9. EXCEPTION HANDLING

```
public class MyClass {
    public static void main(String[] args) {
        RuntimeException re = null;
        throw re;
    }
}
```

What will be the result of attempting to compile and run the above program?

- a. The code will fail to compile, since the main() method does not declare that it throws RuntimeException in its declaration.
- b. The program will compile without error and will run and terminate without any output.
- c. The program will compile without error and will throw java.lang.RuntimeException when run.
- d. The program will compile without error and will throw java.lang.NullpointerException when run.

Given the following:

```
1. System.out.print("Start ");
2. try {
3.     System.out.print("Hello world");
4.     throw new FileNotFoundException();
5. }
6. System.out.print(" Catch Here ");
7. catch(EOFException e) {
8.     System.out.print("End of file exception");
9. }
10. catch(FileNotFoundException e) {
11.     System.out.print("File not found");
12. }
```

and given that EOFException and FileNotFoundException are both subclasses of IOException, and further assuming this block of code is placed into a class, which statement is most true concerning this code?

- a. Code output: Start Hello world Catch Here File not found.
- b. Code output: Start Hello world File Not Found.
- c. The code will not compile.
- d. Code output: Start Hello world End of file exception.

Given the following,

```
1. import java.io.*;
2. public class MyProgram {
3. public static void main(String args[]){
4. FileOutputStream out = null;
5. try {
6. out = new FileOutputStream("test.txt");
7. out.write(122);
8. }
9. catch(IOException io) {
10. System.out.println("IO Error.");
11. }
12. finally {
13. out.close();unhandled exception
14. }
15. }
16. }
```

and given that all methods of class FileOutputStream, including close(), throw an IOException, which one of these is true?

- a. This program fails to compile due to an error at line 6.
- b. This program fails to compile due to an error at line 13.
- c. This program will compile successfully.
- d. This program fails to compile due to an error at line 9.

Given the following:

1. class Base {
2. void display() throws Exception { throw new Exception(); }
3. }
4. public class Derived extends Base {
5. void display() { System.out.println("Derived"); }
6. public static void main(String[] args) {
7. new Derived().display();
8. }
9. }

What is the result ?

- a. Compilation fails because of an error in line 7.
- b. Compilation fails because of an error in line 2.
- c. The code runs with no output.
- d. Derived

Given the following program, which one of the statements is true?

```
public class Exceptions {
    public static void main(String[] args) {
        try {
            if (args.length == 0) return;
            System.out.println(args[0]);
        } finally {
            System.out.println("The end");
        }
    }
}
```

- a. If run with one argument, the program will print the given argument followed by "The end".
- b. If run with one argument, the program will produce no output.
- c. The program will throw an ArrayIndexOutOfBoundsException.
- d. If run with one argument, the program will simply print the given argument.

Both class Error and class Exception are children of this parent:

- a. Problem
- b. Catchable
- c. Runnable
- d. Throwable

Given the following code in the 3 java files:

NewException.java

```
class NewException extends Exception {
}
```

Welcome.java

```
class Welcome {
public String displayWelcome(String name) throws NewException {
if(name == null) {
throw new NewException();
}
return "Welcome "+ name;
}
}
```

TestNewException.java

```
class TestNewException {
public static void main(String... args) {
Welcome w = new Welcome();
System.out.println(w.displayWelcome("Ram"));
}
}
```

What is the result on compiling and executing it ?

- a. Compiles successfully and displays Ram when TestNewException is executed.
- b. Runtime exception occurs on executing the class TestNewException.
- c. Compilation of Welcome.java fails.
- d. Compilation of TestNewException.java fails

Given:

```
1.  public class B {
2.     Integer x;
3.     int sum;
4.     public B(int y) {
5.        sum=x+y;
6.        System.out.println(sum);
7.     }
8.     public static void main(String[] args) {
9.        new B(new Integer(23));
10.    }
11. }
```

What is the expected output?

- a. A NullPointerException occurs at runtime.
- b. Compilation fails because of an error in line 9.
- c. A NumberFormatException occurs at runtime.
- d. The value "23" is printed at the command line.

Which of the following lists exception types from MOST specific to LEAST specific?

- a. Throwable, RunTimeException
- b. ArithmeticException, RunTimeException
- c. Error, Exception
- d. Exception, RunTimeException

What type of exception is thrown by parseInt() if it gets illegal data?

- a. NumberFormatException
- b. ArithmeticException
- c. RunTimeException
- d. NumberError

Which of these statement is true ?

- a. finally block gets executed only when there are exceptions.
- b. finally block gets executed only when there are no exceptions.
- c. Finally gets always executed irrespective of the flow in try catch block.
- d. finally block can be present only when a catch block is present.

Given the following:

```
public class TestIfBoolean {
 public static void main(String[] args) {
  Boolean bFlag = null;
  if (bFlag) {
   System.out.print("A");
  } else if (bFlag == false) {
   System.out.print("B");
  } else {
   System.out.print("C");
  }
 }
}
```

What is the expected output?

- a. A
- b. C
- c. B
- d. java.lang.NullPointerException is thrown at runtime

What is the output of following code?

```
class Main {
  public static void main(String args[]) {
    int x = 0;
    int y = 10;
    int z = y/x;
  }
}
```

- a. Complier Error
- b. Complies and runs fine
- c. Compiles fine but throws ArithmeticException
- d. None of the mentioned

What is the output of following code

```
class Main {
  public static void main(String args[]) {
    try {
      throw 10;
    }
    catch(int e) {
      System.out.println("Got the  Exception " + e);
    }
  }
}
```

- a. Runtime error
- b. Got the exception 0
- c. Got the exception 10
- d. Compiler Error

Given the following:

```
class ShapeException extends Exception {
}

class CircleException extends ShapeException {
}

public class Circle2 {
 void m1() throws ShapeException {
  throw new CircleException();
 }

 public static void main(String[] args) {
  Circle2 circle2 = new Circle2();
  int a = 0, b = 0;

  try {
   circle2.m1();
   a++;
  } catch (ShapeException e) {
   b++;
  }

  System.out.printf("a=%d, b=%d", a, b);
 }
}
```

What is the expected output ?

- a. Compile time error at line 6.
- b. a=0, b=1
- c. a=1, b=0
- d. a=0, b=0

What is the result of compiling and executing the below code with the mentioned arguments ?

java TestInvocation Welcome Year 2009

```
public class TestInvocation
{
   public static void main(String... args)
   {
      String arg1 = args[1];
  String arg2 = args[2];
      String arg3 = args[3];
   }
}
```

- a. Compilation succeeds
- b. Throws exception at runtime
- c. Compilation fails
- d. None of the mentioned.

What is the result of compiling and executing the below code ?

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

- a. Outputs nothing
- b. Runtime Error
- c. Compilation Error
- d. Finally

Given the following,

```
1. public class RTExcept {
2. public static void throwit () {
3. System.out.print("throwit ");
4. throw new RuntimeException();
5. }
6. public static void main(String [] args) {
7. try {
8. System.out.print("hello ");
9. throwit();
10. }
11. catch (Exception re ) {
12. System.out.print("caught ");
13. }
14. finally {
15. System.out.print("finally ");
16. }
17. System.out.println("after ");
18. }
19. }
```

What is the output ?

- a. hello throwit RuntimeException caught after
- b. hello throwit caught
- c. hello throwit caught finally after RuntimeException
- d. hello throwit caught finally after

If a try statement has catch blocks for both Exception and IOException, then which of the following statements is correct?

a. A try statement cannot be declared with these two catch block types because they are incompatible.
b. The catch block for IOException must appear before the catch block for Exception.
c. The catch blocks for these two exception types can be declared in any order.
d. The catch block for Exception must appear before the catch block for IOException.

Given the following code:

```
public class ArithmeticTest {
   public static void main(String[] args){
      try
      {
         int x=0;
         int y=5/x;
         System.out.println👆 ;
      }
      catch (Exception e)
      {
         System.out.println("Exception");
      }
      catch (ArithmeticException ae)
      {
         System.out.println("ArithmeticException");
      }
   }
}
```

What is the output?

a. Compilation Error
b. NaN
c. Exception
d. ArithmeticException

```
class A {
   public static void main (String[] args) {
      Object error = new Error();
      Object runtimeException = new RuntimeException();
      System.out.print((error instanceof Exception) + ",");
      System.out.print(runtimeException instanceof Exception);
}}
```

What is the result of attempting to compile and run the program?

a. Prints: true,false
b. Prints: true,true
c. Prints: false,true
d. Prints: false,false

Given the following code:

```
import java.io.IOException;

public class ExceptionTest {
 public static void main(String[] args) {
  try {
   methodA();
  } catch (IOException e) {
   System.out.println("Caught IO Exception");
  } catch (Exception e) {
   System.out.println("Caught Exception");
  }
 }

 static public void methodA() {
  throw new IOException();
 }
}
```

What is the output ?

a. Program executes normally without printing a message.
b. The output is "Caught Exception".
c. The output is "Caught IO Exception".
d. Code will not compile.

```
public class ExceptionTest {
   public static void main(String[] args)
   {
      try
      {
         ExceptionTest a = new ExceptionTest();
         a.badMethod();
         System.out.println("A");
      }
      catch (Exception e)
      {
         System.out.println("B");
      }
      finally
      {
         System.out.println("C");
      }
   }

   void badMethod()
   {
      throw new Error();
   }
}
```

What is the output?

- a. BC followed by Error exception
- b. Error exception followed by BC
- c. C followed by Error exception
- d. Error exception followed by C

Given:

```
public class TestException {
 public static void main(String... args) {
  try {
   // some piece of code
  } catch (NullPointerException e1) {
   System.out.print("n");
  } catch (RuntimeException e2) {
   System.out.print("r");
  } finally {
   System.out.print("f");
  }
 }
}
```

What is the output if NullPointerException occurs when executing the code in the try block?

- a. nf
- b. nrf
- c. rf
- d. f

Given the following,

```
1. public class MyProgram {
2.    public static void main(String args[]){
3.       try {
4.          System.out.print("Hello world ");
5.       }
6.       finally {
7.          System.out.println("Finally executing ");
8.       }
9.    }
10. }
```

What is the result?

- a. Hello world.
- b. Hello world Finally executing
- c. Nothing. The program will not compile because no exceptions are specified.
- d. Nothing. The program will not compile because no catch clauses are specified.

Which statement is TRUE about catch{} blocks?

- a. A catch{} block need not be present even if there is no finally{} block.
- b. There can only be one catch{} block in a try/catch structure.
- c. The catch{} block for a child exception class must FOLLOW that of a parent exception class.
- d. The catch{} block for a child exception class must PRECEDE that of a parent exception class.

On occurrence of which of the following is it possible for a program to recover?

a. Neither
b. Both errors and exceptions
c. Exceptions
d. Errors

Which statement is TRUE about the try{} block?

a. The statements in a try{} block can only throw one exception type and not several types.
b. It is mandatory for statements in a try{} block to throw at least one exception type.
c. The try{} block can appear after the catch{} blocks.
d. The try{} block can contain loops or branches.

```
class A {A() throws Exception {}} // 1
class B extends A {B() throws Exception {}} // 2
class C extends A {C() {}} // 3
```
Which one of the following statements is true?

a. No compile-time errors.
b. Compile-time error at 1.
c. Compile-time error at 2.
d. Compile-time error at 3.

Given the following,

```
public class MyProgram {
public static void throwit() {
throw new RuntimeException();
}
public static void main(String args[]){
try {
System.out.println("Hello world ");
throwit();
System.out.println("Done with try block ");
}
finally {
System.out.println("Finally executing ");
}
}
}
```
Which answer most closely indicates the behavior of the program?

a. The program will print Hello world, then will print that a RuntimeException has occurred, then will print Done with try block, and then will print Finally executing.
b. The program will print Hello world, then will print that a RuntimeException has occurred, and then will print Finally executing.
c. The program will print Hello world, then will print Finally executing, then will print that a RuntimeException has occurred.
d. The program will not compile.

When is a finally{} block executed?

a. Only when an unhandled exception is thrown in a try{} block.
b. Only when any exception is thrown in a try{} block.
c. Always after execution has left a try catch{} block, no matter for what reason
d. Always just as a method is about to finish.

Given the following:

```
1. class ShapeException extends Exception {}
2.
3. class CircleException extends ShapeException {}
4.
5. public class Circle1 {
6.    void m1() throws CircleException {throw new ShapeException();}
7.
8.    public static void main (String[] args) {
9.       Circle1 circle1 = new Circle1();
10.       int a=1, b=1;
11.
12.       try {circle1.m1(); a--;} catch (CircleException e) {b--;}
13.
14.       System.out.printf("a=%d, b=%d", a, b);
15.    }
16.}
```
What is the expected output?

a. a=1, b=1
b. Compile time error at line 6.
c. a=0, b=1
d. a=1, b=0

Given the following:

```
public class TestDivide {
   public static void main(String[] args) {
      int value=0;
      try {
         int result = 10/value;
      } finally {
         System.out.println("f");
      }
   }
}
```

What is the result ?

- a. Prints only "f" in the output.
- b. Prints an "f" in the output and a runtime error is also displayed.
- c. Only a runtime error is displayed.
- d. Compilation fails since a catch block is not present.

```
class A {
   public static void main (String[] args) {
      Error error = new Error();
      Exception exception = new Exception();
      System.out.print((exception instanceof Throwable) + ",");
      System.out.print(error instanceof Throwable);
}}
```

What is the result of attempting to compile and run the program?

- a. Prints: false,true
- b. Prints: true,true
- c. Prints: true,false
- d. Prints: false,false

Which statement is true?

- a. An overriding method must declare that it throws the same exception classes as the method it overrides.
- b. The main() method of a program cannot declare that it throws checked exceptions.
- c. If an exception is uncaught in a method, the method will terminate and normal execution will resume.
- d. A method declaring that it throws a certain exception class may throw instances of any subclass of that exception class.