**CORE JAVA EXERCISE**

**Problem Statement:**

Find out whether the following file will compile. If it does not compile, how you would fix it?

 public static void main(String[ ] args) {

      int x = 5;

      while (x > 1) {

          x = x + 1;

          if (x < 3) {

               System.out.println(“small x”);

          }

      }

 }

Problem Statement:

Find out whether the following file will compile. If it does not compile, how you would fix it?

                                                          class Digit {

      public static void main(String[ ] args) {

                                   int x = 1;

                                   while (x < 10) {

                              if (x > 3) {

                                   System.out.println(“big x”);

                             }

                        }

                 }

            }

Problem Statement:

Find out whether the following file will compile. If it does not compile, how you would fix it?

                                                          class Loop {

                  int x = 5;

                             while (x  > 1) {

                                   x = x - 1;

                        if (x < 3) {

                              System.out.println(“small x”);

                        }

                   }

             }

**Problem Statement:**

Personalize the Hello World program with your name so that it tells you Hello rather than the somewhat generic "World."

**Problem Statement:**

Write a program that produces the following output:

Hello World!

It's been nice knowing you.

Goodbye world!

**Problem Statement:**

Write a program that draws the following figures one above the other.

\* \* \* \* \*             \*

\* \* \* \* \*            \* \*

\* \* \* \* \*           \* \* \*

\* \* \* \* \*         \* \* \* \* \*

Now modify it to draw them next to each other like earlier.

**Problem Statement:**

Write a program that prints all the integers between zero and 36.

**Problem Statement:**

What does the following program print?

// This is the Hello Rank program in Java

class HelloRank {

      public static void main (String args[ ]) {

                    String name = "Rank";

                    /\* Now let's say hello \*/

                   System.out.println("Hello + name");

                    }

}

**Problem Statement:**

What is wrong with this program?

// This is the Hello program in Java

class Hello {

      public static void main (String args[ ]) {

                    int i;

                    System.out.print("Hello ");       // Say Hello

               i = 0;                               // Initialize loop counter

                       while (i <= args.length) {    // Test and Loop

                    System.out.print(args[i] + " ");

                     i = i + 1;                     // Increment Loop Counter

                     }

                        System.out.println();             // Finish the line

         }

}

**Problem Statement:**

What is the output of the following program?

import java.util.\*;  
public class Area {  
      public static void main(String[ ] args){  
          double a;  
    double r;  
    final double pi = Math.PI;  
  
    r = 1.0;  
    a = pi \* r \* r;  
    display(r,a);  
  
    r = 1.5;  
    a = pi \* r \* r;  
    display(r,a);  
  
    r = 2.0;  
     a = pi \* r \* r;  
    display(r,a);  
      }//end main  
      //-------------------------------------------//  
      static void display(double r, double a){  
           System.out.println("For radius = " + r +  
                                ", area = " + a);  
      }//end print  
}//end Area class

**Problem Statement:**

There are exactly 2.54 centimeters to an inch. Write a program that takes a number of inches from the command line and converts it to centimeters.

**Problem Statement:**

Write a program that reads two numbers from the command line, the number of hours worked by an employee and their base pay rate. Then output the total pay due

**Problem Statement:**

What is the output of the following program?

class Hexy {

     public static void main (String[] args)    {

         Integer i = 42;

         String s = (i<40)?"life"sadi>50)?"universe":"everything";

         System.out.println(s);

     }

}

1. null
2. life
3. universe
4. everything
5. Compilation fails
6. An exception is thrown at runtime.

**Problem Statement:**

Given:

1. class Example {

2.     public static void main(String[] args) {

3.        Short s = 15;

4.        Boolean b;

5.        // insert code here

6.    }

7.}

Which, inserted independently at line 5, will compile? (Choose all that apply.)

1. b  =  (Number instanceof s);
2. b  =  (s instanceof Short);
3. b  =  s.instanceof (Short);
4. b  =  (s.instanceof Number);
5. b  =  s.instanceof (Object);
6. b  =  (s instanceof String);

**Problem Statement:**

What is the output of the following program?

class TryIt {

    public static void main(String[] args) {

        Integer x = 0;

        Integer y = 0;

        for(Short z = 0; z < 5; z++)

            if ((++x > 2) || (++y > 2))

                x++;

        System.out.println(x + " " + y);

    }

}

**Problem Statement:**

What is the output of the following program?

class Titanic {

    public static void main(String[] args)     {

        Boolean b1 = true;

        Boolean b2 = false;

        Boolean b3 = true;

        if ((b1 & b2) | (b2 & b3) & b3)

            System.out.println("alpha ");

        if ((b1 = false) | (b1 & b3) | (b1 | b2))

            System.out.println("beta ");

    }

}

a)       beta

b)      alpha

c)       alpha beta

d)      Compilation fails.

e)       No output is produced.

f)        An exception is thrown at runtime.

**Problem Statement:**

Place the fragments into the code to produce the output 33. Note, you must use each fragment exactly once.

Code:

            class Incr {

           public static void main(String[] args) {

                Integer x = 7;

          int y = 2;

          x    \_\_\_  \_\_\_ ;

          \_\_\_ \_\_\_ \_\_\_ ;

          \_\_\_ \_\_\_ \_\_\_ ;

          \_\_\_ \_\_\_  \_\_\_ ;

          System.out.println(x);

            }

              }

FRAGMENTS:

|  |  |  |  |
| --- | --- | --- | --- |
| y | y | y | y |
| y | x | x |  |
| -= | \*= | \*= | \*= |

**Problem Statement:**

Given the following program:

1. class Maybe {

2.     public static void main(String[] args) {

3.         boolean b1 = true;

4.         boolean b2 = false;

5.         System.out.println(!false ^ false);

6.         System.out.println(" " + (!b1 & (b2 = true)));

7.         System.out.println(" " + (b2 ^ b1));

8.     }

9. }

Which are true?

a)       Line 5 produces true.

b)      Line 5 produces false.

c)       Line 6 produces true.

d)      Line 6 produces false.

e)       Line 7 produces true.

f)        Line 7 produces false.

**Problem Statement:**

What is the output of the following program?

1. class Dims {

2.     public static void main(String[ ] args) {

3.        int[ ][ ] a = {{1,2},{3,4}};

4.        int [ ] b= (int[ ]) a[1];

5.        Object o1 = a;

6.        int[ ][ ] a2 = (int[ ][ ]) o1;

7.        int[ ] b2 = (int[ ]) o1;

8.        System.out.println(b[1]);

9.    }

10. }

**Problem Statement:**

What is the output of the following program?

1. class Zippy {

2.     String[ ] x;

3.     int[ ] a[ ] = {{1,2}, {1}};

4.     Object c = new long[4];

5.     Object[ ] d = x;

6. }

**Problem Statement:**

What is the output of the following program?

1. class InitError {

2.     static int[ ] x = new int[4];

3.     static { x[4] = 5;}

4.     public static void main(String[ ] args) { }

5. }

a)       Compilation fails due to an error on line 3.

b)      An Exception is thrown at runtime.

c)       No exception occurs during the execution.

d)      None of the above.

**Problem Statement:**

What is the output of the following program?

class FloatWrapper {

    public static void main(String[] args) {

        float f2 = new Float(3.14f);  // make a new wrapper object

        short s = f2.shortValue();         // convert f2’s value to a short primitive

        System.out.println(s);

    }

}

A.     3.14

B.     4

C.     3

D.    None of the above

**Problem Statement:**

What is the output of the following program?

class LongWrapper {

    public static void main(String[] args)     {

        long L2 = Long.parseLong("101010", 2);      // binary String to a primitive

        System.out.println("L2 = " + L2);

        Long L3 = Long.valueOf("101010", 2);        // binary String to a Long object

        System.out.println("L3 value = " + L3);

    }

}

1. 42, 42
2. 101010, 42
3. 42, 101010
4. 101010, 101010

**Problem Statement:**

Given the following program:

1. class Convert {

2.     public static void main(String[] args) {

3.         Long xL = new Long(456L);

4.         long x1 = Long.valueOf("123");

5.         Long x2 = Long.valueOf("123");

6.         long x3 = xL.longValue();

7.         Long x4 = xL.longValue();

8.         Long x5 = Long.parseLong("456");

9.         long x6 = Long.parseLong("123");

10.    }

11. }

Which will compile using Java 5, but will *not* compile using Java 1.4? (Choose all that apply.)

A.     Line 4

B.     Line 5

C.     Line 6

D.    Line 7

E.     Line 8

F.      Line 9

**Problem Statement:**

What is the output of the following program?

            class AutoBox1 {

          public static void main(String[ ] args) {

              Integer  iObj = 100;

              Double dObj = 98.6;

               dObj = dObj + iObj;

               System.out.println(“dObj after expression: ”  +  dObj);

         }

     }

**Problem Statement:**

What is the output of the following program?

            class AutoBox2 {

         public static void main(String[ ] args) {

             // Autobox/unbox a boolean

             Boolean b = true;

             // Below, b is auto-unboxed when used in

             // a conditional expression, such as an if.

             if (b)   System.out.println(“b is true”);

             // Autobox/unbox a char.

             Character ch = ‘x’   ;    // box a char

             char ch2 = ch;       // unbox a char

             System.out.println(“ch2 is ”  + ch2);

        }

     }

**Problem Statement:**

What is the output of the following program?

            class UnboxingError {

         public static void main(String[ ] args) {

             Integer iObj = 1000;      // autobox the value 1000

             int num = iObj.byteValue(); // manually unbox as byte !!!

             System.out.println(num);

         }

     }

**Problem Statement:**

Given the following program:

class IfTest {

    public static void main(String[ ] args) {

         boolean boo = false;

         if (boo = true) {  }

    }

}

Which of the following is true?

1. The code compiles and runs fine, and the if test fails because boo is false.
2. The code won’t compile because you are using an assignment (=) rather than an equality test (==).
3. The code compiles and runs fine and the if test succeeds because boo is SET to true (rather than TESTED for true) in the if argument!

**Problem Statement:**

What is the output of the following program?

class ElseIfTest {

    public static void main(String[ ] args) {

        int x = 1;

        if ( x == 3) { }

        else if  (x < 4) { System.out.println(“<4”); }

        else if  (x < 2) { System.out.println(“<2”); }

        else { System.out.println(“else”); }

    }

}

1. 3
2. <4
3. <2
4. <3

**Problem Statement:**

Given the following program. Look for any violation of the rules for switch and case arguments.

class SwitchTest {

    public static void main(String[ ] args) {

        switch (x) {

            case 0 {

                 y = 7;

            }

        }

       switch (x) {

           0: { }

           1: { }

       }

    }

}

1. The code does not compile.
2. The code compiles but results in RuntimeException
3. The code compiles and runs fine.
4. None of the above

**Problem Statement:**

Given the following program, find out which are all legal and illegal enhanced for declarations.

class EnhancedForTest {

    public static void main(String[ ] args) {

         int x;

         long x2;

         Long [ ] La = {4L, 5L, 6L};

         long [ ]  la = {7L, 8L, 9L};

         int [ ] [ ]  twoDee = {{1,2,3}, {4,5,6}, {7,8,9}};

         String [ ]  sNums = {“one”, “two”, “three”};

         Animal [ ] animals = {new Dog[[egg]], new Cat[[egg]]};

     for(long y: la);

     for(long lp: La);

     for(x2 : la);

     for(int[ ] n : twoDee);

     for(int x2 : twoDee);

     for(int n2 : twoDee[2]);

     for(String s : sNums);

     for(int x3 : la);

     for(Object o : sNums);

     for(Animal a : animals);

     for(Dog d : animals);

    }

}

Given:

loop1 : for( int k=0; k< 10; k++){  
          while ( i < 5) {  
               i=a.doSomething();  
               if(i<0) break loop1;   
          }  
          jj = i;  
     }  
     jj = k

What will happen when the control reaches the break statement?

**Problem Statement:**

What is the output of the following program?

class ContinueTest {

    public static void main(String [ ] args) {

         outer:

             for (int i=0; i<5; i++) {

                 for (int j=0; j<5; j++) {

                     System.out.println(“Hello”);

                     continue outer;

                 } // end of inner loop

                 System.out.println(“outer”);

             }

         System.out.println(“Good-Bye”);

    }

}

**Problem Statement:**

What is the output of the following program?

class Eggs {

    int doX(Long x, Long y) { return 1;}

    int doX(long... x) { return 2;}

    int doX(Integer x, Integer y) { return 3; }

    int doX(Number n, Number m) { return 4; }

    public static void main(String[ ] args) {

        new Eggs().go();

    }

    void go() {

        short s = 7;

        System.out.println(doX(s,s) + " ");

        System.out.println(doX(7,7));

    }

}

1. 1  1
2. 2  1
3. 3  1
4. 4  1
5. 2  3
6. 3  3
7. 4  3

**Problem Statement:**

What is the output of the following program?

class Mixer {

    Mixer() { }

    Mixer(Mixer m) { m1 = m; }

    Mixer m1;

    public static void main(String[ ] args) {

        Mixer m2 = new Mixer();

        Mixer m3 = new Mixer(m2);

        m3.go();

        Mixer m4 = m3.m1;

        m4.go();

        Mixer m5 = m2.m1;

        m5.go();

    }

    void go() {

        System.out.println("hi ");

    }

}

1. hi
2. hi hi
3. hi hi hi
4. Compilation fails
5. hi, followed by an exception
6. hi hi, followed by an exception

**Problem Statement:**

What is the output of the following program?

class Alien {

    String invade(short ships) {

        return " a few";

    }

}

class Defender {

    public static void main(String[] args) {

        System.out.println(new Alien().invade(7));

    }

}

1. many
2. a few
3. Compilation fails
4. The output is not predictable
5. An exception is thrown at runtime

**Problem Statement:**

What is the output of the following program?

class Fizz {

    int x = 5;

    public static void main(String[] args) {

        final Fizz f1 = new Fizz();

        Fizz f2 = new Fizz();

        Fizz f3 = FizzSwitch(f1, f2);

        System.out.println((f1 == f3) + " " + (f1.x == f3.x));

    }

    static Fizz FizzSwitch(Fizz x, Fizz y) {

        final Fizz z = x;

        z.x = 6;

        return z;

    }

}

1. true true
2. false true
3. true false
4. false false
5. Compilation fails.
6. An exception is thrown at runtime.

**Problem Statement:**

What is the output of the following program?

class Knowing {

    static final long tooth = 343L;

    static long doIt(long tooth) {

        System.out.print(++tooth + " ");

        return ++tooth;

    }

    public static void main(String[] args) {

        System.out.print(tooth + " ");

        final long tooth = 340L;

        new Knowing().doIt(tooth);

        System.out.println(tooth);

    }

}

1. 343  340  340
2. 343  340  342
3. 343  341  342
4. 343  341  340
5. 343  341  343
6. Compilation fails.
7. An exception is thrown at runtime.

**Problem Statement:**

What is the output of the following program?

class Bird {

    { System.out.print("b1 "); }

    public Bird() {

        System.out.print("b2 ");

    }

}

class Raptor extends Bird {

    static { System.out.print("r1 "); }

    public Raptor() {

        System.out.print("r2 ");

    }

    { System.out.print("r3 "); }

    static { System.out.print("r4 "); }

}

class Hawk extends Raptor {

    public static void main(String[] args) {

        System.out.print("pre ");

        new Hawk();

        System.out.println("hawk ");

    }

}

1. pre b1 b2 r3 r2 hawk
2. pre b2 b1 r2 r3 hawk
3. pre b2 b1 r2 r3 hawk r1 r4
4. r1 r4 pre b1 b2 r3 r2 hawk
5. r1 r4 pre b2 b1 r2 r3 hawk
6. pre r1 r4 b1 b2 r3 r2 hawk
7. pre r1 r4 b2 b1 r2 r3 hawk
8. The order of output cannot be predicted.
9. Compilation fails.

**Problem Statement:**

Which is true? (Choose all that apply.)

1. “X extends Y” is correct if and only if X is a class and Y is an interface.
2. “X extends Y” is correct if and only if X is an interface and Y is a class.
3. “X extends Y” is correct if X and Y are either both classes or both interfaces.
4. “X extends Y” is correct for all combinations of X and Y being classes and/or interfaces.

**Problem Statement:**

What is the output of the following program?

class Top {

    public Top(String s) {

        System.out.println("B");

    }

    public class Bottom2 extends Top {

        public Bootom2(String s) {

            System.out.println("D");

        }

        public static void main(String[] args) {

            new Bottom2("C");

            System.out.println(" ");

        }

    }

}

1. BD
2. DB
3. BDC
4. DBC
5. Compilation fails.

**Problem Statement:**

Using the following fragments, complete the following code so it compiles. Note, you may not have to fill all of the slots.

**Code**:

            class AgedP {

     \_\_\_\_\_\_\_\_\_\_    \_\_\_\_\_\_\_\_\_\_    \_\_\_\_\_\_\_\_\_\_    \_\_\_\_\_\_\_\_\_\_    \_\_\_\_\_\_\_\_\_\_

         public AgedP(int x) {

          \_\_\_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_\_     \_\_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_

         }

            }

     public class Kinder extends AgedP {

         \_\_\_\_\_\_\_\_\_\_    \_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_

             public Kinder(int x) {

                  \_\_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_\_\_   \_\_\_\_\_\_\_\_\_\_\_

             }

         }

**Fragments:** Apply the following fragments zero or more times:

|  |  |  |  |
| --- | --- | --- | --- |
| AgedP | super | this |  |
| ( | ) | { | } |
| ; |  |  |  |

**Problem Statement:**

What is the output of the following program?

class Clidders {

    public final void flipper() {

        System.out.println("Clidder");

    }

}

public class Clidlets  extends Clidders {

    public void flipper() {

        System.out.println("Flip a Clidlet");

        super.flipper();

    }

    public static void main(String[] args) {

        new Clidlets().flipper();

    }

}

1. Flip a Clidlet
2. Flip  a Clidder
3. Flip a Clidder

Flip a Clidlet

1. Flip a Clidlet

Flip a Clidder

1. Compilation fails.

**Problem Statement:**

Given, the following program:

1. class Programmer {

2.     Programmer debug() { return this; }

3. }

4. class SCJP extends Programmer {

5.     // insert code here

6.}

Which method at line 5, will compile? (Choose all that apply.)

1. Programmer debug() { return this; }
2. SCJP debug()  { return this; }
3. Object debug() { return this; }
4. int debug() { return 1; }
5. int debug(int x) { return 1; }
6. Object debug(int x) { return this; }

**Problem Statement:**

What is the output of the following program?

class Clidder {

    private final void flipper() {

        System.out.println("Clidder");

    }

}

public class Clidlet  extends Clidder {

    public final void flipper() {

        System.out.println("Clidlet");

    }

    public static void main(String[] args) {

        new Clidlet().flipper();

    }

}

1. Clidlet
2. Clidder
3. Clidder

Clidlet

1. Clidlet

Clidder

1. Compilation fails.

**Problem Statement:**

Given the following:

  i.     interface Base {

ii.         boolean m1 ();

iii.         byte m2(short s);

iv.     }

Which code fragments will compile? (Choose all that apply.)

a)  interface Base2 implements Base { }

b)  abstract class Class2 extends Base {

c)  public boolean m1 ()  { return true; }  }

d)  abstract class Class2 implements Base { }

e)  abstract class Class2 implements Base {

f)  public boolean m1 () { return (true); }  }

g)  class Class2 implements Base {

h)  boolean m1 () { return false; }

i)  byte m2 (short s) { return 42; } }

**Problem Statement:**

Which of the following declare a compatible abstract class? (Choose all that apply.)

a)  public abstract class Canine { public Bark speak(); }

b)  public abstract class Canine { public Bark speak() { } }

c)  public class Canine { public abstract Bark speak(); }

d)  public class Canine abstract { public abstract Bark speak(); }

**Problem Statement:**

Given:

public abstract interface Frobnicate { public void twiddle(String s); }

Which is a correct class?

a)  public abstract void twiddle(String s) { }

}

b)  public abstract class Frob implements Frobnicate { }

c)  public class Frob extends Frobnicate {

d)              public void twiddle(Integer i) { }

}

e)  public class Frob implements Frobnicate {

f)             public void twiddle(Integer i) { }

}

g)  public class Frob implements Frobnicate {

h)             public void twiddle(String i) { }

i)             public void twiddle(Integer s) { }

**Problem Statement:**

Which statement()s) are true? ()Choose all that apply.)

1. HAS-A relationships always rely on inheritance.
2. HAS-A relationships always rely on instance variables.
3. HAS-A relationships always require at least two class types.
4. HAS-A relationships always rely on polymorphism.
5. HAS-A relationships are always tightly coupled.

**Problem Statement:**

Given:

1. class Plant {

2.     String getName() ) { return “plant”; }

3.     Plant getType() ) { return this; }

4. }

5. class Flower extends Plant {

6.     // insert code here

7. }

8. class Tulip extends Flower { }

Which statement()s), inserted at line 6, will compile? ()Choose all that apply.)

1. **Flower getType() ) { return this; }**
2. String getType() ) { return “this”; }
3. **Plant getType() ) { return this; }**
4. **Tulip getType() ) { return new Tulip() ); }**

**Problem Statement:**

Given:

1.      class Zing {

2.      protected Hmpf h;

3. }

4. class Woop extends Zing {  }

5. class Hmpf { }

Which is true?

A.     Woop IS-A Hmpf and HAS-A zing.

B.     Zing IS-A Woop and HAS-A Hmpf.

C.     Hmpf HAS-A Woop and Woop IS-A Zing.

D.    Woop HAS-A Hmpf and Woop IS-A Zing.

E.     Zing HAS-A Hmpf and Zing IS-A Woop.

**Problem Statement:**

Given:

            public class MyOuter {

         public static class MyInner {

                    public static void foo() ) { }

                }

           }

Which, if placed in a class other than MyOuter or MyInner, instantiates an instance of the nested class?

1. MyOuter.MyInner m = new MyOuter.MyInner() );
2. MyOuter.MyInner m2 = new MyInner() );
3. MyOuter m = new MyOuter() );

MyOuter.MyInner mi = m.new MyOuter.MyInner() );

1. MyInner mi = new MyOuter.MyInner() );

**Problem Statement:**

Given:

            class Boo {

         Boo()String s) { }

         Boo() ) { }

            }

     class Bar extends Boo {

         Bar() ) { }

         Bar()String s)  {super()s); }

         void zoo() ) {

                     // insert code here

                }

            }

Which of the following will create an anonymous inner class from within class Bar? ()Choose all that apply.)

1. Boo  f = new Boo()24) { };
2. Boo  f = new Bar() ) { };
3. Boo  f = Boo() ) {String s; };
4. Bar  f = new Boo()String s) { };
5. Boo  f = new Boo.Bar()String s) { };

**Problem Statement:**

What is the output of the following program?

public abstract class AbstractTest {

    public int getNum()) {

        return 45;

    }

    public abstract class Bar {

        public int getNum()) {

            return 38;

        }

    }

    public static void main()String[] args) {

        AbstractTest t = new AbstractTest()) {

            public int getNum()) {

                return 22;

            }

        };

        AbstractTest.Bar f = t.new Bar()) {

            public int getNum()) {

                return 57;

            }

        };

        System.out.println()f.getNum()) + " " + t.getNum()));

    }

}

1. 57  22
2. 45  38
3. 45  57
4. An exception occurs at runtime.
5. Compilation fails.

**Problem Statement:**

Given:

class Input {

     public static void main(String[ ] args) {

          String s = “-”;

          try {

               doMath(args[0]);

               s += “t ”;      // line 6

          }

          finally {

               System.out.println(s += “f ”);

          }

     }

     public static void doMath(String a) {

          int y = 7 / Integer.parseInt(a);

     }

}

And the command-line invocations:

java Input

java Input 0

Which are true? (Choose all that apply.)

**a)       Line 6 is executed exactly 0 times**

b)      Line 6 is executed exactly 1 time.

c)       Line 6 is executed exactly 2 times.

d)      The finally block is executed exactly 0 times.

e)       The finally block is executed exactly 1 time.

**f)        The finally block is executed exactly 2 times.**

g)       Both invocations produce the same exceptions.

**h)      Each invocation produces a different exception.**

**Problem Statement:**

Given:

try {

     int x = Integer.parseInt(“two”);

}

Which could be used to create an appropriate catch block? (Choose all that apply.)

a)  ClassCastException

b)  IllegalStateException

c)  NumberFormatException

d)  IllegalArgumentException

e)  ExceptionInitializerError

f)  ArrayIndexOutOfBoundsException

**Problem Statement:**

What is the output of the following program?

class Emu {

     static String s = "-";

     public static void main(String[ ] args) {

          try {

               throw new Exception();

          } catch (Exception e) {

               try {

                    try {

                         throw new Exception();

                    } catch (Exception ex) {

                         s += "ic";

                    }

                         throw new Exception();

                    } catch (Exception x) {

                         s += "mc ";

                    } finally {

                         s += "mf ";

               }

          } finally {

               s += "of ";

          }

          System.out.println(s);

     }

}

a)  –ic    of

b)  –mf   of

c)  –mc  mf

d)  –ic    mf    of

e)  –ic    mc   mf    of

f)  –ic    mc   of     mf

**Problem Statement:**

Given:

class Master {

     String doFileStuff() throws FileNotFoundException {

          return "a";

     }

}

class Slave extends Master {

     public static void main(String[] args) {

          String s = null;

          try {

               s = new Slave().doFileStuff();

          } catch (Exception ex) {

               s = "b";

          }

          System.out.println(s);

     }

     // insert code here

}

Which inserted independently at // insert code here, will compile and produce the output b? (Choose all that apply.)

1. **String doFileStuff() { return “b”; }**
2. String doFileStuff() throws IOException { return “b”; }
3. String doFileStuff(int x) throws IOException { return “b”; }
4. **String doFileStuff() throws FileNotFoundException { return “b”; }**
5. **String doFileStuff() throws NumberFormatException { return “b”; }**
6. **String doFileStuff() throws NumberFormatException, FileNotFoundException { return “b”; }**

**Problem Statement:**

What is the output of the following program?

class Plane {

     static String s = "-";

     public static void main(String[] args) {

          new Plane().s1();

          System.out.println(s);

     }

     void s1() {

          try {

               s2();

          } catch (Exception ex) {

               s += "c";

          }

     }

     void s2() throws Exception {

          s3();

          s += "2";

          s3();

          s += "2b";

     }

     void s3() throws Exception {

          throw new Exception();

     }

}

1. –
2. –c
3. –c2
4. -2c
5. –c22b
6. -2c2b
7. -2c2bc
8. Compilation fails.

**Problem Statement:**

Given:

1.      class Ping extends Utils {

2.          public static void main(String[ ]  args) {

3.              Utils u = new Ping();

4.              System.out.print(u.getInt(args[0]));

5.          }

6.          int getInt(String arg) {

7.               return Integer.parseInt(arg);

8.          }

9.      }

10.     class Utils {

11.          int getInt(String x) throws Exception  {

12.              return 7;

13.          }

14.     }

Add the following three possible changes:

a)       C1: Declare that main() throws an Exception.

b)      C2: Declare that Ping.getInt() throws an Exception.

c)       C3: Wrap the invocation of getInt() in a try or catch block.

With change(s) allow the code to compile? (Choose all that apply.)

**A.     Just C1 is sufficient.**

B.     Just C2 is sufficient.

**C.     Just C3 is sufficient.**

D.    Both C1 and C2 are required.

E.     Both C2 and C3 are required.

F.      Both C2 and C3 are required.

G.    All three changes are required.

**Problem Statement:**

What is the output of the following program?

package com.handson;

// Program to define and use user-defined exception

class MyException extends Exception {

     MyException(String msg) {

          super(msg);

     }

}

class TestMyException {

     public static void main(String[ ] args) {

          int x= 3;

          int y=1000;

          try {

               double z = (double) x / (double) y;

               if (z < 0.01) {

                    throw new MyException("Number is too small ");

               }

          } catch (MyException ex) {

               System.out.println(ex.getMessage());

          }

     }

}

And the command line invocation:

java TestMyException

**Problem Statement:**

What is the output of the following program?

class Mineral { }

class Gem extends Mineral { }

class Miner {

     static int x = 7;

     static String s = null;

     public static void getWeight(Mineral m) {

          int y = 0 / x;

          System.out.println(s + " ");

     }

     public static void main(String[ ] args) {

          Mineral[ ] ma = {new Mineral(), new Gem()};

          for(Object o : ma)

               getWeight ((Mineral) o);

     }

}

And the command-line invocation:

java Miner.java

A.     null

B.     null null

C.     A ClasscastException is thrown.

D.    A NullPointerException is thrown.

**E.     A NoClassDefFoundError is thrown.**

F.      An ArithmeticException is thrown.

G.    An IllegalArgumentException is thrown.

H.    An ArrayIndexOutOfBoundsException is thrown.

**Problem Statement:**

Which are most typically thrown by an API developer or an application developer as opposed to being thrown by the JVM?

A. ClassCastException

**B. IllegalStateException**

**C. NumberFormatException**

**D. IllegalArgumentException**

E. ExceptionInitializerError

**Problem Statement:**

What is the output of the following program?

1. class StringTest {

2.    public static void main(String[ ]  args) {

3.        String x = “test”;

4.        String[ ] y = new String[3];

5.        System.out.println(x.length);

6.        System.out.println(y.length());

7.    }

8. }

**Problem Statement:**

What is the output of the following program?

class SubstringTest {

     public static void main(String[ ]  args) {

          String x = “0123456789”;

          System.out.println( x.substring(5) ) ;

          System.out.println(x.substring(5, 8));

     }

}

**Problem Statement:**

What is the output of the following program?

class ChainedMethods {

     public static void main(String[ ]  args) {

          String x = “abc”;

          String y = x.concat(“def”).toUpperCase().replace(‘C’, ‘x’);         // chained methods

          System.out.println(“y = ” + y);

     }

}

**Problem Statement:**

What is the output of the following program?

import java.util.\*;

class SplitTest1 {

     public static void main(String[ ]  args) {

          String[ ] tokens = args[0].split(args[1]);

          System.out.println(“count ” + tokens.length);

          for(String s : tokens)

               System.out.println(“>” + s + “<”);

     }

}

And the command-line invocation

% java SplitTest1  “ab5 ccc 45 @” “\d”

**Problem Statement:**

What is the output of the following program?

class SplitTest2 {

     public static void main(String[ ] args) {

          String str = "ab.cde.fg";

          String[ ] tokens = str.split("\\.");

          System.out.println("total number of splitted tokens: " + tokens.length);

          for (String s : tokens)

               System.out.println("splitted token is: " + s);

     }

}

**Problem Statement:**

What is the output of the following program?

class EscapeSequenceCharacter {

     public static void main(String[ ]  args) {

          System.out.println(“\” \\”);

     }

}

**Problem Statement:**

What is the output of the following program?

class Delete {

     public static void main(String[ ]  args) {

          StringBuilder sb = new StringBuilder(“0123456789”);

          System.out.println(sb.delete(4,6));

     }

}

**Problem Statement:**

What is the output of the following program?

class Insert {

     public static void main(String[ ]  args) {

          StringBuilder  sb = new StringBuilder(“01234567”);

          sb.insert(4, “---”);

          System.out.println(sb);

     }

}

**Problem Statement:**

What is the output of the following program?

class Reverse {

     public static void main(String[ ]  args) {

          StringBuffer sb = new StringBuffer(“A man a plan a canal Panama”);

          sb.reverse();

          System.out.println(sb);

     }

}

**Problem Statement:**

Which about the three java.lang classes String, StringBuilder, and StringBuffer are true? (Choose all that apply.)

A.     **All three classes have a length() method.**

**B.     Objects of type StringBuffer are thread-safe.**

C.     All three classes have overloaded append() methods.

D.    The “+” is an overloaded operator for all three classes.

E.     According to the API, StringBuffer will be faster than StringBuilder under most implementations.

F.      The value of an instance of any of these three types can be modified through various methods in the API.

**Problem Statement:**

What is the output of the following program?

class Polish {

     public static void main(String[ ]  args) {

          int x = 4;

          StringBuffer sb = new StringBuffer(“..fedcba”);

          sb.delete(3, 6);

          sb.insert(3, “az”):

          if (sb.length()  > 6)

               x = sb.indexOf(“b”);

          sb.delete((x-3), (x-2));

          System.out.println(sb);

     }

}

A.     .faza

B.     .fzba

C.     ..azba

D.    .fazba

E.     ..fezba

F.      Compilation fails.

G.    An exception is thrown at runtime.

**Problem Statement:**

What is the output of the following program?

class Append {

     public static void main(String[ ]  args) {

          StringBuffer sbuf = new StringBuffer(“abc”);

          sbuf.append(“def”);

          System.out.println(“sbuf = ” + sbuf);

          StringBuilder sbld = new StringBuilder(“abc”);

          sbld.append(“def”).reverse().insert(3, “---”);

          System.out.println(“sbld = ” +  sbld) ;

     }

}

**Problem Statement:**

Given:

import java.util.\*;

class CollectionTypes {

        public static void main(String[ ]  args) {

        // insert code here

        x.add(“one”);

        x.add(“two”);

        x.add(“TWO”);

        System.out.println(x.poll());

    }

}

Which inserted at // insert code here, will compile? (Choose all that apply.)

a)  List<String> x = new LinkedList<String>();

b)  TreeSet<String> x = new TreeSet<String>();

c)  HashSet<String> x = new HashSet<String>();

**d)  Queue<String> x = new PriorityQueue<String>();**

e)  ArrayList<String> x = new ArrayList<String>();

**f)  LinkedList<String> x = new LinkedList<String>();**

**Problem Statement:**

Given a properly prepared String array containing five elements, which range of results could a proper invocation of Arrays.binarySearch() produce?

a)       zero through four

b)      zero through five

c)       -1 through four

d)      -1 through five

e)       -5 through four

f)        -5 through five

**g)       -6 through four If a match is found, binarysearch() will return the index of the elt that was matched. If no match is found, binarysearch() will return a –ve number that, if inverted and then decreamented, gives you the insertion point at which thte value searched on should be inserted into the array to maintain a proper sort.**

h)      -6 through five

**Problem Statement:**

Given a method declared as:

public static <E extends Number> List<? super E> process(List<E> nums)

A programmer wants to use this method like the following:

// INSERT DECLARATIONS HERE

output = process(input);

Which pairs of declarations could be placed at // INSERT DECLARATIONS HERE to allow the code to compile? (Choose all that apply.)

a)       ArrayList<Integer> input = null;

ArrayList<Integer> output = null;

**b)  ArrayList<Integer> input = null;**

**List<Integer> output = null;**

c)  ArrayList<Integer> input = null;

List<Number> output = null;

d)  List<Number> input = null;

ArrayList<Integer> output = null;

**e)  List<Number> input = null;**

**List<Number> output = null;**

**f)  List<Integer> input = null;**

**List<Integer> output = null;**

g)       None of the earlier.

**Problem Statement:**

Which collection class(es) allows you to grow or shrink its size and provides indexed access to its elements, but whose methods are not synchronized? (Choose all that apply.)

A. java.util.HahSet

B. java.util.LinkedHashSet

C. java.util.List

**D. java.util.ArrayList**

E. java.util.Vector

F. java.util.PriorityQueue

**Problem Statement:**

Given:

            public class Group extends HashSet<Person> {

    public static void main(String[] args) {

        Group g = new Group();

        g.add(new Person("Hans"));

        g.add(new Person("Lotte"));

        g.add(new Person("Jane"));

        g.add(new Person("Hans"));

        g.add(new Person("Jane"));

        System.out.println("Total : " + g.size());

    }

    public boolean add(Object o) {

        System.out.println("Adding: " + o);

        return super.add(o);

    }

}

class Person {

    private final String name;

    public Person(String name) {

        this.name = name;

    }

    public String toString() {

        return name;

    }

}

Which of the following occur at least once when the code is compiled and run? (Choose all that apply.)

A.     Adding Hans

B.     Adding Lotte

C.     Adding Jane

D.    Total: 3

E.     Total: 5

F**.      The code does not compile. The problem here is group’s add() it should have been add(Person), since the class extends HasSet<Person>. So this dosent compile**

G.    An exception is thrown at runtime.

**Problem Statement:**

What is the output of the following program?

package com.handson;

import java.util.HashMap;

import java.util.Map;

class Dog {

    public Dog(String n) {

        name = n;

    }

    public String name;

    public boolean equals(Object o) {

        if ((o instanceof Dog) && (((Dog)o).name == name)) {

            return true;

        } else {

            return false;

        }

    }

    public int hashCode() {

        return name.length();

    }

}

class Cat { }

enum Pets {DOG, CAT, HORSE}

class MapTest {

    public static void main(String[] args) {

        Map<Object, Object> m = new HashMap<Object, Object>();

        m.put("k1", new Dog("aiko"));                // add some key/value pairs

        m.put("k2", Pets.DOG);

        m.put(Pets.CAT, "CAT key");

        Dog d1 = new Dog("clover");                  // let's keep this reference

        m.put(d1, "Dog key");

        m.put(new Cat(), "cat key");

        System.out.println(m.get("k1"));             // #1

        String k2 = "k2";

        System.out.println(m.get(k2));               // #2

        Pets p = Pets.CAT;

        System.out.println(m.get(p));                // #3

        System.out.println(m.get(d1));               // #4

        System.out.println(m.get(new Cat()));        // #5

        System.out.println(m.size());                // #6

    }

}

**Problem Statement:**

Given:

public static void before() {

    Set set = new TreeSet();

    set.add(“2”);

    set.add(3);

    set.add(“1”);

    Iterator it = set.iterator();

    while (it.hasNext())

         System.out.print(it.next() + “  ”);

}

Which of the following statements are true?

A.     The before() method will print 1  2.

B.     The before() method will print 1  2   3.

C.     The before() method will print three numbers, but the order cannot be determined.

D.    The before() method will not compile.

E**.     The before() method will throw an exception at runtime.**

**Problem Statement:**

Given:

TreeSet map = new TreeSet();

map.add(“one”);

map.add(“two”);

map.add(“three”);

map.add(“four”);

map.add(“one”);

Iterator it = map.iterator();

while (it.hasNext()) {

     System.out.print(it.next() + “ “ );

}

What is the result?

A.     Compilation fails.

B.     one    two     three  four

C.     four   three   two    one

D.    four   one     three   two

E.     one    two    three   four  one

F.      one    four    three   two  one

G.    An exception is thrown at runtime.

H.    The print order is not guaranteed.

**Problem Statement:**

What is the output of the following program?

// Modern, generics version using ArrayList and Iterator.

// This program stores a list of strings in the ArrayList and then displays the contents // of the list.

package com.handson;

import java.util.ArrayList;

import java.util.Iterator;

class NewStyle {

    public static void main(String[ ] args) {

        // Now, list holds references of type String.

        ArrayList<String> list = new ArrayList<String>();

        list.add("one");

        list.add("two");

        list.add("three");

        list.add("four");

        // Notice that Iterator is also generic.

        Iterator<String> itr = list.iterator();

        // The following statement will now cause a compile time error.

        // Iterator<Integer> itr = list .iterator();   // Error!

        while(itr.hasNext()) {

            String str = itr.next();    // no cast needed

            // Now, the following line is a compile time, rather than runtime, error

             //Integer i = itr.next();  // this won't compile

            System.out.println(str + " is " + str.length() + " chars long.");

        }

    }

}

**Problem Statement:**

What is the output of the following program?

            import java.text.\*;

     class DateOne {

         public static void main(String[ ] args) {

             Date dt = new Date(1123631685981L);

             DateFormat df = new DateFormat();

             System.out.println(df.format(dt));

         }

     }

And given that 1123631685981L is the number of milliseconds between Jan. 1, 1970, and sometime on Aug. 9, 2005. What is the result? (Note: the time of day in option A may vary.)

A.     4/8/08 12:27 PM

B.     1123631685981L

C.     An exception is thrown at run-time

D.    Compilation fails due to a single error in the code

E.     Compilation fails due to multiple errors in the code

**Problem Statement:**

Which of the following statements are true?

A.     The DateFormat.getDate() is used to convert a String to a Date instance.

**B.     Both DateFormat and NumberFormat objects can be constructed to be Locale specific.**

**C.     Both Currency and NumberFormat objects must be constructed using static methods.**

**D.    If the instance of NumberFormat Locale is to be different than the current Locale, then it must be specified at creation time.**

**E.     A single instance of NumberFormat can be used to create Number objects from Strings and to create formatted numbers from numbers**

**Problem Statement:**

Write a program to display the current date and time using GregorianCalendar class.

**Problem Statement:**

Given:

public class Messenger implements Runnable {

    public static void main(String[ ] args) {

        new Thread(new Messenger(“Wallace”)).start();

        new Thread(new Messenger(“Gromit”)).start();

    }

    private String name;

    public Messenger(String name) {

        this.name = name;

    }

    public void run() {

        message(1);

        message(2);

    }

    private synchronized void message(int n) {

         System.out.print(name + “-” + n + “  ”);

    }

}

Which of the following is a possible result? (Choose all that apply.)

A.     Wallace-1  Wallace-2  Gromit-1

B.     Wallace-1  Gromit-2  Wallace-2  Gromit-1

**C.     Wallace-1  Wallace-2  Gromit-1  Gromit-2**

D.    Gromit-1   Gromit-2

E.     Gromit-2   Wallace-1   Gromit-1  Wallace-2

F.      The code does not compile.x

G.    An error occurs at runtime.

**Problem Statement:**

Given:

public class Letters extends Thread {

    private String name;

    public Letters(String name) { this.name = name; }

    public void write() {

        System.out.print(name);

        System.out.print(name);

    }

     public static void main(String[ ] args) {

          new Letters(“X”).start();

          new Letters(“Y”).start();

     }

}

You want to guarantee that the output can be either XXYY or YYXX, but never XYXY or any other combination. Which of the following method definitions could be added to the Letters class to make this guarantee? (Choose all that apply.)

A.  public void run() { write(); }

B.  public synchronized void run() { write(); }

C.  public static synchronized void run() { write(); }

D.  public void run() { synchronized (this) { write(); } }

**E.  public void run() { synchronized (Letters.class) { write(); } }**

**F.  public void run() { synchronized (System.out) { write(); } }**

G.  public void run() { synchronized (System.out.class) { write(); } }

**Problem Statement:**

The following block of code creates a Thread using a Runnable target:

Runnable  target  = new MyRunnable();

Thread myThread = new Thread(target);

Which of the following classes can be used to create the target, so that the preceding code compiles correctly?

A.  public class MyRunnable extends Runnable { public void run() { } }

B.  public class MyRunnable extends Object { public void run() { } }

**C.  public class MyRunnable implements Runnable { public void run() { } }**

D.  public class MyRunnable implements Runnable { void run() { } }

E.  public class MyRunnable implements Runnable { public void start() { } }

**Problem Statement:**

What is the output of the following program?

            class MyThread extends Thread {

                public static void main(String[ ]  args) {

             MyThread t = new MyThread();

             t.start();

             System.out.print(“one. ”);

             t.start();

             System.out.print(“two. ”);

        }

         public void run() {

             System.out.print(“Thread ”);

         }

     }

A.     Compilation fails.

**B.     An exception occurs at runtime.**

C.     Thread one. Thread two.

D.    The output cannot be determined.

**Problem Statement:**

Which are the methods of the Object class? (Choose all that apply.)

**A.  notify();**

**B.  notifyAll();**

C.  isInterrupted();

D.  synchronized();

E.  interrupt();

**F.  wait(long msecs);**

G.  sleep(long msecs);

H.  yield();

**Problem Statement:**

Given the following program:

1.   public class WaitTest {

2.       public static void main(String[ ]  args) {

3.            System.out.print(“1 ”);

4.            synchronized(args) {

5.                System.out.print(“2 ”);

6.                try {

7.                args.wait();

8.                }

9.         catch (InterruptedException ie) { }

10.            }

11.            System.out.print(“3 ”);

12.      }

13.  }

What is the result of trying to compile and run this program?

A.     It fails to compile because the IllegalMonitorStateException of wait() is not dealt with in line 7.

B.     1  2  3

C.     1  3

D.    At runtime, it throws an IllegalMonitorStateException when trying to wait.

E.     It will fail to compile because it has to be synchronized on this object.

F**.      The program will wait forever.**

**Problem Statement:**

Assume you have a class that holds two private variables: a and b. Which of the following pairs can prevent concurrent access problems in that class? (Choose all that apply.)

a)  public int read() { return a + b; }

public void set (int a, int b) { this.a=a; this.b=b;}

b**)  public synchronized int read() { return a+b; }**

**public synchronized void set (int a, int b) { this.a=a; this.b=b;}**

c)  public int read() { synchronized(a) { return a+b; }}

public void set(int a, int b) {synchronized(b) { this.a=a; this.b=b;}}

d)  public synchronized(this) int read() { return a+b; }

public synchronized(this) void set(int a, int b) {this.a=a; this.b=b;}

e)  **public int read() { synchronized(this) { return a+b; }}**

**public void set(int a, int b) synchronized(this)  {this.a=a; this.b=b;}}**

**Problem Statement:**

Given:

            public class TwoThreads {

    static Thread laurel, hardy;

    public static void main(String[ ] args) {

        laurel = new Thread() {

            public void run() {

                System.out.println("A");

                try {

                    hardy.sleep(1000);

                } catch (Exception ex) {

                    System.out.println("B");

                }

                System.out.println("C");

            }

        };

        hardy = new Thread() {

            public void run() {

                System.out.println("D");

                try {

                    laurel.wait();

                } catch (Exception ex) {

                    System.out.println("E");

                }

                System.out.println("F");

            }

        };

        laurel.start();

        hardy.start();

    }

}

Which letters will eventually appear somewhere in the output? (Choose all that apply.)

A**.     A**

B.     B

C.     **C**

**D.    D**

**E.     E**

**F.      F**

G.    The answer cannot be reliably determined.

H.    The code does not compile.

**Problem Statement:**

Given that bw is a reference to a valid BufferedWriter. and the code snippet is as follows:

15. BufferedWriter b1 = new BufferedWriter(new File(“f”));

16. BufferedWriter b2 = new BufferedWriter(new FileWriter(“f1”));

17. BufferedWriter b3 = new BufferedWriter(new PrintWriter(“f2”));

18. BufferedWriter b4 = new BufferedWriter(new BufferedWriter(bw)):

What is the result?

A.     Compilation fails.

B**.     Compilation fails due only to an error on line 15.**

C.     Compilation fails due only to an error on line 16.

D.    Compilation fails due only to an error on line 17.

E.     Compilation fails due only to an error on line 18.

F.      Compilation fails due to errors on multiple lines.

**Problem Statement:**

Given:

import java.io.\*;

class Directories {

    static String [ ] dirs = {“dir1”, “dir2”};

    public static void main(String[ ] args) {

        for (String d : dirs) {

             // insert code 1 here

             File file = new File(path, args[0]);

             // insert code 2 here

        }

    }

}

and that the invocation java Directories file2.txt is issued from a directory that has two subdirectories, “dir1” and “dir2”, and that “dir1” has a file “file1.txt” and “dir2” has a file “file2.txtx”, and the output is “false true”. Which sets of code fragments must be inserted? (Choose all that apply.)

**a)  String path = d;**

**System.out.print(file.exists() + “  ”);**

**b)  String path = d;**

**System.out.print(file.isFile() + “  ”);**

c)  String path = File.separator + d;

System.out.print(file.exists() + “  ”);

d)  String path = File.separator + d;

System.out.print(file.isFile() + “  ”);

**Problem Statement:**

Write a program to read a string from the console and then display it on the screen.

**Problem Statement:**

What is the output of the following program?

package com.handson;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.io.Serializable;

     class Player {

        /\*\* Creates a new instance of Player \*/

        Player() {

        System.out.print("p");

        }

}

class CardPlayer extends Player implements Serializable {

    CardPlayer() {

        System.out.print("c");

    }

    public static void main(String[ ] args) {

        CardPlayer c1 = new CardPlayer();

        try {

            FileOutputStream fos = new FileOutputStream("play.txt");

            ObjectOutputStream os = new ObjectOutputStream(fos);

            os.writeObject(c1);

            os.close();

            FileInputStream fis = new FileInputStream("play.txt");

            ObjectInputStream is = new ObjectInputStream(fis);

            CardPlayer c2 = (CardPlayer) is.readObject();

            is.close();

        } catch(Exception ex) { }

    }

}

A.     pc

B.     pcc

**C.     pcp - a class to implement serializable even if its superclass dosent. However, when you describe such an object, the non serializable superclass must run its constructor. Constructor don’t run on deserialized classes that implement serializable**

D.    pcpc

E.     Compilation fails.

F.      An Exception is thrown at runtime.

What is the output of the following program? (Choose all that apply.)

     package com.handson;

import java.io.FileOutputStream;

import java.io.ObjectOutputStream;

import java.io.Serializable;

class Keyboard { }

public class Computer implements Serializable {

    private Keyboard k = new Keyboard();

    public static void main(String[ ] args) {

        Computer comp = new Computer();

        comp.storeIt(comp);

    }

    void storeIt(Computer computer) {

        try {

ObjectOutputStream os = new ObjectOutputStream(new    FileOutputStream("myFile"));

            os.writeObject(computer);

            os.close();

            System.out.println("done");

        } catch (Exception ex) {

            System.out.println("exc");

        }

    }

}

**A.     exc --- An instance of type Computer has-a keyboard. Because keyboard dosent implement serialable, any attempt to serialize an instance of computer will cause an exception to be thrown**

B.     done

C.     Compilation fails.

D.    Exactly one object is serialized.

E.     Exactly two object is serialized.

**Problem Statement:**

Given:

package com.handson;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.io.Serializable;

public class TestSer {

    public static void main(String[ ] args) {

        SpecialSerial ss = new SpecialSerial();

        try {

ObjectOutputStream os = new ObjectOutputStream(new FileOutputStream("myFile"));

            os.writeObject(ss);

            os.close();

            System.out.println(++ss.z + " ") ;

ObjectInputStream is = new ObjectInputStream(new FileInputStream("myFile"));

SpecialSerial s2 = (SpecialSerial)is.readObject();

            is.close();

            System.out.println(s2.y + " " + s2.z);

        } catch (Exception ex) {

            System.out.println("exec");

        }

    }

}

class SpecialSerial implements Serializable {

    transient int y = 7;

    static int z = 9;

}

Which are true? (Choose all that apply.)

A.     Compilation fails.

B.     The output is 10  0  9

**C.     The output is 10  0  10**

D.    The output is 10  7  9

E.     The output is 10  7  10

F.      In order to alter the standard deserialization process, you would override the readObject() method in SpecialSerial.

G.    In order to alter the standard deserialization process, you would override the defaultReadObject

**Problem Statement:**

What is the result of the following program?

import java.sql.\*;

public class CreateCoffees {

    public static void main(String args[ ]) {

        String url = "jdbc:mySubprotocol:myDataSource";

        Connection con;

        String createString;

        createString = "create table COFFEES " +

                "(COF\_NAME VARCHAR(32), " +

                "SUP\_ID INTEGER, " +

                "PRICE FLOAT, " +

                "SALES INTEGER, " +

                "TOTAL INTEGER)";

        Statement stmt;

        try {

            Class.forName("myDriver.ClassName");

        } catch(java.lang.ClassNotFoundException e) {

            System.err.print("ClassNotFoundException: ");

            System.err.println(e.getMessage());

        }

        try {

            con = DriverManager.getConnection(url,"myLogin", "myPassword");

            stmt = con.createStatement();

            stmt.executeUpdate(createString);

            stmt.close();

            con.close();

        } catch(SQLException ex) {

            System.err.println("SQLException: " + ex.getMessage());

        }

    }

}

**Problem Statement:**

What is the output of the following program?

import java.sql.\*;

public class InsertCoffees {

   public static void main(String args[]) {

      String url = "jdbc:mySubprotocol:myDataSource";

      Connection con;

      Statement stmt;

      String query = "select COF\_NAME, PRICE from COFFEES";

      try {

         Class.forName("myDriver.ClassName");

         } catch(java.lang.ClassNotFoundException e) {

         System.err.print("ClassNotFoundException: ");

         System.err.println(e.getMessage());

      }

      try {

         con = DriverManager.getConnection(url,

                               "myLogin", "myPassword");

         stmt = con.createStatement();

         stmt.executeUpdate("insert into COFFEES " +

              "values('Colombian', 101, 7.99, 0, 0)");

         stmt.executeUpdate("insert into COFFEES " +

              "values('French\_Roast', 49, 8.99, 0, 0)");

         stmt.executeUpdate("insert into COFFEES " +

              "values('Espresso', 150, 9.99, 0, 0)");

         stmt.executeUpdate("insert into COFFEES " +

              "values('Colombian\_Decaf', 101, 8.99, 0, 0)");

         stmt.executeUpdate("insert into COFFEES " +

              "values('French\_Roast\_Decaf', 49, 9.99, 0, 0)");

         ResultSet rs = stmt.executeQuery(query);

         System.out.println("Coffee Break Coffees and Prices:");

         while (rs.next()) {

            String s = rs.getString("COF\_NAME");

            float f = rs.getFloat("PRICE");

            System.out.println(s + "   " + f);

         }

         stmt.close();

         con.close();

      } catch(SQLException ex) {

         System.err.println("SQLException: " + ex.getMessage());

      }

   }

}

**Problem Statement:**

What is the output of the following program?

import java.sql.\*;

public class CreateSuppliers {

   public static void main(String args[]) {

      String url = "jdbc:mySubprotocol:myDataSource";

      Connection con;

      String createString;

      createString = "create table SUPPLIERS " +

                    "(SUP\_ID INTEGER, " +

                    "SUP\_NAME VARCHAR(40), " +

                    "STREET VARCHAR(40), " +

                    "CITY VARCHAR(20), " +

                    "STATE CHAR(2), ZIP CHAR(5))";

      Statement stmt;

      try {

         Class.forName("myDriver.ClassName");

      } catch(java.lang.ClassNotFoundException e) {

         System.err.print("ClassNotFoundException: ");

         System.err.println(e.getMessage());

      }

      try {

         con = DriverManager.getConnection(url,

                               "myLogin", "myPassword");

         stmt = con.createStatement();

         stmt.executeUpdate(createString);

         stmt.close();

         con.close();

      } catch(SQLException ex) {

         System.err.println("SQLException: " + ex.getMessage());

      }

   }

}

**Problem Statement:**

What is the output of the following program?

import java.sql.\*;

public class InsertSuppliers {

  public static void main(String args[]) {

      String url = "jdbc:mySubprotocol:myDataSource";

      Connection con;

      Statement stmt;

      String query = "select SUP\_NAME, SUP\_ID from SUPPLIERS";

      try {

         Class.forName("myDriver.ClassName");

      } catch(java.lang.ClassNotFoundException e) {

         System.err.print("ClassNotFoundException: ");

         System.err.println(e.getMessage());

      }

      try {

         con = DriverManager.getConnection(url,

                               "myLogin", "myPassword");

         stmt = con.createStatement();

         stmt.executeUpdate("insert into SUPPLIERS " +

              "values(49, 'Superior Coffee', '1 Party Place', " +

             "'Mendocino', 'CA', '95460')");

         stmt.executeUpdate("insert into SUPPLIERS " +

            "values(101, 'Acme, Inc.', '99 Market Street', " +

            "'Groundsville', 'CA', '95199')");

         stmt.executeUpdate("insert into SUPPLIERS " +

"values(150, 'The High Ground', '100 Coffee Lane', " + "'Meadows', 'CA', '93966')");

         ResultSet rs = stmt.executeQuery(query);

            System.out.println("Suppliers and their ID Numbers:");

         while (rs.next()) {

            String s = rs.getString("SUP\_NAME");

            int n = rs.getInt("SUP\_ID");

            System.out.println(s + "   " + n);

         }

         stmt.close();

         con.close();

      } catch(SQLException ex) {

         System.err.println("SQLException: " + ex.getMessage());

      }

   }

}

**Problem Statement:**

What is the output of the following program?

import java.sql.\*;

public class Join {

   public static void main(String args[]) {

      String url = "jdbc:mySubprotocol:myDataSource";

      Connection con;

String query = "select SUPPLIERS.SUP\_NAME, COFFEES.COF\_NAME " +  "from COFFEES, SUPPLIERS " +

"where SUPPLIERS.SUP\_NAME = 'Acme, Inc.' and " +     "SUPPLIERS.SUP\_ID = COFFEES.SUP\_ID";

      Statement stmt;

      try {

         Class.forName("myDriver.ClassName");

      } catch(java.lang.ClassNotFoundException e) {

         System.err.print("ClassNotFoundException: ");

         System.err.println(e.getMessage());

      }

      try {

         con = DriverManager.getConnection (url,

                               "myLogin", "myPassword");

         stmt = con.createStatement();

         ResultSet rs = stmt.executeQuery(query);

         System.out.println("Supplier, Coffee:");

         while (rs.next()) {

            String supName = rs.getString(1);

            String cofName = rs.getString(2);

            System.out.println("    " + supName + ", " + cofName);

         }

         stmt.close();

         con.close();

      } catch(SQLException ex) {

         System.err.print("SQLException: ");

         System.err.println(ex.getMessage());

      }

   }

}

**Problem Statement:**

What is the output of the following program?

import java.sql.\*;

public class TransactionPairs {

   public static void main(String args[]) {

      String url = "jdbc:mySubprotocol:myDataSource";

      Connection con = null;

      Statement stmt;

      PreparedStatement updateSales;

      PreparedStatement updateTotal;

      String updateString = "update COFFEES " +

                     "set SALES = ? where COF\_NAME = ?";

      String updateStatement = "update COFFEES " +

             "set TOTAL = TOTAL + ? where COF\_NAME = ?";

      String query = "select COF\_NAME, SALES, TOTAL from COFFEES";

      try {

         Class.forName("myDriver.ClassName");

      } catch(java.lang.ClassNotFoundException e) {

         System.err.print("ClassNotFoundException: ");

         System.err.println(e.getMessage());

      }

      try {

         con = DriverManager.getConnection(url,

                               "myLogin", "myPassword");

         updateSales = con.prepareStatement(updateString);

         updateTotal = con.prepareStatement(updateStatement);

         int [] salesForWeek = {175, 150, 60, 155, 90};

         String [] coffees = {"Colombian", "French\_Roast",

                          "Espresso", "Colombian\_Decaf",

                          "French\_Roast\_Decaf"};

         int len = coffees.length;

         con.setAutoCommit(false);

         for (int i = 0; i < len; i++) {

            updateSales.setInt(1, salesForWeek[i]);

            updateSales.setString(2, coffees[i]);

            updateSales.executeUpdate();

            updateTotal.setInt(1, salesForWeek[i]);

            updateTotal.setString(2, coffees[i]);

            updateTotal.executeUpdate();

            con.commit();

         }

         con.setAutoCommit(true);

         updateSales.close();

         updateTotal.close();

         stmt = con.createStatement();

         ResultSet rs = stmt.executeQuery(query);

         while (rs.next()) {

            String c = rs.getString("COF\_NAME");

            int s = rs.getInt("SALES");

            int t = rs.getInt("TOTAL");

            System.out.println(c + "     " +  s + "    " + t);

         }

         stmt.close();

         con.close();

      } catch(SQLException ex) {

         System.err.println("SQLException: " + ex.getMessage());

         if (con != null) {

            try {

               System.err.print("Transaction is being ");

               System.err.println("rolled back");

               con.rollback();

            } catch(SQLException excep) {

               System.err.print("SQLException: ");

               System.err.println(excep.getMessage());

            }

         }

      }

   }

}

Problem Statement:

What is the result of the following program?

import java.sql.DriverManager;

import java.sql.Connection;

import java.sql.DatabaseMetaData;

import java.sql.Statement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.SQLWarning;

public class DatabaseMetaDataExample {

final static String driverClass    = "oracle.jdbc.driver.OracleDriver";

final static String connectionURL  = "jdbc:oracle:thin:@localhost:1521:CUSTDB";

final static String userID         = "scott";

final static String userPassword   = "tiger";

Connection   con                   = null;

/\*\*

\* Construct a DatabaseMetaDataExample object. This constructor

\* will create an Oracle database connection.

\*/

public DatabaseMetaDataExample() {

try {

System.out.print("  Loading JDBC Driver  -> " + driverClass + "\n");

Class.forName(driverClass).newInstance();

System.out.print("  Connecting to        -> " + connectionURL + "\n");

this.con = DriverManager.getConnection(connectionURL, userID, userPassword);

System.out.print("  Connected as         -> " + userID + "\n");

} catch (ClassNotFoundException e) {

e.printStackTrace();

} catch (InstantiationException e) {

e.printStackTrace();

} catch (IllegalAccessException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

}

}

private static void prt(Object s) {

System.out.print(s);

}

private static void prt(int i) {

System.out.print(i);

}

private static void prtln(Object s) {

prt(s + "\n");

}

private static void prtln(int i) {

prt(i + "\n");

}

private static void prtln() {

prtln("");

}

/\*\*

\* Run the DatabaseMetaData Example. This method will use a

\* DatabaseMetaData object to obtain the capabilities of the

\* JDBC driver and the database during runtime.

\*/

public void runExample() {

DatabaseMetaData md = null;

try {

// Obtain a DatabaseMetaData object from our current

// connection

md = con.getMetaData();

prtln();

prtln("  DatabaseMetaData Information");

prtln("  ============================");

prt("  - Product Name          : ");

try {

prtln(md.getDatabaseProductName());

} catch (SQLException e) {

prtln("java.sql.SQLException: Unsupported feature");

}

prt("  - Product Version Number : ");

try {

prtln(md.getDatabaseProductVersion());

} catch (SQLException e) {

prtln("java.sql.SQLException: Unsupported feature");

}

prt("  - Database Major Version : ");

try {

prtln(md.getDatabaseMajorVersion());

} catch (SQLException e) {

prtln("java.sql.SQLException: Unsupported feature");

}

prt("  - Database Minor Version : ");

try {

prtln(md.getDatabaseMinorVersion());

} catch (SQLException e) {

prtln("java.sql.SQLException: Unsupported feature");

}

prt("  - Driver Name            : ");

try {

prtln(md.getDriverName());

} catch (SQLException e) {

prtln("java.sql.SQLException: Unsupported feature");

}

prtln("  - Driver Major Version   : " + md.getDriverMajorVersion());

prtln("  - Driver Minor Version   : " + md.getDriverMinorVersion());

prt("  - Username               : ");

try {

prtln(md.getUserName());

} catch (SQLException e) {

prtln("java.sql.SQLException: Unsupported feature");

}

prt("  - Catalogs               : ");

try {

ResultSet catalogs = md.getCatalogs();

while (catalogs.next()) {

prtln("    - " + catalogs.getString(1) );

}

catalogs.close();

} catch (SQLException e) {

prtln("java.sql.SQLException: Unsupported feature");

}

prtln();

} catch (SQLException e) {

e.printStackTrace();

}

}

/\*\*

\* Close down Oracle connection.

\*/

public void closeConnection() {

try {

System.out.print("  Closing Connection...\n");

con.close();

} catch (SQLException e) {

e.printStackTrace();

}

}

/\*\*

\* Sole entry point to the class and application.

\* @param args Array of String arguments.

\* @exception java.lang.InterruptedException

\*            Thrown from the Thread class.

\*/

public static void main(String[] args)

throws java.lang.InterruptedException {

DatabaseMetaDataExample dmde = new DatabaseMetaDataExample();

dmde.runExample();

        dmde.closeConnection();

}

}

Problem Statement:

What is the result of the following program?

import java.sql.DriverManager;

import java.sql.Connection;

import java.sql.Statement;

import java.sql.ResultSet;

import java.sql.CallableStatement;

import java.sql.SQLException;

public class CallPLSQLProc {

final static String driverClass    = "oracle.jdbc.driver.OracleDriver";

final static String connectionURL  = "jdbc:oracle:thin:@localhost:1521:CUSTDB";

final static String userID         = "scott";

final static String userPassword   = "tiger";

Connection   con                   = null;

/\*\*

\* Construct a CallPLSQLProc object. This constructor will

\* create an Oracle database connection.

\*/

public CallPLSQLProc() {

try {

System.out.print("  Loading JDBC Driver  -> " + driverClass + "\n");

Class.forName(driverClass).newInstance();

System.out.print("  Connecting to        -> " + connectionURL + "\n");

this.con = DriverManager.getConnection(connectionURL, userID, userPassword);

System.out.print("  Connected as         -> " + userID + "\n\n");

} catch (ClassNotFoundException e) {

e.printStackTrace();

} catch (InstantiationException e) {

e.printStackTrace();

} catch (IllegalAccessException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

}

}

/\*\*

\* Method to call the PL/SQL procedure "set\_employee\_salary"

\*/

public void performProcCall(  {

        Statement          stmt         = null;

        ResultSet          rset         = null;

        CallableStatement  cstmt        = null;

String   queryString  = "SELECT name, monthly\_salary " + "FROM   emp " + "WHERE  emp\_id = 1001";

        try {

            stmt = con.createStatement ();

            // ---------------------------------------

            // Obtain and print current monthly salary

            // ---------------------------------------

            rset = stmt.executeQuery(queryString);

            rset.next();

System.out.println("  Name           : " + rset.getString(1));

System.out.println("  Monthly Salary : $" + rset.getFloat(2) + "\n");

            // -----------------------------------------------------

            // Call PL/SQL Procedure to give an employee a %20 raise

            // -----------------------------------------------------

System.out.println("  Lets give this employee a 20% raise.\n");

            cstmt = con.prepareCall("{call set\_employee\_salary(?, ?)}");

            cstmt.setInt(1, 1001);

            cstmt.setDouble(2, 1.2);

            cstmt.execute();

            cstmt.close();

            // -------------------------------------------

            // Obtain and print NEW current monthly salary

            // -------------------------------------------

            rset = stmt.executeQuery(queryString);

            rset.next();

System.out.println("  Name           : " + rset.getString(1));

System.out.println("  Monthly Salary : $" + rset.getFloat(2) + "\n");

            rset.close();

            stmt.close();

        } catch (SQLException e) {

            e.printStackTrace();

        }

    }

    /\*\*

     \* Close down Oracle connection.

     \*/

    public void closeConnection() {

        try {

            System.out.print("  Closing Connection...\n");

            con.close();

        } catch (SQLException e) {

            e.printStackTrace();

        }

    }

    /\*\*

     \* Sole entry point to the class and application.

     \* @param args Array of String arguments.

     \* @exception java.lang.InterruptedException

     \*            Thrown from the Thread class.

     \*/

     public static void main(String[] args)

            throws java.lang.InterruptedException {

        CallPLSQLProc mainPrg = new CallPLSQLProc();

        mainPrg.performProcCall();

        mainPrg.closeConnection();

    }

}

Problem Statement:

What is the result of the following program?

import java.sql.DriverManager;

import java.sql.Connection;

import java.sql.Types;

import java.sql.CallableStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import oracle.jdbc.driver.OracleCallableStatement;

import oracle.jdbc.driver.OracleTypes;

import oracle.jdbc.driver.OracleResultSet;

public class RefCursorExample {

final static String driverClass    = "oracle.jdbc.driver.OracleDriver";

final static String connectionURL  = "jdbc:oracle:thin:@localhost:1521:CUSTDB";

    final static String userID         = "scott";

    final static String userPassword   = "tiger";

    Connection   con                   = null;

    /\*\*

     \* Construct a RefCursorExample object. This constructor will create

     \* an Oracle database connection.

     \*/

    public RefCursorExample() {

        try {

            System.out.print("  Loading JDBC Driver  -> " + driverClass             + "\n");

            Class.forName(driverClass).newInstance();

System.out.print("  Connecting to        -> " + connectionURL + "\n");

this.con = DriverManager.getConnection(connectionURL, userID, userPassword);

System.out.print("  Connected as         -> " + userID + "\n\n");

        } catch (ClassNotFoundException e) {

            e.printStackTrace();

        } catch (InstantiationException e) {

            e.printStackTrace();

        } catch (IllegalAccessException e) {

            e.printStackTrace();

        } catch (SQLException e) {

            e.printStackTrace();

        }

    }

    /\*\*

     \* This method is used to return a REF CURSOR that will be used to

     \* retrieve data from a result set. This REF CUSROR is retrieved by

     \* the JDBC program into a ResultSet.

     \* This method Uses the OracleCallableStatement and OracleResultSet

     \* classes.

     \*/

    public void performRefCursor() {

        OracleCallableStatement oraCallStmt   = null;

        OracleResultSet         deptResultSet = null;

System.out.println("Using OracleCallableStatement / OracleResultSet");

System.out.println("-----------------------------------------------");

        try {

            oraCallStmt = (OracleCallableStatement) con.prepareCall(

                "{? = call ref\_cursor\_package.get\_dept\_ref\_cursor(?)}"

            );

            oraCallStmt.registerOutParameter(1, OracleTypes.CURSOR);

            oraCallStmt.setInt(2, 104);

            oraCallStmt.execute();

            deptResultSet = (OracleResultSet) oraCallStmt.getCursor(1);

            while (deptResultSet.next()) {

                System.out.println(

                    " - " +

    deptResultSet.getString(2) + " (" + deptResultSet.getInt(1) + "), " +

                    deptResultSet.getString(3)

                );

            }

            System.out.println();

            oraCallStmt.close();

        } catch (SQLException e) {

            e.printStackTrace();

        }

    }

    /\*\*

     \* This method is used to return a REF CURSOR that will be used to

     \* retrieve data from a result set. This REF CUSROR is retrieved by

     \* the JDBC program into a ResultSet.

     \*

     \* This method Uses the the regular CallableStatement and ResultSet

     \* classes.

     \*/

    public void performRefCursor2() {

        CallableStatement cstmt = null;

        ResultSet         rset  = null;

        System.out.println("Using CallableStatement / ResultSet");

        System.out.println("-----------------------------------");

        try {

            cstmt = con.prepareCall(

                "{? = call ref\_cursor\_package.get\_dept\_ref\_cursor(?)}"

            );

            cstmt.registerOutParameter(1, OracleTypes.CURSOR);

            cstmt.setInt(2, 104);

            cstmt.execute();

            rset = (ResultSet) cstmt.getObject(1);

            while (rset.next()) {

                System.out.println(

                    " - " +

                    rset.getString(2) + " (" + rset.getInt(1) + "), " +

                    rset.getString(3)

                );

            }

            System.out.println();

            cstmt.close();

        } catch (SQLException e) {

            e.printStackTrace();

        }

    }

    /\*\*

     \* Close down Oracle connection.

     \*/

    public void closeConnection() {

        try {

            System.out.print("  Closing Connection...\n");

            con.close();

        } catch (SQLException e) {

            e.printStackTrace();

        }

    }

    /\*\*

     \* Sole entry point to the class and application.

     \* @param args Array of String arguments.

     \* @exception java.lang.InterruptedException

     \*            Thrown from the Thread class.

     \*/

    public static void main(String[] args)

            throws java.lang.InterruptedException {

        RefCursorExample mainPrg = new RefCursorExample();

        mainPrg.performRefCursor();

        mainPrg.performRefCursor2();

        mainPrg.closeConnection();

    }

}

**Problem Statement:**

What is the output of the following program?

package com.handson;

import java.lang.annotation.Retention;

import java.lang.annotation.RetentionPolicy;

import java.lang.reflect.Method;

// A single-member annotation.

@Retention(RetentionPolicy.RUNTIME)

@interface MySingle {

int value();        // this variable name must be value

}

class Single {

// Annotate a method using a marker.

@MySingle(100)

public static void myMeth() {

Single ob = new Single();

try {

Method meth  = ob.getClass().getMethod("myMeth");

MySingle anno = meth.getAnnotation(MySingle.class);

System.out.println(anno.value());

} catch (NoSuchMethodException nsme) {

System.out.println("Method Not Found.");

}

}

public static void main(String args[]) {

myMeth();

}

}

**Problem Statement:**

What is the output of the following program?

package com.handson;

import java.lang.annotation.Retention;

import java.lang.annotation.RetentionPolicy;

import java.lang.reflect.Method;

// An annotation type declaration that includes defaults.

@Retention(RetentionPolicy.RUNTIME)

@interface MyAnno {

String str() default "Testing";

int val() default 9000;

}

class AnnoDefaultValues {

// Annotate a method using the default values.

@MyAnno()

public static void myMeth() {

AnnoDefaultValues ob = new AnnoDefaultValues();

// Obtain the annotation for this method

// and display the values of the members.

try {

Class cls = ob.getClass();

Method meth = cls.getMethod("myMeth");

MyAnno anno = meth.getAnnotation(MyAnno.class);

System.out.println(anno.str() + " " + anno.val());

} catch (NoSuchMethodException nsme) {

System.out.println("Method Not Found.");

}

}

public static void main(String args[]) {

myMeth();

}

**Problem Statement:**

What is the output of the following program?

package com.handson;

import java.lang.annotation.Annotation;

import java.lang.annotation.Retention;

import java.lang.annotation.RetentionPolicy;

import java.lang.reflect.Method;

@Retention(RetentionPolicy.RUNTIME)

@interface MyAnno {

String str();

int val();

}

@Retention(RetentionPolicy.RUNTIME)

@interface What {

String description();

}

@What(description = "An annotation test class")

@MyAnno(str = "AllAnnotations", val=99)

class AllAnnotations {

@What(description = "An annotation test method")

@MyAnno(str = "Testing", val=100)

public static void myMeth() {

AllAnnotations ob = new AllAnnotations();

try {

Annotation annos[] = ob.getClass().getAnnotations();

// Display all annotations for AllAnnotations

System.out.println("All annotations for AllAnnotations: ");

for(Annotation a : annos)

System.out.println(a);

System.out.println();

// Display all annotations for myMeth.

Method meth = ob.getClass().getMethod("myMeth");

annos = meth.getAnnotations();

System.out.println("All annotations for myMeth: ");

for(Annotation a : annos)

System.out.println(a);

} catch (NoSuchMethodException nsme) {

System.out.println("Method Not Found.");

}

}

public static void main(String[] args) {

myMeth();

}

}