OOPJ Mock CCEE-3

Total points 11/40

The respondent's email (**shreyasbhatkar22@gmail.com**) was recorded on submission of this form.

0 of 0 points

Student ID: *

240840520075

Name: *

Shreyas

OOPJ MCQ's

11 of 40 points

All The Best!!!!

✗ Which one of the following statements will compile without errors? * 0/1

- ○ Locale locale1 = new Locale.US;
- ○ Locale locale2 = Locale.US;
- ○ Locale locale3 = new US.Locale();
- ◉ Locale locale4 = Locale("US"); ✗

Correct answer

- ◉ Locale locale2 = Locale.US;

✕ Consider the following program: * 0/1

```
class ClassA {}
interface InterfaceB {}
class ClassC {}
class Test extends ClassA implements InterfaceB { String msg;
 ClassC classC;
}
```

Which one of the following statements is true?

- ○ Class Test is related with String with an IS-A relationship.
- ○ Class Test is related to ClassC with a composition relationship.
- ◉ Class Test is related with ClassA with a HAS-A relationship. ✕
- ○ Class ClassA is related with InterfaceB with an IS-A relationship.

Correct answer

- ◉ Class Test is related to ClassC with a composition relationship.

✓ Which one of the following interfaces declares a single abstract method named iterator()? *1/1
(Note: Implementing this interface allows an object to be the target of the for-each statement.)

- ○ Iterator<T>
- ◉ Iterable<T> ✓
- ○ ForEach<T>
- ○ Enumeration<E>

✗ Given this code segment: * 0/1

```
 Set set = new CopyOnWriteArraySet(); // #1
set.add("2");
set.add("1");
Iterator iter = set.iterator();
set.add("3"); set.add("-1");
while(iter.hasNext()) {
 System.out.print(iter.next() + " ");
}
```

- ○ This code segment prints the following: 2 1
- ○ This code segment the following: 1 2
- ○ This code segment prints the following: -1 1 2 3
- ◉ This code segment prints the following: 2 1 3 -1 ✗

Correct answer

- ◉ This code segment prints the following: 2 1

✗ Which of the following is NOT a problem associated with thread synchronization using mutexes? *0/1

- (●) Deadlock ✗
- ( ) Lock starvation
- ( ) Type erasure
- ( ) Livelock

Correct answer

- (●) Type erasure

✓ Given this class definition: *1/1

```
class Point {
 private int x = 0, y;
 public Point(int x, int y) {
 this.x = x;
 this.y = y;
 }
 // DEFAULT_CTOR
 }
```

Which one of the following definitions of the Point constructor can be replaced without compiler errors in place of the comment DEFAULT_CTOR?

- ○ public Point() { this(0, 0); super();}
- ○ public Point() { super(); this(0, 0); }
- ◉ private Point() { this(0, 0);} ✓
- ○ public Point() { this();}

✓ Which of the following classes in the java.util.concurrent.atomic package inherit from java.lang.Number? *1/1

- (●) AtomicInteger ✓
- ( ) AtomicBoolean
- ( ) AtomicFloat
- ( ) AtomicDouble

✗ Consider the following program: *0/1

```
class Outer {
class Inner {
 public void print() {
 System.out.println("Inner: print");
 }
 } }
class Test {
 public static void main(String []args) {
 // Stmt#1
 inner.print();
 } }
```

Which one of the following statements will you replace with // Stmt#1 to make the program compile and run successfully to print “Inner: print” in console?

- (●) Outer.Inner inner = new Outer().Inner(); ✗
- ( ) Outer.Inner inner = new Outer().new Inner();
- ( ) Outer.Inner inner = new Outer.Inner();
- ( ) Inner inner = new Outer.Inner();

Correct answer

- (●) Outer.Inner inner = new Outer().new Inner();

✗ Consider the following program and choose the right option from the given list: *0/1

```
class Base {
 public void test() {
 protected int a = 10; // #1
 }
}
class Test extends Base { // #2
 public static void main(String[] args) { System.out.printf(null); // #3
 }
}
```

- ○ The compiler will report an error at statement marked with the comment #1
- ○ The compiler will report an error at statement marked with the comment #2
- ○ The compiler will report errors at statement marked with the comment #3
- ◉ The program will compile without any error ✗

Correct answer

- ◉ The compiler will report an error at statement marked with the comment #1

✗ Given this code snippet: *0/1

```
public static Connection connectToDb() throws SQLException {
 String url = "jdbc:mysql://localhost:3306/";
 String database = "addressBook";
 String userName = "root";
 String password = "mysql123";
 // CONNECT_TO_DB
}
```

Which one of the following statements will you replace with the comment CONNECT_TO_DB to create a Connection object?

- ○ return DatabaseManager.getConnection(url, database, userName,password);
- ◉ return Connection.getConnection(url, database, userName,password); ✗
- ○ return DriverManager.getConnection(url + database, userName,password);
- ○ return DatabaseDriver.getConnection(url + database, userName,password);

Correct answer

- ◉ return DriverManager.getConnection(url + database, userName,password);

✕ Given this code segment: * 0/1

```
LocalDate joiningDate = LocalDate.of(2014,Month.SEPTEMBER, 20);
LocalDate now = LocalDate.of(2015, Month.OCTOBER,20);
// GET_YEARS
System.out.println(years);
```

- ○ Duration years = Period.between(joiningDate, now).getYears();
- ○ int years = Period.between(joiningDate, now).getYears();
- ◉ Period years = Period.between(joiningDate, now).getYears(); ✕
- ○ Instant years = Period.between(joiningDate, now).getYears();

Correct answer

- ◉ int years = Period.between(joiningDate, now).getYears();

✓ For the following enumeration definition, which one of the following prints the value 2 in the console? *1/1

enum Pets { Cat, Dog, Parrot, Chameleon };

- (●) System.out.print(Pets.Parrot.ordinal()); ✓
- ( ) System.out.print(Pets.Parrot);
- ( ) System.out.print(Pets.indexAt("Parrot"));
- ( ) System.out.print(Pets.Parrot.value());

✕ Choose the correct option based on this code segment: *0/1

Path path = Paths.get("file.txt"); // READ_FILE
lines.forEach(System.out::println);
Assume that a file named “file.txt” exists in the directory in which this code segment is run and has the content “hello”. Which one of these options can be replaced by the text READ_FILE that will successfully read the “file.txt” and print “hello” on the console?

- ○ List<String> lines = Files.lines(path);
- ○ Stream<String> lines = Files.lines(path);
- ◉ Stream<String> lines = File.readLines(path); ✕
- ○ Stream<String> lines = Files.readAllLines(path);

Correct answer

- ◉ Stream<String> lines = Files.lines(path);

✕ Given this code snippet: *0/1

```
LocalDate dateOfBirth = LocalDate.of(1988,Month.NOVEMBER, 4);
MonthDay monthDay =MonthDay.of(dateOfBirth.getMonth(),
dateOfBirth.getDayOfMonth());
boolean ifTodayBirthday =
monthDay.equals(MonthDay.from(LocalDate.now())); //COMPARE
System.out.println(ifTodayBirthday ? "Happy birthday!" :"Yet another
day!");
```

Assume that today's date is 4th November 2015. Choose the correct answer based on this code segment.

- (●) This code will result in a compiler error in the line marked with the comment COMPARE ✕
- ( ) When executed, this code will throw DateTimeException
- ( ) This code will print: Happy birthday!
- ( ) This code will print: Yet another day!

Correct answer

- (●) This code will print: Happy birthday!

✗ Two friends are waiting for some more friends to come so that they can go to a restaurant for dinner together. Which synchronization construct could be used here to programmatically simulate this situation? *0/1

- ○ java.util.concurrent.RecursiveAction
- ◉ java.util.concurrent.locks.Lock ✗
- ○ java.util.concurrent.RecursiveTask
- ○ java.util.concurrent.CyclicBarrier

Correct answer

- ◉ java.util.concurrent.CyclicBarrier

Given this code segment: *1/1

```
final CyclicBarrier barrier = new CyclicBarrier(3, () ->
System.out.println("Let's play"));
 // LINE_ONE
Runnable r = () -> {
// LINE_TWO
 System.out.println("Awaiting");
 try {
 barrier.await();
 } catch(Exception e) { /* ignore */ }
};
Thread t1 = new Thread(r);
Thread t2 = new Thread(r);
Thread t3 = new Thread(r);
t1.start();
t2.start();
t3.start();
```

Choose the correct option based on this code segment.

- This code segment results in a compiler error in line marked with the comment LINE_ONE
- This code prints: Let's play
- This code prints: Awaiting Awaiting Awaiting Let's play ✓
- This code segment results in a compiler error in line marked with the comment LINE_TWO

✕ Choose the correct option based on this code segment: * 0/1

```
Stream words = Stream.of("eeny", "meeny", "miny", "mo");
// LINE_ONE
String boxedString = words.collect(Collectors.joining(", ", "[", "]"));
// LINE_TWO
System.out.println(boxedString);
```

- ○ This code results in a compiler error in line marked with the comment LINE_ONE
- ◉ This code results in a compiler error in line marked with the comment LINE_TWO ✕
- ○ This program prints: [eeny, meeny, miny, mo]
- ○ This program prints: [eeny], [meeny], [miny], [mo]

Correct answer

- ◉ This program prints: [eeny, meeny, miny, mo]

✓ Choose the correct option based on this code segment: * 1/1

```
List ints = Arrays.asList(1, 2, 3, 4, 5);
ints.replaceAll(i -> i * i); // LINE
System.out.println(ints);
```

- ○ This code segment prints: [1, 2, 3, 4, 5]
- ◉ This program prints: [1, 4, 9, 16, 25] ✓
- ○ This code segment results in a compiler error in the line marked with the comment LINE
- ○ This code segment throws java.lang.UnsupportedOperationException

✗ Consider the following snippet: *0/1

```
int ch = 0;
try (FileReader inputFile = new FileReader(file)) {
 // #1
 System.out.print( (char)ch );
 }
 }
```

Which one of the following statements can be replaced with statement #1 so that the contents of the file are correctly printed on the console and the program terminates.

- ○ while((ch = inputFile.read()) != 0) {
- ○ while((ch = inputFile.read()) != -1) {
- ○ while((ch = inputFile.read()) != null) {
- ◉ while((ch = inputFile.read()) != EOF) { ✗

Correct answer

- ◉ while((ch = inputFile.read()) != -1) {

✗ Consider the following program: *0/1

```
public class Outer {
 private int mem = 10;
 class Inner {
 private int imem = new Outer().mem; // ACCESS1
 }
 public static void main(String []s) { System.out.println(new Outer().new
Inner().imem); //ACCESS2
 }
}
```

Which one of the following options is correct?

- ○ When executed, this program prints 0
- ○ When executed, this program prints 10
- ○ When compiled, this program will result in a compiler error in line marked with comment ACCESS2
- ◉ When compiled, this program will result in a compiler error in line marked with comment ACCESS1 ✗

Correct answer

- ◉ When executed, this program prints 10

✗ Given this code segment: *0/1

IntFunction> func = i -> j -> i * j; // LINE System.out.println(apply);

Which one of these statements when replaced by the comment marked with LINE will print 200?

- ○ Integer apply = func.apply(10).apply(20);
- ○ Integer apply = func.apply(10, 20);
- ◉ Integer apply = func(10 , 20); ✗
- ○ Integer apply = func(10, 20).apply();

Correct answer

- ◉ Integer apply = func.apply(10).apply(20);

✓ Given this code segment: *1/1

```
BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
String integer = br.readLine();
// CODE
System.out.println(val);
```

Which one of the following statements when replaced by the comment CODE will successfully read an integer value from console?

- ○ int val = integer.getInteger();
- ◉ int val = Integer.parseInt(integer); ✓
- ○ int val = String.parseInt(integer);
- ○ int val = Number.parseInteger(integer);

✗ Which one of the following interfaces is empty (i.e., an interface that does not declare any methods)? *0/1

- ○ java.lang.AutoCloseable interface
- ◉ java.util.concurrent.Callable<T> interface ✗
- ○ java.lang.Cloneable interface
- ○ java.lang.Comparator<T> interface

Correct answer

- ◉ java.lang.Cloneable interface

✓ Choose the correct option for this code snippet: * 1/1

```
public static void main(String []files) {
try (FileReader inputFile = new FileReader(new File(files[0]))) {// #1
 inputFile.close(); // #2
 }
 catch (FileNotFoundException | IOException e) { // #3
 e.printStackTrace();
 } }
```

- ○ The code snippet will compile without any errors
- ○ The compiler will report an error at statement marked with the comment #1
- ○ The compiler will report an error at statement marked with the comment #2
- ◉ The compiler will report an error at statement marked with the comment #3 ✓

✗ In the context of Singleton pattern, which one of the following statements is true? *0/1

- ○ A Singleton class has a public constructor
- ○ A Factory class may use Singleton pattern
- ◉ A Singleton class must not have any static members ✗
- ○ All methods of the Singleton class must be private

Correct answer

- ◉ A Factory class may use Singleton pattern

✗ Choose the correct option based on the following code segment: *0/1

```
Comparator comparer = (country1, country2) -
country2.compareTo(country2);
// COMPARE_TO
 String[] brics = {"Brazil", "Russia", "India", "China"};
Arrays.sort(brics, null);
 Arrays.stream(brics).forEach(country ->System.out.print(country + " "));
```

- (●) The program results in a compiler error in the line marked with the comment COMPARE_TO ✗
- ( ) The program prints the following: Brazil Russia India China
- ( ) The program prints the following: Brazil China India Russia
- ( ) The program prints the following: Russia India China Brazil

Correct answer

- (●) The program prints the following: Brazil China India Russia

What will be the output of the following program? *1/1

```
class Base { public Base() {
 System.out.println("Base");
 } }
class Derived extends Base { public Derived() {
System.out.println("Derived");
 } }
class DeriDerived extends Derived { public DeriDerived(){
System.out.println("DeriDerived");
 } }
 class Test { public static void main(String []args) { Derived b = new
DeriDerived();
 } }
```

- ( ) DeriDerived Derived Base
- (•) Base Derived DeriDerived ✓
- ( ) Derived DeriDerived
- ( ) DeriDerived Derived
- ( ) DeriDerived

✓ Given the code segment: *1/1

List integers = Arrays.asList(15, 5, 10, 20, 25, 0);

// GETMAX

Which of the code segments can be replaced for the comment marked with GETMAX to return the maximum value?

- (●) Integer max = integers.stream().max((i, j) -> i - j).get(); ✓
- ( ) Integer max = integers.stream().max().get();
- ( ) Integer max = integers.max();
- ( ) Integer max = integers.stream().mapToInt(i -> i).max();

✗ Which of the following statements is true with respect to enums? * 0/1

- ( ) An enum can have private constructor, enum can have public methods and fields
- ( ) An enum can extend a class
- ( ) An enum can have public constructor
- (●) An enum cannot implement an interface ✗

Correct answer

- (●) An enum can have private constructor, enum can have public methods and fields

✗ Choose the correct option based on this program: *0/1

```
class base1 {
 protected int var;
}
interface base2 {
 int var = 0; // #1
}
class Test extends base1 implements base2 { // #2 public static void
main(String args[]) {
System.out.println("var:" + var); // #3
 } }
```

- ○ The program will compile without any errors
- ◉) The program will report a compilation error at statement marked with the comment #1 ✗
- ○ The program will report a compilation error at statement marked with the comment #2
- ○ The program will report a compilation error at statement marked with the comment #3

Correct answer

- ◉ The program will report a compilation error at statement marked with the comment #3

Choose the correct option based on this program: *0/1

```
 import java.util.stream.Stream;
public class Reduce {
 public static void main(String []args) {
 Stream words = Stream.of("one", "two", "three");
 int len = words.mapToInt(String::length).reduce(0, (len1, len2) -> len1 +
len2);
 System.out.println(len);
 } }
```

- This program does not compile and results in compiler error(s)
- This program prints: onetwothree
- This program prints: 11
- This program throws an IllegalArgumentException

Correct answer

- This program prints: 11

✓ Given this code segment: *1/1

```
DateTimeFormatter fromDateFormat
=DateTimeFormatter.ofPattern("MM/dd/yyyy");
// PARSE_DATE
DateTimeFormatter toDateFormat
=DateTimeFormatter.ofPattern("dd/MMM/YY");
System.out.println(firstOct2015.format(toDateFormat));
```

Which one of the following statements when replaced with the comment PARSE_DATE will result in the code to print “10/Jan/15”?

- (●) LocalDate firstOct2015 = LocalDate.parse("01/10/2015", fromDateFormat); ✓
- ( ) Period firstOct2015 = Period.parse("01/10/2015", fromDateFormat);
- ( ) DateTimeFormatter firstOct2015 = DateTimeFormatter.parse("01/10/2015", fromDateFormat);
- ( ) LocalTime firstOct2015 = LocalTime.parse("01/10/2015",fromDateFormat);

✗ Consider the following program: * 0/1

```
class WildCard {
 interface BI {}
 interface DI extends BI {}
 interface DDI extends DI {}
 static class C {}
 static void foo(C arg) {}
public static void main(String []args) {
 foo(new C()); // ONE
 foo(new C()); // TWO
 foo(new C()); // THREE
 foo(new C()); // FOUR
 } }
```

Which of the following options are correct?

- (●) Line marked with comment ONE will result in a compiler error ✗
- ( ) Line marked with comment TWO will result in a compiler error
- ( ) Line marked with comment THREE will result in a compiler error
- ( ) Line marked with comment FOUR will result in a compiler error

Correct answer

- (●) Line marked with comment THREE will result in a compiler error

✗ Choose the correct option based on this code segment: * 0/1

```
Stream ints = Stream.of(1, 2, 3, 4);
boolean result =ints.parallel().map(Function.identity()).isParallel();
System.out.println(result);
```

- ○ This code segment prints: false
- ○ This code segment results in compiler error(s)
- ○ This code segment prints: true
- ◉ This code segment throws InvalidParallelizationException for the call parallel() ✗

Correct answer

- ◉ This code segment prints: true

Consider the following code segment: *0/1

```
while( (ch = inputFile.read()) != VALUE) {outputFile.write( (char)ch );
}
```

Assume that inputFile is of type FileReader, and outputFile is of type FileWriter, and ch is of type int. The method read() returns the character if successful, or VALUE if the end of the stream has been reached. What is the correct value of this VALUE checked in the while loop for end-of-stream?

- Integer.MAX_VALUE
- -1
- 0 ✗
- 255

Correct answer

- -1

✗ Given the class definition: * 0/1

```
class Student{
 public Student(int r) {
 rollNo = r;
 }
 int rollNo;
}
```

Choose the correct option based on this code segment:

```
 HashSet students = new HashSet<>();
students.add(new Student(5));
students.add(new Student(10));
 System.out.println(students.contains(new Student(10)));
```

- (●) This program prints the following: true ✗
- ( ) This program prints the following: false
- ( ) This program results in compiler error(s)
- ( ) This program throws NoSuchElementException

Correct answer

- (●) This program prints the following: false

✗ What will be the result of executing this code segment? *0/1

```
Stream.of("ace ", "jack ", "queen ", "king ", "joker ") .mapToInt(card ->
card.length()) .filter(len -> len > 3) .peek(System.out::print) .limit(2);
```

- ○ This code segment does not print anything on the console
- ○ This code segment prints: jack queen king joker
- ◉ This code segment prints: jack queen ✗
- ○ This code segment prints: king joker

Correct answer

- ◉ This code segment does not print anything on the console

✗ Consider the following definitions: *0/1

interface BI {}
interface DI extends BI {}

The following options provide definitions of a template class X. Which one of the options specifies class X with a type parameter whose upper bound declares DI to be the super type from which all type arguments must be derived?

- ○ class X <T super DI> { }
- ○ class X <T implements DI> { }
- ○ class X <T extends DI> { }
- ◉ class X <T extends ? & DI> { } ✗

Correct answer

- ◉ class X <T extends DI> { }

✗ Select all the statements that are true about streams (supported in java.util.stream.Stream interface)? *0/1

- [x] Computation on source data is performed in a stream only when the terminal operation is initiated, i.e., streams are “lazy” ✓
- [x] If the stream source is modified when the computation in the stream is being performed, then it may result in unpredictable or erroneous results ✓
- [x] Once a stream is created as a sequential stream, its execution mode cannot be changed to parallel stream (and vice versa) ✗
- [x] Once a terminal operation is invoked on a stream, it is considered consumed and cannot be used again ✓

Correct answer

- [x] Computation on source data is performed in a stream only when the terminal operation is initiated, i.e., streams are “lazy”
- [x] Once a terminal operation is invoked on a stream, it is considered consumed and cannot be used again
- [x] If the stream source is modified when the computation in the stream is being performed, then it may result in unpredictable or erroneous results

✗ Which one of the following options is best suited for generating random numbers in a multi-threaded application? *0/1

- (●) Using java.lang.Math.random() ✗
- ( ) Using java.util.concurrent.ThreadLocalRandom
- ( ) Using java.util.RandomAccess

Correct answer

- (●) Using java.util.concurrent.ThreadLocalRandom