# Free Questions for 1Z0-809 by ebraindumps

## Shared by Hamilton on 06-06-2022

For More Free Questions and Preparation Resources

Check the Links on Last Page

# Question 1

## Question Type: **MultipleChoice**

Given:

```
class Student {
String course, name, city;
public Student (String name, String course, String city) {
this.course = course; this.name = name; this.city = city;
}
public String toString() {
return course + ":" + name + ":" + city;
}
public String getCourse() {return course;}
public String getName() {return name;}
public String getCity() {return city;}
```

and the code fragment:

```
List stds = Arrays.asList(
new Student ("Jessy", "Java ME", "Chicago"),
new Student ("Helen", "Java EE", "Houston"),
new Student ("Mark", "Java ME", "Chicago"));
stds.stream()
.collect(Collectors.groupingBy(Student::getCourse))
.forEach(src, res) -> System.out.println(scr));
```

What is the result?

## Options:

**A-** A compilation error occurs.

**B-** Java EEJava ME

**C-** [Java EE: Helen:Houston][Java ME: Jessy:Chicago, Java ME: Mark:Chicago]

**D-** [Java ME: Jessy:Chicago, Java ME: Mark:Chicago][Java EE: Helen:Houston]

### Answer:

B

## Question 2

**Question Type: MultipleChoice**

Given:

```
public class Product {

int id; int price;

public Product (int id, int price) {

this.id = id;

this.price = price;

}

Public String toString () { return id + ":" + price;)

}
```

and the code fragment:

```
List products = new ArrayList (Arrays.asList(new Product(1, 10),
new Product (2, 30),
new Product (3, 20));
Product p = products.stream().reduce(new Product (4, 0), (p1, p2) -> {
p1.price+=p2.price;
return new Product (p1.id, p1.price);});
products.add(p);
products.stream().parallel()
.reduce((p1, p2) - > p1.price > p2.price ? p1 : p2)
.ifPresent(System.out: :println);
```

What is the result?

## Options:

**A-** 4:60

**B-** 2:30

**C-** 4:602:303:201:10

**D-** 4:0

**E-** The program prints nothing

### Answer:

C

## Question 3

**Question Type:** **MultipleChoice**

Given the code fragment:

```
List doubles = Arrays.asList (100.12, 200.32);

DoubleFunction funD = d --> d + 100.0;

doubles.stream (). forEach (funD); // line n1

doubles.stream(). forEach(e --> System.out.println(e)); // line n2
```

What is the result?

## Options:

**A-** A compilation error occurs at line n2.

**B-** 200.12300.32

**C-** 100.12200.32

**D-** A compilation error occurs at line n1.

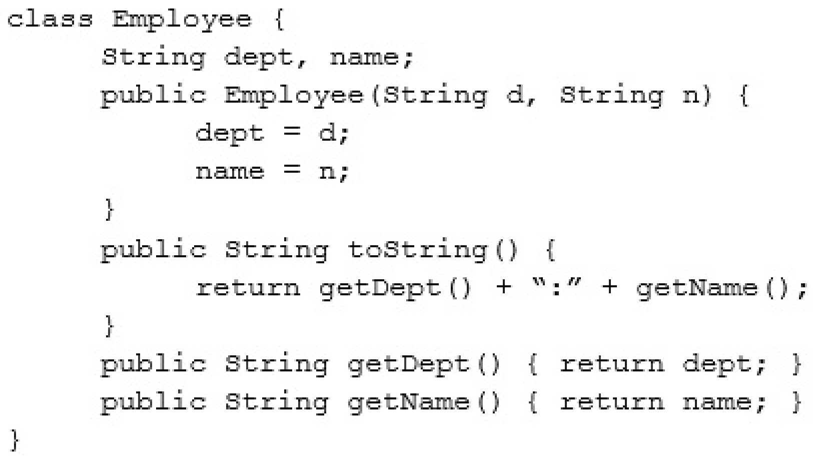
## Answer:

A

# Question 4

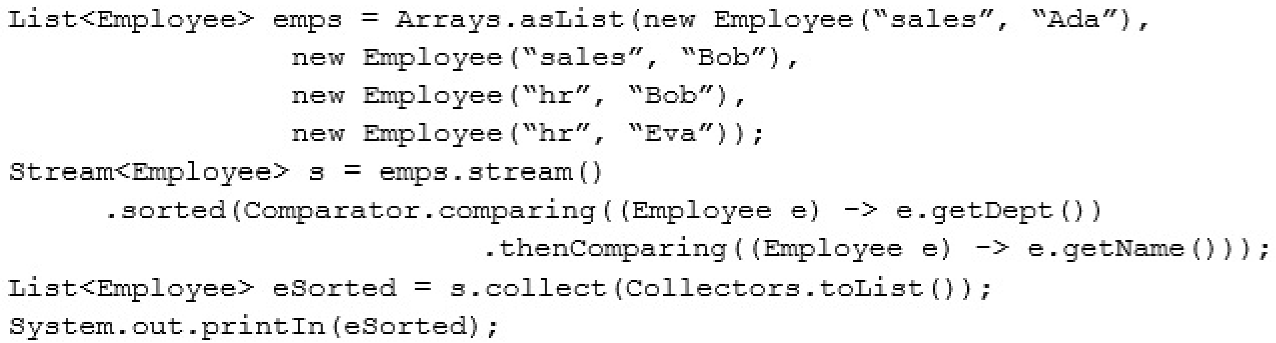
**Question Type:** **MultipleChoice**

Given the definition of the Employee class:



```
class Employee {
      String dept, name;
      public Employee(String d, String n) {
            dept = d;
            name = n;
      }
      public String toString() {
            return getDept() + “:” + getName();
      }
      public String getDept() { return dept; }
      public String getName() { return name; }
}
```

and this code fragment:



```
List<Employee> emps = Arrays.asList(new Employee(“sales”, “Ada”),
                  new Employee(“sales”, “Bob”),
                  new Employee(“hr”, “Bob”),
                  new Employee(“hr”, “Eva”));
Stream<Employee> s = emps.stream()
      .sorted(Comparator.comparing((Employee e) -> e.getDept())
                        .thenComparing((Employee e) -> e.getName()));
List<Employee> eSorted = s.collect(Collectors.toList());
System.out.printIn(eSorted);
```

What is the result?

## Options:

**A-** [sales:Ada, hr:Bob, sales:Bob, hr:Eva]

**B-** [Ada:sales, Bob:sales, Bob:hr, Eva:hr]

**C-** [hr:Eva, hr:Bob, sales:Bob, sales:Ada]

**D-** [hr:Bob, hr:Eva, sales:Ada, sales:Bob]

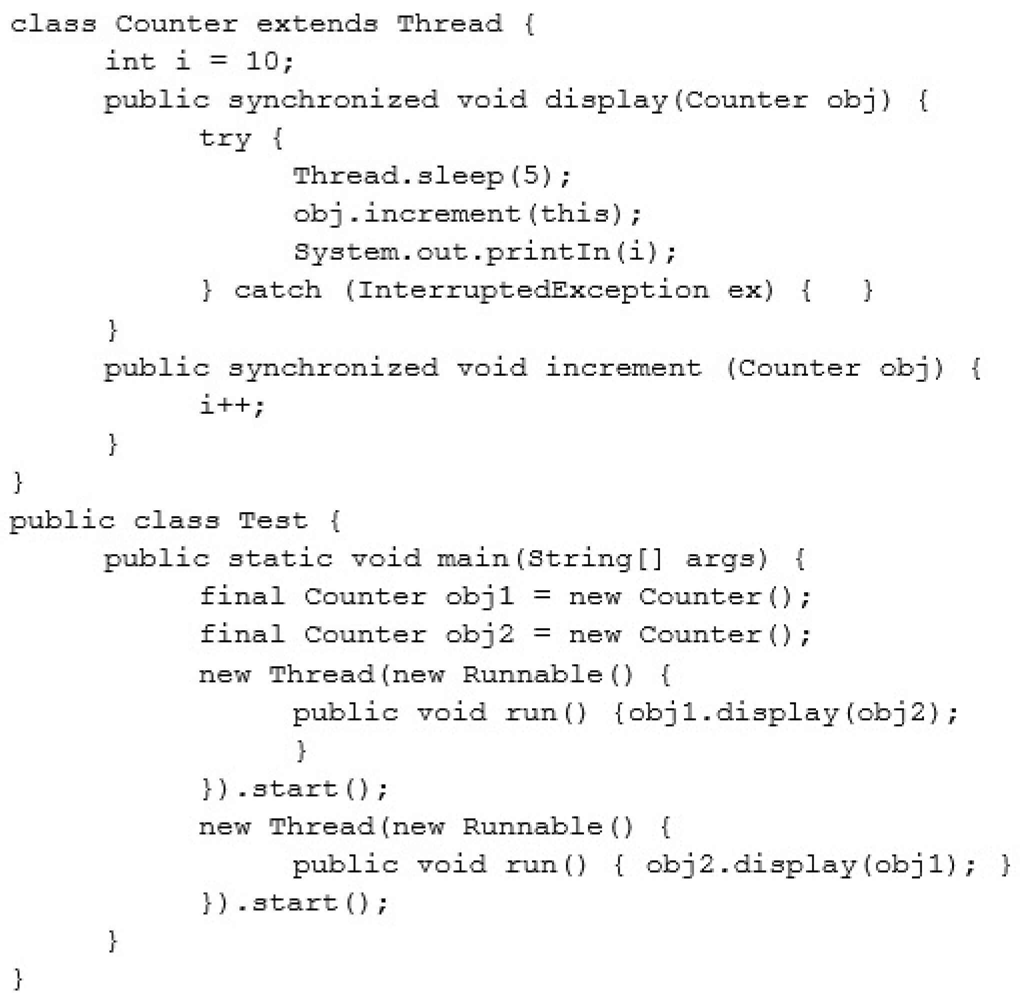
### Answer:

A

## Question 5

**Question Type: MultipleChoice**

Given:



```
class Counter extends Thread {
      int i = 10;
      public synchronized void display(Counter obj) {
            try {
                  Thread.sleep(5);
                  obj.increment(this);
                  System.out.printIn(i);
            } catch (InterruptedException ex) {   }
      }
      public synchronized void increment (Counter obj) {
            i++;
      }
}
public class Test {
      public static void main(String[] args) {
            final Counter obj1 = new Counter();
            final Counter obj2 = new Counter();
            new Thread(new Runnable() {
                  public void run() {obj1.display(obj2);
                  }
            }).start();
            new Thread(new Runnable() {
                  public void run() { obj2.display(obj1); }
            }).start();
      }
}
```

From what threading problem does the program suffer?

## Options:

**A-** race condition

**B-** deadlock

**C-** starvation

**D-** livelock

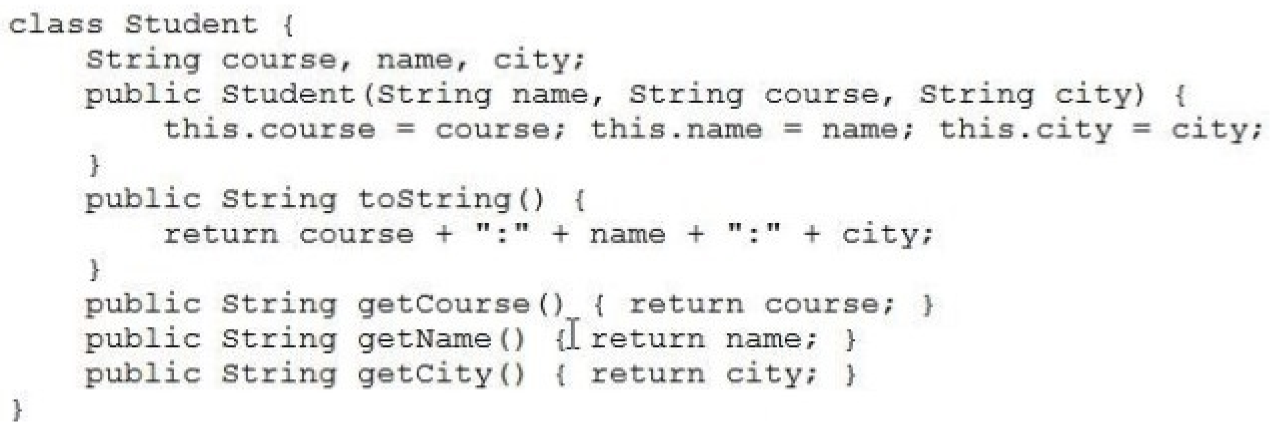
## Answer:

B

# Question 6

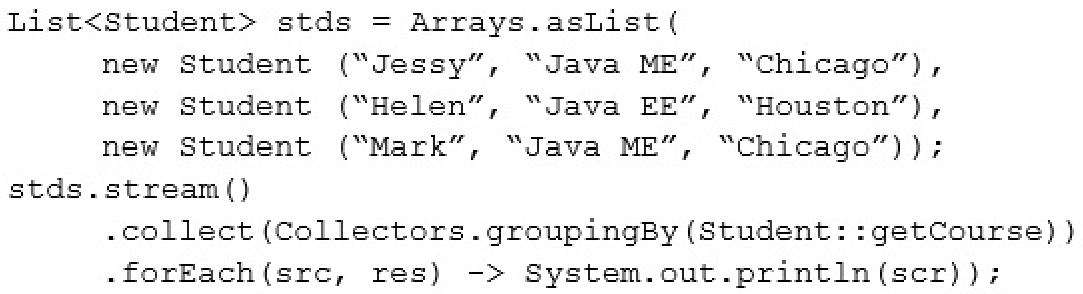
**Question Type:** **MultipleChoice**

Given:



```
class Student {
    String course, name, city;
    public Student(String name, String course, String city) {
        this.course = course; this.name = name; this.city = city;
    }
    public String toString() {
        return course + ":" + name + ":" + city;
    }
    public String getCourse() { return course; }
    public String getName() { return name; }
    public String getCity() { return city; }
}
```

and the code fragment:



```
List<Student> stds = Arrays.asList(
     new Student (“Jessy”, “Java ME”, “Chicago”),
     new Student (“Helen”, “Java EE”, “Houston”),
     new Student (“Mark”, “Java ME”, “Chicago”));
stds.stream()
     .collect(Collectors.groupingBy(Student::getCourse))
     .forEach(src, res) -> System.out.println(scr));
```

What is the result?

## Options:

**A-** [Java EE: Helen:Houston][Java ME: Jessy:Chicago, Java ME: Mark:Chicago]

**B-** Java EEJava ME

**C-** [Java ME: Jessy:Chicago, Java ME: Mark:Chicago][Java EE: Helen:Houston]

**D-** A compilation error occurs.

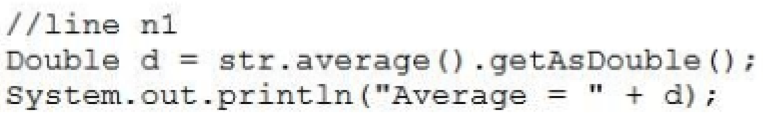
### Answer:

D

## Question 7

**Question Type:** MultipleChoice

Given the code fragment:



```
//line n1
Double d = str.average().getAsDouble();
System.out.println("Average = " + d);
```

Which should be inserted into line n1 to print Average = 2.5?

### Options:

**A-** IntStream str = Stream.of (1, 2, 3, 4);

**B-** IntStream str = IntStream.of (1, 2, 3, 4);

**C-** DoubleStream str = Stream.of (1.0, 2.0, 3.0, 4.0);

**D-** Stream str = Stream.of (1, 2, 3, 4);

### Answer:

C

## Question 8

**Question Type:** MultipleChoice

Which two statements are true about synchronization and locks? (Choose two.)

### Options:

**A-** A thread automatically acquires the intrinsic lock on a synchronized statement when executed.

**B-** The intrinsic lock will be retained by a thread if return from a synchronized method is caused by an uncaught exception.

**C-** A thread exclusively owns the intrinsic lock of an object between the time it acquires the lock and the time it releases it.

**D-** A thread automatically acquires the intrinsic lock on a synchronized method's object when entering that method.

**E-** Threads cannot acquire intrinsic locks on classes.

## Answer:

A, B

20%
DISCOUNT