Java Quizmaster for Beginners

LEARN JAVA IN 17 DAYS AND MASTER JAVA CODE BY SOLVING 105 QUIZZES AND 117 ASSIGNMENTS

Sar Maroof

ISBN: 978-1-975-78178-1

Table of contents

About the author

Sar Maroof is graduated from HBO Amsterdam "higher professional education" when he already had a Bachelor of Science degree in Physics. He is a SUN certified JSP as well as EJB. He has experience with Java since 2001 and worked for several big as well as small companies and later as a freelancer.
The combination of his experiences and skills as a teacher and as a Java web developer inspired him to share his knowledge with enthusiastic younger generations through writing books. He handles his own method of teaching programming by focusing on a practical way to learn it.

About this book

This book is organized to learn Java in 17 days, and it guides you to master Java code by solving 105 quizzes and 117 assignments. It has already been published both in English and Dutch.

Any prior background in coding does not require to start with this book. It explains Java in an easy way with simple examples and many exercises. That makes it ideal for beginners.
If you have already experience with Java or other programming languages, this book helps you to enrich your experience by solving many quizzes and executing assignments.

Read below the explanation of how this book is organized to learn standard Java step by step in 17 days..

1. This book contains 17 chapters, and each chapter covers a Java topic that starts with a simple explanation and examples.

2. The next step allows you to solve the quizzes regarding each specific chapter. For each quiz, there is a step by step explanation of the answer..

3. By each quiz, there are one or more assignments. You will be asked to change the code or add your own code to the quiz to achieve a specific goal..

4. It is your time from chapter 5 to write your own Java code. You will be asked to execute a certain assignment and write code from scratch regarding each chapter.

5. You can download the source code of this book at www.sarmaroof.com/book-en-java-eclipse-set-up-code. There is also a step by step explanation of how to set up the code in Eclipse. For more information visit the website of the author: www.SarMaroof.com.

Sar Maroof

Required knowledge

This book focuses on learning how to program by practicing with code. There will be little theoretical explanation – just enough to solve the quizzes. That is a challenge because it is not easy to write a tiny executable program about only one Java topic. In every executable program, you see different topics. All the programs demonstrate a particular Java topic. This book divides the Java programming language into some subjects. In each chapter, a number of small executable programs is offered regards that specific topic. To make this possible, it is important to be familiar with the following basics. It's not required that you understand all the points below properly because details will be covered later in this book.

The only thing you need to learn to start with is what the role of these points is for the program to compile and run. Each chapter of this book begins with a brief explanation, after which the quizzes follow. The codes are complete that you can test, compile, and run.

1. Java editor

We use as Java editor, Eclipse including JDK 7 (Java Development Kit) or higher. Eclipse is a free Java IDE (integrated development environment). You can find and download Eclipse on the Internet. The JDK tools are included, which is necessary when editing Java-programs. The source codes of Java are files with the extension .java. The compiled bytecode files have the extension .class. Java is a platform-independent programming language, which is why you can run Java programs on every operating system.

2. Compiling programs

Compiling is the translation of source code into machine language with the help of a tool (compiler). After compiling a source code, the program is directly executable.

3. Java classes and interfaces

Java programs contain classes and interfaces. These two concepts are covered in details later in this book. You only need for now is to learn that a class in Java begins with a statement as class Myclass. Myclass is the name of the class, and you can decide the name of the class. Every class in Java is stored in a file with the name of the class. The class Myclass should be stored in a file with the name of the class and the extension .java. The name of the file in our example is thus MyClass.java. Each class name

begins with the keyword class, and each interface name begins with the keyword interface. Classes and interfaces also have members like variables and methods. Methods are a block code between curly braces.

Example 1

```
class MyClass
{
  // code
}
```

The following interface must also be stored in a file with the name MyInterface.java.

Example 2

```
interface MyInterface
{
  // code
}
```

4. Statements

Statements in Java are similar to sentences in natural languages, and they are executable units. A statement is usually ended with a semicolon (;).

5. Code block in Java

Code in Java is within a start and an end brace. This is called block of code. Below are some examples of block codes.

Block type	Block style

Class	class MyClass
	{
	  // code
	}

Method	void myMethod
	{
	  // code

	} The keyword void means that the method doesn't return any values.

Conditional	if(x > 3)
	{
	// code
	}

Iteration	for(int i=0; i<5; i ++)
	{
	// code
	}

6. The main method

The main method is a method that is needed to execute a Java program. The main method begins as follows:

public static void main(String [] args)

For the moment it is important to learn the following about the main method. All executable programs must have a main method such as below. The statements within the main method are executed. The execution of the statements is done from top to bottom. In the next example statement 1 is first executed, then statement 2 and at last statement 3. The following code writes 253 to the standard output.

Example 3

```
public class MyClass
{
  // statements;
  // main method
  public static void main(String[] args)
  {
    System.out.print(2); // statement 1;
    System.out.print(5); // statement 2;
    System.out.print(3); // statement 3;
  }
}
```

7. Writing values of variables and texts to the standard output

The statement System.out.println(); is used to test your program. It is not necessary for now to understand all the details about this statement, but it is important to know what you can do with this. You can use this statement to write values of variables and texts to the standard output as shown in example 4.

Example 4

```
public class MyClass
{
  public static void main(String[] args)
  {
    int x1 = 25; // Variable x1 is an integer
    int x2 = 99; // Variable x2 is an integer
    // statement 1 writes the value of x1 to the standard output
    System.out.println(x1);
    // statement 2 writes the value of x2 to the standard output
    System.out.println(x2);
    // statement 3 writes the text "My name is Emma." to the standard output
    System.out.println("My name is Emma.");
    /*
     *  statement 4 writes a combination of a text and
     *  a variable to the standard output
     */
    System.out.println("Age: " + x1 + " year");
  }
}
```

If you compile and run the code of example 4, the following will be written to the standard output:

```
25
99
My name is Emma.
Age: 25 year
```

To write a text to the standard output, it must be between quotation marks, but for the value of variables that is not required, see statement 1, 2 and 3 in the previous example.

To write a combination of texts and variables to the standard output, you need to use the plus (+) sign, see statement 4. To write the texts and the variables on one line to the standard output you need to use **print** instead of **println.**

8. Comments

Comments are ignored by the compiler. Below are two ways to add comments to your code.

1. Comment of a single line begins with two slashes //.
   Everything on the right side of these characters, see Java as comments:

   ```
   // comment of a single-line
   ```

2. Comment of multiple lines starts with /* and ends with */.
   Everything between these two characters sees Java as comments:

   ```
   /* here is a comment
         of multiple lines */
   ```

9. Keywords

There are important keywords in Java that we use in the quiz codes such as static and public. We use those keywords from the beginning because they help to write small programs. The above-mentioned keywords will be explained in details later in this book.

The Keyword static

The concept of static is important in Java, and it is treated in a separate chapter. The only thing you need to learn for now is that this keyword helps to write small executable programs. That is why we use the keyword static before the name of some of the variables in the quiz codes. The following example makes this idea clear.

Example 5

What happens when the following program is compiled and run?

```
public class MyClass
{
  // static keyword before the variable name
  static int x = 4;
  // the main method to execute the program
  public static void main(String[] args)
  {
    // statement 1: writes the value of the variable x
    System.out.print(x);
    // statement 2: writes the text "My Java code"
    System.out.print(" My Java code");
  }
}
```

The previous program writes **4 My Java code** to the standard output. If you remove the keyword static in the previous example, you get the following error message:

Cannot make a static reference to the non-static field x.

In order to solve this problem, we need to create an object. Creating objects is explained in chapter 5. Therefore, we declared the variable x static. This little trick helps to avoid create objects in the beginning. That helps to create a small executable program, and we focus on the subject that matter. The first statement in the main method writes the value of the variable x to the standard output. The second statement writes the text **My Java code** to the standard output. Don't forget that we used here print instead of println, therefor the values of the variable and the text are written on one line to the standard output.

The Keyword public

Java supports controlling access to the classes and class members by using special keywords. Using the **public** keyword for the name of the class or the class members (variables and methods) indicates that the class or the members are accessible from other classes. This will be explained later in details.

10. Java standard API (Application Programming Interface)

Java provides a lot of code that can be reused by programmers. It is important for every Java programmers to use this free rich library, which is why it is introduced in this book. For some of the assignments, you need to open this document which helps how to use the code. You can find Java standard API here:

https://docs.oracle.com/javase/8/docs/api/

11. Escape sequences

An escape sequence is a character preceded by a backslash. These characters have a special meaning for the compiler. For example, if you have a double quote within a double quote, you must use escape sequences. See the following example. If you need to write the text "He says: "I go to London."" to the standard output, you should do that as follows.

System.out.print("He says: \"I go to Amsterdam\".");

Example 6

```
public class MyClass
{
  public static void main(String[] args)
  {
    {
      System.out.println("Apostrophe: " + "abcde\'fghij");
      System.out.println("Double quotation mark   : " + "abcde\"fghij");
      System.out.println("Backslash: " + "abcde\\fghij");
      System.out.println("New line: " + "abcde\nfghij");
      System.out.println("New line 2: " + "abcde\ffghij");
      System.out.println("Tab : " + "abcde\tfghij");
      System.out.print("It was written \"Parking is not Allowed.\".");
    }
  }
}
```

If this code is compiled and run, the following is written to the standard output.

```
Apostrophe              : abcde'fghij
Double quotation mark   : abcde"fghij
Backslash               : abcde\fghij
New line                : abcde
fghij
New line 2              : abcdefghij
Tab                     : abcde     fghij
It was written "Parking is not Allowed.".
```

The table of the escape sequences:

Escape sequence	Description
\'	single quote
\”	double quote
\\	backslash character
\b	backspace
\n	newline
\r	carriage return
\t	tab
\f	formfeed

Object Oriented Programming (OOP)

Java is an object oriented programming language. The basic idea of the **object-oriented programming** is creating a number of objects that communicate with each other. Each object contains variables, and they communicate thorough methods. The important principles of the object-oriented programming are inheritance, polymorphism, and Encapsulation.

1. **Inheritance:** Inheritance is an important principle of object oriented programming. Java supports inheritance to allow reusing code and extending new classes based on exiting ones. Read chapter 11 for more details about inheritance.
2. **Polymorphism:** It is the ability to use one method name to invoke many different methods. There are two types of polymorphism.
    a. *Overriding or Dynamic Polymorphism:* For method overriding the choice is determined at runtime.
    b. *Overloading or Static Polymorphism:* For method overloading the choice is determined at compile time. Read chapter 11 for more details about Overriding methods and overloading methods.
3. **Encapsulation:** It is the ability of hiding data implementation by restricting access to public accessor and mutator methods. Accessors are used to get information about the state of an object, while mutators allow you to modify the state of an object.

Chapter 1—Data Types & Variables

There are 8 primitive data types in Java, which can be declared by programmers. Those types are: **byte, short, int, long, float, double, char and boolean,** which are divided into 4 categories as shown below.

Data Type (bits)	**Range, Description and Examples**
	Integer Type
byte (8 bits)	-2^7 to $2^7 - 1$, -128 to 127 The byte type is small and can be used to save memory. Its default value is 0. **Example: byte b = 20;**
short (16 bits)	-2^{15} to $2^{15} - 1$ The short type can also be used to save memory. Its default value is 0. **Example: short s = 500;**
int (32 bits)	-2^{31} to $2^{31} - 1$ The int type can be used for bigger values. Its default value is 0. **Example: int i = 2500;**
long (64 bits)	-2^{63} to $2^{63} - 1$ The long type can be used when you need a range of values wider than those of int. Its default value is 0. **Example: long l = 23333333333;**

Floating-point Type	
float (32)	~ -3.4 x 10^{38} to ~ 3.4 x 10^{38} The float type can be used when floating-point types and values are needed. **Example: float f = 1.4f**
double (64)	~ -1.8 x 10^{308} to ~1.8 x 10^{308} The double type can be used when floating-point types and values are needed. double is a default choice for decimal values. **Example: double d = 22.3;**
Character Type	
char (16)	0 to 65,535 The char type can be used by character types like a, b, c, d, $ Characters of type char are always inside single quotes. For the type char you can use a unicode character as 'B' or as a hexadecimal number of '\u0000' to '\uFFFF'. Examples: '\u03A9' = Ω '\u0045' = E '\u20AC' = € **Example: char letter = 'd';**
Boolean Type	
boolean (1)	The boolean type has two possible values either **true** or **false**. It is false by default. **Example: boolean bool = true;**

String	A string is used for the texts in Java, it is not a primary variable, but it is an object. This book covers string in a separate chapter. Texts of the type string are between double quotes. **Example: String text = "My name is John";**

Data Type	Default Value
byte, short, int, long	0
float, double	0.0
char	'\u0000' (the null character)
boolean	false
non-primitive data types (object)	null Objects are explained later in this book.

Declaring and initializing variables

1. Variable type (always required)
2. Identifier (always required)
3. Initial value (not always required)

Examples

```
double price = 44.00;
int height;
```

The variable height has the default value of 0, because it is not initialized.

There are three types of variables in Java, namely local variables, instance variables and class variables.

1. *Instance variables:* An instance variable is declared within a class, but outside of the methods, constructors or other blocks.
2. *Local variables:* A local variable is a variable that is declared within a method, constructor or a block.
3. *Class variables:* Class variables are also called static variables. They are declared once inside a class but outside the methods, constructors or blocks. There is only one copy of the class variable is available.

Example

```
public class MyClass
{
  double wage; // instance variable
  static int counter; // class variable
```

```
  void myMethod()
  {
    char gender = 'm'; // local variable
  }
}
```

Quiz 1: Primitive data types and variables

What happens when the following program is compiled and run?

```
public class Worker
{
  static boolean isMarried;

  public static void main(String[] args)
  {
    int age = 29;
    long bankAccount = 6552;
    double wage = 110.30;
    char gender = 'm'; // female: f, male: m
    System.out.print(age + ", ");
    System.out.print(bankAccount + ", ");
    System.out.print(wage + ", ");
    System.out.print(isMarried + ", ");
    System.out.print(gender);
  }
}
```

Select the correct answer:

a. This code writes "29, 6552, 110.3, false, m" to the standard output.
b. This code writes "29, 6552, 110.3, true, m" to the standard output.

Explanation

All the values of the variables are printed to the standard output.
Since boolean variable "isMarried" is not initialized, its value is by default false.

The correct answer is a.

Assignments

1. Declare a variable with the name **isForeigner** to know which workers are foreigners.
2. We assume that the most workers are foreigners.
3. Add a statement to the program to write the value of the variable **isForeigner** to the standard output.
4. Change the position of your previous statement in the code to see what happens.
5. Try to assign the new values 45, 298888888, 124.89, to the variables age, bank account, and wages. What is written to the standard output if the program is compiled and run?

Quiz 2: Primitive data types and variables

What happens when the following program is compiled and run?

```
public class MyVariable
{
  static byte b = 80;
  static short s;
  static float f1 = 3.50f;
  static float f2;
  static double d;

  public static void main(String[] args)
  {
    System.out.print(b + ", ");
    System.out.print(s + ", ");
    System.out.print(f1 + ", ");
    System.out.print(f2 + ", ");
    System.out.print(d);
  }
}
```

Select the correct answer:

a. This code writes "80, 0, 3.5, 0.0, 0.0" to the standard output.
b. This code writes "80, 0, 3.5, 0, 0" to the standard output.

c. This code writes "80, 0, 3.5, 0.0, 0" to the standard output.

Explanation

The default value of integers is "0", but the default value of floats and doubles are "0.0".

The correct answer is a.

Assignments

1. Assign the new values 122, 43.9f, 335.35 to the variables b, f2, d, and execute the program to see what happens.
2. Declare a character type variable called "myChar".
3. Assign the value "Q" to the variable "myChar".
4. Add a statement to the code to print the value of myChar to the standard output.
5. Change the position of your statement in the code to see what happens.

Quiz 3: Primitive data types default values

What happens when the following program is compiled and run?

```
public class MyClass
{
  static int i;
  static double d;
  static boolean b;

  public static void main(String[] args)
  {
    System.out.print(i + ", ");
    System.out.print(d + ", ");
    System.out.print(b);
  }
}
```

Select the correct answer:

a. This code writes "0, 0, false" to the standard output.

b. This code writes "0, 0.0, false" to the standard output.

Explanation

The correct answer is b, because the default value of double is "0.0".

The correct answer is b.

Assignments

1. Declare a variable called "myVar", and assign the value 1344.98 to it.
2. Declare a variable called "myVar2" directly under myVar, and assign the value "g" to it.
3. Declare a variable called "myVar3" directly under myVar2, and assign the value 766 to it.
4. Add three statements to the the program to write the values of myVar, myVar2 and myVar3 to the standard output.

Quiz 4: Assigning values to variables

What happens when the following program is compiled and run?

```
public class MyClass
{
  static int i1 = 7;
  static int i2 = 12;

  public static void main(String[] args)
  {
    i1 = i1 - 3;
    i2 = i2 + i1;
    System.out.print(i1 + ", ");
    System.out.print(i2 + " ");
  }
}
```

Select the correct answer:

a. This code writes "7, 12" to the standard output.

b. This code writes "4, 19" to the standard output.
c. This code writes "4, 16" to the standard output.
d. This code writes "7, 19" to the standard output.
e. This program does not compile.

Explanation

i1 = i1 - 3 = 7- 3 = 4. the new value of i1 is 4, and that is why we use this value in the second equation, i2 = i2 + i1 = 12 + 4 = 16.

The correct answer is c.

Assignments

1. Add the statement "i1 = 9;" directly under the statement "public static void main(String[] args)".
2. Add the statement "i2 = 8;" directly under the previous statement.
3. What is written to the standard output if you compile and run the program?

Chapter 2—Operators

Operators are special symbols that are used to operate on one, two or three operands and return a result. An example is:

age >= 40

In the previous example, age and 40 are operands of >=.

Java supports different types of operators such as: **arithmetic**, **relational**, **conditional, assignment, unary** and **comparison**.

Arithmetic operators

Arithmetic operators are used by working with numbers and mathematical expressions.

Arithmetic operators	**Description**
+ (Addition)	3 + 6 returns 9
- (subtraction)	8 – 6 returns 2
* (Multiplication)	4 * 5 returns 20
/ (Division)	20/4 returns 5
% (Modulus)	Divides left operand by right operand and returns remainder 17 % 5 returns 2, because 17/5 = 3 and the rest is 17 – (5 * 3) = 2

Relational operators

Relational operators are used by evaluating two operands for equality. The answer is either true or false.

Relational Operators	Description
== Equal	Checks the value of two operands. If they are equal returns true else returns false. int x = 5, int y = 6; (x == y) returns false.
!= Not equal	Checks the value of two operands. If they are not equal returns true else returns false. int x = 5, int y = 6; (x != y) returns true.
> Greater than	If the left operand value greater than the right one returns true else returns false. (8 > 5) returns true.
< Less than	If the left operand value is smaller than the right one returns true else returns false. (8 < 5) returns false.
>= Greater or equal	If the left operand value is greater or equal to the right one, returns true else returns false. (7 >= 7) returns true.
<= Less than or equal	If the left operand value is smaller or equal to the right one returns true else returns false. (6 <= 9) returns true

Conditional operators

Conditional operators are used by conditional statements. The conditional statement types that Java supports are AND (&&) and OR(| |)

Conditional operators	Description

&& AND	&& combines two boolean variables and returns true only if both of its operands are true.
	if (3 > 2 && 4 < 6) returns true
	if(6 > 3 && 3 < 2) returns false

| | OR	| | combines two boolean variables and returns

	true if one or both of its operands are true. If(1 > 2 | | 6 < 13) returns true

? :	Ternary operator
	Shorthand formulas
	if-then-else statement
	int n = 6;
	int p = (n == 6) ? 4 :5;
	The above statement means the following.
	
	if(n == 6) {
	p = 4;
	}
	else {
	p = 5;
	}
	Returns 4, because n is equal to 6.

Assignment operators

Assignment operators	**Description**
= Assignment operator	Assigns values to variables. x = a - b assigns the value of a - b to x
+= Addition Assignment	x += 5 is equivalent to x = x + 5
-= Subtraction Assignment	x -= 4 is equivalent to x = x – 4
*= Multiplication Assignment	x *= 3 is equivalent to x = x * 3
/= Division Assignment	x /= 2 is equivalent to x = x/2
%= Modulus Assignment	x %= 2 is equivalent to x = x % 2 example 1 int x = 21; x % 4; is equivalent to x = x % 4 = 21 % 4 = 1. x = 21 % 4 = the rest of 21 divided by 4 = 1. example 2

	int x = 17; x % 3; means x = de rest van 17/3 = 2.

Unary opeators

Unary operators	Description
++ Increment	Increments a value by 1. int x = 20; ++ x returns 21
- - Decrement	Decrements a value by 1 int x = 20; - - x returns 19
!	! reverses the value of a boolean expression. boolean isDefected = false; ! isDefected returns true.

Type comparison operator

Comparison operator	Description
instanceof	Compares an object to a specified type. Objects are later explained in this book.

The if-block

The blocks of if-statements are only executed if the condition is true.

The following program writes XU to the standard output.

Example

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

```
    if (13 == 6)
    {
     System.out.print("N");
      /*
       * This Block is ignored. N is not written to
       * the standard output, because 13 is not equal to 6
       */
    }
    if (12 <= 22)
    {
     System.out.print("X");
      /*
       * Writes X to the standard output,
       * because 12 is less than 22
       */
    }
    if (21 > 8 && 3 != 5)
    {
     System.out.print("U");
      /*
       * Writes U to the standard output, because
       * 21 is greater than 8 and 3 is not equal to 5
       */
    }
  }
}
```

Quiz 1: Arithmetic operators

What happens when the following program is compiled and run?

```
public class Calculate
{
  public static void main(String[] args)
  {
    int x = 20;
    int y = 5;
    int z = 3;
    double d = 2.2;
```

```
    double d2 = 3.7;
    System.out.print(x / y + ", ");
    System.out.print(x * z + ", ");
    System.out.print(x + y - z + ", ");
    System.out.print(x / y + z * 2 + ", ");
    System.out.print(d2 - d);
  }
}
```

Select the correct answer:

a. This code writes "4, 60, 22, 10, 1.5" to the standard output.
b. This code writes "4, 60, 22, 14, 1.5 " to the standard output.

Explanation

This exercise is a simple calculation.

x/y = 20/5 = 4;
x*z = 20*3 = 60;
x+y-z = 20+5-3 = 22;
x/y + z*2 = 20/5 + 3*2 = 4 + 6 = 10;
d2- d = 3.7-2.2 = 1.5;

The correct answer is a.

Assignments

What does each of the following three statements write to the standard output if you add them directly under the statement System.out.println(d2 - d);

```
System.out.print(x * y / 10 + ", ");
System.out.print(2 * d2 + 2.5 + ", ");
System.out.print(z * 3 - 6);
```

Quiz 2: Modulus

What happens when the following program is compiled and run?

```
public class MyClass
{
 public static void main(String[] args)
 {
  System.out.print(21 % 7 + ", ");
  System.out.print(12 % 5 + ", ");
  System.out.print(23 % 6);
 }
}
```

Select the correct answer:
a. This code writes "3, 2, 3" to the standard output.
b. This code writes "0, 2, 5" to the standard output.
c. This code writes "3, 2, 5" to the standard output.

Explanation

Modulus % calculates the remainder of the left-hand operand divided by the right one.

21/7 = 3. The remainder is 21 - (3 * 7) = 0.
12/5 = 2. The remainder is 12 - (5 * 2) = 12 -10 = 2.
23 % 6 = 3. The remainder is 23 - (6 * 3) = 23 -18 = 5

The correct answer is b.

Assignments

What does each of the following three statements write to the standard output if you add them directly under the statement System.out.println(23 % 6);.

```
System.out.print(44 % 10 + ", ");
System.out.print(7 % 2 + ", ");
System.out.print(30 % 3);
```

Quiz 3: Increments & decrements

What happens when the following program is compiled and run?

```
public class MyClass
```

```
{
  public static void main(String[] args)
  {
    int x = 4;
    int y = 6;
    x--;
    y++;
    System.out.print(x + ", " + y);
  }
}
```

Select the correct answer:

a. This code writes "4, 7" to the standard output.
b. This code writes "4, 6" to the standard output.
c. This code writes "3, 7" to the standard output.
d. This code writes "3, 6" to the standard output.

Explanation
x = 4 and y = 6.
x-- decrements the value of x by 1.
x = x - 1 = 4 -1 = 3;
y++ increments the value of y by 1.
y = y + 1 = 6 + 1 = 7

The correct answer is c.

Assignments

Research the following and execute the program to check out your expectation.

1. Add the statement x++; directly under the statement System.out.print(x + ", " + y);.
2. Change the position of the statement x++; directly above the statement System.out.print(x + ", " + y);.

Does the position of the statement x++; in de code make any difference?

Quiz 4: Relational operators

What happens when the following program is compiled and run?

```
public class MyClass
{
  public static void main(String[] args)
  {
    int x = 3;
    int y = 8;
    int z = 3;
    if (x == z)
    {
      System.out.print("N");
    }
    if (x >= y)
    {
      System.out.print("O");
    }
    if (x <= z)
    {
      System.out.print("P");
    }
    if (z > y)
    {
      System.out.print("Q");
    }
    if (y != z)
    {
      System.out.print("R");
    }
  }
}
```

Select the correct answer:

a. This code writes "NOPQR" to the standard output.
b. This code writes "NR" to the standard output.
c. This code writes "NPR" to the standard output.
d. This code writes nothing to the standard output.

Explanation

The conditional statement if(x == z) returns true, because both variables are equal to 3.
N will be printed to the standard output.
The conditional statement if(x >= y) returns false because x is not greater or equal to y.
The conditional statement if(x <= z) returns true, because x is equal to z and equal to 3.
The letter P is written to the standard output.
The conditional statement if(z > y) is false because z = 3, but y = 8.
The conditional statement if(y != z) is true because z doesn't equal to y.
The letter R is written to the standard output.

The correct answer is c.

Assignments

1. What is written to the standard output if you make the following changes?
2. Assign a new value 15 to the variable x and add the statement
3. System.out.print("Z"); directly under the statement System.out.print("O");
4. Execute the program to check out your expectation.

Quiz 5: Conditional operators

What happens when the following program is compiled and run?

```
public class MyClass
{
 public static void main(String[] args)
 {
  boolean isDefect = true;
  int x = 2;
  int y = 7;
  int z = 9;
  if (x < y && x > 1)
  {
   System.out.print("N");
  }
  if (z > y || x > y)
```

```
    {
      System.out.print("O");
    }
    if (!isDefect)
    {
      System.out.print("P");
    }
  }
}
```

Select the correct answer:

a. This code writes "NO" to the standard output.
b. This code writes "OP" to the standard output.
c. This code writes "NP" to the standard output.

Explanation

The condition if(x < y && x > 1) returns true, because both operands are true.
The condition if(z > y || x > y) returns true, because the operand z > y is true.
|| (conditional or) returns true if one or both operands are true.
The condition if(! isDefect) returns false, because isDetected is true and the sign ! reverses the value of the boolean.

The correct answer is a.

Assignments

What is written to the standard output if you make the following changes to the program?
Compile and run the code to check out your expectation.

1. Assign the value false to the variable isDefect.
2. Assign the value 1 to the variable x.

Quiz 6: Conditional operators

What happens when the following program is compiled and run?

```
public class MyClass
```

```
{
  public static void main(String[] args)
  {
    boolean isOld = false;
    int x = 5;
    int y = 14;
    int z = 18;
    if (y > x && z > y && (x + 12) >= z)
    {
      System.out.print("P");
    }
    if (x >= 6 || z <= y || z <= 18)
    {
      System.out.print("Q");
    }
    if (!isOld || y > z)
    {
      System.out.print("R");
    }
  }
}
```

Select the correct answer:

a. This code writes "PR" to the standard output.
b. This code writes "QR" to the standard output.
c. This code writes "PQR" to the standard output.
d. This code writes "PQ" to the standard output.

Explanation

The condition if(y > x && z > y && (x + 12) >= z) returns false, because one of the operands namely (x + 12) >= z returns false.
The condition if(x >= 4 || z <= y || z <= 18) returns true, because one of the operands return true namely z <= 18.
if(! isOld || y > z) returns true, because one of the operands namely !isOld is true.

The correct answer is b.

Assignments

What is written to the standard output if you make the following changes to the program?
Compile and run the code to check out your expectation.

1. Assign the value "true" to the variable isOld.
2. Assign the value "17" to the variable "z".

Quiz 7: Assignment operators

What happens when the following program is compiled and run?

```
public class MyClass
{
  public static void main(String[] args)
  {
    int i1 = 3;
    int i2 = 5;
    int i3 = 12;
    int i4 = 20;
    i1 += 4;
    i2 *= 3;
    i3 /= 3;
    i4 -= 12;
    System.out.print(i1 + ", ");
    System.out.print(i2 + ", ");
    System.out.print(i3 + ", ");
    System.out.print(i4 + " ");
  }
}
```

Select the correct answer:
a. This code writes "3, 5, 12, 20" to the standard output.
b. This code writes "4, 3, 3, 12" to the standard output.
c. This code writes "7, 15, 3, 8" to the standard output.
d. This code writes "7, 15, 4, 8 " to the standard output.

Explanation

i1 += 4 is equivalent to i1 = i1 + 4 = 3 + 4 = 7
i2 *= 3 is equivalent to i2 = i2 * 3 = 5 * 3 = 15
i3 /= 3 is equivalent to i3 = i3 / 3 = 12 / 3 = 4
i4 -= 12 is equivalent to i4 = i4 - 12 = 20 - 12 = 8

The correct answer is d.

Assignments

What is written to the standard output if you add the following statements to the program?
Compile and run the code to check out your expectation.
Add the following statements directly under the statement i4 -= 12;.

1. i1 ++ ;
2. i2 -= 3;
3. i3 *= 2;
4. i4 /= 4;

Quiz 8: Assignment operators

What happens when the following program is compiled and run?

```
public class MyClass
{
 public static void main(String[] args)
 {
  int i1 = 22;
  int i2 = 17;
  int i3 = 30;
  i1 %= 6;
  i2 %= 5;
  i3 %= 6;
  System.out.print(i1 + " ");
  System.out.print(i2 + " ");
  System.out.print(i3 + " ");
 }
}
```

Select the correct answer:
a. This code writes "3, 3, 5" to the standard output.
b. This code writes "4, 2, 0" to the standard output.
c. This code writes "3, 3, 1" to the standard output.
d. This code writes "2, 2, 0" to the standard output.

Explanation

22 %= 6. is equal to 4
17%= 5 is equal to 2
30%= 3 is equal to 0

The correct answer is b.

Assignments

What is written to the standard output if you add the following statements to the program?
Compile and run the program to check out your expectation.

Add the following statements directly under the statement i3 %= 6;.

1. i1 %= 3;
2. i2 %= 7;

Quiz 9: Ternary operator

What happens when the following program is compiled and run?

```
public class MyClass
{
 public static void main(String[] args)
 {
  int x = 3;
  int x2 = 8;
  int y = (x == 3) ? 24 : 8;
  int z = (x2 == 4) ? 33 : 21;
  System.out.print(y);
  System.out.print(z);
```

```
  }
}
```

Select the correct answer:

a. This code writes "833" to the standard output.
b. This code writes "2433" to the standard output.
c. This code writes "821" to the standard output.
d. This code writes "2421" to the standard output.

Explanation

The statement int y = (x == 3) ? 24: 8; is equivalent to:
if x is equal to 3, y is equal to 24;
if x is not equal to 3, y is equal to 8.
This returns 24, because x is equal to 3.
The statement int y = (x2 == 4) ? 33 : 21; returns 21, because x2 is not equal to 4.

The correct answer is d.

Assignments

Assign the value 6 to the variable x and 4 to the variable x2. Wat is written to the standard output if you compile and run the program?