Encapsulation /Inheriance/Polymorphism

1. QUESTION

```
class X {}
class Y {Y () {}}
class Z {z(int i ) {} }
```

Which class has a default constructor?

[✓ A.] X only
B. Y only
C. Z only
D. X and Y
E. Y and Z
F. X and Z
G. X, Y and Z

2. QUESTION

Which statement is true about the default constructor of a class?

A. It can take arguments.
B. It has private access modifier in its declaration.
C. It can be overloaded.
[✓ D.] The default constructor of a subclass always invokes the no-argument constructor of its superclass.

3. QUESTION

```
public class ComputeSum{
	public int x;
	public int y;
	public int sum;
	public ComputeSum (int nx, int ny){
		x = nx; y =ny;
		updateSum();
	}
	public void setX(int nx){
		x = nx;
		updateSum();
	}
	public void setY(int ny){
		x = ny;
		updateSum();
	}
	void updateSum(){
		sum = x + y;
	}
}
```

This class needs to protect an invariant on the sum field. Which three members must have the private access modifier to ensure that this invariant is maintained?

- A. The x field
- B. The y field
- [✓ C.] The sum field
- [✓ D.] The ComputerSum ( ) constructor
- E. The setX ( ) method
- [✓ F.] The setY ( ) method

4. QUESTION

```
public class SampleClass {
    public static void main(String[] args) {
        AnotherSampleClass asc = new AnotherSampleClass();
        SampleClass sc = new SampleClass();
        // TODO code application logic here
    }}
class AnotherSampleClass extends SampleClass {
}
```

Which statement, when inserted into line "// TODO code application logic here ", is valid change?

A. asc = sc ;
[✓ B.] sc = asc ;
C. asc = (object) sc ;
D. asc = sc.clone () ;

5. QUESTION

```
public class SampleClass {
    public static void main(String[] args) {
        AnotherSampleClass asc = new AnotherSampleClass();
        SampleClass sc = new SampleClass();
        sc = asc;
        System.out.println("sc: " + sc.getClass());
        System.out.println("asc: " + asc.getClass());
        }
    }
class AnotherSampleClass extends SampleClass {
}
```

What is the result?

A. sc: class.Object asc: class.AnotherSampleClass
B. sc: class.SampleClass asc: class.AnotherSampleClass
C. sc: class.AnotherSampleClass asc: class.SampleClass
[✓ D.] sc: class.AnotherSampleClass asc: class.AnotherSampleClass

6. QUESTION

```
public class SampleClass {
    public static void main(String[] args) {
        SampleClass sc, scA, scB;
        sc = new SampleClass();
        scA = new SampleClassA();
        scB = new SampleClassB();
        System.out.println("Hash is : " + sc.getHash() + ", " + scA.getHash() + ", " + scB.getHash());
    }
    public int getHash() {
        return 111111;
    }
}
class SampleClassA extends SampleClass {
    public long getHash() {
        return 44444444;
    }
}
class SampleClassB extends SampleClass {
    public long getHash() {
        return 999999999;
    }
}
```

What is the result?

[✓ A.] Compilation fails
B. An exception is thrown at runtime
C. There is no result because this is not correct way to determine the hash code
D. Hash is: 111111, 44444444, 999999999

7. QUESTION

```
class Base {
    public static void main (String[] args) {
      System.out.println("Base" + args[2]);
    }
}
public class Sub extends Base {
    public static void main (String[] args) {
      System.out.println("Overriden" + args[1]);
}
```

And the commands:

```
javac Sub.java
java Sub 10 20 30
```

What is the result?

A. Base 30
[✓ B. Overridden 20]
C. Overridden 20 Base 30
D. Base 30 Overridden 20

8. QUESTION

Given the class definitions:

```
class Shape { }
class Square extends Shape { }
```

Given the variable declarations:

```
Shape shape1 = null;
Square square1 = null;
```

Which four compile?

[✓ A.] Shape1 = (Square) new Square();

[✓ B.] Shape1 = new Square();

[✓ C.] Square1= (Square) new Shape();

D. Square1= new Shape();

[✓ E.] Square1= new Square();
shape1 = square1;

F. Shape1 = new Shape();
square1 = shape1;

9. QUESTION

Given:

```
class X {
    public void mX() {
        System.out.println("Xm1");
    }
}
class Y extends X {
    public void mX() {
        System.out.println("Xm2");
    }
    public void mY() {
        System.out.println("Ym");
    }
}
public class Test {
    public static void main(String[] args) {
        X xRef = new Y();
        Y yRef = (Y) xRef;
        yRef.mY();
        xRef.mX();
    }
}
```

What is the result?

[✓ A.] Ym Xm2
B. Ym Xm1
C. Compilation fails.
D. Xm1 Ym

10. QUESTION

Given:

```
class X{
    int x1, x2, x3;
}
class Y extends X{
    int y1;
    Y(){
        x1 = 1;
        x2 = 2;
        y1 = 10;
    }
}
class Z extends Y{
    int z1;
    Z(){
        x1 = 3;
        y1 = 20;
        z1 = 100;
    }
}
public class Test3{
    public static void main(String[] args){
        Z obj = new Z();
        System.out.println(obj.x3 + "," + obj.y1 + "," +obj.z1);
    }
}
```

Which constructor initializes the variable x3?

A. Only the default constructor of class X

B. Only the no-argument constructor of class Y

[✓ C.] Only the no-argument constructor of class Z

D. Only the default constructor of object class

11. QUESTION

```
class Base {
    // insert code here
}
public class Derived extends Base{
    public static void main(String[] args) {
        Derived obj = new Derived();
        obj.setNum(3);
        System.out.println("Square = " + obj.getNum() * obj.getNum());
    }
}
```

Which two options, when inserted independently inside class Base, ensure that the class is being properly encapsulated and allow the program to execute and print the square of the number?

[✓ A.] private int num;
public int getNum() {return num;}
public void setNum(int num) {this.num = num;}

B. public int num;
protected public int getNum() {return num;}
protected public void setNum(int num) {this.num = num;}

C. private int num;
public int getNum() {return num;}
private void setNum(int num) {this.num = num;}

[✓ D.] protected int num;
public int getNum() {return num;}
public void setNum(int num) {this.num = num;}

E. protected int num;
private int getNum() {return num;}
public void setNum(intnum) {this.num = num;}

12. QUESTION

```
public class Circle {
    double radius;
    public double area;
    public Circle (double r) { radius = r;}
    public double getRadius() {return radius;}
    public void setRadius(double r) { radius = r;}
    public double getArea() { return /* ??? */;}
}
class App {
    public static void main(String[] args) {
        Circle c1 = new Circle(17.4);
        c1.area = Math.PI * c1.getRadius() * c1.getRadius();
    }
}
```

The class is poorly encapsulated. You need to change the circle class to compute and return the area instead. Which two modifications are necessary to ensure that the class is being properly encapsulated?

A. Remove the area field.

[✓ B.] Change the getArea( ) method as follows: public double getArea ( ) { return Math.PI * radius * radius; }

C. Add the following method:
public double getArea ( )
{ area = Math.PI * radius * radius; }

[✓ D.] Change the cacess modifier of the SerRadius ( ) method to be protected.

13. QUESTION

Which three statements are benefits of encapsulation?

[✓ A.] Allows a class implementation to change without changing the clients

[✓ B.] Protects confidential data from leaking out of the objects

C. Prevents code from causing exceptions

[✓ D.] Enables the class implementation to protect its invariants

E. Permits classes to be combined into the same package

F. Enables multiple instances of the same class to be created safely

1. QUESTION

Which two statements are true?

[✓ A.] An abstract class can implement an interface.
B. An abstract class can be extended by an interface.
C. An interface CANNOT be extended by another interface.
D. An interface can be extended by an abstract class.
[✓ E.] An abstract class can be extended by a concrete class.
F. An abstract class CANNOT be extended by an abstract class.

2. QUESTION

Which one is a valid abstract class?

A. public abstract class Car { protected void accelerate(); }
[✓ B.] public interface Car { protected abstract void accelerate(); }
C. public abstract class Car { protected final void accelerate(); }
D. public abstract class Car { protected abstract void accelerate(); }
E. public abstract class Car { protected abstract void accelerate() { //more car can do }}

3. QUESTION

Class A {}

Class B { }

Interface X { }

Interface Y { }

Which two definitions of class C are valid?

[✓ A.] Class C extends A implements X { }
B. Class C implements Y extends B { }
C. Class C extends A, B { }
D. Class C implements X, Y extends B { }
[✓ E.] Class C extends B implements X, Y { }

4. QUESTION

Which code fragment is illegal?

A. class Base1 {
   abstract class Abs1 { ...}
   }

B. abstract class Abs1 {
   void doit () { }
   }

C. class Basel {
   abstract class Abs1 extends Basel {...}
   }

[✓ D.] abstract int var1 = 89;

5. QUESTION

interface Pet{}

class Dog implements Pet{}

public class Beagle extends Dog{}

Which three are valid?

[✓ A.] Pet a = new Dog();

B. Pet b = new Pet();

C. Dog f = new Pet();

[✓ D.] Dog d = new Beagle();

[✓ E.] Pet e =new Beagle() ;

F. Beagle c = new Dog();

6. QUESTION

```
public abstract class wow {
    private int wow;
    public wow (int wow) {
        this.wow = wow;
    }
    public void wow () {
    }
    private void wowza () {
    }
}
```

What is the true about the class Wow?

[✓ A.] It compiles without error.
B. It does't compile because an abstract class cannot have private methods.
C. It does't compile because an abstract class cannot have instance variables.
D. It does't compile because an abstract class must have at least one abstract method.
E. It does't compile because an abstract class must have a constructor with no arguments.

7. QUESTION

```
abstract class X {
    public abstract void methodX();
}
interface Y {
    public void methodY();
}
```

Which two code fragments are valid?

A. class Z extends X implements Y {
```
    public void methodZ() { }
}
```

[✓ B.] abstract class Z extends X implements Y {
```
    public void methodZ() { }
}
```

C. class Z extends X implements Y {
```
    public void methodX() { }
}
```

[✓ D.] abstract class Z extends X implements Y {
```
}
```

E. class Z extends X implements Y {
```
    public void methodY() { }
}
```

8. QUESTION

```
abstract class A1 {
    public abstract void m1();
    public void m2() {
        System.out.println("Green");
    } }
abstract class A2 extends A1 {
    public abstract void m3();
    public void m1() {
        System.out.println("Cyan");
    }
    public void m2() {
        System.out.println("Blue");
    } }
public class A3 extends A2 {
    public void m1() {
        System.out.println("Yellow");
    }
    public void m2() {
        System.out.println("Pink");
    }
    public void m3() {
        System.out.println("Red");
    }
    public static void main(String[] args) {
        A2 tp = new A3();
        tp.m1();
        tp.m2();
        tp.m3();
    } }
```

What is the result?

[✓ A. Yellow Pink Red]
B. Cyan Blue Red
C. Cyan Green Red
D. Compilation fails

9. QUESTION

```
public class X implements Z {
    public String toString() {
        return "I am X";
    }
    public static void main(String[] args) {
        Y myY = new Y();
        X myX = myY;
        Z myZ = myX;
        System.out.println(myZ);
    }
}
class Y extends X {
    public String toString() {
        return "I am Y";
    }
}
interface Z {}
```

What is the reference type of myZ and what is the type of the object it references?

A. Reference type is Z; object type is Z.

B. Reference type is Y; object type is Y.

[✓ C.] Reference type is Z; object type is Y.

D. Reference type is X; object type is Z.

10. QUESTION

Given:

```
public class Bark {
    // Insert code here - Line 5
        public abstract void bark(); // Line 6
    } // Line 7
}// Line 8
    // Insert code here - Line 9
    public void bark() {
        //System.out.println("woof");
    }
}
```

What code should be inserted?

A. 5. class Dog {
   9. public class Poodle extends Dog {

B. 5. abstract Dog {
   9. public class poodle extends Dog {

[✓ C.] 5. abstract class Dog {
   9. public class Poodle extends Dog {

D. 5. abstract Dog {
   9. public class Poodle implements Dog {

E. 5. abstract Dog {
   9. public class Poodle implements Dog {

F. 5. abstract class Dog {
   9. public class Poodle implements Dog {

11. QUESTION

Given:

```
public class X implements Z {
      public String toString() {
            return "X ";
      }
      public static void main(String[] args) {
            Y myY = new Y();
            X myX = myY;
            Z myZ = myX;
            System.out.print(myX);
            System.out.print((Y) myX);
            System.out.print(myZ);
      }
}
class Y extends X {
      public String toString() {
            return "Y ";
      }
}
interface Z {}
```

What is the result?

A. X X X

B. X Y X

C. Y Y X

D. Y Y Y

12. QUESTION

Given:

```
class Star {
          public void doStuff() { System.out.println("Twinkling Star");
     }
}
interface Universe {
     public void doStuff();
}
class Sun extends Star implements Universe {
     public void doStuff() {
          System.out.println("Shining Sun");
     }
}
     public class Bob {
     public static void main(String[] args) {
     Sun obj2 = new Sun();
     Star obj3 = obj2;
     ((Sun) obj3).doStuff();
     ((Star) obj2).doStuff();
     ((Universe) obj2).doStuff();
     }
}
```

What is the result?

A. Shining Sun Shining Sun Shining Sun

B. Shining Sun Twinkling Star Shining Sun

C. Compilation fails.

D. A ClassCastException is thrown at runtime.

13. QUESTION

Given:

```
public abstract class Shape{
    private int x;
    private int y;
    public abstract void draw();
    public void setAnchor(int x,int y){
     this.x = x;
     this.y = y;
     }}
```

Which two classes use the class correctly?

A. public class Circle implements Shape{

```
        private int radius; }
```

B. public abstract class Cricle extends Shape{

```
        private int radius; }
```

C. public class Cricle extends Shape{

```
       private int radius;
       public void draw();}
```

D. public abstract class Circle implements Shape{

```
        private int radius;
        public void draw();}
```

E. public class Cricle extends Shape{

```
        private int radius;
        public void draw(){
        /* code here */
        }}
```

F. public abstract class Circle implements Shape{

```
        private int radius;
        public void draw(){
        /* code here */
        }}
```

1. QUESTION

```
public class Main {
    public static void main (String[] args) {
        doSomething();
    }
    private static void doSomething() {
        doSomeThingElse();
    }
    private static void doSomeThingElse() {
        throw new Exception();
    }
}
```

Which approach ensures that the class can be compiled and run?

A. Put the throw new Exception() statement in the try block of try catch
B. Put the doSomethingElse() method in the try block of a try catch
C. Put the doSomething() method in the try block of a try catch
D. Put the doSomething() method and the doSomethingElse() method in the try block of a try catch

2. QUESTION

The catch clause argument is always of type__________.

A. Exception
B. Exception but not includ RuntimeException
C. Throwable
D. RuntimeException
E. CheckedException
F. Error

3. QUESTION

public Class Test { }

Which two packages are automatically imported into the java source file by the java compiler?

A. Java.lang
B. Java.awt
C. Javax.net
D. Java.*
E. The package with no name

4. QUESTION

The protected modifier on a field declaration within a public class means that the field ____________

A. Cannot be modified
B. Can be read but not written from outside the class
C. Can be read and written from this class and its subclasses only within the same package
D. Can be read and written from this class and its subclasses defined in any package

5. QUESTION

Which two are Java Exception classes?

A. SercurityException
B. DuplicatePathException
C. IllegalArgumentException
D. TooManyArgumentsException

6. QUESTION

An unchecked exception occurs in a method dosomething()

Should other code be added in the dosomething() method for it to compile and execute?

A. The Exception must be caught

B. The Exception must be declared to be thrown

C. The Exception must be caught or declared to be thrown

D. No othercode needs to be added.

7. QUESTION

A method doSomething () that has no exception handling code is modified to trail a method that throws a checked exception. Which two modifications, made independently, will allow the program to compile?

A. Catch the exception in the method doSomething()

B. Declare the exception to be thrown in the doSomething() method signature

C. Cast the exception to a RunTimeException in the doSomething() method

D. Catch the exception in the method that calls doSomething()

8. QUESTION

```
interface SampleClosable {
  public void close () throws java.io.IOException;
}
```

Which three implementations are valid?

A.
```
public class Test implements SampleCloseable{
  public void close() throws java.io.IOException{
        //do something
  }
}
```

B.
```
public class Test implements SampleCloseable{
  public void close() throws Exception{
        //do something
  }
}
```

C.
```
public class Test implements SampleCloseable{
  public void close() throws java.io.FileNotFoundException{
        //do something
  }
}
```

D.
```
public class Test extends SampleCloseable{
  public void close() throws java.io.IOException{
  //do something
  }
}
```

E.
```
public class Test implements SampleCloseable{
  public void close() {
        //do something
  }
}
```

9. QUESTION

```
public class Two {
    public static void main(String[] args) {
        try {
            doStuff();
            System.out.println("1");
        }
        catch (RuntimeException e)   {
            System.out.println("2");
        }
    }
public static void doStuff() {
    if (Math.random() > 0.5)
    throw new RuntimeException();
        doMoreStuff();
    System.out.println("3 ");
    }
 public static void doMoreStuff() {
    System.out.println("4");
        }
    }
```

Which two are possible outputs?

A. 2

B. 4 3 1

C. 1

D. 1 2

10. QUESTION

Given the classes:

* AssertionError
* ArithmeticException
* ArrayIndexOutofBoundsException
* FileNotFoundException
* IllegalArgumentException
* IOError
* IOException
* NumberFormatException
* SQLException

Which option lists only those classes that belong to the unchecked exception category?

A. ArithmeticException, ArrayIndexOutOfBoundsException, ArithmeticException

B. AssertionError, IOError, IOException

C. ArithmeticException, FileNotFoundException, NumberFormatException

D. FileNotFoundException, IOException, SQLException

E. ArrayIndexOutOfBoundException, IllegalArgumentException, FileNotFoundException

11. QUESTION

Given:

```
package p1;
public interface DoInterface {
    void method1(int n1); // line n1
}
package p3;
import p1.DoInterface;
class DoClass implements DoInterface {
    public DoClass(int p1) { }
    public void method1(int p1) { } // line n2
    private void method2(int p1) { } // line n3
}
public class Test {
    public static void main(String[] args) {
        DoInterface doi= new DoClass(100); // line n4
        doi.method1(100);
        doi.method2(100);
    }
}
```

Which change will enable the code to compile?

A. Adding the public modifier to the declaration of method1 at line n1

B. Removing the public modifier from the definition of method1 at line n2

C. Changing the private modifier on the declaration of method 2 public at line n3

D. Changing the line n4 DoClass doi = new DoClass ( );

12. QUESTION

Given:

```
public class Main {
    public static void main(String[] args) throws Exception {
        doSomething();
    }
    private static void doSomething() throws Exception {
        System.out.println("Before if clause");
        if (Math.random() > 0.5) {
            throw new Exception();
            }
System.out.println ("After if clause");
    }
}
```

Which two are possible outputs?

A. Before if clause
   Exception in thread "main" java.lang.Exception
   At Main.doSomething (Main.java:8)
   At Main.main (Main.java:3)

B. Before if clause
   Exception in thread "main" java.lang.Exception
   At Main.doSomething (Main.java:8)
   At Main.main (Main.java:3)
   After if clause

C. Exception in thread "main" java.lang.Exception
   At Main.doSomething (Main.java:8)
   At Main.main (Main.java:3)

D. Before if clause
   After if clause

13. QUESTION

Given:

```
import java.io.IOException;
public class Y {
      public static void main(String[] args) {
            try {
                  doSomething();
      }
      catch (RuntimeException e) {
            System.out.println(e);
      }
}
      static void doSomething() {
            if (Math.random() > 0.5)
            throw new IOException();
            throw new RuntimeException();
      }
}
```

Which two actions, used independently, will permit this class to compile?

A. Adding throws IOException to the main() method signature

B. Adding throws IOException to the doSoomething() method signature

C. Adding throws IOException to the main() method signature and to the dosomething() method

D. Adding throws IOException to the dosomething() method signature and changing the catch argument to IOException

E. Adding throws IOException to the main() method signature and changing the catch argument to IOException

14. QUESTION

Given the following four Java file definitions:

```
//Foo.java
package facades;
public interface Foo { }

//Boo.java
package facades;
public interface Boo extends Foo { }

// Woofy.java
package org.domain
// line n1
public class Woofy implements Boo, Foo { }

// Test.java
package.org;
// line n2
public class Test {
public static void main(String[] args) {
Foo obj=new Woofy();}
}
```

Which set modifications enable the code to compile and run?

A. At line n1, Insert: import facades; At line n2, insert: import facades; import org.domain;

B. At line n1, Insert: import facades.*; At line n2, insert: import facades; import org.*;

C. At line n1, Insert: import facades.*; At line n2, insert: import facades.Boo; import org.*;

D. At line n1, Insert: import facades.Foo, Boo; At line n2, insert: import org.domain.Woofy;

E. At line n1, Insert: import facades.*;At line n2, insert: import facades.*; import org.domain.Woofy;

15. QUESTION

Given:

```
Test.java
public class Test {
      public static void main (String[] args) {
            Integer num = Integer.parseInt (args[1]);
         System.out.println (“Number is : ” + num);
      }
 }
```

And the commands:

```
Javac Test.java Java Test
```

What is the result?

A. Number us : 12345
B. Null Pointer Exception is thrown at runtime
C. Number Format Exception is thrown at runtime
D. An Array Index Out Of Bound Exception is thrown at runtime.

16. QUESTION

Given the code fragment:

```
interface SampleCloseable {
      public void close () throws java.io.IOException;
}
```

Which three implementations are valid?

A.
```
public class Test implements SampleCloseable {
  public void close() throws java.io.IOException{
        // do something
  }
}
```

B.
```
public class Test implements SampleCloseable {
  public void close() throws Exception {
        // do something
  }
}
```

C.
```
public class Test implements SampleCloseable {
  public void close() throws java.io.FileNotFoundException {
        // do something
  }
}
```

D.
```
public class Test extends SampleCloseable{
  public void close() throws java.IO.IOException {
        // do something
  }
}
```

E.
```
public class Test implements SampleCloseable {
  public void close() {
        // do something
  }
}
```

17. QUESTION

Given:

```
import java.io.Error;
public class TestApp{
      public static void main(String[] args){
            TestApp t = new TestApp();
      try{
            t.doPrint();
            t.doList();
      }catch(Exception e2){
         System.out.println("Caught " + e2);
      }
   }
   public void doList() throws Exception{
      throw new Error("Error");
   }
   public void doPrint() throws Exception{
      throw new RuntimeException("Exception");
   }
}
```

what is the result?

A. Caught java.lang.RuntimeException: Exception
   Exception in thread "main" java.lang.Error: Error
   at TestApp.doList(TestApp.java: 14)
   at TestApp.main(TestApp.java: 6)
B. Exception in thread "main" java.lang.Error: Error
   at TestApp.doList(TestApp.java: 14)
   at TestApp.main(TestApp.java: 6)
C. Caught java.lang.RuntimeException: Exception
   Caught java.lang.Error: Error
D. Caught java.lang.RuntimeException: Exception

18. QUESTION

Which three are advantages of the Java exception mechanism?

A. Improves the program structure because the error handling code is separated from the normal program function

B. Provides a set of standard exceptions that covers all the possible errors

C. Improves the program structure because the programmer can choose where to handle exceptions

D. Improves the program structure because exceptions must be handled in the method in which they occurred

E. allows the creation of new exceptions that are tailored to the particular program being

19. QUESTION

Which two statements correctly describe checked exception?

A. These are exceptional conditions that a well-written application should anticipate and recover from.

B. These are exceptional conditions that are external to the application, and that the application usually cannot anticipate or recover from.

C. These are exceptional conditions that are internal to the application, and that the application usually cannot anticipate or recover from.

D. class that is a subclass of RuntimeException and Error is categorized as checked exception.

E. Every class that is a subclass of Exception, excluding RuntimeException and its subclasses, is categorized as checked exception.

20. QUESTION

Given:

```
public class Test{
      static void disResult(int[] num){
            try{
                  System.out.println(num[1] / (num[1] – num[2]));
            } catch(ArithmeticException e) {
                  System.out.println(“first exception”);
            }
            System.out.println(“Done”);
      }
      public static void main(String[] args){
            try{
                  int[] arr = (100, 100);
            } catch(IllegalArgumentException e) {
                  System.err.println(“second exception”);
            } catch(Exception e) {
                  System.err.println(“third exception”);
            }
      }
}
```

What is the result?

A. 0 Done
B. First Exception Done
C. Second Exception
D. Done Third Exception
E. Third Exception

21. QUESTION

Given:

```
class MarksOutOfBoundsException extends IndexOutOfBoundsException {}
public class GradingProcess {
    void verify(int marks) throws IndexOutOfBoundsException {
    if (marks > 100) {
    throw new MarksOutOfBoundsException();
    }
    if (marks > 50) {
    System.out.print("Pass");
    }
    else {
    System.out.print("Fail");
    }
    }
    public static void main(String[] args) {
        int marks = Integer.parseInt(args[2]);
        try {
            new GradingProcess().verify(marks); }
        catch (Exception e) {
        System.out.print(e.getClass());
        }
    }
}
```

And the command line invocation:

```
java  Grading Process   89   50   104
```

What is the result?

A. Pass
B. Fail
C. class MarksOutOfBoundsException
D. class IndexOutOfBoundsException
E. class Exception

22. QUESTION

Given:

```
#1
package handy.dandy; public class KeyStroke {
      public void typeExclamation() {
            System.out.println("!")
      }
}
#2
package handy; /* Line 1 */
public class greet { /* Line 2 */
      public static void main(String[] args) { /* Line 3 */
            String greeting = "Hello"; /* Line 4 */
            System.out.print(greeting); /* Line 5 */
            Keystroke stroke = new Keystroke; /* Line 6 */
            stroke.typeExclamation(); /* Line 7 */
      } /* Line 8 */
} /* Line 9 */
```

What three modifications, made independently, made to class greet, enable the code to compile and run?

A. Line 6 replaced with handy.dandy.Keystroke stroke = new KeyStroke ( );
B. Line 6 replaced with handy.*.KeyStroke = new KeyStroke ( );
C. Line 6 replaced with handy.dandy.KeyStroke Stroke = new handy.dandy.KeyStroke();
D. import handy.*; added before line 1
E. import handy.dandy.*; added after line 1
F. import handy.dandy.KeyStroke; added after line 1
G. import handy.dandy.KeyStroke.typeException(); added before line 1

23. QUESTION

Given:

```
public class Test {
    public static void main(String[] args) {
        int ax = 10, az = 30; int aw = 1, ay = 1;
        try {
            aw= ax % 2; ay = az / aw;
        } catch (ArithmeticException e1){
            System.out.println("Invalid Divisor");
        } catch(Exception e2) {
            aw = 1;
            System.out.println("Divisor Changed");
        }
        ay = az /aw; // Line 14
        System.out.println("Succesful Division " + ay);
    }
}
```

What is the result?

A. Invalid Divisor Divisor Changed Successful Division 30

B. Invalid Divisor Successful Division 30

C. Invalid Divisor Exception in thread "main" java.lang.ArithmeticException: / by zero at test.Teagle.main(Teagle.java:14)

D. Invalid Divisor Exception in thread "main" java.lang.ArithmeticException: / by zero at test.Teagle.main(Teagle.java:14) Successful Division 1 C

24. QUESTION

Given:

```
public class x{
    public static void main (String [] args){
        String theString = "Hello World";
        System.out.println(theString.charAt(11));
    }
}
```

What is the result?

A. There is no output
B. d is output
C. A String Index Out Of Bounds Exception is thrown at runtime
D. An ArrayIndexOutOfBoundsException is thrown at runtime
E. A NullPointException is thrown at runtime
F. A StringArrayIndexOutOfBoundsException is thrown at runtime