OOPJ - CCEE Practice Test 1

Total points 13/20

Time: 30 Mins
Experience section: 5 Mins

The respondent's email (**harshal.tarmale.cmaug25@gmail.com**) was recorded on submission of this form.

0 of 0 points

Name *

Harshal Vilas Tarmale

MCQ

13 of 20 points

✗ class Demo { * 0/1

```
    void display() {
        System.out.println("Hello");
    }
    static void call() {
        display();
    }
}
```

- ○ Hello
- ○ Compilation error
- ◉ Runtime error ✗
- ○ Nothing prints

Correct answer

- ◉ Compilation error

✗ class Demo { * 0/1

```
    static int a = 10;
    void show() {
        int a = 5;
        System.out.println(a);
    }
}
```

- ○ 10
- ○ 5
- ◉ Compilation error ✗
- ○ 10 and 5

Correct answer

- ◉ 5

✓ class Test { * 1/1

```
    void display() {
        System.out.println("Hi");
    }
}
public class Main {
    public static void main(String[] args) {
        Test.display();
    }
}
```

- ○ Hi
- ◉ Compilation error ✓
- ○ Exception
- ○ Runtime error

✓ class Test { * 1/1

```
    static int x = 10;
    void show() {
        x++;
        System.out.println(x);
    }
}
public class Main {
    public static void main(String[] args) {
        Test t1 = new Test();
        Test t2 = new Test();
        t1.show();
        t2.show();
    }
}
```

- ○ 10 10
- ○ 11 11
- ◉ 11 12 ✓
- ○ 12 12

✓ What is the role of the **class loader** in the JVM? * 1/1

- ○ The class loader is responsible for executing bytecode in the JVM.
- ◉ The class loader loads class files into memory and verifies them. ✓
- ○ The class loader stores instances of objects in memory.
- ○ The class loader compiles Java source code into bytecode.

✗ class Test { * 0/1

```
    void display() {
        System.out.println("Hello");
    }
    static void call() {
        Test t;
        t.display();
    }
}
```

- ○ Hello
- ○ Compilation error
- ◉ Runtime error - NullPointerException ✗
- ○ Nothing prints

Correct answer

- ◉ Compilation error

✓ class Demo { * 1/1

```
    static int x;
    static {
        x = 5;
        System.out.println("Static block executed");
    }
    Demo() {
        System.out.println("Constructor executed");
    }
}
public class Main {
    public static void main(String[] args) {
        Demo d1;
        Demo d2 = new Demo();
    }
}
```

- ○ Static block executed
- ◉ Static block executed, Constructor executed ✓
- ○ Constructor executed
- ○ Nothing prints

✗ class Test { * 0/1

```
    static void count() {
        for(int i=0; i<5; ) {
            System.out.print(i + " ");
        }
    }
}
public class Main {
    public static void main(String[] args) {
        Test.count();
    }
}
```

- ○ 0 1 2 3 4
- ○ Infinite loop
- ◉ Compilation error ✗
- ○ 0

Correct answer

- ◉ Infinite loop

✗ class Demo { * 0/1

```
    static int x = 100;
    void show() {
        int x = 50;
        System.out.println(x);
    }
}
public class Main {
    public static void main(String[] args) {
        new Demo().show();
    }
}
```

- ○ 100
- ○ 50
- ◉ Compilation error ✗
- ○ 0

Correct answer

- ◉ 50

✓ class Counter { * 1/1

```
    static int count;
    Counter() {
        count++;
    }
}
public class Main {
    public static void main(String[] args) {
        Counter c1;
        c1 = new Counter;
        Counter c2 = new Counter();
        System.out.println(Counter.count);
    }
}
```

- ○ 2
- ◉ Compilation error ✓
- ○ 1
- ○ Runtime error

✓ What is stored in the **method area** in JVM memory? * 1/1

- (●) Method area stores static variables and method code. ✓
- ( ) Method area stores the heap memory addresses of objects.
- ( ) Method area stores local variables and method calls.
- ( ) Method area stores only static methods but not variables.

✓ class Sample { * 1/1

```
    void display() {
        System.out.println("Hello");
    }
    static void call() {
        Sample s = new Sample();
        s.display();
    }
}
public class Main {
    public static void main(String[] args) {
        call();
    }
}
```

- (●) Compilation error ✓
- ( ) Hello
- ( ) Runtime error
- ( ) null

✓ class Sample { * 1/1

```
    static {
        System.out.println("Static block");
    }
}
public class Main {
    public static void main(String[] args) {
        Sample s;
        s = new Sample();
    }
}
```

- (●) Static block ✓
- ( ) Nothing prints
- ( ) Compilation error
- ( ) Runtime error

✗ class Demo { * 0/1

```
int x = 10;
  static void show() {
    System.out.println(x);
  }
}
```

- ○ 10
- ○ Compilation error
- ◉ Runtime error ✗
- ○ Nothing prints

Correct answer

- ◉ Compilation error

✓ what will be the sequence of JVM components involved in execution after the program is loaded? *1/1

- ○ Class Loader → Execution Engine → Heap → Stack
- ◉ Class Loader → Method Area → Execution Engine ✓
- ○ Class Loader → Stack → Execution Engine
- ○ Method Area → Class Loader → Stack

✓ class Test { * 1/1

```
    static void display(int x) {
        System.out.println(x);
    }
    static void display(double x) {
        System.out.println(x);
    }
}
public class Main {
    public static void main(String[] args) {
        display(5);
    }
}
```

- ○ 5
- ◉ Compilation error ✓
- ○ 5.0
- ○ Runtime error

✓ Which of the following statements about JVM memory is correct, given the following code? *1/1

- ○ Stack stores instance variables, and heap stores local variables.
- ◉ Stack stores local variables, and heap stores instance variables and objects. ✓
- ○ Stack stores method references, and heap stores primitive variables.
- ○ Stack stores method calls, and heap stores all class-level information.

✗ class Test { * 0/1

```
    static void printNumbers() {
        for(int i; i<5; i++) {
            System.out.println(i);
        }
    }
}
```

- ○ 0 1 2 3 4
- ○ Compilation error
- ○ Infinite loop
- ◉ Prints nothing ✗

Correct answer

- ◉ Compilation error

✓ class Demo { * 1/1

```
    static int x = 0;
    Demo() {
        x++;
    }
}
public class Main {
    public static void main(String[] args) {
        Demo d1 = new Demo();
        Demo d2 = new Demo();
        Demo d3 = new Demo();
        System.out.println(d2.x);
    }
}
```

- ○ 1
- ○ 2
- ◉ 3 ✓
- ○ Compilation error

✓ class Sample { * 1/1

```
    static int x;
    Sample() {
        x += 5;
    }
}
public class Main {
    public static void main(String[] args) {
        Sample s1 = new Sample();
        Sample s2 = new Sample();
        Sample s3 = new Sample();
        System.out.println(x);
    }
}
```

- ○ 15
- ◉ Compilation error ✓
- ○ 5
- ○ Runtime error

Experience Section 0 of 0 points

Level of Exam *

- ○ Easy
- ◉ Medium
- ○ Hard

Do you feel that you are becoming better with consistency? *

- ◉ Yes
- ○ No

How is your experience? (No one word) *

Good way to test my doubting and my firmness on my knowledge. as similar type questions with little difference makes you think more. Good exam to test my knowledge. will be going through my mistake to correct my way of thinking if i have done many errors.