| | |
|---|---|
| **Started on** | Thursday, 5 June 2025, 11:30 AM |
| **State** | Finished |
| **Completed on** | Thursday, 5 June 2025, 12:10 PM |
| **Time taken** | 39 mins 48 secs |
| **Marks** | 38.00/50.00 |
| **Grade** | **76.00** out of 100.00 |

**Question 1**

Complete

Mark 0.00 out of 1.00

What will be the output of the following code?

```
class A {
    int x = 10;
    A() {
        print();
    }
    void print() {
        System.out.println("A: " + x);
    }
}
class B extends A {
    int x = 20;
    void print() {
        System.out.println("B: " + x);
    }
}
public class Main {
    public static void main(String[] args) {
        A obj = new B();
    }
}
```

a. B: 0

b. A: 10

c. Runtime Error

d. B: 20

**Question 2**

Complete

Mark 1.00 out of 1.00

What is the result of the following code?

```
interface I1 {
    default void display() {
        System.out.println("I1");
    }
}
interface I2 {
    default void display() {
        System.out.println("I2");
    }
}
class C implements I1, I2 {
    public void display() {
        I1.super.display();
    }
}
public class Test {
    public static void main(String[] args) {
        new C().display();
    }
}
```

- a. I1
- b. I1 I2
- c. I2
- d. Compilation error

**Question 3**

Complete

Mark 1.00 out of 1.00

What will this code print?

```
class Super {
    static void method() {
        System.out.println("Super");
    }
}
class Sub extends Super {
    static void method() {
        System.out.println("Sub");
    }
}
public class Demo {
    public static void main(String[] args) {
        Super obj = new Sub();
        obj.method();
    }
}
```

- a. Runtime error
- b. Super
- c. Compilation error
- d. Sub

**Question 4**

Complete

Mark 0.00 out of 1.00

Which one is not allowed in Java?

- a. Abstract class with constructor
- b. Final class with abstract methods
- c. Abstract class with static methods
- d. A class with both abstract and non-abstract methods

**Question 5**

Complete

Mark 0.00 out of 1.00

What is the output?

```
class Test {
    private void display() {
        System.out.println("Private");
    }
}
class Sub extends Test {
    public void display() {
        System.out.println("Public");
    }
}
public class Demo {
    public static void main(String[] args) {
        Test t = new Sub();
        t.display();
    }
}
```

- a. Compilation error
- b. Runtime error
- c. Private
- d. Public

**Question 6**

Complete

Mark 1.00 out of 1.00

What is the output of the following code?

```
class Base {
    void show() {
        System.out.println("Base show()");
    }
}
class Derived extends Base {
    void show(int x) {
        System.out.println("Derived show(" + x + ")");
    }
}
public class Main {
    public static void main(String[] args) {
        Base b = new Derived();
        b.show();
    }
}
```

- a. Runtime error
- b. Compilation error
- c. Derived show()
- d. Base show()

**Question 7**

Complete

Mark 1.00 out of 1.00

What is the output?

```
class A {
    int i = 10;
    A() {
        System.out.println(i);
        i = 20;
    }
}
public class Test {
    public static void main(String[] args) {
        A a = new A();
        System.out.println(a.i);
    }
}
```

- a. 10 20
- b. 20 20
- c. 0 10
- d. 10 10

**Question 8**

Complete

Mark 1.00 out of 1.00

What is true about constructors in Java?

- a. A constructor can call another constructor in the same class using this()
- b. Constructors must be public
- c. Constructor name can be different from class name
- d. Constructors can be inherited

**Question 9**

Complete

Mark 1.00 out of 1.00

What will be the output?

```
class Animal {
    void sound() { System.out.println("Generic sound"); }
}
class Dog extends Animal {
    void sound() { System.out.println("Bark"); }
}
public class Main {
    public static void main(String[] args) {
        Animal a = new Dog();
        a.sound();
    }
}
```

- a. Compilation error
- b. Bark
- c. Runtime error
- d. Generic sound

**Question 10**

Complete

Mark 0.00 out of 1.00

Which of the following is not true for method overriding?

- a. Method name must be same
- b. Return type must be same or subtype
- c. Access modifier can be more restrictive
- d. Method must be inherited

**Question 11**

Complete

Mark 1.00 out of 1.00

What will happen?

```
class A {
    final void show() {}
}
class B extends A {
    void show() {}
}
```

- a. Compilation error
- b. Compiles and runs
- c. Runtime error
- d. Shows nothing

**Question 12**

Complete

Mark 1.00 out of 1.00

What is the output?

```
try {
    throw new IllegalArgumentException("Illegal");
} catch (IllegalArgumentException e) {
    System.out.println(e.getMessage());
}
```

- a. Exception
- b. IllegalArgumentException
- c. null
- d. Illegal

**Question 13**

Complete

Mark 1.00 out of 1.00

What is true about finally block?

- a. It must be used with catch
- b. It executes only if no exception occurs
- c. It always executes regardless of exceptions
- d. It is optional

**Question 14**

Complete

Mark 1.00 out of 1.00

What will be printed?

```
try {
    int[] arr = new int[5];
    arr[5] = 100;
} catch (ArrayIndexOutOfBoundsException e) {
    System.out.print("Caught ");
} finally {
    System.out.print("Finally");
}
```

- a. Caught
- b. Runtime Error
- c. Caught Finally
- d. Finally

**Question 15**

Complete

Mark 1.00 out of 1.00

What happens if catch block is missing?

```
try {
    System.out.println(10 / 0);
} finally {
    System.out.println("Done");
}
```

- a. Done
- b. Compilation error
- c. Program compiles but doesn't run
- d. ArithmeticException is thrown after finally

**Question 16**

Complete

Mark 0.00 out of 1.00

What is the output of the following code?

```
try {
    throw new Exception("Check");
} catch (RuntimeException e) {
    System.out.println("Runtime");
} catch (Exception e) {
    System.out.println("Exception");
}
```

- a. Exception
- b. Runtime
- c. Compilation error
- d. Runtime error

**Question 17**

Complete

Mark 1.00 out of 1.00

Which of the following is a checked exception?

- a. ArithmeticException
- b. IllegalArgumentException
- c. FileNotFoundException
- d. NullPointerException

**Question 18**

Complete

Mark 1.00 out of 1.00

What is the output?

```
try {
    int a = 5 / 0;
} catch (Exception e) {
    System.out.print("Catch ");
} finally {
    System.out.print("Finally");
}
```

- a. Catch Finally
- b. Runtime Error
- c. Finally
- d. Catch

**Question 19**

Complete

Mark 1.00 out of 1.00

What happens if exception is thrown in finally block?

- a. finally block never throws exception
- b. Compile-time error
- c. Original exception is suppressed
- d. Both exceptions are printed

**Question 20**

Complete

Mark 1.00 out of 1.00

What is the output?

```
public class Test {
    public static void main(String[] args) {
        try {
            System.out.println("Try");
            return;
        } finally {
            System.out.println("Finally");
        }
    }
}
```

- a. Compilation error
- b. Try Finally
- c. Try
- d. Finally

**Question 21**

Complete

Mark 0.00 out of 1.00

What is the output of the following code?

```
LocalDate date = LocalDate.of(2024, Month.FEBRUARY, 29);
System.out.println(date.plusYears(1));
```

- a. DateTimeException
- b. 2025-02-28
- c. 2025-02-29
- d. 2025-03-01

**Question 22**

Complete

Mark 1.00 out of 1.00

What does this print?

```
LocalTime time = LocalTime.of(23, 59, 59);
System.out.println(time.plusSeconds(2));
```

- a. 00:01:00
- b. 00:00
- c. 23:59:61
- d. 00:00:01

**Question 23**

Complete

Mark 0.00 out of 1.00

Which class would you use to represent a date and time with time zone?

- a. ZonedDateTime
- b. Instant
- c. OffsetDateTime
- d. LocalDateTime

**Question 24**

Complete

Mark 0.00 out of 1.00

What will this code output?

```
DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy/MM/dd");
LocalDate date = LocalDate.parse("2023/08/15", formatter);
System.out.println(date);
```

- a. 2023/08/15
- b. 2023-08-15
- c. August 15, 2023
- d. Compilation Error

**Question 25**

Complete

Mark 0.00 out of 1.00

What is the output?

```
Instant instant = Instant.now();
ZonedDateTime zdt = instant.atZone(ZoneId.of("UTC"));
System.out.println(zdt.getOffset());
```

- a. null
- b. +00:00
- c. Throws Exception
- d. System timezone offset

**Question 26**

Complete

Mark 1.00 out of 1.00

What does the JVM use the heap memory for?

- a. Method calls
- b. Object allocation
- c. Thread-local storage
- d. Stack frames

**Question 27**

Complete

Mark 1.00 out of 1.00

What happens when an object becomes unreachable in Java?

- a. It becomes eligible for GC
- b. It's immediately deleted
- c. JVM throws NullPointerException
- d. StackOverFlowError occurs

**Question 28**

Complete

Mark 0.00 out of 1.00

Which generation in GC typically contains short-lived objects?

- a. Eden Space
- b. PermGen
- c. Tenured
- d. Old Generation

**Question 29**

Complete

Mark 0.00 out of 1.00

What is true about finalize() in Java?

- a. It may or may not be called
- b. It guarantees memory cleanup
- c. It is used for performance
- d. It is always called before GC

**Question 30**

Complete

Mark 1.00 out of 1.00

What is the role of System.gc()?

- a. Prevents GC
- b. Immediately triggers GC
- c. Deletes static data
- d. Requests GC but does not force it

**Question 31**

Complete

Mark 1.00 out of 1.00

What is the output?

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

- a. Only one "Thread running"
- b. Runtime error
- c. Compilation error
- d. Thread running Thread running

**Question 32**

Complete

Mark 1.00 out of 1.00

Which method causes the current thread to wait until another completes?

- a. sleep()
- b. yield()
- c. wait()
- d. join()

**Question 33**

Complete

Mark 1.00 out of 1.00

What will this code output?

```
public class Main {
    public static void main(String[] args) {
        Thread t = new Thread(() -> {
            for (int i = 0; i < 2; i++) {
                System.out.print(Thread.currentThread().getName() + " ");
            }
        });
        t.setName("Worker");
        t.start();
    }
}
```

- a. Main Main
- b. Worker Worker
- c. Compiler Error
- d. Thread-0 Thread-0

**Question 34**

Complete

Mark 1.00 out of 1.00

What does this print?

```
class MyThread extends Thread {
    public void run() {
        System.out.print("Hello ");
    }
    public static void main(String[] args) {
        MyThread t1 = new MyThread();
        t1.start();
        t1.start();
    }
}
```

- a. Hello
- b. Compilation Error
- c. Hello Hello
- d. Runtime Exception

**Question 35**

Complete

Mark 0.00 out of 1.00

Which of the following is true about thread priorities?

- a. Priority affects the order of execution deterministically
- b. Thread priority is a hint to the thread scheduler
- c. Priorities are always respected by JVM
- d. Higher priority means more CPU time guaranteed

**Question 36**

Complete

Mark 1.00 out of 1.00

What does this code print?

```
import java.util.concurrent.*;
public class Main {
    public static void main(String[] args) {
        ExecutorService service = Executors.newFixedThreadPool(2);
        service.submit(() -> System.out.println("Task 1"));
        service.submit(() -> System.out.println("Task 2"));
        service.shutdown();
    }
}
```

- a. Any order of Task 1 and Task 2
- b. Compilation error
- c. Task 1 Task 2
- d. Task 2 Task 1

**Question 37**

Complete

Mark 1.00 out of 1.00

What is the key feature of ReentrantLock over synchronized?

- a. Fairness policy and tryLock() capability
- b. Can't be interrupted
- c. Simpler syntax
- d. No need to unlock manually

**Question 38**

Complete

Mark 1.00 out of 1.00

```
What will happen here?
ConcurrentHashMap<String, Integer> map = new ConcurrentHashMap<>();
map.put("A", 1);
map.compute("A", (k, v) -> v + 1);
System.out.println(map.get("A"));
```

- a. 2
- b. 1
- c. NullPointerException
- d. Compilation error

**Question 39**

Complete

Mark 1.00 out of 1.00

What is ForkJoinPool best suited for?

- a. Dividing tasks recursively and processing in parallel
- b. Thread communication
- c. UI event handling
- d. Long I/O-bound tasks

**Question 40**

Complete

Mark 1.00 out of 1.00

What is the default parallelism level of ForkJoinPool?

- a. Number of processors - 1
- b. 1
- c. Number of available processors
- d. Fixed at 4

**Question 41**

Complete

Mark 1.00 out of 1.00

What will this code print?

```
List<String> list = Arrays.asList("apple", "banana", "cherry");
list.stream()
    .filter(s -> s.length() > 5)
    .map(String::toUpperCase)
    .forEach(System.out::print);
```

- a. banana cherry
- b. APPLEBANANACHERRY
- c. BANANACHERRY
- d. CHERRYBANANA

**Question 42**

Complete

Mark 1.00 out of 1.00

What is the output of the following?

```
Stream.of(1, 2, 3, 4, 5)
      .filter(i -> i % 2 == 0)
      .map(i -> i * i)
      .findFirst()
      .ifPresent(System.out::print);
```

- a. 4
- b. 2
- c. 16
- d. 1

**Question 43**

Complete

Mark 1.00 out of 1.00

Which operation is terminal in streams?

- a. peek()
- b. map()
- c. forEach()
- d. filter()

**Question 44**

Complete

Mark 1.00 out of 1.00

What is the result?

```
List<String> list = Arrays.asList("a", "bb", "ccc", "dd");
String result = list.stream()
    .filter(s -> s.length() == 2)
    .collect(Collectors.joining("-"));
System.out.println(result);
```

- a. bb
- b. bb-dd
- c. a-bb-ccc-dd
- d. bbd

**Question 45**

Complete

Mark 1.00 out of 1.00

What will this output?

```
Stream<String> s = Stream.of("java", "lambda", "stream");
long count = s.map(String::length).filter(l -> l > 5).count();
System.out.println(count);
```

- a. 3
- b. 0
- c. 2
- d. 1

**Question 46**

Complete

Mark 1.00 out of 1.00

What is the output?

```
List<String> list = new ArrayList<>();
list.add("a");
list.add("b");
list.add(1, "c");
System.out.println(list);
```

- a. [a, b, c]
- b. [a, c, b]
- c. Compilation error
- d. [c, a, b]

**Question 47**

Complete

Mark 1.00 out of 1.00

Which collection is synchronized?

- a. Vector
- b. ArrayList
- c. HashMap
- d. LinkedList

**Question 48**

Complete

Mark 1.00 out of 1.00

What does this print?

```
Map<String, String> map = new HashMap<>();
map.put("a", "apple");
map.put("b", "banana");
map.put("a", "avocado");
System.out.println(map.get("a"));
```

- a. apple
- b. banana
- c. avocado
- d. null

**Question 49**

Complete

Mark 1.00 out of 1.00

What is true about HashSet?

- a. Uses hashCode and equals
- b. Maintains insertion order
- c. Implements List
- d. Allows duplicate elements

**Question 50**

Complete

Mark 1.00 out of 1.00

Which Map maintains insertion order?

- a. Hashtable
- b. TreeMap
- c. HashMap
- d. LinkedHashMap