Lambdas introduced in Java 8 allow us to treat -

a. Data as code  
b. Code as data  
c. none  
d. all

b

Lambda expressions in java 8 are based on

1. Procedural programming
2. Functional programming
3. Data programming
4. All

b

The Java 8 API with a sequence of elements which of these supports sequential and parallel aggregate operations -

1. Hadoop
2. Streams
3. SequenceProgramming
4. Big-data

b

Stream operations in java 8 can be divided into  
a. Terminal types  
b. Intermediate types  
c. All  
d. None

c

Which package contains Date/Time (JSR 310) API in Java 8 -

a. java.time  
b. java.util.time  
c. java.timedate  
d. java.util.calendar

a

Which of these represents a process that accepts one argument and produces a result in Java 8

1. Function
2. Process
3. Method
4. JavaFunctions

a

PermGen space has been replaced with which of these in Java 8 -

1. PermSpace
2. PermSpaceGen
3. Metaspace
4. MetaGenSpace

c

What can help us in avoiding NullPointeExceptions and null checks in java 8 -

a.Optional  
b.Required  
c.NotNull  
d.NotRequired

a

In java 8, R apply(T t) is a method of-

a.Function  
b.Process  
c.Predicate  
d.None

a

What is Predicate in Java 8 -  
a.method  
b.class  
c.Interface  
d.Framework

c

How sorting speed has been improved significantly on multi-core machines by using -

a.Arrays.parallelSort  
b.Arrays.sort  
c.Collection.parallelSort  
d.Arrays.sortParallelly

a

Which is aggregate operation in Java 8

a.filter  
b.map   
c.forEach  
d.All

d

Example of functional interfaces in Java 8 -

a.java.util.concurrent.Callable  
b.java.lang.Runnable  
c.All  
d.None

c

We need to override which Predicate method in Java 8 -

a.predict(T t)  
b.predictable(T t)  
c.testable(T t)  
d.test(T t)

d

What needs to be implemented to use lambda expression

a.Functional interface  
b.Functional class  
c.Functional method  
d.Functional object

a

void accept(T t) is method of -

a.Consumer  
b.Producer  
c.Both  
d.None

a

Which of these does Stream filter() operates on

a.Predicate  
b.Interface  
c.Class  
d.Methods

a

Which of these does Stream map() operates on

a.Class  
b.Interface  
c.Predicate  
d.Function

d

Which of these does forEach() operates on

a.Methods  
b.Consumer  
c.Producer  
d.Predicate

b

What is Optional object used for ?  
a) Optional is used for optional runtime argument  
b) Optional is used for optional spring profile  
c) Optional is used to represent null with absent value  
d) Optional means it’s not mandatory for method to return object

c

Which type of exception you would usually prefer while creating custom exceptions in your application

1. Checked Exception
2. Unchecked Exception

B

In java 8 parallel streams are almost always performs better then sequential stream.

1. True
2. False

B

An interface with only one abstract method is always a functional interface whether it is annotated with a @FunctionalInterface annotation or not.

1. True
2. False

A

The following types of methods in an interface do not count for defining a functional interface:

• Default methods

• Static methods

• Public methods inherited from the Object class

• All of above

D

An interface may have more than one abstract method, and can still be a functional interface if all but one of them is a re-declaration of the methods in the Object class.

T/F

T

A stream implementation may throw below exception if it detects that the stream is being reused.

IllegalStateException

StreamIndexOutOfBoundException

IllegalOperationException

NullPointerException

IllegalStateException

Which of these statements compile? (Choose all that apply.)

A. HashSet<Number> hs = new HashSet<Integer>();

B. HashSet<? super ClassCastException> set = new HashSet<Exception>();

C. List<String> list = new Vector<String>();

D. List<Object> values = new HashSet<Object>();

E. List<Object> objects = new ArrayList<? extends Object>();

F. Map<String, ? extends Number> hm = new HashMap<String, Integer>();

B,C,F

What is the result of the following code?

1: public class Hello<T> {

2: T t;

3: public Hello(T t) { this.t = t; }

4: public String toString() { return t.toString(); }

5: public static void main(String[] args) {

6: System.out.print(new Hello<String>("hi"));

7: System.out.print(new Hello("there"));

8: } }

A. hi

B. hi followed by a runtime exception

C. hithere

D. Compiler error on line 4

E. Compiler error on line 6

F. Compiler error on line 7

C

What is the result of the following program?

public class MyComparator implements Comparator<String> {

public int compare(String a, String b) {

return b.toLowerCase().compareTo(a.toLowerCase());

}

public static void main(String[] args) { String[] values = { "123", "Abb", "aab" }; Arrays.sort(values, new MyComparator()); for (String s: values)

System.out.print(s + " "); }

}

A. Abb aab 123

B. aab Abb 123

C. 123 Abb aab

D. 123 aab Abb

E. The code does not compile.

F. A runtime exception is thrown.

What code change is needed to make the method compile?

public static T identity(T t) { return t;

}

A. Add <T> after the public keyword.

B. Add <T> after the static keyword.

C. Add <T> after T.

D. Add <?> after the public keyword.

E. Add <?> after the static keyword.

F. No change required. The code already compiles.

Which are true statements about terminal operations in a stream? (Choose all that apply.)

A. At most one terminal operation can exist in a stream pipeline.

B. Terminal operations are a required part of the stream pipeline in order to get a result.

C. Terminal operations have Stream as the return type.

D. The referenced Stream may be used after the calling a terminal operation.

E. The peek() method is an example of a terminal operation.

A and B

Which terminal operations on the Stream class are reductions? (Choose all that apply.)

A. collect()

B. count()

C. findFirst()

D. map()

E. peek()

F. sum()

**A, B.**

We have a method that returns a sorted list without changing the original. Which of the following can replace the method implementation to do the same with streams?

private static List<String> sort(List<String> list) {

List<String> copy = new ArrayList<>(list);

Collections.sort(copy, (a, b) -> b.compareTo(a));

return copy;

}

A. return list.stream()

.compare((a, b) -> b.compareTo(a)) .collect(Collectors.toList());

B. return list.stream()

.compare((a, b) -> b.compareTo(a)) .sort();

C. return list.stream()

.compareTo((a, b) -> b.compareTo(a)) .collect(Collectors.toList());

D. return list.stream()

.compareTo((a, b) -> b.compareTo(a)) .sort();

E. return list.stream()

.sorted((a, b) -> b.compareTo(a)) .collect();

F. return list.stream()

.sorted((a, b) -> b.compareTo(a)) .collect(Collectors.toList());

F

Which of the following are true given the declaration IntStream is = IntStream. empty()? (Choose all that apply.)

A. is.average() returns the type int.

B. is.average() returns the type OptionalInt.

C. is.findAny() returns the type int.

D. is.findAny() returns the type OptionalInt.

E. is.sum() returns the type int.

F. is.sum() returns the type OptionalInt.

DE

Which of the following are true statements? (Choose all that apply.)

A. A traditional try statement without a catch block requires a finally block.

B. A traditional try statement without a finally block requires a catch block.

C. A traditional try statement with only one statement can omit the {}.

D. A try-with-resources statement without a catch block requires a finally block.

E. A try-with-resources statement without a finally block requires a catch block.

F. A try-with-resources statement with only one statement can omit the {}.

A,B

Given an instance of a Stream, s, and a Collection, c, which are valid ways of creating a parallel stream? (Choose all that apply.)

A. new ParallelStream(s) B. c.parallel()

C. s.parallelStream()

D. c.parallelStream()

E. new ParallelStream(c) F. s.parallel()

D,F

Which happens when more tasks are submitted to a thread executor than available threads?

A. The thread executor will throw an exception when a task is submitted that is over its

thread limit.

B. The task will be added to an internal queue and completed when there is an available thread.

C. The thread executor will discard any task over its thread limit.

D. The call to submit the task to the thread executor will wait until there is a thread avail-

able before continuing.

E. The thread executor creates new temporary threads to complete the additional tasks.

B

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

Integer i1 = Arrays.asList(1,2,3,4,5).stream().findAny().get(); synchronized(i1) { // y1

Integer i2 = Arrays.asList(6,7,8,9,10) .parallelStream()

.sorted() // y2

.findAny().get(); // y3

System.out.println(i1+" "+i2); }

A. It outputs 1 6.

B. It outputs 1 10.

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

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

E. The code will not compile because of line y3.

F. It compiles but throws an exception at runtime.

G. The output cannot be determined ahead of time.

H. It compiles but waits forever at runtime.

G

Which of the following properties of concurrency are true? (Choose all that apply.)

A. By itself, concurrency does not guarantee which task will be completed first.

B. Concurrency always improves the performance of an application.

C. Computers with a single processor do not benefit from concurrency.

D. Applications with many resource-heavy tasks tend to benefit more from concurrency than ones with CPU-intensive tasks.

E. Concurrent tasks do not share the same memory.

A,D

Why does Console.readPassword() return a char[] array instead of a String object? (Choose all that apply.)

A. It improves performance.

B. It is more secure.

C. To encrypt the password data.

D. To support all character encodings.

E. Because Java puts all String values in a reusable pool.

F. So that the value can be removed from memory immediately after use.

B, E, F.

Spring

In Spring @ControllerAdvice is used for:

A create Rest controller

B exception handling

C Auditing

D Logging

B

Which one is default Embedded Container in Spring Boot?

1. Tomcat
2. Jetty
3. Undertow
4. None

A

The @SpringBootApplication annotation is equivalent to using

1. @Configuration
2. @EnableAutoConfiguration
3. @Configuration and @EnableAutoConfiguration
4. @Configuration, @EnableAutoConfiguration and @ComponentScan

D

By default, Spring Boot configures logging via

1. Logback
2. Log4j
3. slf4j
4. log4j2

A.

In Spring Boot 2.x, default connection pool is:

1. HikariCP
2. Tomcat connection pool
3. [Commons DBCP2](https://commons.apache.org/proper/commons-dbcp/)
4. SpringDB Pool

a

Spring Boot supports distributed JTA transactions across multiple XA resources by using :

1. **Atomikos**
2. **Bitronix**
3. **Narayana**.
4. ALL

d

Service Registration and Discovery is implemented in Spring Cloud by using:  
Hystrix

Eureka

Zuul

Turbine

Zipkin

Ans Eureka

Circuit Breaker and fault tolerance is implemented in Spring Cloud by using:  
Hystrix

Eureka

Zuul

Turbine

Zipkin

Ans Hystrix

in Spring Cloud Zuul is:

API gateway

Service Registration and Discovery

Circuit Breaker

Config Server

Ans API gateway

In Spring Cloud Ribbon is used for:

Load balancing

Circuit Breaker and fault tolerance

API gateway

Logging and tracing

Ans Load balancing