NPTEL Online Certification Courses

Indian Institute of Technology Kharagpur

NOC24-CS105 (July-2024 24A)

PROGRAMMING IN JAVA

Assignment 5

TYPE OF QUESTION: MCQ

Number of questions: 10 Total marks**: 10 × 1 = 10**

QUESTION 1:

Which exception will be thrown by parseInt() method in Java?

a. **IntegerOutOfBoundException**

b. **IntegerFormatException**

c. **ArithmeticException**

d. **NumberFormatException**

Integer.parseInt(String s): Converts the string s to a decimal integer.
Integer.parseInt(String s, int radix): Converts s to an integer using the specified radix (base). For example, if radix is 16, the method interprets the string as a hexadecimal number.

Correct Answer:

d. **NumberFormatException**

Detailed Solution:

parseInt() method parses input into integer. This method will throw NumberFormatException.

If the string does not contain a valid integer representation, parseInt will throw a NumberFormatException. For example:

```
String invalidStr = "abc";
int number = Integer.parseInt(invalidStr);  // This will throw NumberFormatException
```

QUESTION 2:

What will be the output of the following program?

```java
interface P {
    String p = "PPPP";
    String methodP();
}
interface Q extends P {
    String q = "QQQQ";
    String methodQ();
}
class R implements P, Q {
    public String methodP() {
        return q + p;
    }
    public String methodQ() {
        return p + q;
    }
}
public class Main{
    public static void main(String[] args) {
        R r = new R();
        System.out.println(r.methodP());
        System.out.println(r.methodQ());
    }
}
```

a. QQQQPPPP
PPPPQQQQ

b. PPPPQQQQ
QQQQPPPP

c. PPPPPPPP
QQQQQQQQ

d. Compilation error

Correct Answer:

a. QQQQPPPP
PPPPQQQQ

Detailed Solution:

methodP() returns the concatenation of q and p, resulting in "QQQQPPPP".
methodQ() returns the concatenation of p and q, resulting in "PPPPQQQQ".

QUESTION 3:

What will be the output of the following code?

```
class A implements B {
    public int methodB(int i) {
        return i = +i * i;
    }
}

interface B {
    int methodB(int i);
}

public class MainClass {
    public static void main(String[] args) {
        B b = new A();

        System.out.println(b.methodB(2));
    }
}
```

a. 4

b. 6

c. 2

d. 8

Correct Answer:

a. 4

Detailed Solution:

methodB(2) calculates 2 * 2, which equals 4. This value is then returned.

QUESTION 4:

A method that potentially generates a checked exception must include this keyword in its method signature:

- **a. throw**
- **b. extend**
- **c. throws**
- **d. extends**

Unchecked exceptions
• Exceptionsof class Error and RunTimeExceptionand all of their descendants are called unchecked exceptions
• Allotherexceptionsare calledcheckedexceptions
• These exceptions are not checked by the compiler, and hence, need not be caught or declared to be thrown in the program
• They are programming logical errors that can be fixed in compiled time, rather than leaving it to runtime exception handling.

Correct Answer:

c. throws

Checked exceptions :–
They are checked by the compiler and must be caught or declared to be thrown. – Listed in the throwsclause,or– Handledusing try catch in the method

Detailed Solution:

Any Java class that generates a checked exception and does not handle it internally must use the throws keyword to alert other methods of its instability.

1. Checked Exceptions:
Definition: Checked exceptions are exceptions that the compiler requires you to handle explicitly, either with a try-catch block or by declaring them in the method signature with the throws keyword.

Examples: IOException, SQLException, FileNotFoundException, ClassNotFoundException.

Purpose: These exceptions are generally beyond the control of the program and are often related to external resources (like files or databases), which may fail due to external factors.

2.Unchecked Exception:
Definition: Unchecked exceptions are exceptions that the compiler does not force you to handle. They occur due to programming errors, such as logic issues or improper usage of APIs, which can typically be fixed in the code itself.

Examples: ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException, IllegalArgumentException.

Purpose: These exceptions usually result from coding errors, and while they can be handled, the compiler doesn’t enforce it because they indicate issues that may be fixed by adjusting the code.

Summary
Checked Exceptions are checked at compile-time and are usually caused by issues outside the program’s direct control. Examples include file access or network errors.
Unchecked Exceptions occur due to programming errors, like dividing by zero or accessing an invalid array index. They are not checked by the compiler, but they can still be handled at runtime.

QUESTION 5:

Which of the following statements is true about Java's finally block?

a. **The finally block is only executed if an exception is thrown in the try block**

b. **The finally block is only executed if an exception is thrown in the catch block**

c. **The finally block is only executed if an exception is not thrown in the try or catch block**

d. **The finally block is executed regardless of whether an exception is thrown in the try or catch block**

Correct Answer:

d. **The finally block is executed regardless of whether an exception is thrown in the try or catch block**

Detailed Solution:

The finally block always executes, regardless of whether or not Java throws an exception in the try block.

QUESTION 6:

Which of the following statements is true about exception handling in Java:

a. **A try block can have many catch blocks but only one finally block** True!!!!

b. **A try block can have many catch blocks and many finally blocks** False!!

c. **A try block must have one finally block for each catch block** False!!

d. **A try block must have at least one catch block to have a finally block** False!!

Correct Answer:

a. **A try block can have many catch blocks but only one finally block**

Detailed Solution:

A try block can only have one finally block. However, multiple catches are allowed. It is even allowable to have no catch blocks and only a finally block.

No, a catch block cannot exist without a try block in Java. The catch block is designed to handle exceptions thrown by code inside the associated try

QUESTION 7:

What will be the output of the following program?

```
class Output {
    public static void main(String args[]) {
        try {
            int a = 0;
            int b = 5;
            int c = b / a;
            System.out.print("Hello");
        } catch (Exception e) {
            System.out.print("World");
        } finally {
            System.out.print("World");
        }
    }
}
```

Yes, in Java, the finally block is executed last after the try and catch blocks, regardless of whether an exception was thrown or caught.

a. **Hello**

b. **World**

c. **HelloWOrld**

d. **WorldWorld**

Correct Answer:

d. **WorldWorld**

Detailed Solution:

finally block is always executed after tryblock, no matter exception is found or not. catch block is executed only when exception is found. Here divide by zero exception is found hence both catch and finally are executed.

QUESTION 8:

What will be the output of the following Java code?

```
class Output {
    public static void main(String args[]) {
        try {
            int a = 0;
            int b = 5;
            int c = a / b;
            System.out.print("Hello");
        } finally {
            System.out.print("World");
        }
    }
}
```

a. **Hello**

b. **World**

c. **HelloWOrld**

d. **Compilation Error**

Correct Answer:

c. **HelloWOrld**

Detailed Solution:

finally block is always executed after try block, no matter exception is found or not.

QUESTION 9:

What will be the output of the following Java program?

```
interface calculate {
    void cal(int item);
}

class displayA implements calculate {
    int x;

    public void cal(int item) {
        x = item * item;
    }
}

class displayB implements calculate {
    int x;

    public void cal(int item) {
        x = item / item;
    }
}

class interfaces {
    public static void main(String args[]) {
        displayA arr1 = new displayA();
        displayB arr2 = new displayB();
        arr1.x = 0;
        arr2.x = 0;
        arr1.cal(2);
        arr2.cal(2);
        System.out.print(arr1.x + " " + arr2.x);
    }
}
```

a. 0 0

b. 2 2

c. 4 1

d. 1 4

Correct Answer:

c. 4 1

Detailed Solution:

`class displayA` implements the interface calculate by doubling the value of item, where as class displayB implements the interface by dividing item by item, therefore variable x of class displayA stores 4 and variable x of class displayB stores 1.

QUESTION 10:

Which of the following exceptions is not a subclass of the RuntimeException class?

others are RunTime exception!!

a. **NullPointerException**

b. **ArrayIndexOutOfBoundsException**

c. **IOException**

d. **ArithmeticException**

Correct Answer:

c. **IOException**

Detailed Solution:

IOException is not a subclass of the RuntimeException class. It is a checked exception in Java.