

# SCJP Mock Exams by JavaChamp.com

Open Certification Plattform

Authors: N. Ibrahim, Y. Ibrahim

Copyright (c) 2009

# Introducing JavaChamp.com Website

JavaChamp.com is a Java Open Certification Platform.

It provides an interactive web interface for the Java community to learn, interact and certify its acquired java experience.

JavaChamp.com helps the Java developers to achieve the highest impact of thier learning activities.

Start JavaChamp.com online exams now and prepare yourself for the next interview or certification!

Visit <http://www.JavaChamp.com>



- ↓ Services
- ↓ About us
- ↓ Contac Us

- Home
- Take Exam
- Log in
- Register
- Book

Search



→ Stop

## ★ About us

JavaChamp.com is a Java Open Certification Plattform!

- It provides unique services to help you develop yourself in the field of computer science and programming
- It provides a web interface for the Java community to learn,interact and certify its acquired java experience.
- JavaChamp.com helps the Java developers to achieve the

## ★ Why to register in our community?

**Registration is optional and free but brings many advantages**

- By registering in JavaChamp.com you can save and keep track of your exams for later revision and hence to help you monitor your progress.

## CHECK OUR eBook

JavaChamp.com's eBook encompasses a large number of multiple questions, which cover the subjects java,C and assembly. The book doesn't ...

More →

## ★ Exam in Session

Question 3 / 20

What is the expected output?

```
01. public class OuterTest {
02.
03.     public static void main(String args[]) {
04.         Airplane.BlackBox box = new Airplane().new BlackBox(); // line 1
05.         box.printVariables();
06.
07.     }
08. }
09.
10. class Airplane {
11.     String code = "11";
12.
13.     class BlackBox {
14.         String code = "22";
15.
16.         public void printVariables() {
17.             System.out.print(code);
18.             System.out.print(Airplane.this.code); // line 20
19.
20.         }
21.     }
22.
23. }
```

- Compile error because of line 1 (incorrect instantiation)
- Compile error because of line 20 (can't access Airplane's variables)
- 2222
- 1111
- 2211

[Back](#) [Next](#)[Finish and evaluate](#) [Abort Exam](#)

### CHECK OUR eBook

JavaChamp.com's eBook encompasses a large number of multiple questions, which cover the subjects java, C and assembly. The book doesn't ...

[More](#) →

# Copyright

Copyright 2009 JavaChamp.com

Online version published by JavaChamp.com Germany.

## DISCLAIMER

All services and content of JavaChamp.com are provided under JavaChamp.com terms of use on an "as is" basis, without warranty of any kind, either expressed or implied, including, without limitation, warranties that the provided services and content are free of defects, merchantable, fit for a particular purpose or non-infringing. The entire risk as to the quality and performance of the provided services and content is with you. In no event shall JavaChamp.com be liable for any damages whatsoever arising out of or in connection with the use or performance of the services. Should any provided services and content prove defective in any respect, you (not the initial developer, author or any other contributor) assume the cost of any necessary servicing, repair or correction. This disclaimer of warranty constitutes an essential part of these "terms of use". No use of any services and content of JavaChamp.com is authorized hereunder except under this disclaimer.

The detailed "terms of use" of JavaChamp.com can be found under:

<http://www.javachamp.com/public/termsOfUse.xhtml>

This work is licensed under a Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 license.

The full license legal code can be found under:

<http://creativecommons.org/licenses/by-nc-nd/3.0/legalcode>

And a human-readable summary of the this license can be found under:

<http://creativecommons.org/licenses/by-nc-nd/3.0/>

According to the Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 license You agree to the following:

You are free to share, copy, distribute and transmit the work under the following conditions:

- You must attribute the work to JavaChamp.com with a link to <http://www.javachamp.com>.
- You may not use this work for commercial purposes.
- You may not alter, transform, or build upon this work.

# Table of Contents

## 1. Java Programming Language

- Strings
- Constructors
- Inner class
- Flow Control
- Declarations and Access Control
- Interfaces and Abstract Classes
- Exceptions
- Inheritance
- Arrays
- Operators
- Variables and Methods
- Wrappers and Autoboxing
- Garbage Collection
- Overriding and Overloading
- Collections
- Generics
- Formatting
- I/O
- Threads
- Enums
- Data Types
- Static and init blocks
- Serialization

# 1. Chapter: Java Programming Language

## Chapter Description and Objectives

### 1. Threads

#### Exam Category Description and Objectives

##### 1.1.1. How java thread start running?

What is considered an impossible output of running the following program?

```
public class Tester extends Thread {  
    int code = 9;  
  
    public void run() {  
        this.code = 7;  
    }  
  
    public static void main(String[] args) {  
  
        Tester thread = new Tester();  
        thread.start();  
  
        for (int i = 0; i < 5; i++) {  
            System.out.print(thread.code);  
        }  
  
    }  
}
```

Please choose only one answer:

- 99777
- 97777
- 77777
- 79999
- 99999

Check this Question online on JavaChamp.com: [How java thread start running?](#)

### 1.1.2. When java IllegalMonitorStateException is thrown?

What is the result of compiling and running the following program?

```
public class Tester {  
  
    public void validate() {  
        int i = 0;  
        while (++i < 3) {  
            try {  
                wait();  
            } catch (InterruptedException e) {  
                e.printStackTrace();  
            }  
            System.out.print(i);  
        }  
    }  
  
    public static void main(String[] args) {  
        new Tester().validate();  
    }  
}
```

Please choose only one answer:

- Compilation error because of calling wait() outside a synchronized block
- Compilation error because IllegalMonitorStateException is not handled
- At runtime, it throws an IllegalMonitorStateException when trying to wait
- 12

Check this Question online on JavaChamp.com: [When java IllegalMonitorStateException is thrown?](#)

### 1.1.3. Can java thread invoke start more than once?

What is the result of compiling and running the following code?

```
public class Tester extends Thread {  
    public void run() {  
        System.out.print("run");  
    }  
  
    public static void main(String[] args) {  
        Tester thread = new Tester();  
        new Thread(thread).start();  
        new Thread(thread).start();  
    }  
}
```

Please choose only one answer:

- Compilation error, can't invoke start() twice
- runrun
- IllegalStateException will be thrown because of the second invoke to start()
- run

Check this Question online on JavaChamp.com: [Can java thread invoke start more than once?](#)

#### 1.1.4. When java InterruptedException is thrown?

What could be a part of the output of compiling and running the following code?

```
public class Tester extends Thread {  
  
    public void run() {  
        System.out.println("run");  
    }  
  
    public static void main(String[] args) {  
        Thread thread = new Tester();  
        thread.run();  
        thread.start();  
        thread.start();  
    }  
}
```

Please choose all the answers that apply:

- Compilation error
- Prints "run" twice, not necessarily consecutively
- InterruptedException will be thrown
- InterruptedException will be thrown
- Prints "run" three times

Check this Question online on JavaChamp.com: [When java InterruptedException is thrown?](#)

### 1.1.5. which thread methods are static?

Which of the following methods defined in Thread class are static?

Please choose all the answers that apply:

- sleep()
- start()
- yield()
- join()
- run()

Check this Question online on JavaChamp.com: [which thread methods are static?](#)

### 1.1.6. which methods are defined in calss object?

Which methods are defined in calss Object?

Please choose all the answers that apply:

- wait()
- sleep()
- toString()
- finalize()
- notify()

Check this Question online on JavaChamp.com: [which methods are defined in calss object?](#)

### 1.1.7. when to override thread method run()?

When a class implements interface Runnable, it must provide implementation for method start():

Please choose only one answer:

- False
- True

Check this Question online on JavaChamp.com: [when to override thread method run\(\)?](#)

### 1.1.8. when a thread can call wait()?

A thread that calls the wait() method of an object, must own the lock of the object.

Please choose only one answer:

- True
- False

Check this Question online on JavaChamp.com: [when a thread can call wait\(\)?](#)

### 1.1.9. how can a thread own the lock of an object?

How can a thread own the lock of an object?

Please choose all the answers that apply:

- When the thread executes a synchronized instance method of that object
- When the thread executes a synchronized statement block
- When the thread calls wait() on this object

Check this Question online on JavaChamp.com: [how can a thread own the lock of an object?](#)

### 1.1.10. how to instantiate a thread object?

Which of the following Thread instantiations are correct?

```
public static void main(String[] args) {  
    new Thread(); // line 1  
    new Thread("myThread"); // line 2  
    new Thread(new Long(14)); // line 3  
    new Thread(new Runnable(){public void run(){} }); // line 4  
    Thread.getInstance(); // line 5  
}
```

Please choose all the answers that apply:

- line 1
- line 2
- line 3
- line 4
- line 5

Check this Question online on JavaChamp.com: [how to instantiate a thread object?](#)

### 1.1.11. How to create a Thread object?

Thread objects are created by:

Please choose all the answers that apply:

- direct instantiation from java.lang.Thread class
- instantiation from a class which implements the java.lang.Thread class
- instantiation a Runnable object (from a class which implements the Runnable interface)
- instantiation a Runnable object (from a class which implements the Runnable interface) and passing this Runnable object to a Thread constructor.

Check this Question online on JavaChamp.com: [How to create a Thread object?](#)

### 1.1.12. Create a thread by implementing Runnable

When creating a thread by implementing Runnable interface :

Please choose only one answer:

- you must implement the method run()
- you can override run(), but in case you don't you'll be restricted to the provided run() method, which contains no code and does nothing

Check this Question online on JavaChamp.com: [Create a thread by implementing Runnable](#)

### 1.1.13. instantiate a thread by implementing Runnable interface

How many times the statement "we are painting" would be printed in this program?

```
public class Test{  
    public static void main(String[] args) {  
        Painter painter1 = new Painter();  
        painter1.start();  
  
        Painter painter2 = new Painter();  
        painter2.start();  
  
    }  
}  
  
class Painter implements Runnable {  
    public void run() {  
        System.out.println("we are painting");  
    }  
}
```

Please choose only one answer:

- two times
- zero times
- the program will not compile

Check this Question online on JavaChamp.com: [instantiate a thread by implementing Runnable interface](#)

### 1.1.14. starting many threads together in java

What is the possible output of compiling and running the following code?

```
public class Test {  
  
    public static void main(String[] args) {  
        Thread request1 = new Thread(new InternetRequest (), "request# 1");  
        Thread request2 = new Thread(new InternetRequest (), "request# 2");  
  
        request1.start();  
        request2.start();  
  
    }  
}  
  
class InternetRequest implements Runnable {  
  
    public void run() {  
        System.out.print(Thread.currentThread().getName());  
  
    }  
}
```

Please choose all the answers that apply:

- request# 2 request# 1
- request# 1 request# 2
- compilation error

Check this Question online on JavaChamp.com: [starting many threads together in java](#)

### 1.1.15. instantiate a thread by implementing Runnable in java

What is the output of compiling and running the following code?

```
class Writer extends Thread {
    public void run() {
        System.out.println("Writer run");
    }

    public static void main(String[] args) {
        Runnable c = new Writer();
        Thread t = new Thread(c);
        t.run();
    }
}
```

Please choose only one answer:

- Compilation error
- compiles fine, but no output produced
- compiles fine and prints "Writer run"
- compiles fine but throws an exception

Check this Question online on JavaChamp.com: [instantiate a thread by implementing Runnable in java](#)

### 1.1.16. handling a checked exception

What is the output of compiling and running the following code?

```
public class Test extends Thread {
    static int count = 0;

    public static void main(String argv[]) {
        Test t = new Test ();
        t.increment(count);
        t.start();
        Thread.sleep(1000);
        System.out.println(count);
    }

    public void increment(int count) {
        ++count;
    }

    public void run() {
        count = count + 5;
    }
}
```

Please choose only one answer:

- 5
- 6
- Compilation error

Check this Question online on JavaChamp.com: [handling a checked exception](#)

### 1.1.17. running a thread in java

What is the output of compiling and running the following code?

```
public class Test extends Thread {
    static int count = 0;

    public static void main(String argv[]) throws InterruptedException {
        Test t = new Test ();
        t.increment(count);
        t.start();
        Thread.sleep(1000);
        System.out.println(count);
    }

    public void increment(int count) {
        ++count;
    }

    public void run() {
        count = count + 5;
    }
}
```

Please choose all the answers that apply:

- 5
- 6
- Compilation error
- An Exception is thrown

Check this Question online on JavaChamp.com: [running a thread in java](#)

### 1.1.18. synchronized static method and threads in java

What is the possible output of running this program once as it is and once by marking swim() synchronized?

```
class Swimmer implements Runnable{
    String name ;
    Swimmer(String name){
        this.name = name;
    }
    public void run() {
        Test.swim(name);
    }
}

public class Test {

    public static void swim(String name) {
        System.out.print(name);
        System.out.print(name);
    }

    public static void main(String[] args) {
        new Thread(new Swimmer("Tom")).start();
        new Thread(new Swimmer("Hanks")).start();
    }
}
```

Please choose all the answers that apply:

- as it is, a possible output is TomHanksTomHanks
- as it is, a possible output is TomHanksHanksTom
- by synchronizing swim(), a possible output is TomHanksHanksTom
- by synchronizing swim(), a possible output is HanksHanksTomTom
- by synchronizing swim(), a possible output is TomHanksTomHanks

Check this Question online on JavaChamp.com: [synchronized static method and threads in java](#)

### 1.1.19. synchronized method and threads in java

What are the possible outputs of running this program once as it is, and second with marking swim() synchronized?

```
class Swimmer implements Runnable{
    String name ;
    Swimmer(String name){
        this.name = name;
    }
    public void run() {
        new Test().swim(name);
    }
}

public class Test {

    public void swim(String name) {
        System.out.print(name);
        System.out.print(name);
    }
    public static void main(String[] args) {
        new Thread(new Swimmer("Tom")).start();
        new Thread(new Swimmer("Hanks")).start();
    }
}
```

Please choose all the answers that apply:

- as it is, the output is always consecutive two Tom then two Hanks or consecutive two Hanks then two Tom
- as it is, the output is undetermined. It is in any order of two Tom and two Hanks
- by synchronizing swim(), the output is always consecutive two Tom then two Hanks or consecutive two Hanks then two Tom
- by synchronizing swim(), the output is undetermined. It is in any order of two Tom and two Hanks

Check this Question online on JavaChamp.com: [synchronized method and threads in java](#)

## 1.1.20. synchronized method in java

What are the possible outputs of running this program once as it is, and second with marking swimIn() synchronized?

```
class Swimmer implements Runnable {
    String name;
    Pool pool;

    Swimmer(String name, Pool pool) {
        this.name = name;
        this.pool = pool;
    }

    public void run() {
        pool.swimIn(name);
    }
}

public class Pool {

    public void swimIn(String name) {
        System.out.print(name);
        System.out.print(name);
    }

    public static void main(String[] args) {
        Pool pool = new Pool();
        new Thread(new Swimmer("Tom", pool)).start();
        new Thread(new Swimmer("Hanks", pool)).start();
    }
}
```

Please choose all the answers that apply:

- as it is, the output is always two consecutive "Tom" followed by two consecutive "Hanks" or viceversa
- as it is, the output could be TomHanksTomHanks
- by synchronizing swimIn(), the output is always two consecutive "Tom" followed by two consecutive "Hanks" or viceversa
- by synchronizing swimIn(), the output could be TomHanksTomHanks
- by synchronizing swimIn(), the output could be TomHanksHanksTom

Check this Question online on JavaChamp.com: [synchronized method in java](#)

### 1.1.21. how to synchronize a method in java

What is the possible result of compiling and running the following code?

```
class Swimmer implements Runnable {
    String name;
    Pool pool;

    Swimmer(String name, Pool pool) {
        this.name = name;
        this.pool = pool;
    }

    public void run() {
        pool.swimIn(name);
    }
}

public class Pool {

    public void swimIn(String name) {
        synchronized {
            System.out.print(name);
            System.out.print(name);
        }
    }

    public static void main(String[] args) {
        Pool pool = new Pool();
        new Thread(new Swimmer("Tom", pool)).start();
        new Thread(new Swimmer("Hanks", pool)).start();
    }
}
```

Please choose only one answer:

- TomTomHanksHanks
- HanksHanksTomTom
- HanksTomHanksTom
- undetermined order
- compilation error

Check this Question online on JavaChamp.com: [how to synchronize a method in java](https://www.java-champ.com/question/how-to-synchronize-a-method-in-java/)

## 1.1.22. thread and synchronized method in java

What is the possible result of compiling and running the following code?

```
public class Test implements Runnable {
    Integer id;

    public static void main(String[] args) {
        new Thread(new Test()).start();
        new Thread(new Test()).start();
    }

    public void run() {
        press(id);
    }

    synchronized void press(Integer id) {
        System.out.print(id.intValue());
        System.out.print(++id.intValue());
    }
}
```

Please choose only one answer:

- 0101
- 0011
- -10-10
- -1-100
- compilation error
- an exception is thrown at run time

Check this Question online on JavaChamp.com: [thread and synchronized method in java](https://www.java-champ.com/question/thread-and-synchronized-method-in-java/)

### 1.1.23. synchronized non static method in java

What is the possible result of compiling and running the following code?

```
public class Test implements Runnable {
    Integer id = 0;

    public static void main(String[] args) {
        new Thread(new Test()).start();
        new Thread(new Test()).start();
    }

    public void run() {
        press(id);
    }

    synchronized void press(Integer id) {
        System.out.print(id.intValue());
        System.out.print(++id.intValue());
    }
}
```

Please choose only one answer:

- 0011
- 0101
- 0123
- compilation error
- an exception is thrown at run time

Check this Question online on JavaChamp.com: [synchronized non static method in java](#)

## 1.1.24. thread join java

What is true?

```
public class Test implements Runnable {  
  
    public static void main(String[] args) throws InterruptedException {  
        Test test = new Test();  
        Thread t= new Thread(test);  
        t.start();  
        t.join();  
        System.out.print("main");  
    }  
  
    public void run() {  
        System.out.print("run");  
    }  
}
```

Please choose only one answer:

- the output could be "mainrun"
- the output could be "runmain"
- the output could be "run" then an exception is thrown at run time
- compilation error

Check this Question online on JavaChamp.com: [thread join java](#)

## 1.1.25. join thread in java

What is true?

```
public class Test implements Runnable {  
  
    public static void main(String[] args) {  
        Test test = new Test();  
        Thread thread = new Thread(test);  
        thread.start();  
        thread.join();  
        System.out.print("main");  
    }  
  
    public void run() {  
        System.out.print("run");  
    }  
}
```

Please choose only one answer:

- the program could prints runmain
- the program could prints mainrun
- the compilation fails
- an exception is thrown at run time

Check this Question online on JavaChamp.com: [join thread in java](#)

## 1.1.26. run vs start in threads in java

What are the possible results of compiling and running the following code?

```
public class Test implements Runnable {  
    int id;  
  
    Test(int id) {  
        this.id = id;  
    }  
  
    public static void main(String[] args) throws InterruptedException {  
        Thread thread1 = new Thread(new Test(1));  
        Thread thread2 = new Thread(new Test(2));  
        thread1.run();  
        thread2.start();  
        System.out.print("main");  
    }  
  
    public void run() {  
        System.out.print(id);  
    }  
}
```

Please choose all the answers that apply:

- 12main
- 21main
- 2main1
- 1main2
- compilation error, cannot invoke run() directly on thread1

Check this Question online on JavaChamp.com: [run vs start in threads in java](#)

### 1.1.27. sleep in thread in java

What does sleep(long millis) in Thread class do?

Please choose only one answer:

- causes the thread, which sleep is invoked on, to sleep (temporarily cease execution) for the specified number of milliseconds
- causes the currently executing thread to sleep (temporarily cease execution) for the specified number of milliseconds
- causes the main() thread to sleep for the specified number of milliseconds
- Causes the currently executing thread to wait(temporarily cease execution) for the specified number of milliseconds then brings it back to run.

Check this Question online on JavaChamp.com: [sleep in thread in java](#)

### 1.1.28. thread in java

What are the possible results of compiling and running the following code?

```
public class Test extends Thread {  
    int id;  
  
    Test(int id) {  
        this.id = id;  
        start();  
    }  
  
    public static void main(String[] args) {  
  
        Thread t = new Thread(new Test(2));  
        t.start();  
        System.out.print("main");  
    }  
  
    public void run() {  
        System.out.print(id);  
    }  
}
```

Please choose all the answers that apply:

- main22
- 0main2
- main02
- 2main2
- compilation error for calling start twice
- `IllegalThreadStateException` is thrown for calling start twice

Check this Question online on JavaChamp.com: [thread in java](#)

## 1.1.29. using wait and sleep in threads in java

What are the possible results of compiling and running the following code?

```
public class Test {  
    public static void main(String[] args) throws InterruptedException {  
        Runnable t1 = new Runnable() {  
            public void run() {  
                try {  
                    System.out.print("t1before");  
                    Thread.sleep(100);  
                    System.out.print("t1after");  
                } catch (InterruptedException e) {  
                }  
            }  
        };  
  
        final Thread t2 = new Thread() {  
            public void run() {  
                try {  
                    System.out.print("t2before");  
                    wait();  
                    System.out.print("t2after");  
                } catch (InterruptedException e) {  
                }  
            }  
        };  
  
        t2.start();  
        new Thread(t1).start();  
    }  
}
```

Please choose all the answers that apply:

- t1before may be part of th output
- t1after may be part of th output
- t2before may be part of th output
- t2after may be part of th output
- an exception will be thrown at run time
- compilation fails

Check this Question online on JavaChamp.com: [using wait and sleep in threads in java](#)

### 1.1.30. using sleep and wait in thread in java

What is the possible results of compiling and running the following code?

```
public class Test {  
    static Runnable t1 = new Runnable() {  
        public void run() {  
            try {  
                System.out.print("t1before");  
                Thread.sleep(100);  
                System.out.print("t1after");  
            } catch (InterruptedException e) {  
            }  
        }  
    };  
  
    static Thread t2 = new Thread() {  
        public void run() {  
            try {  
                System.out.print("t2before");  
                synchronized (this) {  
                    wait();  
                }  
                System.out.print("t2after");  
            } catch (InterruptedException e) {  
            }  
        }  
    };  
  
    public static void main(String[] args) throws InterruptedException {  
        new Thread(t1).start();  
        t2.start();  
    }  
}
```

Please choose all the answers that apply:

- t1before may be part of the output
- t1after may be part of the output
- t2before may be part of the output
- t2after may be part of the output
- compilation fails
- IllegalMonitorStateException is thrown at run time

Check this Question online on JavaChamp.com: [using sleep and wait in thread in java](https://www.java-champ.com/question/using-sleep-and-wait-in-thread-in-java/)

### 1.1.31. synchronizing and multithreading in java

The idea of this program is to allow two workers to build a wall (which consists of bricks and cement) , given the following code, what necessarily modifications are needed to guarantee that the two workers will have a chance to work and the wall will be built by alternating between bricks and cement (brickcementbrickcement...) ? (choose all what apply)

```
class Worker extends Thread {
    Contract contract;

    Worker(Contract contract) {
        this.contract = contract;
    }

    public void run() {
        contract.work();
    }
}

public class Contract {

    StringBuilder wall = new StringBuilder("brick");
    boolean isCementLastAdded = false;

    public void putBrick() {
        if (isCementLastAdded) {
            wall.append("brick");
            isCementLastAdded = false;
        }
    }

    public void putCementLayer() {
        if (!isCementLastAdded) {
            wall.append("cement");
            isCementLastAdded = true;
        }
    }

    public boolean isWallDone() {
        return wall.length() >= 100;
    }

    public void work() {
        while (!isWallDone()) {
            putCementLayer();
            putBrick();
        }
    }

    public static void main(String[] args) {
        Contract contract = new Contract();
        new Worker(contract).start();
        new Worker(contract).start();
    }
}
```

Please choose all the answers that apply:

- synchronize putBrick()
- synchronize putCementLayer()
- synchronize work()
- add Thread.sleep() after putBrick() in work() method
- synchronize isWallDone()

Check this Question online on JavaChamp.com: [synchronizing and multithreading in java](#)

