# LANGUAGE FUNDAMENTALS

1. Which four options describe the correct default values for array elements of the types indicated?

    1. int -> 0
    2. String -> "null"
    3. Dog -> null
    4. char -> '\u0000'
    5. float -> 0.0f
    6. boolean -> true

    **A.** 1, 2, 3, 4

    **B.** 1, 3, 4, 5

    **C.** 2, 4, 5, 6

    **D.** 3, 4, 5, 6

2. Which one of these lists contains only Java programming language keywords?

    **A.** class, if, void, long, Int, continue

    **B.** goto, instanceof, native, finally, default, throws

    **C.** try, virtual, throw, final, volatile, transient

    **D.** strictfp, constant, super, implements, do

    **E.** byte, break, assert, switch, include

3. Which will legally declare, construct, and initialize an array?

    **A.** int [] myList = {"1", "2", "3"};

    **B.** int [] myList = (5, 8, 2);

    **C.** int myList [] [] = {4,9,7,0};

    **D.** int myList [] = {4, 3, 7};

4. Which is a reserved word in the Java programming language?

**A.** method

**B.** native

**C.** subclasses

**D.** reference

**E.** array

5. Which is a valid keyword in java?

**A.** interface

**B.** string

**C.** Float

**D.** unsigned

6. Which three are legal array declarations?

1. int [] myScores [];
2. char [] myChars;
3. int [6] myScores;
4. Dog myDogs [];
5. Dog myDogs [7];

**A.** 1, 2, 4

**B.** 2, 4, 5

**C.** 2, 3, 4

**D.** All are correct.

7.

```
public interface Foo
{
   int k = 4; /* Line 3 */
}
```

Which three piece of codes are equivalent to line 3?

1. final int k = 4;
2. public int k = 4;
3. static int k = 4;
4. abstract int k = 4;
5. volatile int k = 4;
6. protected int k = 4;

**A.** 1, 2 and 3

**B.** 2, 3 and 4

**C.** 3, 4 and 5

**D.** 4, 5 and 6

8. Which one of the following will declare an array and initialize it with five numbers?

**A.** Array a = new Array(5);

**B.** int [] a = {23,22,21,20,19};

**C.** int a [] = new int[5];

**D.** int [5] array;

9. Which three are valid declarations of a char?

1. char c1 = 064770;
2. char c2 = 'face';
3. char c3 = 0xbeef;
4. char c4 = \u0022;
5. char c5 = '\iface';
6. char c6 = '\uface';

**A.** 1, 2, 4

**B.** 1, 3, 6

**C.** 3, 5

**D.** 5 only

10. Which is the valid declarations within an interface definition?

**A.** public double methoda();

**B.** public final double methoda();

**C.** static void methoda(double d1);

**D.** protected void methoda(double d1);

11. Which one is a valid declaration of a boolean?

**A.** boolean b1 = 0;

**B.** boolean b2 = 'false';

**C.** boolean b3 = false;

**D.** boolean b4 = Boolean.false();

**E.** boolean b5 = no;

12. Which three are valid declarations of a float?

1. float f1 = -343;
2. float f2 = 3.14;
3. float f3 = 0x12345;
4. float f4 = 42e7;
5. float f5 = 2001.0D;
6. float f6 = 2.81F;

**A.** 1, 2, 4

**B.** 2, 3, 5

**C.** 1, 3, 6

**D.** 2, 4, 6

13. Which is a valid declarations of a String?

**A.** String s1 = null;

**B.** String s2 = 'null';

**C.** String s3 = (String) 'abc';

**D.** String s4 = (String) '\ufeed';

14. What is the numerical range of a char?

**A.** -128 to 127

**B.** $-(2^{15)}$ to $(2^{15}) - 1$

**C.** 0 to 32767

**D.** 0 to 65535

www.vtucs.com

# INNER CLASSES

1. Which is true about an anonymous inner class?

   **A.** It can extend exactly one class and implement exactly one interface.

   **B.** It can extend exactly one class and can implement multiple interfaces.

   **C.** It can extend exactly one class or implement exactly one interface.

   **D.** It can implement multiple interfaces regardless of whether it also extends a class.

2.
```
class Boo
{
   Boo(String s) { }
   Boo() { }
}
class Bar extends Boo
{
   Bar() { }
   Bar(String s) {super(s);}
   void zoo()
   {
   // insert code here
   }
}
```

   which one create an anonymous inner class from within class Bar?

   **A.** Boo f = new Boo(24) { };

   **B.** Boo f = new Bar() { };

   **C.** Bar f = new Boo(String s) { };

   **D.** Boo f = new Boo.Bar(String s) { };

3. Which is true about a method-local inner class?

   **A.** It must be marked final.

**B.** It can be marked abstract.

**C.** It can be marked public.

**D.** It can be marked static.

4. Which statement is true about a static nested class?

**A.** You must have a reference to an instance of the enclosing class in order to instantiate it.

**B.** It does not have access to nonstatic members of the enclosing class.

**C.** It's variables and methods must be static.

**D.** It must extend the enclosing class.

5. Which constructs an anonymous inner class instance?

**A.** Runnable r = new Runnable() { };

**B.** Runnable r = new Runnable(public void run() { });

**C.** Runnable r = new Runnable { public void run(){}};

**D.** System.out.println(new Runnable() {public void run() { }});

6.

```
class Foo
{
   class Bar{ }
}
class Test
{
   public static void main (String [] args)
   {
      Foo f = new Foo();
      /* Line 10: Missing statement ? */
   }
```

```
}
```

which statement, inserted at line 10, creates an instance of Bar?

**A.** Foo.Bar b = new Foo.Bar();

**B.** Foo.Bar b = f.new Bar();

**C.** Bar b = new f.Bar();

**D.** Bar b = f.new Bar();

7.

```
public class MyOuter
{
   public static class MyInner
   {
      public static void foo() { }
   }
}
```

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

**A.** MyOuter.MyInner m = new MyOuter.MyInner();

**B.** MyOuter.MyInner mi = new MyInner();

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

**D.** MyInner mi = new MyOuter.MyInner();

# DECLARATIONS AND ACCESS CONTROL

1. You want subclasses in any package to have access to members of a superclass. Which is the most restrictive access that accomplishes this objective?

   **A.** public

   **B.** private

   **C.** protected

   **D.** transient

2.
```
public class Outer
{
   public void someOuterMethod()
   {
      //Line 5
   }
   public class Inner { }

   public static void main(String[] argv)
   {
      Outer ot = new Outer();
      //Line 10
   }
}
```

   Which of the following code fragments inserted, will allow to compile?

   **A.** new Inner(); //At line 5

   **B.** new Inner(); //At line 10

   **C.** new ot.Inner(); //At line 10

   **D.** new Outer.Inner(); //At line 10

3.
```
interface Base
{
   boolean m1 ();
   byte m2(short s);
}
```

   which two code fragments will compile?

1. interface Base2 implements Base {}
2. abstract class Class2 extends Base
   { public boolean m1(){ return true; }}
3. abstract class Class2 implements Base {}
4. abstract class Class2 implements Base
   { public boolean m1(){ return (7 > 4); }}
5. abstract class Class2 implements Base
   { protected boolean m1(){ return (5 > 7) }}

**A.** 1 and 2 **B.** 2 and 3

**C.** 3 and 4 **D.** 1 and 5

4. Which three form part of correct array declarations?

   1. public int a [ ]
   2. static int [ ] a
   3. public [ ] int a
   4. private int a [3]
   5. private int [3] a [ ]
   6. public final int [ ] a

**A.** 1, 3, 4 **B.** 2, 4, 5

**C.** 1, 2, 6 **D.** 2, 5, 6

5.
```
public class Test { }
```
What is the prototype of the default constructor?

**A.** Test( ) **B.** Test(void)

**C.** public Test( ) **D.** public Test(void)

6. What is the most restrictive access modifier that will allow members of one class to have access to members of another class in the same package?

**A.** public **B.** abstract

**C.** protected

**D.** synchronized

**E.** default access

7. Which of the following is/are legal method declarations?

   1. protected abstract void m1();
   2. static final void m1(){}
   3. synchronized public final void m1() {}
   4. private native void m1();

   **A.** 1 and 3

   **B.** 2 and 4

   **C.** 1 only

   **D.** All of them are legal declarations.

8. Which cause a compiler error?

   **A.** int[ ] scores = {3, 5, 7};

   **B.** int [ ][ ] scores = {2,7,6}, {9,3,45};

   **C.** String cats[ ] = {"Fluffy", "Spot", "Zeus"};

   **D.** boolean results[ ] = new boolean [] {true, false, true};

   **E.** Integer results[ ] = {new Integer(3), new Integer(5), new Integer(8)};

9. Which three are valid method signatures in an interface?

   1. private int getArea();
   2. public float getVol(float x);
   3. public void main(String [] args);
   4. public static void main(String [] args);

5. boolean setFlag(Boolean [] test);

**A.** 1 and 2 **B.** 2, 3 and 5

**C.** 3, 4, and 5 **D.** 2 and 4

10. You want a class to have access to members of another class in the same package. Which is the most restrictive access that accomplishes this objective?

**A.** public **B.** private

**C.** protected **D.** default access

11. What is the widest valid returnType for methodA in line 3?

```
public class ReturnIt
{
   returnType methodA(byte x, double y) /* Line 3 */
   {
      return (long)x / y * 2;
   }
}
```

**A.** int **B.** byte

**C.** long **D.** double

12.

```
class A
{
   protected int method1(int a, int b)
   {
      return 0;
   }
}
```

Which is valid in a class that extends class A?

**A.** public int method1(int a, int b) {return 0; }

**B.** private int method1(int a, int b) { return 0; }

**C.** public short method1(int a, int b) { return 0; }

**D.** static protected int method1(int a, int b) { return 0; }

13. Which one creates an instance of an array?

**A.** int[ ] ia = new int[15];

**B.** float fa = new float[20];

**C.** char[ ] ca = "Some String";

**D.** int ia[ ] [ ] = { 4, 5, 6 }, { 1,2,3 };

14. Which two of the following are legal declarations for nonnested classes and interfaces?

1. final abstract class Test {}
2. public static interface Test {}
3. final public class Test {}
4. protected abstract class Test {}
5. protected interface Test {}
6. abstract public class Test {}

**A.** 1 and 4

**B.** 2 and 5

**C.** 3 and 6

**D.** 4 and 6

15. Which of the following class level (nonlocal) variable declarations will not compile?

**A.** protected int a;

**B.** transient int b = 3;

**C.** private synchronized int e;

**D.** volatile int d;

16. Which two cause a compiler error?

```
1. float[ ] f = new float(3);
2. float f2[ ] = new float[ ];
3. float[ ]f1 = new float[3];
4. float f3[ ] = new float[3];
5. float f5[ ] = {1.0f, 2.0f, 2.0f};
```

**A.** 2, 4

**B.** 3, 5

**C.** 4, 5

**D.** 1, 2

17. Given a method in a protected class, what access modifier do you use to restrict access to that method to only the other members of the same class?

**A.** final

**B.** static

**C.** private

**D.** protected

**E.** volatile

18. Which is a valid declaration within an interface?

**A.** public static short stop = 23;

**B.** protected short stop = 23;

**C.** transient short stop = 23;

**D.** final void madness(short stop);

# EXCEPTIONS

1. What will be the output of the program?

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

**A.** Finally

**B.** Compilation fails.

**C.** The code runs with no output.

**D.** An exception is thrown at runtime.

2. What will be the output of the program?

```
try
{
   int x = 0;
   int y = 5 / x;
}
catch (Exception e)
{
   System.out.println("Exception");
}
catch (ArithmeticException ae)
{
   System.out.println(" Arithmetic Exception");
}
```

```
System.out.println("finished");
```

**A.** finished

**B.** Exception

**C.** Compilation fails.

**D.** Arithmetic Exception

3. What will be the output of the program?

```
public class X
{
   public static void main(String [] args)
   {
      try
      {
         badMethod();
         System.out.print("A");
      }
      catch (Exception ex)
      {
         System.out.print("B");
      }
      finally
      {
         System.out.print("C");
      }
      System.out.print("D");
   }
   public static void badMethod()
   {
      throw new Error(); /* Line 22 */
   }
}
```

**A.** ABCD

**B.** Compilation fails.

**C.** C is printed before exiting with an error message.

**D.** BC is printed before exiting with an error message.

4. What will be the output of the program?

```
public class X
{
   public static void main(String [] args)
   {
      try
      {
         badMethod();
         System.out.print("A");
      }
      catch (RuntimeException ex) /* Line 10 */
      {
         System.out.print("B");
      }
      catch (Exception ex1)
      {
         System.out.print("C");
      }
      finally
      {
         System.out.print("D");
      }
      System.out.print("E");
   }
   public static void badMethod()
   {
      throw new RuntimeException();
   }
}
```

**A.** BD

**B.** BCD

**C.** BDE

**D.** BCDE

5. What will be the output of the program?

```
public class RTExcept
{
   public static void throwit ()
   {
      System.out.print("throwit ");
      throw new RuntimeException();
```

```
    }
    public static void main(String [] args)
    {
        try
        {
            System.out.print("hello ");
            throwit();
        }
        catch (Exception re )
        {
            System.out.print("caught ");
        }
        finally
        {
            System.out.print("finally ");
        }
        System.out.println("after ");
    }
}
```

**A.** hello throwit caught

**B.** Compilation fails

**C.** hello throwit RuntimeException caught after

**D.** hello throwit caught finally after

6. What will be the output of the program?

```
public class Test
{
    public static void aMethod() throws Exception
    {
        try /* Line 5 */
        {
            throw new Exception(); /* Line 7 */
        }
        finally /* Line 9 */
        {
            System.out.print("finally "); /* Line 11 */
        }
    }
```

```
    public static void main(String args[])
    {
        try
        {
            aMethod();
        }
        catch (Exception e) /* Line 20 */
        {
            System.out.print("exception ");
        }
        System.out.print("finished"); /* Line 24 */
    }
}
```

**A.** finally

**B.** exception finished

**C.** finally exception finished

**D.** Compilation fails

7. What will be the output of the program?

```
public class X
{
    public static void main(String [] args)
    {
        try
        {
            badMethod();
            System.out.print("A");
        }
        catch (Exception ex)
        {
            System.out.print("B");
        }
        finally
        {
            System.out.print("C");
        }
        System.out.print("D");
    }
```

```
   public static void badMethod() {}
}
```

**A.** AC

**B.** BC

**C.** ACD

**D.** ABCD

8. What will be the output of the program?

```
public class X
{
   public static void main(String [] args)
   {
      try
      {
         badMethod(); /* Line 7 */
         System.out.print("A");
      }
      catch (Exception ex) /* Line 10 */
      {
         System.out.print("B"); /* Line 12 */
      }
      finally /* Line 14 */
      {
         System.out.print("C"); /* Line 16 */
      }
      System.out.print("D"); /* Line 18 */
   }
   public static void badMethod()
   {
      throw new RuntimeException();
   }
}
```

**A.** AB

**B.** BC

**C.** ABC

**D.** BCD

9. What will be the output of the program?

```
public class MyProgram
```

```
{
   public static void main(String args[])
   {
      try
      {
         System.out.print("Hello world ");
      }
      finally
      {
         System.out.println("Finally executing ");
      }
   }
}
```

**A.** Nothing. The program will not compile because no exceptions are specified.

**B.** Nothing. The program will not compile because no catch clauses are specified.

**C.** Hello world.

**D.** Hello world Finally executing

10. What will be the output of the program?

```
class Exc0 extends Exception { }
class Exc1 extends Exc0 { } /* Line 2 */
public class Test
{
   public static void main(String args[])
   {
      try
      {
         throw new Exc1(); /* Line 9 */
      }
      catch (Exc0 e0) /* Line 11 */
      {
         System.out.println("Ex0 caught");
      }
      catch (Exception e)
      {
         System.out.println("exception caught");
      }
   }
```

```
}
```

**A.** Ex0 caught

**B.** exception caught

**C.** Compilation fails because of an error at line 2.

**D.** Compilation fails because of an error at line 9.

www.vtucs.com

# THREADS

1. What is the name of the method used to start a thread execution?

   **A.** init();  **B.** start();

   **C.** run();  **D.** resume();

2. Which two are valid constructors for Thread?

   1. Thread(Runnable r, String name)
   2. Thread()
   3. Thread(int priority)
   4. Thread(Runnable r, ThreadGroup g)
   5. Thread(Runnable r, int priority)

   **A.** 1 and 3  **B.** 2 and 4

   **C.** 1 and 2  **D.** 2 and 5

3. Which three are methods of the Object class?

   1. notify();
   2. notifyAll();
   3. isInterrupted();
   4. synchronized();
   5. interrupt();
   6. wait(long msecs);
   7. sleep(long msecs);
   8. yield();

   **A.** 1, 2, 4  **B.** 2, 4, 5

   **C.** 1, 2, 6  **D.** 2, 3, 4

4.
```
class X implements Runnable
{
   public static void main(String args[])
   {
```

```
    /* Missing code? */
  }
  public void run() {}
}
```

Which of the following line of code is suitable to start a thread ?

**A.** Thread t = new Thread(X);

**B.** Thread t = new Thread(X); t.start();

**C.** X run = new X(); Thread t = new Thread(run); t.start();

**D.** Thread t = new Thread(); x.run();

5. Which cannot directly cause a thread to stop executing?

**A.** Calling the SetPriority() method on a Thread object.

**B.** Calling the wait() method on an object.

**C.** Calling notify() method on an object.

**D.** Calling read() method on an InputStream object.

6. Which two of the following methods are defined in class Thread?

1. start()
2. wait()
3. notify()
4. run()
5. terminate()

**A.** 1 and 4

**B.** 2 and 3

**C.** 3 and 4

**D.** 2 and 4

7. Which three guarantee that a thread will leave the running state?

1. yield()

2. wait()
3. notify()
4. notifyAll()
5. sleep(1000)
6. aLiveThread.join()
7. Thread.killThread()

**A.** 1, 2 and 4 **B.** 2, 5 and 6

**C.** 3, 4 and 7 **D.** 4, 5 and 7

8. Which of the following will directly stop the execution of a Thread?

**A.** wait() **B.** notify()

**C.** notifyall() **D.** exits synchronized code

9. Which method must be defined by a class implementing the java.lang.Runnableinterface?

**A.** void run() **B.** public void run()

**C.** public void start() **D.** void run(int priority)

10. Which will contain the body of the thread?

**A.** run(); **B.** start();

**C.** stop(); **D.** main();

11. Which method registers a thread in a thread scheduler?

**A.** run(); **B.** construct();

**C.** start(); **D.** register();

12. Assume the following method is properly synchronized and called from a thread A on an object B:

wait(2000);

After calling this method, when will the thread A become a candidate to get another turn at the CPU?

**A.** After thread A is notified, or after two seconds.

**B.** After the lock on B is released, or after two seconds.

**C.** Two seconds after thread A is notified.

**D.** Two seconds after lock B is released.

13. Which of the following will not directly cause a thread to stop?

**A.** notify() **B.** wait()

**C.** InputStream access **D.** sleep()

14. Which class or interface defines the wait(), notify(),and notifyAll() methods?

**A.** Object **B.** Thread

**C.** Runnable **D.** Class

15.
```
public class MyRunnable implements Runnable
{
   public void run()
   {
      // some code here
   }
}
```

which of these will create and start this thread?

**A.** new Runnable(MyRunnable).start();

**B.** new Thread(MyRunnable).run();

**C.** new Thread(new MyRunnable()).start();

**D.** new MyRunnable().start();

www.vtucs.com

# OPERATORS AND ASSIGNMENTS

1. What will be the output of the program?

```
class PassA
{
   public static void main(String [] args)
   {
      PassA p = new PassA();
      p.start();
   }

   void start()
   {
      long [] a1 = {3,4,5};
      long [] a2 = fix(a1);
      System.out.print(a1[0] + a1[1] + a1[2] + " ");
      System.out.println(a2[0] + a2[1] + a2[2]);
   }

   long [] fix(long [] a3)
   {
      a3[1] = 7;
      return a3;
   }
}
```

**A.** 12 15

**B.** 15 15

**C.** 3 4 5 3 7 5

**D.** 3 7 5 3 7 5

2. What will be the output of the program?

```
class Test
{
   public static void main(String [] args)
   {
      Test p = new Test();
      p.start();
   }

   void start()
   {
```

```
        boolean b1 = false;
        boolean b2 = fix(b1);
        System.out.println(b1 + " " + b2);
    }

    boolean fix(boolean b1)
    {
        b1 = true;
        return b1;
    }
}
```

**A.** true true

**B.** false true

**C.** true false

**D.** false false

3. What will be the output of the program?

```
class PassS
{
    public static void main(String [] args)
    {
        PassS p = new PassS();
        p.start();
    }

    void start()
    {
        String s1 = "slip";
        String s2 = fix(s1);
        System.out.println(s1 + " " + s2);
    }

    String fix(String s1)
    {
        s1 = s1 + "stream";
        System.out.print(s1 + " ");
        return "stream";
    }
}
```

**A.** slip stream

**B.** slipstream stream

**C.** stream slip stream

**D.** slipstream slip stream

4. What will be the output of the program?

```
class BitShift
{
   public static void main(String [] args)
   {
      int x = 0x80000000;
      System.out.print(x + " and  ");
      x = x >>> 31;
      System.out.println(x);
   }
}
```

**A.** -2147483648 and 1

**B.** 0x80000000 and 0x00000001

**C.** -2147483648 and -1

**D.** 1 and -2147483648

5. What will be the output of the program?

```
class Equals
{
   public static void main(String [] args)
   {
      int x = 100;
      double y = 100.1;
      boolean b = (x = y); /* Line 7 */
      System.out.println(b);
   }
}
```

**A.** true

**B.** false

**C.** Compilation fails

**D.** An exception is thrown at runtime

6. What will be the output of the program?

```
class Test
{
   public static void main(String [] args)
   {
      int x=20;
      String sup = (x < 15) ? "small" : (x < 22)? "tiny" : "huge";
      System.out.println(sup);
   }
}
```

**A.** small

**B.** tiny

**C.** huge

**D.** Compilation fails

7. What will be the output of the program?

```
class Test
{
   public static void main(String [] args)
   {
      int x= 0;
      int y= 0;
      for (int z = 0; z < 5; z++)
      {
         if (( ++x > 2 ) && (++y > 2))
         {
            x++;
         }
      }
      System.out.println(x + " " + y);
```

```
    }
}
```

**A.** 5 2

**B.** 5 3

**C.** 6 3

**D.** 6 4

8. What will be the output of the program?

```
class Test
{
    public static void main(String [] args)
    {
        int x= 0;
        int y= 0;
        for (int z = 0; z < 5; z++)
        {
            if (( ++x > 2 ) || (++y > 2))
            {
                x++;
            }
        }
    System.out.println(x + " " + y);
    }
}
```

**A.** 5 3

**B.** 8 2

**C.** 8 3

**D.** 8 5

9. What will be the output of the program?

```
class Bitwise
{
    public static void main(String [] args)
    {
        int x = 11 & 9;
        int y = x ^ 3;
        System.out.println( y | 12 );
    }
```

```
}
```

**A.** 0

**B.** 7

**C.** 8

**D.** 14

10. What will be the output of the program?

```
class SSBool
{
   public static void main(String [] args)
   {
      boolean b1 = true;
      boolean b2 = false;
      boolean b3 = true;
      if ( b1 & b2 | b2 & b3 | b2 ) /* Line 8 */
         System.out.print("ok ");
      if ( b1 & b2 | b2 & b3 | b2 | b1 ) /*Line 10*/
         System.out.println("dokey");
   }
}
```

**A.** ok

**B.** dokey

**C.** ok dokey

**D.** No output is produced

**E.** Compilation error

11. What will be the output of the program?

```
class SC2
{
   public static void main(String [] args)
   {
      SC2 s = new SC2();
      s.start();
```

```
    }

    void start()
    {
        int a = 3;
        int b = 4;
        System.out.print(" " + 7 + 2 + " ");
        System.out.print(a + b);
        System.out.print(" " + a + b + " ");
        System.out.print(foo() + a + b + " ");
        System.out.println(a + b + foo());
    }

    String foo()
    {
        return "foo";
    }
}
```

**A.** 9 7 7 foo 7 7foo

**B.** 72 34 34 foo34 34foo

**C.** 9 7 7 foo34 34foo

**D.** 72 7 34 foo34 7foo

12. What will be the output of the program?

```
class Test
{
    static int s;
    public static void main(String [] args)
    {
        Test p = new Test();
        p.start();
        System.out.println(s);
    }

    void start()
    {
        int x = 7;
        twice(x);
```

```
        System.out.print(x + " ");
    }

    void twice(int x)
    {
        x = x*2;
        s = x;
    }
}
```

**A.** 7 7

**B.** 7 14

**C.** 14 0

**D.** 14 14

13. What will be the output of the program?

```
class Two
{
    byte x;
}

class PassO
{
    public static void main(String [] args)
    {
        PassO p = new PassO();
        p.start();
    }

    void start()
    {
        Two t = new Two();
        System.out.print(t.x + " ");
        Two t2 = fix(t);
        System.out.println(t.x + " " + t2.x);
    }

    Two fix(Two tt)
    {
        tt.x = 42;
        return tt;
    }
}
```

**A.** null null 42

**B.** 0 0 42

**C.** 0 42 42

**D.** 0 0 0

14. What will be the output of the program?

```
class BoolArray
{
   boolean [] b = new boolean[3];
   int count = 0;

   void set(boolean [] x, int i)
   {
      x[i] = true;
      ++count;
   }

   public static void main(String [] args)
   {
      BoolArray ba = new BoolArray();
      ba.set(ba.b, 0);
      ba.set(ba.b, 2);
      ba.test();
   }

   void test()
   {
      if ( b[0] && b[1] | b[2] )
         count++;
      if ( b[1] && b[(++count - 2)] )
         count += 7;
      System.out.println("count = " + count);
   }
}
```

**A.** count = 0

**B.** count = 2

**C.** count = 3

**D.** count = 4

15. What will be the output of the program?

```
public class Test
{
   public static void leftshift(int i, int j)
   {
      i <<= j;
   }
   public static void main(String args[])
   {
      int i = 4, j = 2;
      leftshift(i, j);
      System.out.printIn(i);
   }
}
```

**A.** 2

**B.** 4

**C.** 8

**D.** 16

www.vtucs

# OBJECTS AND COLLECTIONS

1. Suppose that you would like to create an instance of a new Map that has an iteration order that is the same as the iteration order of an existing instance of a Map. Which concrete implementation of the Map interface should be used for the new instance?

   **A.** TreeMap

   **B.** HashMap

   **C.** LinkedHashMap

   **D.** The answer depends on the implementation of the existing instance.

2. Which class does not override the equals() and hashCode() methods, inheriting them directly from class Object?

   **A.** java.lang.String **B.** java.lang.Double

   **C.** java.lang.StringBuffer **D.** java.lang.Character

3. Which collection class allows you to grow or shrink its size and provides indexed access to its elements, but whose methods are not synchronized?

   **A.** java.util.HashSet **B.** java.util.LinkedHashSet

   **C.** java.util.List **D.** java.util.ArrayList

4. You need to store elements in a collection that guarantees that no duplicates are stored and all elements can be accessed in natural order. Which interface provides that capability?

   **A.** java.util.Map **B.** java.util.Set

   **C.** java.util.List **D.** java.util.Collection

5. Which interface does java.util.Hashtable implement?

   **A.** Java.util.Map **B.** Java.util.List

**C.** Java.util.HashTable **D.** Java.util.Collection

6. Which interface provides the capability to store objects using a key-value pair?

**A.** Java.util.Map **B.** Java.util.Set

**C.** Java.util.List **D.** Java.util.Collection

7. Which collection class allows you to associate its elements with key values, and allows you to retrieve objects in FIFO (first-in, first-out) sequence?

**A.** java.util.ArrayList **B.** java.util.LinkedHashMap

**C.** java.util.HashMap **D.** java.util.TreeMap

8. Which collection class allows you to access its elements by associating a key with an element's value, and provides synchronization?

**A.** java.util.SortedMap **B.** java.util.TreeMap

**C.** java.util.TreeSet **D.** java.util.Hashtable

9. Which is valid declaration of a float?

**A.** float f = 1F; **B.** float f = 1.0;

**C.** float f = "1"; **D.** float f = 1.0d;

10.

```
/* Missing Statement ? */
public class foo
{
   public static void main(String[]args)throws Exception
   {
      java.io.PrintWriter out = new java.io.PrintWriter();
      new java.io.OutputStreamWriter(System.out,true);
      out.println("Hello");
   }
```

```
}
```

What line of code should replace the missing statement to make this program compile?

**A.** No statement required.

**B.** import java.io.*;

**C.** include java.io.*;

**D.** import java.io.PrintWriter;

11. What is the numerical range of char?

**A.** 0 to 32767

**B.** 0 to 65535

**C.** -256 to 255

**D.** -32768 to 32767

12. Which of the following are Java reserved words?

1. run
2. import
3. default
4. implement

**A.** 1 and 2

**B.** 2 and 3

**C.** 3 and 4

**D.** 2 and 4

# GARBAGE COLLECTIONS

1.

```
void start() {
   A a = new A();
   B b = new B();
   a.s(b);
   b = null; /* Line 5 */
   a = null;  /* Line 6 */
   System.out.println("start completed"); /* Line 7 */
}
```

When is the B object, created in line 3, eligible for garbage collection?

**A.** after line 5

**B.** after line 6

**C.** after line 7

**D.** There is no way to be absolutely certain.

2.

```
class HappyGarbage01
{
   public static void main(String args[])
   {
      HappyGarbage01 h = new HappyGarbage01();
      h.methodA(); /* Line 6 */
   }
   Object methodA()
   {
      Object obj1 = new Object();
      Object [] obj2 = new Object[1];
      obj2[0] = obj1;
      obj1 = null;
      return obj2[0];
   }
}
```

Where will be the most chance of the garbage collector being invoked?

**A.** After line 9

**B.** After line 10

**C.** After line 11

**D.** Garbage collector never invoked in methodA()

3.

```
class Bar { }
class Test
{
   Bar doBar()
   {
      Bar b = new Bar(); /* Line 6 */
      return b; /* Line 7 */
   }
   public static void main (String args[])
   {
      Test t = new Test();  /* Line 11 */
      Bar newBar = t.doBar();  /* Line 12 */
      System.out.println("newBar");
      newBar = new Bar(); /* Line 14 */
      System.out.println("finishing"); /* Line 15 */
   }
}
```

At what point is the Bar object, created on line 6, eligible for garbage collection?

**A.** after line 12

**B.** after line 14

**C.** after line 7, when doBar() completes

**D.** after line 15, when main() completes

4.

```
class Test
{
   private Demo d;
   void start()
   {
      d = new Demo();
      this.takeDemo(d); /* Line 7 */
   } /* Line 8 */
   void takeDemo(Demo demo)
```

```
    {
        demo = null;
        demo = new Demo();
    }
}
```

When is the Demo object eligible for garbage collection?

**A.** After line 7

**B.** After line 8

**C.** After the start() method completes

**D.** When the instance running this code is made eligible for garbage collection.

5.

```
public class X
{
    public static void main(String [] args)
    {
        X x = new X();
        X x2 = m1(x); /* Line 6 */
        X x4 = new X();
        x2 = x4; /* Line 8 */
        doComplexStuff();
    }
    static X m1(X mx)
    {
        mx = new X();
        return mx;
    }
}
```

After line 8 runs. how many objects are eligible for garbage collection?

**A.** 0

**B.** 1

**C.** 2

**D.** 3

6.

```
public Object m()
{
    Object o = new Float(3.14F);
```

```
   Object [] oa = new Object[l];
   oa[0] = o; /* Line 5 */
   o = null;  /* Line 6 */
   oa[0] = null; /* Line 7 */
   return o; /* Line 8 */
}
```

When is the Float object, created in line 3, eligible for garbage collection?

**A.** just after line 5

**B.** just after line 6

**C.** just after line 7

**D.** just after line 8

6.

```
public Object m()
{
   Object o = new Float(3.14F);
   Object [] oa = new Object[l];
   oa[0] = o; /* Line 5 */
   o = null;  /* Line 6 */
   oa[0] = null; /* Line 7 */
   return o; /* Line 8 */
}
```

When is the Float object, created in line 3, eligible for garbage collection?

**A.** just after line 5

**B.** just after line 6

**C.** just after line 7

**D.** just after line 8

**Answer & Explanation**

**Answer:** Option **C**

**Explanation:**

Option A is wrong. This simply copies the object reference into the array.

Option B is wrong. The reference o is set to null, but, oa[0] still maintains the reference to the Float object.

Option C is correct. The thread of execution will then not have access to the object.

View Answer Workspace Report Discuss in Forum

7.

```
class X2
{
   public X2 x;
   public static void main(String [] args)
   {
      X2 x2 = new X2();  /* Line 6 */
      X2 x3 = new X2();  /* Line 7 */
      x2.x = x3;
      x3.x = x2;
      x2 = new X2();
      x3 = x2; /* Line 11 */
      doComplexStuff();
   }
}
```

after line 11 runs, how many objects are eligible for garbage collection?

**A.** 0

**B.** 1

**C.** 2

**D.** 3

8. What allows the programmer to destroy an object x?

**A.** x.delete()

**B.** x.finalize()

**C.** Runtime.getRuntime().gc()

**D.** Only the garbage collection system can destroy an object.