1. Which digits, and in what order, will be printed when the following program is run?

```
public class MyClass {
public static void main(String[] args) {
int k=0;
try {
int i = 5/k;
} catch (ArithmeticException e) {
System.out.println("1");
} catch (RuntimeException e) {
System.out.println("2");
return;
} catch (Exception e) {
System.out.println("3");
} finally {
System.out.println("4");
}
System.out.println("5");
}
}
```

Select the one correct answer.

(a) The program will only print 5.

(b) The program will only print 1 and 4, in that order.

(c) The program will only print 1, 2, and 4, in that order.

[✓ (d) The program will only print 1, 4, and 5, in that order.]

(e) The program will only print 1, 2, 4, and 5, in that order.

(f) The program will only print 3 and 5, in that order.

2. Given the following program, which statements are 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");
}
}
}
```

Select the two correct answers.

(a) If run with no arguments, the program will produce no output.

(b) If run with no arguments, the program will print "The end".

(c) The program will throw an ArrayIndexOutOfBoundsException.

(d) If run with one argument, the program will simply print the given argument.

(e) If run with one argument, the program will print the given argument followed by "The end".

3. What will be the result of attempting to compile and run the following program?

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

Select the one correct answer.

(a) The code will fail to compile because the main() method does not declare that it throws RuntimeException in its declaration.

(b) The program will fail to compile because it cannot throw re.

(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.

(e) The program will compile without error and will run and terminate without any output.

4. Which statements are true?

Select the two correct answers.

(a) If an exception is not caught in a method, the method will terminate and normal execution will resume.

(b) An overriding method must declare that it throws the same exception classes as the method it overrides.

(c) The main() method of a program can declare that it throws checked exceptions.

(d) A method declaring that it throws a certain exception class may throw instances of any subclass of that exception class.

(e) finally blocks are executed if, and only if, an exception gets thrown while inside the corresponding try block.

5. Which digits, and in what order, will be printed when the following program is compiled and run?

```
public class MyClass {
public static void main(String[] args) {
        try {
        f();
        } catch (InterruptedException e) {
        System.out.println("1");
        throw new RuntimeException();
        } catch (RuntimeException e) {
        System.out.println("2");
        return;
        } catch (Exception e) {
        System.out.println("3");
        } finally {
        System.out.println("4");
        }
        System.out.println("5");
        }
// InterruptedException is a direct subclass of
Exception.
static void f() throws InterruptedException {
throw new InterruptedException("Time for lunch.");
}
}
```

Select the one correct answer.

(a) The program will print 5.

[✓ (b) The program will print 1 and 4, in that order.]

(c) The program will print 1, 2, and 4, in that order.

(d) The program will print 1, 4, and 5, in that order.

(e) The program will print 1, 2, 4, and 5, in that order.

(f) The program will print 3 and 5, in that order.

6. Which digits, and in what order, will be printed when the following program is run?

```
public class MyClass {
public static void main(String[] args) throws
InterruptedException {
try {
f();
System.out.println("1");
} finally {
System.out.println("2");
}
}
System.out.println("3");
}
// InterruptedException is a direct subclass of
Exception.
static void f() throws InterruptedException {
throw new InterruptedException("Time to go
home.");
}
```

Select the one correct answer.

(a) The program will print 2 and throw InterruptedException.

(b) The program will print 1 and 2, in that order.

(c) The program will print 1, 2, and 3, in that order.

(d) The program will print 2 and 3, in that order.

(e) The program will print 3 and 2, in that order.

(f) The program will print 1 and 3, in that order.

7. What is wrong with the following code?

```
public class MyClass {
public static void main(String[] args) throws A {
try {
f();
} finally {
System.out.println("Done.");
} catch (A e) {
throw e;
}
}
public static void f() throws B {
throw new B();
}
}
class A extends Throwable {}
class B extends A {}
```

Select the one correct answer.

(a) The main() method must declare that it throws B.

(b) The finally block must follow the catch block in the main() method.

(c) The catch block in the main() method must declare that it catches B rather than A.

(d) A single try block cannot be followed by both a finally and a catch block.

(e) The declaration of class A is illegal.

8. What is the minimal list of exception classes that the overriding method f() in the following code must declare in its throws clause before the code will compile correctly?

```
class A {
// InterruptedException is a
//direct subclass of //Exception.
void f() throws
ArithmeticException,
InterruptedException {

div(5, 5);
}
int div(int i, int j) throws
ArithmeticException {
return i/j;
}
}
public class MyClass extends A {
void f() /* throws [...list of
exceptions...] */ {
try {
div(5, 0);
} catch (ArithmeticException e) {
return;
}
throw new
RuntimeException("ArithmeticE
xception was expected.");
}
}
```

Select the one correct answer.

(a) Does not need to specify any exceptions.

(b) Needs to specify that it throws ArithmeticException.

(c) Needs to specify that it throws InterruptedException.

(d) Needs to specify that it throws RuntimeException.

(e) Needs to specify that it throws both ArithmeticException and InterruptedException.

9. What, if anything, would cause the following code not to compile?

```
class A {
void f() throws ArithmeticException {
//...
}
}
public class MyClass extends A {
public static void main(String[] args) {
A obj = new MyClass();
try {
obj.f();
} catch (ArithmeticException e) {
return;
} catch (Exception e) {
System.out.println(e);
throw new RuntimeException("Something wrong here");
}
}
// InterruptedException is a direct subclass of Exception.
void f() throws InterruptedException {
//...
}
```

Select the one correct answer.

(a) The main() method must declare that it throws RuntimeException.

(b) The overriding f() method in MyClass must declare that it throws ArithmeticException, since the f() method in class A declares that it does.

(c) The overriding f() method in MyClass is not allowed to throw InterruptedException, since the f() method in class A does not throw this exception.

(d) The compiler will complain that the catch(ArithmeticException) block shadows the catch(Exception) block.

(e) You cannot throw exceptions from a catch block.

(f) Nothing is wrong with the code, it will compile without errors.

10. Find the below code Snippet and answer:

```
class Base extends Exception {}
class Derived extends Base  {}

public class Main {
  public static void main(String args[]) {
   // some other stuff
   try {
       // Some monitored code
       throw new Derived();
    }
    catch(Base b)     {
       System.out.println("Caught base class exception");
    }
    catch(Derived d)  {
       System.out.println("Caught derived class exception");
    }
  }
}
```

a. Caught base class exception

b. Caught derived class exception

c. Compiler Error because derived is not throwable

d. Compiler Error because base class exception is caught before derived class

11. Find the below code Snippet and answer:

```
class Test
{
    public static void main (String[] args)
    {
        try
        {
            int a = 0;
            System.out.println ("a = " + a);
            int b = 20 / a;
            System.out.println ("b = " + b);
        }

        catch(ArithmeticException e)
        {
            System.out.println ("Divide by zero error");
        }

        finally
        {
            System.out.println ("inside the finally block");
        }
    }
}
```

a. Compile error

b. Divide by zero error

c. a = 0

   Divide by zero error

   inside the finally block

d. a = 0

e. inside the finally block