Back

Module quiz: Getting started with Java
Graded Assignment • 30 min

English

Due Dec 21, 11:59 PM WIB

Your grade: 100%

Your latest: **100%** • Your highest: **100%** • To pass you need at least 80%. We keep your highest score.

Next item

1. If `total = 50`, what is the new value of total after `total += 10`?

1 / 1 point

- [✓ 60]
- 10
- 5010
- 40

Correct
That's correct. When you use the compound assignment operator **+=**, it adds the assignee's existing value to the value on the right and assigns the value back to the assignee. So, total `+=10` is the same as `total = total + 10` or `total = 50 + 10`, which is **60**.

2. Select the correct output for the following code:

1 / 1 point

```java
public class Main {
    public static void main(String[] args) {
        int num = 10;
        boolean result = num == 10;
        System.out.println(result);
    }
}
```

- 10
- [✓ true]
- false
- null

Correct
That's correct. The expression *num*`== 10` checks whether the value of num is equal to **10**. Since `num` is indeed **10**, the result is **true**.

3. Anwar is writing a program to manage user profiles. He needs to declare a boolean variable to indicate whether a user is a student. How would Anwar declare a boolean variable named `isStudent` and initialize it to **true**?

1 / 1 point

- `isStudent = true;`
- [✓ `boolean isStudent = true;`]
- `boolean isStudent = "true";`
- `bool isStudent = true;`

Correct
That's correct. When you want to declare and initialize the boolean variable in a single line, then boolean `isStudent = true` is correct.

4. Andrea is working on a text processing application and needs to extract a specific part of a String. What method would she use to extract the substring "Dark" from the String `sentence = "The Dark Web"`?

1 / 1 point

- `sentence.sub(4, 8)`
- `sentence.subString(4, 8)`
- [✓ `sentence.substring(4, 8)`]
- `sentence.slice(4, 8)`

Correct
That's correct. The String class has a method called `substring(int start, int end)`. The `substring()` method extracts characters from a String between two specified indices (positions) and returns the resulting substring.

5. Imagine you are writing a welcome message for a user in an application. Which code snippet would you use to combine the Strings `greeting = "Hello"` and `name = "Alice"` with a comma and a space in between?

1 / 1 point

- `greeting.concat(", ", name)`

- `greeting.concat(", " + name)`
- `greeting + name + ", "`
- [✓ `greeting + ", " + name`]

Correct
That's correct. The **+** operator concatenates greeting and name with `", "` in between, resulting in the output **"Hello, Alice"**. The other options either misuse the **concator** or incorrectly place the comma and space.

6. Tom is debugging his Java program and comes across a line of code that is causing an error. What is wrong with the following code snippet that Tom identifies? **1 / 1 point**

`int number = 10.5;`

- The value should be enclosed in double quotes.
- There's nothing wrong; the code is correct.
- The variable name is incorrect.
- [✓ The variable type int cannot hold decimal values.]

Correct
That's correct. The int data type is used for whole numbers (integers) and cannot store decimal values.

7. Which data type is used to store whole numbers in Java? **1 / 1 point**

- float
- [✓ int]
- double
- char

Correct
That's correct. Int can store only whole numbers. If you want to store decimal numbers, use the double data type.

8. The logical **NOT** operator is denoted by ________ in Java. **1 / 1 point**

- `-`
- [✓ `!`]
- `not`
- `~`

Correct
That's correct. **!** is the logical **NOT** operator. It inverts the boolean value: if the value is true, it becomes false, and if it is false, it becomes true.

9. Select the correct output for the following code: **1 / 1 point**

```
public class Main {
    public static void main(String[] args) {
        int remainder = 9 % 4;
        System.out.println(remainder);
    }
}
```

- 4
- 3
- 2
- [✓ 1]

Correct
That's correct. `9 % 4` will result in 1. The modulo (or remainder) operator **%** returns the remainder after dividing the first number by the second number. In this case, 9 divided by 4 is 2, with a remainder of 1.

10. True or False: Java Strings are immutable, meaning methods like `toUpperCase()` return a new String rather than modifying the original. **1 / 1 point**

- False

False

[✓ True]

Correct

That's correct. Strings are immutable in Java, meaning their content cannot be changed once created. Any modification to a String, like the `toUpperCase()` method, will create a new String object.