.stream().parallel()
  .mapToInt(s -> s)
  .peek(System.out::print)
  .forEachOrdered(System.out::print);
```

A. The peek() method will print the entries in the sorted order: 12589.

B. The peek() method will print the entries in the original order: 25198.

C. The peek() method will print the entries in an order that cannot be determined ahead of time.

D. The forEachOrdered() method will print the entries in the sorted order: 12589.

E. The forEachOrdered() method will print the entries in the original order: 25198.

F. The forEachOrdered() method will print the entries in an order that cannot be determined ahead of time.

G. The code does not compile.

7. Fill in the blanks: __________ occur(s) when two or more threads are blocked forever but both appear active. _______ occur(s) when two or

more threads try to complete a related task at the same time, resulting in invalid or unexpected data.

A. Livelock, Deadlock

B. Deadlock, Starvation

C. Race conditions, Deadlock

D. Livelock, Race conditions

E. Starvation, Race conditions

F. Deadlock, Livelock

8. Assuming this class is accessed by only a single thread at a time, what is the result of calling the countIceCreamFlavors() method?

```
import java.util.stream.LongStream;
public class Flavors {
  private static int counter;
  public static void countIceCreamFlavors() {
    counter = 0;
    Runnable task = () -> counter++;
    LongStream.range(0, 500)
      .forEach(m -> new Thread(task).run());
    System.out.println(counter);
  }}
```

A. The method consistently prints a number less than 500.

B. The method consistently prints 500.

C. The method compiles and prints a value, but that value cannot be determined ahead of time.

D. The method does not compile.

E. The method compiles but throws an exception at runtime.

F. None of the above

9. Which happens when a new task is submitted to an ExecutorService in which no threads are available?

A. The executor throws an exception when the task is submitted.

B. The executor discards the task without completing it.

C. The executor adds the task to an internal queue and completes when there is an available thread.

D. The thread submitting the task waits on the submit call until a thread is available before continuing.

E. The executor stops an existing task and starts the newly submitted one.

10. What is the result of executing the following code snippet?

```
List<Integer> lions = new ArrayList<>(List.of(1,2,3));
List<Integer> tigers = new CopyOnWriteArrayList<>(lions);
Set<Integer> bears = new ConcurrentSkipListSet<>();
bears.addAll(lions);
for(Integer item: tigers) tigers.add(4); // x1
for(Integer item: bears) bears.add(5);  // x2
System.out.println(lions.size() + " " + tigers.size()
  + " " + bears.size());
```

A. It outputs 3 6 4.

B. It outputs 6 6 6.

C. It outputs 6 3 4.

D. The code does not compile.

E. It compiles but throws an exception at runtime on line x1.

F. It compiles but throws an exception at runtime on line x2.

G. It compiles but enters an infinite loop at runtime.

11. What statements about the following code are true? (Choose all that apply.)

```
Integer i1 = List.of(1, 2, 3, 4, 5).stream().findAny().get();
synchronized(i1) { // y1
  Integer i2 = List.of(6, 7, 8, 9, 10)
    .parallelStream()
    .sorted()
    .findAny().get(); // y2
  System.out.println(i1 + " " + i2);
}
```

A. The first value printed is always 1.

B. The second value printed is always 6.

C. The code will not compile because of line y1.

D. The code will not compile because of line y2.

E. The code compiles but throws an exception at runtime.

F. The output cannot be determined ahead of time.

G. It compiles but waits forever at runtime.

12. Assuming each call to takeNap() takes five seconds to execute without throwing an exception, what is the expected result of executing the following code snippet?

```
ExecutorService service = Executors.newFixedThreadPool(4);
try {
  service.execute(() -> takeNap());
  service.execute(() -> takeNap());
  service.execute(() -> takeNap());
} finally {
  service.shutdown();
}
service.awaitTermination(2, TimeUnit.SECONDS);
System.out.println("DONE!");
```

A. It will immediately print DONE!.

B. It will pause for 2 seconds and then print DONE!.

C. It will pause for 5 seconds and then print DONE!.

D. It will pause for 15 seconds and then print DONE!.

E. It will throw an exception at runtime.

F. None of the above, as the code does not compile.

13. What statements about the following code are true? (Choose all that apply.)

```
System.out.print(List.of("duck","flamingo","pelican")
 .parallelStream().parallel()  // q1
 .reduce(0,
  (c1, c2) -> c1.length() + c2.length(),  // q2
  (s1, s2) -> s1 + s2));    // q3
```

A. It compiles and runs without issue, outputting the total length of all strings in the stream.

B. The code will not compile because of line q1.

C. The code will not compile because of line q2.

D. The code will not compile because of line q3.

E. It compiles but throws an exception at runtime.

F. None of the above

14. What statements about the following code snippet are true? (Choose all that apply.)

```
Object o1 = new Object();
Object o2 = new Object();
```

```
var service = Executors.newFixedThreadPool(2);
var f1 = service.submit(() -> {
  synchronized (o1) {
    synchronized (o2) { System.out.print("Tortoise"); }
  }
});
var f2 = service.submit(() -> {
  synchronized (o2) {
    synchronized (o1) { System.out.print("Hare"); }
  }
});
f1.get();
f2.get();
```

A. The code will always output Tortoise followed by Hare.

B. The code will always output Hare followed by Tortoise.

C. If the code does output anything, the order cannot be determined.

D. The code does not compile.

E. The code compiles but may produce a deadlock at runtime.

F. The code compiles but may produce a livelock at runtime.

G. It compiles but throws an exception at runtime.

15. Which statement about the following code snippet is correct?

```
2: var cats = Stream.of("leopard", "lynx", "ocelot", "puma")
3:   .parallel();
4: var bears = Stream.of("panda","grizzly","polar").parallel();
5: var data = Stream.of(cats,bears).flatMap(s -> s)
6:   .collect(Collectors.groupingByConcurrent(
7:     s -> !s.startsWith("p")));
8: System.out.println(data.get(false).size()
9:   + " " + data.get(true).size());
```

A. It outputs 3 4.

B. It outputs 4 3.

C. The code will not compile because of line 6.

D. The code will not compile because of line 7.

E. The code will not compile because of line 8.

F. It compiles but throws an exception at runtime.

16. Assuming one minute is enough time for all the threads within this program to complete, what are the possible results of executing the following program? (Choose all that apply.)

```
public class RocketShip {
  private volatile int fuel;
  private void launch(int checks) {
    var p = new ArrayList<Thread>();
    for (int i = 0; i < checks; i++)
      p.add(new Thread(() -> fuel++));
    p.forEach(Thread::interrupt);
    p.forEach(Thread::start);
    p.forEach(Thread::interrupt);
  }
  public static void main(String[] args) throws Exception {
    var ship = new RocketShip();
    ship.launch(100);
    Thread.sleep(60*1000);
    System.out.print(ship.fuel);
  } }
```

A. It prints a number less than 100.

B. It prints 100.

C. It prints a number greater than 100.

D. It does not compile.

E. It compiles but throws an InterruptedException at runtime.

17. Which statements about methods in ReentrantLock are correct? (Choose all that apply.)

A. The lock() method will attempt to acquire a lock without waiting indefinitely for it.

B. The testLock() method will attempt to acquire a lock without waiting indefinitely for it.

C. The attemptLock() method will attempt to acquire a lock without waiting indefinitely for it.

D. By default, a ReentrantLock fairly releases to each thread in the order in which it was requested.

E. Calling the unlock() method once will release a resource so that other threads can obtain the lock.

F. None of the above

18. Which of the following are valid Callable expressions? (Choose all that apply.)

A. a -> {return 10;}

B. () -> {String s = "";}

C. () -> 5

D. () -> {return null}

E. () -> "The" + "Zoo"

F. (int count) -> count+1

G. () -> {System.out.println("Giraffe"); return 10;}

19. What is the result of executing the following application? (Choose all that apply.)

```
import java.util.concurrent.*;
import java.util.stream.*;
public class PrintConstants {
 public static void main(String[] args) {
   var s = Executors.newScheduledThreadPool(10);
   DoubleStream.of(3.14159,2.71828)  // b1
     .forEach(c -> s.submit(       // b2
       () -> System.out.println(10*c)));  // b3
   s.execute(() -> System.out.println("Printed"));
 } }
```

A. It compiles and outputs the two numbers followed by Printed.

B. The code will not compile because of line b1.

C. The code will not compile because of line b2.

D. The code will not compile because of line b3.

E. It compiles, but the output cannot be determined ahead of time.

F. It compiles but throws an exception at runtime.

G. It compiles but waits forever at runtime.

20. What is the result of executing the following program? (Choose all that apply.)

```
import java.util.*;
import java.util.concurrent.*;
import java.util.stream.*;
public class PrintCounter {
 static int count = 0;
 public static void main(String[] args) throws
         InterruptedException, ExecutionException {
   var service = Executors.newSingleThreadExecutor();
   try {
     var r = new ArrayList<Future<?>>();
     IntStream.iterate(0,i -> i+1).limit(5).forEach(
    i -> r.add(service.execute(() -> {count++;})) // n1
   );
   for(Future<?> result : r) {
    System.out.print(result.get()+" "); // n2
   }
  } finally { service.shutdown(); }
 }}
```

A. It prints 0 1 2 3 4

B. It prints 1 2 3 4 5

C. It prints null null null null null

D. It hangs indefinitely at runtime.

E. The output cannot be determined.

F. The code will not compile because of line n1.

G. The code will not compile because of line n2.

21. Given the following code snippet and blank lines on p1 and p2, which values guarantee that 1 is printed at runtime? (Choose all that apply.)

```
var data = List.of(List.of(1,2),
 List.of(3,4),
 List.of(5,6));
data.___________  // p1
 .flatMap(s -> s.stream())
 .___________  // p2
 .ifPresent(System.out::print);
```

A. stream() on line p1, findFirst() on line p2

B. stream() on line p1, findAny() on line p2

C. parallelStream() on line p1, findAny() on line p2

D. parallelStream() on line p1, findFirst() on line p2

E. The code does not compile regardless of what is inserted into the blanks.

F. None of the above

22. Assuming one minute is enough time for the tasks submitted to the service executor to complete, what is the result of executing countSheep()? (Choose all that apply.)

```
import java.util.concurrent.*;
import java.util.concurrent.atomic.*;
public class BedTime {
 private AtomicInteger s1 = new AtomicInteger(0); // w1
private int s2 = 0;

private void countSheep() throws InterruptedException {
  var service = Executors.newSingleThreadExecutor(); // w2
  try {
    for (int i = 0; i < 100; i++)
    service.execute(() -> {
      s1.getAndIncrement(); s2++; }); // w3
    Thread.sleep(60*1000);
    System.out.println(s1 + " " + s2);
  } finally { service.shutdown(); }
}
public static void main(String... nap) throws InterruptedException {
  new BedTime().countSheep();
} }
```

A. The method consistently prints 100 99.

B. The method consistently prints 100 100.

C. The output cannot be determined ahead of time.

D. The code will not compile because of line w1.

E. The code will not compile because of line w2.

F. The code will not compile because of line w3.

G. It compiles but throws an exception at runtime.

23. What is the result of executing the following application? (Choose all that apply.)

```
import java.util.concurrent.*;
import java.util.stream.*;
public class StockRoomTracker {
  public static void await(CyclicBarrier cb) { // j1
    try { cb.await(); } catch (Exception e) {}
  }
  public static void main(String[] args) {
    var cb = new CyclicBarrier(10,
      () -> System.out.println("Stock Room Full!")); // j2
    IntStream.iterate(1, i -> 1).limit(9).parallel()
      .forEach(i -> await(cb)); // j3
} }
```

A. It outputs Stock Room Full!

B. The code will not compile because of line j1.

C. The code will not compile because of line j2.

D. The code will not compile because of line j3.

E. It compiles but throws an exception at runtime.

F. It compiles but waits forever at runtime.

24. What statements about the following class definition are true? (Choose all that apply.)

```
public final class TicketManager {
  private int tickets;
  private static TicketManager instance;
  private TicketManager() {}
  static synchronized TicketManager getInstance() {     // k1
    if (instance==null) instance = new TicketManager(); // k2
    return instance;
  }

  public int getTicketCount() { return tickets; }
  public void addTickets(int value) {tickets += value;}  // k3
  public void sellTickets(int value) {
    synchronized (this) {                    // k4
      tickets -= value;
    }}}
```

A. It compiles without issue.

B. The code will not compile because of line k2.

C. The code will not compile because of line k3.

D. The locks acquired on k1 and k4 are on the same object.

E. The class correctly protects the tickets data from race conditions.

F. At most one instance of TicketManager will be created in an application that uses this class.

25. Assuming an implementation of the performCount() method is provided prior to runtime, which of the following are possible results of executing the following application? (Choose all that apply.)

```
import java.util.*;
import java.util.concurrent.*;
public class CountZooAnimals {
  public static void performCount(int animal) {
    // IMPLEMENTATION OMITTED
  }
  public static void printResults(Future<?> f) {
    try {
      System.out.println(f.get(1, TimeUnit.DAYS)); // o1
  } catch (Exception e) {
    System.out.println("Exception!");
  }
}
public static void main(String[] args) throws Exception {
  final var r = new ArrayList<Future<?>>();
  ExecutorService s = Executors.newSingleThreadExecutor();
  try {
    for(int i = 0; i < 10; i++) {
      final int animal = i;
      r.add(s.submit(() -> performCount(animal))); // o2
    }
    r.forEach(f -> printResults(f));
  } finally { s.shutdown(); }
}}
```

A. It outputs a number 10 times.

B. It outputs a Boolean value 10 times.

C. It outputs a null value 10 times.

D. It outputs Exception! 10 times.

E. It hangs indefinitely at runtime.

F. The code will not compile because of line o1.

G. The code will not compile because of line o2.