Quiz Submissions - Java Review Reading Quiz

×

Chork Hieng (username: gt9182iu)

Retaken Attempt 2

Written: Jan 13, 2022 11:04 AM - Jan 13, 2022 11:06 AM

Submission View

Your quiz has been submitted successfully.

Show All Questions

Section 1.1

Question 1 Correct on previous attempt(s) **1 / 1 point**

A Java source file may contain only a single primary class.

✓ ○ True

○ False

Question 2 Correct on previous attempt(s) **1 / 1 point**

A Java source file may have a name that is different from the name of the class defined in it.

○ True

✓ ○ False

Question 3 Correct on previous attempt(s) **1 / 1 point**

Which of these is a proper declaration of a **`main()`** function in Java?

○ **`public static void Main(String[] args)`**

✓ ○ **`public static void main(String[] args)`**

○ **`public static void main(String args)`**

○ **`public void main(String[] args)`**

○ **`static void main(String[] args)`**

○ `public void Main(String[] args)`

▷ View Feedback

Question 4 Correct on previous attempt(s) **5 / 5 points**

Which of these are legal identifiers in Java?

✓ □ i

✓ □ num_students

✓ □ N

✓ □ numStudents

✓ □ sys%input

✓ □ student.name

✓ □ 3students

✓ □ shazam!

✓ □ NumStudents

✓ □ number of students

▷ View Feedback

Question 5 Correct on previous attempt(s) **4 / 4 points**

Which of the following operators may be used with `int`s?

✓ □ %

✓ □ +

✓ □ *

✓ □ &&

✓ □ -

✓ □ ||

✓ □ /

✓ □ !

▷ View Feedback

→ **Question 6** Retaken **4 / 4 points**

Which of the following operators may be used with `doubles`?

✓ □ %

✓ □ &&

✓ □ *

✓ □ ||

✓ □ /

✓ □ +

✓ □ !

✓ □ -

▷ View Feedback

Question 7 Correct on previous attempt(s) **4 / 4 points**

Which of the following operators may be used with `bools`?

✓ □ /

✓ □ *

✓ □ %

✓ □ !

✓ □ ||

✓ □ +

✓ □ -

✓ □ &&

▷ View Feedback

→ **Question 8** Retaken **2.5 / 3 points**

Which of the follow are valid `if` statements? (Assume all variables are declared elsewhere in the code.)

➡ ✓ □
```
if (x < 15)
    x = 15;
```

➡ ✓ □
```
if (x < 15)
{
    x = 15;
}
```

✓ □
```
if (x < 15)
    x = 15;
}
```

✓ □
```
if x < 15:
    x = 15;
```

✓ □
```
if x < 15
    x = 15;
```

➡ ✗ □
```
if (x < 15) {
    x = 15;
```

```
        }
```

▷ View Feedback

→ **Question 9** Retaken **3 / 3 points**

Which of the following are valid `if-else` statements? (Assume all variables are declared elsewhere in the code.)

✓ □
```
if (x < y) {
    min = x;
}
else {
    min = y;
}
```

✓ □
```
if (x < y) {
    min = x;
} else {
    min = y;
}
```

✓ □
```
if (x < y) min = x; else min = y;
```

✓ □
```
if (x < y) {
    min = x;
else
    min = y;
}
```

✓ □
```
if (x < y)
    min = x;
else if
    min = y;
```

✓ □
```
if (x < y)
    min = x;
else
    min = y;
```

▷ View Feedback

Question 10 Correct on previous attempt(s) **2 / 2 points**

Which of the follow are valid **while** statements? (Assume all variables are declared elsewhere in the code, and that both subsequent statements are meant to be part of the loop contents.)

✓ □
```
while x < 100
{
     sum += x;
     x += 1;
}
```

✓ □
```
while (x < 100)
     sum += x;
     x += 1;
```

✓ □
```
while (x < 100)
{
     sum += x;
     x += 1;
}
```

✓ □
```
while (x < 100) {
     sum += x;
     x += 1;
}
```

▷ View Feedback

Question 11 Correct on previous attempt(s) **1 / 1 point**

What are the possible values for a **bool** value?

○ 1 and 0

○ **True** and **False**

✓ ○ **true** and **false**

○ **T** and **F**

▷ View Feedback

Question 12 Correct on previous attempt(s) **1 / 1 point**

Java has two special loop exit statements. Match the statement with what it does.

✓ __1__	Exits the loop; the next statement to execute is the one following the loop.	1. **break**
✓ __2__	Exits this iteration of the loop, and returns to the top of the loop to check the condition.	2. **continue**

→ Question 13 Retaken 1.5 / 3 points

Which of the following are valid **for** loops? (Assume all necessary variables are declared elsewhere in the code.)

✓ □
```
for i in range(20):
    System.out.printf("%d %d\n", i, i*i);
```

✓ □
```
for (i < 20; int i = 0; i++) {
    System.out.printf("%d %d\n", i, i*i);
}
```

➡ ✗ □
```
for (int i = 0; i < 20; i++) {
    System.out.printf("%d %d\n", i, i*i);
}
```

✗ □
```
for int i = 0; i < 20; i++
    System.out.printf("%d %d\n", i, i*i);
```

➡ ✗ □
```
for (int i = 0; i < 20; i++)
{
    System.out.printf("%d %d\n", i, i*i);
}
```

➡ ✓ □
```
for (int i = 0; i < 20; i++)
    System.out.printf("%d %d\n", i, i*i);
```

▷ View Feedback

Question 14 Correct on previous attempt(s) 1 / 1 point

Which of the following lines of code properly allocates an int array called **dailyHighTemps** with 31 elements?

✓ ○ `int[] dailyHighTemps = new int[31];`

○ `int dailyHighTemps[31];`

○ `int[31] dailyHighTemps;`

○ `int[] dailyHighTemps = int[31];`

○ `int * dailyHighTemps = new int[31];`

▷ View Feedback

Question 15 Correct on previous attempt(s) **1 / 1 point**

Which of these properly creates an array of `int`s with 5 initial values:

○ `int[] collatz = new int{ 27, 82, 41, 124, 62 };`

○ `int collatz = { 27, 82, 41, 124, 62 };`

○ `int[] collatz{ 27, 82, 41, 124, 62 };`

✓ ○ `int[] collatz = { 27, 82, 41, 124, 62 };`

▷ View Feedback

Question 16 Correct on previous attempt(s) **1 / 1 point**

Which of the follow properly assigns the number of elements in array `a` to the variable `len`?

○ `double len = a.length;`

✓ ○ `int len = a.length;`

○ `int len = a.length();`

○ `int len = a.size;`

▷ View Feedback

Question 17 Correct on previous attempt(s) **1 / 1 point**

Which of these shows the proper signature of a static function called `find` which takes an array of `double`s and a `double` value (in that order), and returns a `int`?

○ `static find(double[] a, double val, int result)`

✓○ `static int find(double[] a, double val)`

○ `int find(double[] a, double val)`

○ `static int find(double val, double[] a)`

▷ View Feedback

Question 18 Correct on previous attempt(s) **1 / 1 point**

Which of these properly defines a static function called `count` which counts the number of times `val` appears in the `int` array `a`? (Just check the syntax, not whether the function actually works.)

✓○

```
public static int count(int[] a, int val) {
    int c = 0;
    for (int i = 0; i < a.length; i++)
        if (a[i] == val)
            c += 1;
    return c;
}
```

○

```
public static count(int[] a, int val) {
    int c = 0;
    for (int i = 0; i < a.length; i++)
        if (a[i] == val)
            c += 1;
    return c;
}
```

○

```
public static int count(int[] a, int val)
    int c = 0;
    for (int i = 0; i < a.length; i++)
        if (a[i] == val)
            c += 1;
    return c;
```

○

```
int count(int[] a, int val) {
    int c = 0;
    for (int i = 0; i < a.length; i++)
        if (a[i] == val)
            c += 1;
    return c;
}
```

▷ View Feedback

Question 19 Correct on previous attempt(s) **1 / 1 point**

`Math.sqrt` is a function in the Java library that takes a `double` and returns a `double`. Which of the following is a correct call to `Math.sqrt`?

○ `double sqrtOf2 = Math.sqrt("two");`

○ `String sqrtOf2 = Math.sqrt(2.0);`

✓ ○ `double sqrtOf2 = Math.sqrt(2.0);`

○ `double sqrtOf2 = sqrt(2.0);`

▷ View Feedback

Question 20 Correct on previous attempt(s) **1 / 1 point**

In Java (and other object-oriented languages) functions are often called methods.

✓ ○ True

○ False

→ **Question 21** Retaken **1 / 1 point**

A function may have only one return statement.

○ True

✓ ○ False

Question 22 Correct on previous attempt(s) **1 / 1 point**

Java allows multiple functions with the same name, as long as they have different parameters.

✓ ○ True

○ False

Question 23 Correct on previous attempt(s) **1 / 1 point**

A function with a return value of `void` generally has some sort of side effect, such as generating some output.

✓ ○ True

○ False

Question 24 Correct on previous attempt(s) **1 / 1 point**

What is wrong with this recursive definition of a factorial function?

```
public static int factorial(int n) {
    return n * factorial(n-1);
}
```

✓ ○ There is no base case.

○ There is no recursive call.

○ The recursive call does not reduce the problem size.

▷ View Feedback

Question 25 Correct on previous attempt(s) **1 / 1 point**

What is wrong with this recursive definition of a factorial function?

```
public static int factorial(int n) {
    if (n <= 1) return 1;
    return n * factorial(n);
}
```

○ There is no recursive call.

○ There is no base case.

✓ ○ The recursive call does not reduce the problem size.

▷ View Feedback

Question 26 Correct on previous attempt(s) **1 / 1 point**

Which of these correctly declares a string variable called name with the value "John Doe"?

○ `String name "John Doe";`

○ `String name = 'John Doe';`

○ `string name = "John Doe";`

✓○ `String name = "John Doe";`

▷ View Feedback

Question 27 Correct on previous attempt(s) **1 / 1 point**

What is the result of the follow code:

```
String unknownName = "John" + "Doe";
```

○ The variable `unknownName` is assigned the value "John Doe".

✓○ The variable `unknownName` is assigned the value "JohnDoe".

○ Nothing; this is a syntax error, because you cannot "add" strings.

▷ View Feedback

→ **Question 28** Retaken **1 / 1 point**

What is the result of this code:

```
int answer = 42;
String msg = "The answer is " + answer;
```

○ The variable `msg` is assigned the value "The answer is answer".

✓○ The variable `msg` is assigned the value "The answer is 42".

○ Nothing: you cannot "add" a String to an int.

▷ View Feedback

Question 29 Correct on previous attempt(s) **1 / 1 point**

What line of code creates a `Scanner` object to read from `System.in`?

○ `Scanner sc = new System.in;`

○ `Scanner sc = new Scanner();`

○ `Scanner sc = Scanner(System.in);`

✓○ `Scanner sc = new Scanner(System.in);`

▷ View Feedback

Question 30 Correct on previous attempt(s) **1 / 1 point**

Which code snippet reads integers from standard input until the input is exhausted?

○

```
Scanner sc = new Scanner(System.in);
while (sc.hasNextInt()) {
    int n = sc.next();
    // Do something with n
}
```

✓○

```
Scanner sc = new Scanner(System.in);
while (sc.hasNextInt()) {
    int n = sc.nextInt();
    // Do something with n
}
```

○

```
Scanner sc = new Scanner(System.in);
while (sc.hasNext()) {
    int n = sc.nextInt();
    // Do something with n
}
```

▷ View Feedback

Attempt Score:	48 / 50 - 96 %
Overall Grade (average of all attempts)**:**	47 / 50 - 94 %

Done