Questions B

1. **Functional interfaces can be annotated as**

a.@Function
b.@FunctionalInterface
c.@Functional
d.@Interface

2. **Functional interfaces method is**

a.static
b.abstract
c.final
d.none

3. **What needs to be implemented to use lambda expression**

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

4. **Functional interface methods can be declared as**

a.static
b.abstract
c.final
d.All

5. **What will be output of following code -**

```
private interface Defaultable {
    default String notRequired() {
        return "Default implementation";
    }
}

private static class DefaultableImpl implements Defaultable {
}

private static class OverridableImpl implements Defaultable {
    @Override
    public String notRequired() {
        return "Overridden implementation";
    }
}
```

a.compilation error
b.Runtime Exception
c.Compile and execute without any Exception or error
d.None

6. **void accept(T t) is method of -**

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

7. **Which of these does Stream filter() operates on**

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

8. **Which of these does Stream map() operates on**

a.Class
b.Interface

c.Predicate
d.Function

9. Which of these does forEach() operates on

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

10. A pipeline is a sequence of what operations in java 8

a.multi-threading
b.concurrent
c.consequent
d.stream

11. How can we obtain source of objects in java 8?

a.Stream stream()
b.Stream obtain()
c.Stream obtainSource()
d.All

12. What will be output of following program -

```
1  package javaqas.java8;
2
3  import java.time.Clock;
4
5  public class DateTimeTest {
6
7      public static void main(String[] args) {
8          // Get the system clock as UTC offset
9          final Clock clock = Clock.systemUTC();
10         System.out.println(clock.instant());
11         System.out.println(clock.millis());
12     }
13 }
```

a.compilation error
b.Runtime Exception
c.Compile and execute without any Exception or error
d.None

13. What will be output of following program -

```java
package javaqas.java8;

import java.time.Clock;
import java.time.ZoneId;
import java.time.ZonedDateTime;

public class ZonedDateTimeTest {

    public static void main(String[] args) {

        final Clock clock = Clock.systemUTC();

        final ZonedDateTime zonedDatetime = ZonedDateTime.now();
        final ZonedDateTime zonedDatetimeFromClock = ZonedDateTime.now(clock);
        final ZonedDateTime zonedDatetimeFromZone = ZonedDateTime.now(ZoneId.of("America"));

        System.out.println(zonedDatetime);
        System.out.println(zonedDatetimeFromClock);
        System.out.println(zonedDatetimeFromZone);
    }
}
```

a.compilation error
b.Runtime Exception
c.Compile and execute without any Exception or error
d.None

14. What will be output of following code -

```java
package javaqas.java8;

public class Calculator {

    interface IntegerMath {
        int operation(int a, int b);
    }

    public int operateBinary(int a, int b, IntegerMath op) {
        return op.operation(a, b);
    }

    public static void main(String... args) {

        Calculator myApp = new Calculator();
        IntegerMath addition = (a, b) -> a + b;
        IntegerMath subtraction = (a, b) -> a - b;
        System.out.println("40 + 2 = " + myApp.operateBinary(40, 2, addition));
        System.out.println("20 - 10 = " + myApp.operateBinary(20, 10, subtraction));
    }
}
```

a.compilation error
b.Runtime Exception

c.Compile and execute without any Exception or error
d.None

15. What will be output of following code -

```
1  package javaqas.java8;
2
3  import java.lang.reflect.Method;
4  import java.lang.reflect.Parameter;
5
6  public class ParameterNames {
7
8      public static void main(String[] args) throws Exception {
9          Method method = ParameterNames.class.getMethod("main", String[].class);
10         for (final Parameter parameter : method.getParameters()) {
11             System.out.println("Parameter: " + parameter.getName());
12         }
13     }
14 }
```

a.compilation error
b.Runtime Exception
c.Compile and execute without any Exception or error
d.None

16. What will be output of following code -

```
1  package javaqas.java8;
2
3  public class Test {
4
5      public static void main(String[] args) {
6
7          final String str = "test";
8          str.chars().forEach(ch -> System.out.println(ch));
9      }
10 }
```

a.Strings
b.Integers
c.Characters
d.Float

17. Lambda expressions are _____ scoped

a.Lexically
b.Semantically
c.Binary
d.None

18. On which of these does annotations can be used on in Java 8

a.Local variables
b.Super classes
c.Generic types
d.All of these

19. What will be output of following code -

```
1  package javaqas.java8;
2
3  import java.lang.annotation.ElementType;
8
9  public class RepeatingAnnotations {
10
11     @Target(ElementType.TYPE)
12     @Retention(RetentionPolicy.RUNTIME)
13     public @interface Filters {
14         Filter[]value();
15     }
16
17     @Target(ElementType.TYPE)
18     @Retention(RetentionPolicy.RUNTIME)
19     @Repeatable(Filters.class)
20     public @interface Filter {
21         String value();
22     };
23
24     @Filter("filter1")
25     @Filter("filter2")
26     public interface Filterable {
27     }
28
29     public static void main(String[] args) {
30         for (Filter filter : Filterable.class.getAnnotationsByType(Filter.class)) {
31             System.out.println(filter.value());
32         }
33     }
34 }
```

a.compilation error
b.Runtime Exception
c.Compile and execute without any Exception or error
d.None

20. What will be output of following code -

```java
package javaqas.java8;

import java.lang.annotation.ElementType;

public class Annotations {
    @Retention(RetentionPolicy.RUNTIME)
    @Target({ ElementType.TYPE_USE, ElementType.TYPE_PARAMETER })
    public @interface NonEmpty {
    }

    public static class Holder<@NonEmpty T> extends @NonEmpty Object {
        public void method() throws @NonEmpty Exception {
        }
    }

    @SuppressWarnings("unused")
    public static void main(String[] args) {
        final Holder<String> holder = new @NonEmpty Holder<String>();
        @NonEmpty
        Collection<@NonEmpty String> strings = new ArrayList<>();
        strings.add("string1");
        strings.add("string2");
        for(String s: strings) {
            System.out.println(s);
        }
    }
}
```

a.compilation error
b.compilation error
c.Compile and execute without any Exception or error
d.None

21. What will be output of following code -

```java
import java.util.Random;
/** Copyright (c), AnkitMittal JavaMadeSoEasy.com */
public class outputProg_GenerateRandomIntegersInJava8_ {

    public static void main(String[] args) {

        Random rand = new Random();
        for (int i = 0; i < 3; i++)
            System.out.print(rand.ints(1, 5, 11).findFirst().getAsInt());

    }

}
```

a.output may be 125
b.output may be 695

c.output may be 1511
d.output may be 456

22. What will be output of following code -

```java
package javaqas.java8;

import java.util.Optional;

public class OptionalTest {

    public static void main(String[] args) {
        Optional<String> fullName = Optional.of("Tom");
        fullName = Optional.ofNullable(null);
        System.out.println("Full Name is set? " + fullName.isPresent());
        System.out.println("Full Name: " + fullName.orElseGet(() -> "[none]"));
        System.out.println(fullName.map(s -> "Hey " + s + "!").orElse("Hey Stranger!"));
    }
}
```

a.Full Name: [none]
b.
c.compilation error
d.compilation error

23. What will be output of following code -

```java
package javaqas.java8;

import java.time.Clock;
import java.time.LocalDate;
import java.time.LocalTime;

public class LocalDateTimeTest {

    public static void main(String[] args) {

        final Clock clock = Clock.systemUTC();

        final LocalDate date = LocalDate.now();
        final LocalDate dateFromClock = LocalDate.now(clock);

        System.out.println(date);
        System.out.println(dateFromClock);

        final LocalTime time = LocalTime.now();
        final LocalTime timeFromClock = LocalTime.now(clock);

        System.out.println(time);
        System.out.println(timeFromClock);
    }
}
```

a.compilation error
b.Runtime Exception
c.Compile and execute without any Exception or error
d.None

24. What will be output of following code -

```java
package javaqas.java8;

import java.time.Duration;
import java.time.LocalDateTime;
import java.time.Month;

public class DurationTest {

    public static void main(String[] args) {

        // Get duration between two dates
        final LocalDateTime from = LocalDateTime.of(2014, Month.APRIL, 16, 0, 0, 0);
        final LocalDateTime to = LocalDateTime.of(2015, Month.APRIL, 16, 23, 59, 59);

        final Duration duration = Duration.between(to, from);
        System.out.println("Duration in days: " + duration.toDays());
        System.out.println("Duration in hours: " + duration.toHours());
    }
}
```

a.compilation error
b.Runtime Exception
c.Compile and execute without any Exception or error
d.None

25. What will be output of following code -

```java
package javaqas.java8;

import javax.script.ScriptEngine;
import javax.script.ScriptEngineManager;

public class NashornTest {

    public static void main(String[] args) {
        ScriptEngineManager manager = new ScriptEngineManager();
        ScriptEngine engine = manager.getEngineByName("JavaScript");

        System.out.println(engine.getClass().getName());
        System.out.println("Result:" + engine.eval("function f() { return 1; }; f() + 1;"));
    }
}
```

a.compilation error
b.Runtime Exception
c.Compile and execute without any Exception or error

d.None