Advanced Java Class Design - 0

Given code:

```
class A {
    public void someMethod(final String name) {
        /*INSERT*/ {
            void print() {
                System.out.println("Hello " + name);
            }
        }
        new B().print();

    }
}

public class Test {
    public static void main(String[] args) {
        new A().someMethod("World!");
    }
}
```

Which of the following options can replace /*INSERT*/ such that on executing Test class, “Hello World!” is displayed in the output?

A - abstract class B

B - class B

C - public class B

D - final class B

E - protected class B

F - private class B

Advanced Java Class Design - 1

What is the purpose of below lambda expression?

```
(x, y) -> x + y;
```

A - It accepts two String arguments, concatenates them and returns the String instance

B - It accepts two int arguments, adds them and returns the int value

C - It accepts a String and an int arguments, concatenates them and returns the String instance

D - Not possible to define the purpose

Advanced Java Class Design - 2

What will be the result of compiling and executing Test class?

```
@FunctionalInterface
interface I5 {
    void print();
}

public class Test {
    int i = 100;

    I5 obj1 = new I5() {
        int i = 200;
        public void print() {
            System.out.println(this.i);
        }
    };

    I5 obj2 = () -> {
        int i = 300;
        System.out.println(this.i);
    };

    public static void main(String[] args) {
        Test ques = new Test();
        ques.obj1.print();
        ques.obj2.print();
    }
}
```

A -

```
100
100
```

B -

```
200
100
```

C -

```
200
300
```

D -

```
100
300
```

Advanced Java Class Design - 3

Below is the code of Test.java file:

```
class Outer {
    abstract static class Animal { //Line 2
        abstract void eat();
    }

    static class Dog extends Animal { //Line 6
        void eat() { //Line 7
            System.out.println("Dog eats biscuits");
        }
    }
}

public class Test {
    public static void main(String[] args) {
        Outer.Animal animal = new Outer.Dog(); //Line 15
        animal.eat();
    }
}
```

What will be the result of compiling and executing Test class?

A - Compilation error at Line 2

B - Compilation error at Line 6

C - Compilation error at Line 7

D - Dog eats biscuits

E - Compilation error at Line 15

Advanced Java Class Design - 4

Below is the code of Test.java file:

```
interface Flyable {
    void fly();
}

public class Test {
    public static void main(String[] args) {
        /*INSERT*/
    }
}
```

Which of the following options can replace /*INSERT*/ such that there are no compilation errors?

A -

```
Flyable flyable = new Flyable() {
  public void fly() {
    System.out.println("Flying high");
  }
};
```

B -

```
Flyable flyable = new Flyable() {
  public void fly() {
    System.out.println("Flying high");
  }
}
```

C - Flyable flyable = new Flyable();

D - Flyable flyable = new Flyable(){};

Advanced Java Class Design - 5

Below is the code to Test.java file:

```
enum ShapeType {
    CIRCLE, SQUARE, RECTANGLE;
}

abstract class Shape {
    private ShapeType type = ShapeType.SQUARE; //default ShapeType

    Shape(ShapeType type) {
        this.type = type;
    }

    public ShapeType getType() {
        return type;
    }

    abstract void draw();
}

public class Test {
    public static void main(String[] args) {
        Shape shape = new Shape() {
            @Override
            void draw() {
                System.out.println("Drawing a " + getType());
            }
        };
        shape.draw();
    }
}
```

What will be the result of compiling and executing Test class?

A - Drawing a CIRCLE

B - Compilation error

C - Drawing a SQUARE

D - Drawing a RECTANGLE

Advanced Java Class Design - 6

Given code of Test.java file:

```
interface Printer1 {
    default void print() {
        System.out.println("Printer1");
    }
}

class Printer2 {
    public void print() {
        System.out.println("Printer2");
    }
}

class Printer extends Printer2 implements Printer1 {

}

public class Test {
    public static void main(String[] args) {
        Printer printer = new Printer();
        printer.print();
    }
}
```

What will be the result of compiling and executing Test class?

A - Compilation error for Printer

B - Compilation error for Printer2

C - Compilation error for Printer1

D - Printer1

E - Printer2

Advanced Java Class Design - 7

Given code:

```
class Outer {
    private String name = "James Gosling";
    //Insert inner class definition here
}

public class Test {
    public static void main(String [] args) {
        new Outer().new Inner().printName();
    }
}
```

Which of the following Inner class definition inserted in the Outer class, will print 'James Gosling' in the output on executing Test class?

A -

```
inner class Inner {
  public void printName() {
    System.out.println(name);
  }
}
```

B -

```
class Inner {
  public void printName() {
    System.out.println(name);
  }
}
```

C -

```
class Inner {
  public void printName() {
    System.out.println(this.name);
  }
}
```

D -

```
abstract class Inner {
  public void printName() {
    System.out.println(name);
  }
}
```

Advanced Java Class Design - 8

Consider the code of Test.java file:

```
enum Flags {
    TRUE, FALSE;

    Flags() {
        System.out.println("HELLO");
    }
}

public class Test {
    public static void main(String[] args) {
        Flags flags = new Flags();
    }
}
```

What will be the result of compiling and executing Test class?

A - None of the other options

B - HELLO is printed twice

C - Exception is thrown at runtime

D - HELLO is printed once

Advanced Java Class Design - 9

What will be the result of compiling and executing class M?

```
class M {
    private int num1 = 100;
    class N {
        private int num2 = 200;
    }

    public static void main(String[] args) {
        M outer = new M();
        M.N inner = outer.new N();
        System.out.println(outer.num1 + inner.num2);
    }
}
```

A - 200

B - 300

C - Compilation error

D - 100

Advanced Java Class Design - 10

Which of the following statement is correct about java enums?

A - An enum can extend another class

B - All java enums implicitly extend from java.util.Enum class

C - An enum can extend another enum

D - An enum can implement interfaces

Advanced Java Class Design - 11

What will be the result of compiling and executing Test class?

```
class Outer {
    public void print(int x) {
        class Inner {
            public void getX() {
                System.out.println(++x);
            }
        }
        Inner inner = new Inner();
        inner.getX();
    }
}

public class Test {
    public static void main(String[] args) {
        new Outer().print(100);
    }
}
```

A - 100

B - Runtime exception

C - 101

D - Compilation error

Advanced Java Class Design - 12

Given code of Test.java file:

```
interface Printer1 {
    default void print() {
        System.out.println("Printer1");
    }
}

interface Printer2 {
    default void print() {
        System.out.println("Printer2");
    }
}

class Printer implements Printer1, Printer2 {

}

public class Test {
    public static void main(String[] args) {
        Printer printer = new Printer();
        printer.print();
    }
}
```

What will be the result of compiling and executing Test class?

A - Printer1

B - Printer2

C - Compilation error for Printer1

D - Compilation error for Printer

E - Compilation error for Printer2

Advanced Java Class Design - 13

For the given code:

```
interface Operator {
    int operate(int i, int j);
}

public class Test {
    public static void main(String[] args) {
        Operator opr = new Operator() {
            public int operate(int i, int j) {
                return i + j;
            }
        };
        System.out.println(opr.operate(10, 20));
    }
}
```

Which of the following options successfully replace anonymous inner class code with lambda expression code?

A - Operator opr = (x, y) -> x + y;

B - Operator opr = (int x, int y) -> { return x + y; };

C - Operator opr = (x, y) -> { return x + y; };

D - Operator opr = (x, y) -> return x + y;

E - Operator opr = x, y -> x + y;

Advanced Java Class Design - 14

Below is the code of Test.java file:

```
public class Test {
    public static void main(String [] args) {
        System.out.println(new Object() {
            public String toString() {
                return "Anonymous";
            }
        });
    }
}
```

What will be the result of compiling and executing Test class?

A - Anonymous

B - Compilation error

C - Some text containing @ symbol

D - Runtime exception

Advanced Java Class Design - 15

Given code:

```
abstract class Greetings {
    abstract void greet();
}

public class Test {
    public static void main(String[] args) {
        Greetings obj = new Greetings() {
            @Override
            public void greet() {
                System.out.println("Hello");
            }
        };
        obj.greet();
    }
}
```

What will be the result of compiling and executing Test class?

A - Hello

B - Compilation error

C - NullPointerException

D - Nothing is printed on to the console

Advanced Java Class Design - 16

What will be the result of compiling and executing Test class?

```
public class Test {
    enum TrafficLight {
        private String message;
        GREEN("go"), AMBER("slow"), RED("stop");

        TrafficLight(String message) {
            this.message = message;
        }

        public String getMessage() {
            return message;
        }
    }

    public static void main(String[] args) {

         System.out.println(TrafficLight.AMBER.getMessage().toUpperCase());
    }
}
```

A - SLOW

B - slow

C - Compilation error

D - NullPointerException is thrown at runtime

Advanced Java Class Design - 17

What will be the result of compiling and executing Test class?

```
interface I9 {
    void print();
}

public class Test {
    public static void main(String[] args) {
        int i = 400;
        I9 obj = () -> System.out.println(i);
        obj.print();
        System.out.println(++i);
    }
}
```

A - Compilation error

B -

```
400
401
```

C - Exception is thrown at runtime

D -

```
400
400
```

Advanced Java Class Design - 18

Consider the code of Test.java file:

```
enum Flags {
    TRUE, FALSE;

    Flags() {
        System.out.println("HELLO");
    }
}

public class Test {
    public static void main(String[] args) {
        Flags flags = Flags.TRUE;
    }
}
```

What will be the result of compiling and executing Test class?

A - HELLO is printed twice

B - HELLO is printed once

C - Exception is thrown at runtime

D - None of the other options

Advanced Java Class Design - 19

What will be the result of compiling and executing class Test?

```
class X {
    class Y {
        private void m() {
            System.out.println("INNER");
        }
    }

    public void invokeInner() {
        Y obj = new Y(); //Line 9
        obj.m(); //Line 10
    }
}

public class Test {
    public static void main(String[] args) {
        new X().invokeInner();
    }
}
```

A - Compilation error at Line 10 as private method m() cannot be invoked outside the body of inner class (Y)

B - Exception is thrown at runtime

C - INNER

D - Compilation error at Line 9 as instance of outer class (X) is needed to create the instance of inner class (Y)

Advanced Java Class Design - 20

Below is the code of Test.java file:

```
class Outer {
    static class Inner {
        static void greetings(String s) {
            System.out.println(s);
        }
    }
}

public class Test {
    public static void main(String[] args) {
        /*INSERT*/
    }
}
```

Which of the following 2 options can replace /*INSERT*/ such that there on executing class Test, output is: HELLO!?

A - Inner.greetings(“HELLO!”);

B - Outer.Inner.greetings(“HELLO!”);

C -

```
Outer.Inner inner2 = new Outer.Inner();
inner2.greetings("HELLO! ");
```

D -

```
Outer.Inner inner1 = new Outer().new Inner();
inner1.greetings( HELLO! ");
```

Advanced Java Class Design - 21

Given code:

```
public class Test {
    class A {
        void m() {
            System.out.println("INNER");
        }
    }

    public static void main(String [] args) {
        //Insert statement here
    }
}
```

Which statement when inserted in the main(String []) method will print “INNER” in the output?

A -

```
Test.A a2 = new Test().new A();
a2.m();
```

B -

```
Test.A a4 = this.new A();
a4.m();
```

C -

```
A a3 = this.new A();
a3.m();
```

D -

```
A a1 = new Test().new A();
a1.m();
```

Advanced Java Class Design - 22

Will below code compile successfully?

```
interface I1 {
    void m1();

    interface I2 {
        void m2();
    }

    abstract class A1 {
        public abstract void m3();
    }

    class A2 {
        public void m4() {
            System.out.println(4);
        }
    }
}
```

A - Yes

B - No

Advanced Java Class Design - 23

What will be the result of compiling and executing Test class?

```
class P {
    private int var = 100;
    class Q {
        String var = "Java";
        void print() {
            System.out.println(var);
        }
    }
}

public class Test {
    public static void main(String[] args) {
        new P().new Q().print();
    }
}
```

A - 100

B - Exception is thrown at runtime

C - Java

D - Compilation error

Advanced Java Class Design - 24

What will be the result of compiling and executing Test class?

```
public class Test {
    enum JobStatus {
        SUCCESS, FAIL; //Line 3
    }

    enum TestResult {
        PASS, FAIL; //Line 7
    }

    public static void main(String[] args) {
        JobStatus js = JobStatus.FAIL;
        TestResult tr = TestResult.FAIL;

        System.out.println(js.equals(tr)); //Line 14
        System.out.println(js == tr); //Line 15
    }
}
```

A - Compilation error at Line 15

B -

```
false
false
```

C - Compilation error at Line 14

D -

```
true
true
```

Advanced Java Class Design - 25

Given code:

```
class Message {
    public void printMessage() {
        System.out.println("Hello!");
    }
}

public class Test {
    public static void main(String[] args) {
        Message msg = new Message() {
            @Override
            public void PrintMessage() {
                System.out.println("HELLO!");
            }
        };
        msg.printMessage();
    }
}
```

What will be the result of compiling and executing Test class?

A - Compilation error

B - Runtime error

C - HELLO!

D - Hello!

Advanced Java Class Design - 26

Given code of Test.java file:

```
interface Printer1 {
    default void print() {
        System.out.println("Printer1");
    }
}

interface Printer2 {
    public static void print() {
        System.out.println("Printer2");
    }
}

class Printer implements Printer1, Printer2 {

}

public class Test {
    public static void main(String[] args) {
        Printer printer = new Printer();
        printer.print();
    }
}
```

What will be the result of compiling and executing Test class?

A - Compilation error for Printer2

B - Printer1

C - Compilation error for Printer1

D - Printer2

E - Compilation error for Printer

Advanced Java Class Design - 27

Which of the annotation is used for Functional Interface?

A - @FunctionalInterface

B - @Functional

C - @Functional Interface

D - @FI

Advanced Java Class Design - 28

Given code:

```
class Outer {
    class Inner {
        public void m() {
            System.out.println("WELCOME!");
        }
    }
}

public class Test {
    public static void main(String[] args) {
        //Insert statement here
    }
}
```

Which statement when inserted in the main(String []) method will print “WELCOME!” in the output?

A -

```
Inner obj2 = new Outer().new Inner();
obj2.m();
```

B -

```
Outer.Inner obj1 = new Outer().new Inner();
obj1.m();
```

C -

```
Inner obj4 = this.new Inner();
obj4.m();
```

D -

```
Outer.Inner obj3 = this.new Inner();
obj3.m();
```

Advanced Java Class Design - 29

Below is the code of TestSellable.java file:

```
interface Sellable {
    double getPrice();
}

public class TestSellable {
    private static void printPrice(Sellable sellable) {
        System.out.println(sellable.getPrice());
    }

    public static void main(String[] args) {
        /*INSERT*/
    }
}
```

Which of the following options can replace /*INSERT*/ such that there are no compilation errors?

A -

```
printPrice(new Sellable() {

});
```

B -

```
printPrice(new Sellable() {
  @0verride
  public double getPrice() {
    return 45.34;
  }
});
```

C - printPrice(null);

D - printPrice(new Sellable());

Advanced Java Class Design - 30

Given code of Test.java file:

```
interface I10 {
    void m(String s);
}

public class Test {
    public static void main(String[] args) {
        method(new I10() {
            @Override
            public void m(String s) {
                System.out.println(s.toUpperCase());
            }

        }, "good morning!");
    }

    private static void method(I10 obj, String text) {
        obj.m(text);
    }
}
```

Which of the following code replaces the anonymous inner class code with lambda expression?

A - method(s -> s.toUpperCase(), “good morning!”);

B - method(s -> { System.out.printtn(s.toUpperCase()) }, “good morning!”);

C - method(s -> System.out.println(s.toUpperCase()), “good morning!”);

D - method(s -> System.out.println(s.toUpperCase()));

Advanced Java Class Design - 31

Below is the code of Test.java file:

```
public class Test {
    enum TrafficLight {
        RED, YELLOW, GREEN;
    }

    public static void main(String[] args) {
        TrafficLight tl = TrafficLight.valueOf(args[1]);
        switch(tl) {
            case TrafficLight.RED:
                System.out.println("STOP");
                break;
            case TrafficLight.YELLOW:
                System.out.println("SLOW");
                break;
            case TrafficLight.GREEN:
                System.out.println("GO");
                break;
        }
    }
}
```

What will be the result of compiling and executing Test class by using the commands:

javac Test.java java Test RED AMBER

A - IllegalArgumentException is thrown

B - STOP

C - No output

D - None of the other options

Advanced Java Class Design - 32

Given Code:

```
class Outer {
    public static void sayHello() {}
    static {
        class Inner {
            /*INSERT*/
        }
        new Inner();
    }
}

public class TestOuter {
    public static void main(String[] args) {
        Outer.sayHello();
    }
}
```

Which of the following options can replace /*INSERT*/ such that on executing TestOuter class, “HELLO” is printed in the output?

A -

```
static {
  System.out.println( HELLO”);
}
```

B -

```
{
  System.out.println( HELLO”);
}
```

C -

```
Inner(String s) {
  Systen.out.println(s);
}
```

D -

```
Inner() {
  System.out.println( HELLO”);
}
```

Advanced Java Class Design - 33

What will be the result of compiling and executing Test class?

```
class Foo {
    static { //static initialization block
        System.out.print(1);
    }
    class Bar {
        static { //static initialization block
            System.out.print(2);
        }
    }
}

public class Test {
    public static void main(String [] args) {
        new Foo().new Bar();
    }
}
```

A - Exception is thrown at runtime

B - 21

C - 12

D - Compilation error

Advanced Java Class Design - 34

Given:

```
package com.training.ocp;

enum TrafficLight {
    RED, YELLOW, GREEN;
}

public class Test {
    public static void main(String[] args) {
        TrafficLight tl1 = TrafficLight.GREEN;
        TrafficLight tl2 = tl1.clone(); //Line 10
        System.out.println(tl2); //Line 11
    }
}
```

What will be the result of compiling and executing Test class?

A - Compilation error at Line 10

B - Line 11 throws CloneNotSupportedException at runtime

C - GREEN

Advanced Java Class Design - 35

Given code:

```
class Message {
    public void printMessage() {
        System.out.println("Hello!");
    }
}

public class Test {
    public static void main(String[] args) {
        Message msg = new Message() {
            public void PrintMessage() {
                System.out.println("HELLO!");
            }
        };
        msg.PrintMessage();
    }
}
```

What will be the result of compiling and executing Test class?

A - HELLO!

B - Hello!

C - Compilation error

D - Runtime error

Advanced Java Class Design - 36

For the given code:

```
abstract class Greetings {
    abstract void greet(String s);
}

public class Test {
    public static void main(String[] args) {
        Greetings obj = new Greetings() {
            public void greet(String s) {
                System.out.println(s);
            }
        };
        obj.greet("Happy New Year!");
    }
}
```

Which of the following options successfully replace anonymous inner class code with lambda expression code?

A - Greetings obj = (String s) -> {System.out.printin(s.toUpperCase());};

B - Lambda expression cannot be used in this case

C - Greetings obj = s -> {System.out.println(s.toUpperCase());};

D - Greetings obj = s -> System.out.printin(s.toUpperCase());

Advanced Java Class Design - 37

Below is the code of Test.java file:

```
class Outer {
    private static int i = 10;
    private int j = 20;

    static class Inner {
        void add() {
            System.out.println(i + j);
        }
    }
}

public class Test {
    public static void main(String[] args) {
        Outer.Inner inner = new Outer.Inner();
        inner.add();
    }
}
```

What will be the result of compiling and executing Test class?

A - Compilation error in Test class code

B - Compilation error in Inner class code

C - Exception is thrown at runtime

D - 30

Advanced Java Class Design - 38

Given code of Test.java file:

```
interface Printer {
    default void print() {
        System.out.println("Printer");
    }
}

@FunctionalInterface
interface ThreeDPrinter extends Printer {
    @Override
    void print();
}

public class Test {
    public static void main(String[] args) {
        ThreeDPrinter p = () -> System.out.println("3DPrinter");
        p.print();
    }
}
```

What will be the result of compiling and executing Test class?

A - Printer

B - Compilation error in ThreeDPrinter class

C - 3DPrinter

D - Compilation error in Test class

Advanced Java Class Design - 39

Consider the code of Test.java file:

```
enum Flags {
    TRUE, FALSE;

    public Flags() {
        System.out.println("HELLO");
    }
}

public class Test {
    public static void main(String[] args) {
        Flags flags = Flags.TRUE;
    }
}
```

What will be the result of compiling and executing Test class?

A - HELLO is printed once

B - Compilation error

C - HELLO is printed twice

D - Exception is thrown at runtime

Advanced Java Class Design - 40

Given code:

```
class A {
    private String str = "Hello";
    public class B {
        public B(String s) {
            if(s != null)
                str = s;
        }
        public void m1() {
            System.out.println(str);
        }
    }
}

public class Test {
    public static void main(String[] args) {
        //Insert statement here
    }
}
```

Which statement when inserted in the main(String []) method will print “Hello” in the output?

A - new A.B().m1();

B - new A().new B(“hello”).m1();

C - new A().new B(null).m1();

D - new A().new B().m1();

Advanced Java Class Design - 41

Can an anonymous inner class implement multiple interfaces?

A - Yes

B - No

Advanced Java Class Design - 42

Consider below interface:

```
interface I2 {
    int calc(int x);
}
```

Which of the following is the correct lambda expression for I2?

A - I2 obj4 = x -> x*x;

B - I2 obj2 = (x) -> return x*x;

C - I2 obji = x -> return x*x;

D - I2 obj3 = x - > x*x;

Advanced Java Class Design - 43

Given code:

```
class A {
    public void print(String name) {
        class B {
            B() {
                System.out.println(name); //Line 5
            }
        }
    }
    B obj = new B(); //Line 9
}

public class Test {
    public static void main(String[] args) {
        new A().print("OCP"); //Line 14
    }
}
```

What will be the result of compiling and executing Test class?

A - Compilation error at Line 14

B - Compilation error at Line 5

C - Compilation error at Line 9

D - OCP

Advanced Java Class Design - 44

Is below functional interface correctly defined?

```
@FunctionalInterface
interface I8 {
    boolean equals(Object obj);
}
```

A - Yes

B - No

Advanced Java Class Design - 45

What will be the result of compiling and executing Test class?

```
class A {
    A() {
        System.out.print(1);
    }
    class B {
        B() {
            System.out.print(2);
        }
    }
}

public class Test {
    public static void main(String [] args) {
        B obj = new A().new B();
    }
}
```

A - 21

B - Compilation error

C - 2

D - 12

Advanced Java Class Design - 46

Will below code compile successfully?

```
class Outer {
    interface I1 {
        void m1();
    }
}
```

A - No

B - Yes

Advanced Java Class Design - 47

Below is the code of Test.java file:

```
class A {
    static class B {

    }
}

public class Test {
    /*INSERT*/
}
```

Which of the following options can replace /*INSERT*/ such that there are no compilation errors?

A - A.B obj = new A.B();

B - B obj = new B();

C - A.B obj = new A().new B();

D - B obj = new A.B();

Advanced Java Class Design - 48

For the given code:

```
interface Printable {
    void print(String msg);
}

public class Test {
    public static void main(String[] args) {
        Printable obj = new Printable() {
            public void print(String msg) {
                System.out.println(msg);
            }
        };
        obj.print("Welcome!");
    }
}
```

Which of the following options successfully replace anonymous inner class code with lambda expression code?

Select ALL that apply.

A - Printable obj = (String msg) -> {System.out.println(msg);};

B - Printable obj = x -> System.out.printtln(x);

C - Printable obj = (msg) -> System.out.println(msg);

D - Printable obj = y - > System.out.printtin(y);

E - Printable obj = msg -> System.out.println(msg);

F - Printable obj = (msg) -> {System.out.println(msg);};

Advanced Java Class Design - 49

Below is the code of Test.java file:

```
public class Test {
    enum TrafficLight {
        RED, YELLOW, GREEN;
    }

    public static void main(String[] args) {
        TrafficLight tl = TrafficLight.valueOf(args[0]);
        switch(tl) {
            case RED:
                System.out.println("STOP");
                break;
            case YELLOW:
                System.out.println("SLOW");
                break;
            case GREEN:
                System.out.println("GO");
                break;
        }
    }
}
```

What will be the output if Test class is executed by the commands: javac Test.java java Test GREEN AMBER

A - NullPointerException is thrown at runtime

B - Compilation error

C - IllegalArgumentException is thrown at runtime

D - GO

Advanced Java Class Design - 50

Consider the code of Test.java file:

```
enum Flags {
    TRUE;

    Flags() {
        System.out.println("HELLO");
    }
}

public class Test {
    public static void main(String[] args) {
        Flags f1 = Flags.TRUE;
        Flags f2 = Flags.TRUE;
        Flags f3 = Flags.TRUE;
    }
}
```

What will be the result of compiling and executing Test class?

A - HELLO is printed three times

B - HELLO is printed twice

C - HELLO Is not printed on to the console

D - HELLO is printed once

Advanced Java Class Design - 51

Given code:

```
class Outer {
    Outer() {
        System.out.print(2);
    }
    /*INSERT 1*/

    class Inner {
        Inner() {
            System.out.print(4);
        }
        /*INSERT 2*/
    }
}

public class Test {
    public static void main(String[] args) {
        new Outer().new Inner();
    }
}
```

Currently on executing Test class, 24 is printed in the output.

Which of the following pairs will correctly replace /*INSERT 1*/ and /*INSERT 2*/ so that on executing Test class, 1234 is printed in the output?

A -

```
Replace /*INSERT 1*/ with static {System.out.print(1);}
Replace /*INSERT 2*/ with static {System.out.print(3);}
```

B -

```
Replace /*INSERT 1*/ with {System.out.print(1);}
Replace /*INSERT 2*/ with static {System.out.print(3);}
```

C -

```
Replace /*INSERT 1*/ with {System.out.print(1);}
Replace /*INSERT 2*/ with {System.out.print(3);}
```

D -

```
Replace /*INSERT 1*/ with static {System.out.print(1);}
Replace /*INSERT 2*/ with {System.out.print(3);}
```

Advanced Java Class Design - 52

What will be the result of compiling and executing Test class?

```
interface Operation {
    int operate(int x, int y);
}

public class Test {
    public static void main(String[] args) {
        int x = 10;
        int y = 20;
        Operation o1 = (x, y) -> x * y;
        System.out.println(o1.operate(5, 10));
    }
}
```

A - Exception is thrown at runtime

B - 50

C - Compilation error

D - 200

Advanced Java Class Design - 53

Given code:

```
class Outer {
    private String msg = "A";
    public void print() {
        final String msg = "B";
        class Inner {
            public void print() {
                System.out.println(this.msg);
            }
        }
        Inner obj = new Inner();
        obj.print();
    }
}

public class Test {
    public static void main(String[] args) {
        new Outer().print();
    }
}
```

What will be the result of compiling and executing Test class?

A - Exception is thrown at runtime

B - Compilation error

C - B

D - A

Advanced Java Class Design - 54

Given code:

```
class Message {
    public void printMessage() {
        System.out.println("Hello!");
    }
}

public class Test {
    public static void main(String[] args) {
        Message msg = new Message() {
            public void PrintMessage() {
                System.out.println("HELLO!");
            }
        };
        msg.printMessage();
    }
}
```

What will be the result of compiling and executing Test class?

A - Hello!

B - Compilation error

C - Runtime error

D - HELLO!

Advanced Java Class Design - 55

What will be the result of compiling and executing Test class?

```
@FunctionalInterface
interface I7 {
    void print();
}

public class Test {
    String var = "Lambda";
    class Inner {
        int var = 1000;
        I7 obj = () -> System.out.println(this.var);
    }

    public static void main(String[] args) {
        Inner inner = new Test().new Inner();
        inner.obj.print();
    }
}
```

A - Lambda

B - 1000

C - Compilation Error

D - None of the other options

Advanced Java Class Design - 56

What will be the result of compiling and executing Test class?

```
@FunctionalInterface
interface I4 {
    void print();
    boolean equals(Object obj);
}

public class Test {
    public static void main(String[] args) {
        I4 obj = () -> System.out.println("Lambda expression");
        obj.print();
    }
}
```

A - Lambda expression

B - Compilation error

C - No output

D - Runtime error

Advanced Java Class Design - 57

Which of the following operator is used in lambda expressions?

A - ->

B - = >

C - - >

D - =>

Advanced Java Class Design - 58

What will be the result of compiling and executing Test class?

```
interface I1 {
    void print();
}

public class Test {
    public static void main(String[] args) {
        I1 obj = () -> System.out.println("Hello");
    }
}
```

A - Program compiles and executes successfully but nothing is printed on to the console

B - Compilation error

C - Hello

D - Runtime error

Advanced Java Class Design - 59

What will be the result of compiling and executing Test class?

```
interface I6 {
    void m6();
}

public class Test {
    public static void main(String[] args) {
        I6 obj = () ->  {
            int i = 10;
            i++;
            System.out.println(i);
        };
        obj.m6();
    }
}
```

A - Compilation error

B - Exception is thrown at runtime

C - 11

D - 10

Advanced Java Class Design - 60

Given code:

```
interface I1 {
    void m1();
}

public class Test {
    public static void main(String[] args) {
        I1 i1 = new I1() {
            @Override
            public void m1() {
                System.out.println(1234);
            }
        }
        i1.m1();
    }
}
```

What will be the result of compiling and executing Test class?

A - 1234

B - No output

C - Compilation error

D - Runtime exception

Advanced Java Class Design - 61

Which of the following are Functional Interface in JDK 8?

A - java.lang.Cloneable

B - java.lang.Runnable

C - java.awt.event.ActionListener

D - java.io. Serializable

E - java.util.Comparator

Advanced Java Class Design - 62

What will be the result of compiling and executing Test class?

```
enum Status {
    PASS, FAIL, PASS;
}

public class Test {
    public static void main(String[] args) {
        System.out.println(Status.FAIL);
    }
}
```

A - Fail

B - FAIL

C - None of the other options

D - fail

Advanced Java Class Design - 63

Consider the code of Test.java file:

```
public class Test {
    enum Directions {
        NORTH("N"), SOUTH("S"), EAST("E"), WEST("W")

        private String notation;

        Directions(String notation) {
            this.notation = notation;
        }

        public String getNotation() {
            return notation;
        }
    }

    public static void main(String[] args) {
        System.out.println(Test.Directions.NORTH.getNotation());
    }
}
```

What will be the result of compiling and executing Test class?

A - Exception is thrown at runtime

B - NORTH

C - Compilation error

D - N

Advanced Java Class Design - 64

What will be the result of compiling and executing Test class?

```
class Foo {
    public static void m1() {
        System.out.println("Foo : m1()");
    }
    class Bar {
        public static void m1() {
            System.out.println("Bar : m1()");
        }
    }
}

public class Test {
    public static void main(String [] args) {
        Foo foo = new Foo();
        Foo.Bar bar = foo.new Bar();
        bar.m1();
    }
}
```

A - Compilation error

B - Runtime exception

C - Bar : m1()

D - Foo : m1()

Advanced Java Class Design - 65

Below the code of A.java file:

```
public class A {
    private static class B {
        private void log() {
            System.out.println("static nested class");
        }
    }

    public static void main(String[] args) {
        /*INSERT*/
    }
}
```

Which of the following options can replace /*INSERT*/ such that there on executing class A, output is: static nested class?

A -

```
A.B obj2 = new A.B();
obj2.log();
```

B -

```
B obj3 = new A().new BC);
obj3.log();
```

C -

```
A.B obj4 = new A().new BC);
obj4.1og();
```

D -

```
B obj1 = new B();
obj1.log();
```

Advanced Java Class Design - 66

Does below code compile successfully?

```
@FunctionalInterface
interface I1 {
    void print();
    boolean equals();
}
```

A - Yes

B - No

Advanced Java Class Design - 67

Given code:

```
class Message {
    public void printMessage() {
        System.out.println("Hello!");
    }
}

public class Test {
    public static void main(String[] args) {
        Message msg = new Message() {}; //Line 9
        msg.printMessage(); //Line 10
    }
}
```

What will be the result of compiling and executing Test class?

A - NullPointerException is thrown by Line 10

B - HELLO!

C - Compilation error at Line 9

D - Hello!

Advanced Java Class Design - 68

What will be the result of compiling and executing Test class?

```
interface Formatter {
    public abstract String format(String s1, String s2);
}

public class Test {
    public static void main(String[] args) {
        Formatter f1 = (str1, str2) -> str1 + "_" +
         str2.toUpperCase();
        System.out.println(f1.format("Training", "Ocp"));
    }
}
```

A - TRAINING_Ocp

B - Training_OCP

C - Training_Ocp

D - TRAINING_OCP