

OCA Junior Java Foundations Exam Preparation (1Z0-811)

Rizqi Ardiansyah

SMKN 2 Surabaya, 3 – 7 Desember 2017

Exam Number: 1Z0-811

Exam Title: Java Foundations (novice-level exam)

Associated Certification Paths

Passing this exam is required to earn these certifications. Select each certification title below to view full requirements. [More Info](#)

➤ [Java Foundations Certified Junior Associate \(novice-level certification\)](#)

Exam Details

Duration:	150 minutes
Number of Questions:	75
Passing Score:	65% View passing score policy
Validated Against:	This exam has been validated for version JDK 1.8.
Format:	Multiple Choice
Exam Price:	IDR 1188545 More on exam pricing

— Review Exam Topics

This is a novice-level exam for candidates who are students at secondary schools, 2-year colleges and 4-year colleges and universities. If you are seeking a more career-level certification, you may consider Java SE 8 Programmer I | 1Z0-808 or Java SE 7 Programmer I | 1Z0-803. Those exams are more appropriate for candidates who have completed training through Oracle University or Oracle's WDP program.

What Is Java?

- Describe the features of Java
- Describe the real-world applications of Java

Java Basics

- Describe the Java Development Kit (JDK) and the Java Runtime Environment (JRE)
- Describe the components of object-oriented programming
- Describe the components of a basic Java program
- Compile and execute a Java program

Basic Java Elements

- Identify the conventions to be followed in a Java program
- Use Java reserved words
- Use single-line and multi-line comments in Java programs
- Import other Java packages to make them accessible in your code
- Describe the java.lang package

Working with Java Data Types

- Declare and initialize variables including a variable using final
- Cast a value from one data type to another including automatic and manual promotion
- Declare and initialize a String variable

Working with Java Operator

- Use basic arithmetic operators to manipulate data including +, -, *, /, and %
- Use the increment and decrement operators
- Use relational operators including ==, !=, >, >=, <, and <=
- Use arithmetic assignment operators
- Use conditional operators including &&, ||, and ?
- Describe the operator precedence and use of parenthesis

Working with the String Class

- Develop code that uses methods from the String class
- Format Strings using escape sequences including %d, %n, and %s

Working with the Random and Math Classes

- Use the Random class
- Use the Math class

Using Decision Statements

- Use the decision making statement (if-then and if-then-else)
- Use the switch statement
- Compare how == differs between primitives and objects
- Compare two String objects by using the compareTo and equals methods

Using Looping Statements

- Describe looping statements
- Use a for loop including an enhanced for loop
- Use a while loop
- Use a do- while loop
- Compare and contrast the for, while, and do-while loops
- Develop code that uses break and continue statements

Debugging and Exception Handling

- Identify syntax and logic errors
- Use exception handling
- Handle common exceptions thrown
- Use try and catch blocks

Arrays and ArrayLists

- Use a one-dimensional array
- Create and manipulate an ArrayList
- Traverse the elements of an ArrayList by using iterators and loops including the enhanced for loop
- Compare an array and an ArrayList

Classes and Constructors

- Create a new class including a main method
- Use the private modifier
- Describe the relationship between an object and its members
- Describe the difference between a class variable, an instance variable, and a local variable
- Develop code that creates an object's default constructor and modifies the object's fields
- Use constructors with and without parameters
- Develop code that overloads constructors

Java Methods

- Describe and create a method
- Create and use accessor and mutator methods
- Create overloaded methods
- Describe a static method and demonstrate its use within a program

1. Given the code fragment:

```
String name = "Angel";
```

```
int score = 70;
```

```
System.out.println("Miss. %s's score is %d ", name, score);
```

What is the result?

- A. Miss. 70's score is Angel
- B. Miss. Angel's score is 70
- C. An exception is thrown at run time
- D. A compilation error occurs

2. Given the code fragment:

```
public static void main(String [] args) {  
    Boolean gotMilk = true;  
    ArrayList<String> shoppingList = new ArrayList<>();  
    shoppingList.add("Sugar");  
    shoppingList.add("Butter");  
    shoppingList.add("Eggs");  
    gotMilk = shoppingList.contains("Milk");  
    if(gotMilk) {  
        System.out.println("The shopping list is complete");  
    } else {  
        System.out.println("You forgot the milk");  
    }  
}
```

What is the result?

- A. A compilation error occurs
- B. A runtime exception is thrown
- C. You forgot the milk
- D. The shopping list is complete

3. Given the code fragment:

```
// line n1
```

```
num = new int[10];
```

Which code fragment can be inserted at line n1 to enable the code to compile?

- A. `int [] num;`
- B. `int num [10];`
- C. `int [10] num;`
- D. `new int num [];`

4. Which statement is valid?

- A. `int 2totalScore = 0;`
- B. `int total score = 0;`
- C. `int totalScore2 = 0;`
- D. `int total-score = 0;`

5. Given the code fragment:

```
int a = 3;  
int b = 2;  
int c = -1;  
if(a + b % c > (a + (-b) * (-c))) {  
    System.out.println(a + b - c);  
}
```

What is the result?

- A. A compilation error occurs
- B. There is no output
- C. 4
- D. 6

6. Given the code fragment

```
String [] flowers =  
{ "lotus", "lily", "rose", "jasmine" };  
for (String c : flowers) {  
    if (c.length() < 4) {  
        continue;  
    }  
    System.out.print(c + " ");  
    if (c.length() == 4) {  
        break;  
    }  
}
```

- A. A compilation error occurs.
- B. lotus
- C. lotus lily
- D. lotus jasmine

7. Given the code fragment

```
String s1 = "cat";
```

```
String s2 = new String(s1);
```

```
System.out.println(s1.equals(s2) + ":" + (s1 == s2) + ":" +  
s1.compareTo(s2));
```

What is the result?

- A. true:true:1
- B. false:true:0
- C. true:false:0
- D. True:0:true

8. Given the code fragment

```
int num = 100;  
int count = 0;  
do {  
    num--;  
    count++;  
} while (count > 1);  
System.out.println("num = " +  
num);
```

- A. num = 100;
- B. num = 0;
- C. num = 99;
- D. The program executes indefinitely

9. Given the code fragment

```
int value = 10;
int a = ++value;
int b = value;
int c = value++;
if(a <= b && a <= c) {
    if(b <= c) {
        a = ++b;
    } else {
        a = ++c;
    }
}
```

System.out.println(a);

What is the result?

- A. 13
- B. 11
- C. 10
- D. 12

10. Given the code fragment

```
class Ball {  
    double weight;  
}  
  
public class App {  
    public static void main(String [] args) {  
        // line n1  
        System.out.println(b.weight);  
    }  
}
```

Which code fragment can be inserted at line1 to enable code to print 0.0?

- A. Ball.weight = 0.0;
- B. Ball b = null;
 b.weight = 0.0;
- C. Ball b = new Ball(0.0);
- D. Ball b = new Ball();

11. Given the code fragment

```
1. class App {  
2.  
3. }
```

Which two code fragments are valid at line 2?

```
A. {  
    private int num;  
}
```

```
B. private String name = "John";  
   public void display() {  
       System.out.print(name);  
   }
```

```
C. package p1;
```

```
D. import java.util.*;  
   public void display(){  
       List<Integer> nums =  
           new ArrayList<>();  
   }
```

```
E. for(int count = 0; count < 5; count++) {  
    System.out.print(count);  
}
```

12. Given the code fragment

```
List<String> names = new ArrayList<>();  
names.add("Julia");  
names.add("Peter");  
for(Iterator<String> itr = names.iterator(); itr.hasNext();) {  
    System.out.println(itr.next());  
}
```

What is the result?

- | | | |
|----------|----------|----------------------------------|
| A. Julia | B. Peter | C. A compilation error occurs |
| Peter | Julia | D. A runtime exception is thrown |

13. Which statement is true about Java applications?

- A. They can run only on any Java Virtual Machine
- B. They depend on computer architecture
- C. They can run only if the appropriate Java Development Kit is available
- D. They can run only on the Java Virtual Machine upon which the application was developed

14. Given the code fragment

Given the code fragment:

```
boolean checkOut = true;
```

```
int days = 0;
```

```
while(checkOut) {
```

```
    days++;
```

```
    if(days > 3) {
```

```
        checkout = false;
```

```
    }
```

```
}
```

```
System.out.println(days);
```

What is the result?

A. 2

B. 3

C. 4

D. The program executes an infinite number of times

15. Given the code fragment

```
int a = 3;
```

```
a = ++a + a++;
```

```
a = --a - a--;
```

```
System.out.println(a);
```

What is the output?

- A. A compilation error occurs
- B. 3
- C. 8
- D. 0

16. Given the code fragment

```
public static void main(String [ ] args) {  
    int a = 10, b = 15;  
    boolean result = false;  
    // line n1  
    System.out.println(result);  
}
```

Which two statements, when inserted at line n1 independently, enable the code to print true?

- A. `result = a == b;`
- B. `result = a != b;`
- C. `result = a > b;`
- D. `result = ! (a > b);`
- E. `result = (a ! > b)`

17. Given the code fragment

```
short x = 1;
```

```
String y = "2";
```

```
System.out.println(y + x);
```

What is the output?

- A. arrays
- B. char
- C. String
- D. short

18. Which statement prints a random number with values only from 1 to 10?

- A. `System.out.println(Math.random() * 10);`
- B. `System.out.println(Math.round(Math.random() % 9));`
- C. `System.out.println(Math.round(Math.random() * 10));`
- D. `System.out.println(1 + Math.round(Math.random() * 9));`

19. Given the Car.java file:

```
public class Car {  
    public static void main(String [ ] args) {  
        System.out.print(args[0]);  
    }  
}
```

Which option enables you to print Win! ?

- A. javac Car.java
java Car.class Win!
- B. javac Car.java Win!
java Car.class
- C. javac Car.java
java Car Win!
- D. javac Car.java Win!
java Car

20. Given:

```
public class Test {  
    int var1;           // line n1  
    public static void main(String [ ] args) {  
        int var2;       // line n2  
        Test obj = new Test();  
        int var3 = var2 + obj.var1;  
        System.out.println(var3);  
    }  
}
```

What is the result?

- A. Compilation fails. To make it compile, replace line n1 with `var1 = 0;`
- B. 0
- C. Nothing is printed
- D. Compilation fails. To make it compile, replace line2 with `var2 = 0;`

21. Given:

```
String test = "a";  
for( ; test.compareTo("aaa") == 0 ; test = test + "a")  
    System.out.print(test.length() + " ");  
System.out.print(test);
```

What is the output?

- A. Compilation fails
- B. 1 2 3 aaaa
- C. a
- D. 1 2 aaa

22. Given

```
class Product {  
    String color = null;  
    Product (Product p) {  
        this.color = p.color;  
    }  
}
```

And the code fragment:

```
Product p1 = new Product();           // line n1  
p1.color = "White";  
Product p2 = new Product(p1);  
System.out.println(p1.color + " | " + p2.color);
```

What is the result?

- A. A compilation error occurs at line n1
- B. null | null
- C. White | null
- D. White | White

23. Given

```
class Messenger {  
    String msg;  
    Messenger(String msg) {  
        this.msg = msg;  
    }  
    public void writeMsg() {  
        System.out.println(msg);  
    }  
    public void readMsg() {  
        // line n1  
    }  
}
```

And the code fragment:

```
Messenger m = new Messenger("All the best");  
m.readMsg();
```

Which code fragment can be inserted at line n1 to enable the code to print All the best?

- A. Messenger.writeMsg();
- B. m.writeMsg();
- C. writeMsg();
- D. void writeMsg();

24. Given

```
class Product {
    int id;
    Product(int id) {
        this.id = id;
    }
}

public class Test {
    public static void main(String [] args) {
        List<Product> pts = new ArrayList<>();
        pts.add(new Product(100));
        pts.add(new Product(200));
        // line n1
    }
}
```

Which code fragment, when inserted at line n1, enable the code to print 100:200?

- A.

```
Iterator<Product> i = pts.iterator();
while(i.hasNext()) {
    System.out.print(i.next() + " : ");
}
```
- B.

```
for(int cn = 0; cn < pts.size(); cn++) {
    System.out.print(pts.id + " : ");
}
```
- C.

```
for(int id : pts) {
    System.out.print(pts.id + " : ");
}
```
- D.

```
for(Product pt : pts) {
    System.out.print(pts.id + " : ");
}
```

25. Given

```
class Bus {  
    String type = "default";  
    // line n1  
    Bus(String type) {  
        // line n2  
        this.type = type;  
    }  
}  
  
public class App {  
    public static void main(String [ ] args) {  
        Bus b1 = new Bus();  
        System.out.println(b1.type);  
        Bus b2 = new Bus("luxury");  
    }  
}
```

Which is the result?

- A. The code compiles and prints:
 default
 luxury
- B. The code fails to compile. To make it compile,
 at line n1 insert:
 this() { }
- C. The code fails to compile. To make it compile,
 at line n1 insert:
 this();
- D. The code fails to compile. To make it compile,
 at line n1 insert:
 Bus() { }

26. Given the code fragment

```
int [ ] num = new int[2];  
num[0] = 10;  
num[1] = 15;  
List<Integer> lst =  
    new ArrayList<>(2);  
lst.add(10);  
lst.add(15);  
num[1] = 20;  
lst.add(20);
```

```
for(int x : num) { System.out.print(x + " "); }  
System.out.println("");  
for(int y : lst) { System.out.print(y + " "); }
```

What is the result?

- A. A compilation error occurs
- B. 10 20
10 20
- C. 10 20
10 15 20
- D. A runtime exception is thrown

27. Given

```
public class App {  
    public void find(int x, int y) {  
        try {  
            int x = x / y;  
        } catch (Exception e) {  
            System.out.println("find() -  
exception");  
        }  
    }  
}
```

```
public static void main(String [ ] args) {  
    App obj = new App();  
    int [ ] array = {10, 0};  
    try {  
        int x = x / y;  
    } catch (Exception e) {  
        System.out.println("main() -  
exception");  
    }  
}
```

27. Contd.

What is the result?

- A. A compilation error occurs
- B. find() – exception
- C. main() – exception
- D. find() – exception
main() – exception

28. Given

```
public class Course {  
    String courseName;  
}  
public class Student {  
    String stuName;  
    public static void main(String [ ] args) {  
        Student s = new Student();  
        s.stuName = args[0];  
        Course c = new Course();  
        c.courseName = args[1];  
        System.out.println(s.stuName + " is studying " + c.courseName);  
    }  
}
```

28. Contd.

Which statement is true?

A. The commands:

```
javac Student.java
```

```
java Student Richard William Java
```

are used to print Richard William is studying Java

B. The commands:

```
javac Student.java
```

```
java Student "Richard William" Java
```

are used to print Richard William is studying Java

28. Contd.

C. The commands:

```
javac Student.java
```

```
java Student Richard William Java
```

throw an error about missing Course.class file

D. The commands:

```
javac Course.java
```

```
javac Student.java
```

```
java Course
```

```
java Student "Richard William" Java
```

are used to print Richard William is studying Java

29. Which statement is true about static methods?

- A. Static methods and variables can be invoked by using the class name directly
- B. Static methods are accessed with objects
- C. Static methods can be inherited by derived class
- D. Non-static variables and methods can not access static variables and methods

30. Which three statements are true about the structure of a Java class?

- A. A class can have only one private constructor.
- B. A method can have the same name as a field.
- C. A class can have overloaded static methods.
- D. A public class must have a main method.
- E. The methods are mandatory components of a class.
- F. The fields need not be initialized before use.

31. Which of the following are the advantages of Instance methods?

- A. Instance methods can be overridden and overloaded.
- B. They can be accessed by using an instance of the class using the dot operator.
- C. Static variables cannot be accessed from the instance methods.
- D. Methods having private access in a class are not inherited.

