14.1 What will be the result of attempting to compile and run the following code?

```
public class RQ100_50 {
  public static void main(String[] args) {
    List<Integer> lst = new ArrayList<Integer>();  // (1)
    lst.add(2007);
    lst.add(2008);
    List<Number> numList = lst;                    // (2)
    for(Number n : numList)                        // (3)
      System.out.println(n + " ");
  }
}
```

Select the one correct answer.

(a) The code will fail to compile because of an error in (1).
(b) The code will fail to compile because of an error in (2).
(c) The code will fail to compile because of an error in (3).
(d) The code will compile, but throw a ClassCastException at runtime in (2).
(e) The code will compile and will print "2007 2008 ", when run.

14.2 What will be the result of attempting to compile and run the following code?

```
class Fruit {}
class Apple extends Fruit {}
class Orange extends Fruit {}
public class RQ100_60 {
  public static void main(String[] args) {
    ArrayList<Apple> aList = new ArrayList<Apple>();
    aList.add(new Apple());
    ArrayList bList = aList;          // (1)
    ArrayList<Orange> oList = bList;  // (2)
    oList.add(new Orange());
    System.out.println(aList);
  }
}
```

Select the one correct answer.

(a) The code will fail to compile because of errors in (1) and (2).
(b) The code will fail to compile because of an error in (1).
(c) The code will fail to compile because of an error in (2).
(d) The code will compile with an unchecked warning in both (1) and (2).
(e) The code will compile with an unchecked warning in (1).
(f) The code will compile with an unchecked warning in (2).
(g) The code will compile without warnings, but throw a ClassCastException at runtime in (2).
(h) The code will compile without warnings and will print "[Apple@hhhhhh, Orange@HHHHHHH]", when run. (hhhhhh and HHHHHHH represent some hash code.)

14.3 What will be the result of attempting to compile and run the following code?

```
public class RQ100_40 {
  public static void main(String[] args) {
    List <? super Integer> sList = new ArrayList<Number>(); //(1)
    int i = 2007;
    sList.add(i);
    sList.add(++i);              //(2)
    Number num = sList.get(0);              //(3)
  }
}
```

Select the one correct answer.

(a) The code will fail to compile because of an error in (1).
(b) The code will fail to compile because of an error in (2).
(c) The code will fail to compile because of an error in (3).
(d) The code will compile, but throw a ClassCastException at runtime at (3).
(e) The code will compile and execute normally.

14.4 What will be the result of attempting to compile the following code?

```
public class RQ100_70 {
  public static void main(String[] args) {
    List<Integer> glst1 = new ArrayList();       //(1)
    List nglst1 = glst1;                         //(2)
    List nglst2 = nglst1;                        //(3)
    List<Integer> glst2 = glst1;                 //(4)
  }
}
```

Select the one correct answer.

(a) The code will compile without any warnings.
(b) The code will compile with an unchecked warning in (1).
(c) The code will compile with an unchecked warning in (2).
(d) The code will compile with an unchecked warning in (3).
(e) The code will compile with an unchecked warning in (4).

14.5 Which occurrences of the type parameter T are illegal?

```
public class Box<T> {
  private T item;                                  // (1)
  private static T[] storage = new T[100];         // (2)
  public Box(T item) { this.item = item; }         // (3)
  public T getItem() { return item; }                    // (4)
  public void setItem(T newItem) { item = newItem; }     // (5)
  public static void getAllItems(T newItem) {            // (6)
    T temp;                                              // (7)
  }
}
```

Select the three correct answers.

(a) Occurrence of the type parameter T in (1).
(b) Occurrences of the type parameter T in (2).
(c) Occurrence of the type parameter T in (3).
(d) Occurrence of the type parameter T in (4).
(e) Occurrence of the type parameter T in (5).
(f) Occurrence of the type parameter T in (6).
(g) Occurrence of the type parameter T in (7).

14.6 Which statements are true about the following code?

```
public class RQ100_14 {
  public static void main(String[] args) {
    List legacyList = new ArrayList<Integer>(); // (1)
    List<?> anyList = legacyList;               // (2)
    legacyList = anyList;                       // (3)
  }
}
```

Select the one correct answer.

(a) The compiler will report errors in the code.
(b) The code will compile with an unchecked warning in (1).
(c) The code will compile with an unchecked warning in (2).
(d) The code will compile with an unchecked warning in (3).
(e) The code will compile without unchecked warnings.

14.7 Which declarations will compile without warnings?

Select the four correct answers.

(a) Map<Integer, Map<Integer, String>> map1 = new HashMap<Integer, HashMap<Integer, String>>();
(b) Map<Integer, HashMap<Integer, String>> map2 = new HashMap<Integer, HashMap<Integer, String>>();
(c) Map<Integer, Integer> map3 = new HashMap<Integer, Integer>();
(d) Map<? super Integer, ? super Integer> map4 = new HashMap<? super Integer, ? super Integer>();
(e) Map<? super Integer, ? super Integer> map5 = new HashMap<Number, Number>();
(f) Map<? extends Number, ? extends Number> map6 = new HashMap<Number, Number>();
(g) Map <?,?> map7 = new HashMap<?,?>();

14.8 Which statement is true about the following code?

```
class Fruit {}
class Apple extends Fruit {}
public class RQ100_15 {
  public static void main(String[] args) {
    List<? extends Apple> lst1 = new ArrayList<Fruit>(); // (1)
    List<? extends Fruit> lst2 = new ArrayList<Apple>(); // (2)
    List<? super Apple> lst3 = new ArrayList<Fruit>();   // (3)
    List<? super Fruit> lst4 = new ArrayList<Apple>();   // (4)
    List<?> lst5 = lst1;            // (5)
    List<?> lst6 = lst3;            // (6)
    List lst7 = lst6;               // (7)
    List<?> lst8 = lst7;            // (8)
  }
}
```

Select the one correct answer.

(a) (1) will compile, but (2) will not.
(b) (3) will compile, but (4) will not.
(c) (5) will compile, but (6) will not.
(d) (7) will compile, but (8) will not.
(e) None of the above.

14.9 Which statements can be inserted at (1) without the compiler reporting any errors?

```
public class RQ100_12 {
  public static void main(String[] args) {
    List lst = new ArrayList<String>();
    // (1) INSERT HERE
  }
}
```

Select the four correct answers.

(a) lst.add(null);
(b) lst.add("OK");
(c) lst.add(2007);
(d) String v1 = lst.get(0);
(e) Object v2 = lst.get(0);

14.10 Which declaration can be inserted at (1) so that the program compiles without errors?

```
public class RQ100_84 {
  // (1) INSERT DECLARATION HERE...
  public long getNum(String name) {
    Long number = accounts.get(name);
    return number == null ? 0 : number;
  }
  public void setNum(String name, long number) {
    accounts.put(name, number);
  }
}
```

Select the one correct answer.

(a) private Map<String, long> accounts = new HashMap<String, long>();
(b) private Map<String, Long> accounts = new HashMap<String, Long>();
(c) private Map<String<Long>> accounts = new HashMap<String<Long>>();
(d) private Map<String, Long> accounts = new Map<String, Long>();
(e) private Map accounts = new HashMap();

14.11 What will be the result of attempting to compile and run the following code?

```
public class RQ100_11 {
  public static void main(String[] args) {
    Set set = new TreeSet<String>();
    set.add("one");    set.add(2);
    set.add("three");  System.out.println(set);
  } }
```

Select the one correct answer.

(a) The program does not compile.
(b) The program compiles with unchecked warnings and prints the elements in the set.
(c) The program compiles without unchecked warnings and prints the elements in the set.
(d) The program compiles with unchecked warnings and throws an exception at runtime.
(e) The program compiles without unchecked warnings and throws an exception at runtime.

14.12 What will be the result of attempting to compile the following code?

```
class Vehicle { }
class Car extends Vehicle { }
class Sedan extends Car { }
class Garage<V> {
  private V v;
  public V get() { return this.v; }
  public void put(V v) { this.v = v; }
}
public class GarageAdmin {
  private Object object = new Object();
  private Vehicle vehicle = new Vehicle();
  private Car car = new Car();
  private Sedan sedan = new Sedan();
  public void doA(Garage g) {
    g.put(object);         // (1)
    g.put(vehicle);        // (2)
    g.put(car);           // (3)
    g.put(sedan);         // (4)
    object  = g.get();         // (5)
    vehicle = g.get();         // (6)
    car     = g.get();         // (7)
    sedan   = g.get();         // (8)
  }
}
```

Select the two correct answers.

(a) The call to the put() method in statements (1) - (4) will compile.
(b) The assignment statement (5) will compile.
(c) The assignment statements (6), (7), and (8) will compile.

14.13 What will be the result of attempting to compile the following code?

```
class Vehicle { }  class Car extends Vehicle { }
class Sedan extends Car { }
class Garage<V> {
  private V v;
  public V get() { return this.v; }
  public void put(V v) { this.v = v; }
}
public class GarageAdmin {
  private Object object = new Object();
  private Vehicle vehicle = new Vehicle();
  private Car car = new Car();
  private Sedan sedan = new Sedan();
  public void doB(Garage<Car> g) {
    g.put(object);             // (1)
    g.put(vehicle);            // (2)
    g.put(car);                // (3)
    g.put(sedan);              // (4)
    object  = g.get();         // (5)
    vehicle = g.get();         // (6)
    car     = g.get();         // (7)
    sedan   = g.get();         // (8)
  }
}
```

Select the two correct answers.

(a) The call to the put() method in statements (1) - (2) will compile.
(b) The call to the put() method in statements (3) - (4) will compile.
(c) The assignment statements (5), (6) and (7) will compile.
(d) The assignment statement (8) will compile.

14.14 What will be the result of attempting to compile the following code?

```
class Vehicle { }
class Car extends Vehicle { }
class Sedan extends Car { }
class Garage<V> {
  private V v;
  public V get() { return this.v; }
  public void put(V v) { this.v = v; }
}
public class GarageAdmin {
  private Object object = new Object();
  private Vehicle vehicle = new Vehicle();
  private Car car = new Car();
  private Sedan sedan = new Sedan();
  public void doC(Garage<?> g) {
    g.put(object);        // (1)
    g.put(vehicle);       // (2)
    g.put(car);           // (3)
    g.put(sedan);         // (4)
    object  = g.get();        // (5)
    vehicle = g.get();        // (6)
    car     = g.get();        // (7)
    sedan   = g.get();        // (8)
  }
}
```

Select the one correct answer.

(a) The call to the put() method in statements (1) - (2) will compile.
(b) The call to the put() method in statements (3) - (4) will compile.
(c) The assignment statement (5) will compile.
(d) The assignment statements (6), (7), and (8) will compile.

14.15 What will be the result of attempting to compile the following code?

```
class Vehicle { }     class Car extends Vehicle { }
class Sedan extends Car { }
class Garage<V> {
  private V v;
  public V get() { return this.v; }
  public void put(V v) { this.v = v; }
}
public class GarageAdmin {
  private Object object = new Object();
  private Vehicle vehicle = new Vehicle();
  private Car car = new Car();
  private Sedan sedan = new Sedan();
  public void doD(Garage<? extends Car> g) {
    g.put(object);        // (1)
    g.put(vehicle);       // (2)
    g.put(car);         // (3)
    g.put(sedan);        // (4)
    object  = g.get();        // (5)
    vehicle = g.get();        // (6)
    car     = g.get();        // (7)
    sedan   = g.get();        // (8)
  }
}
```

Select the one correct answer.

(a) The call to the put() method in statements (1) - (2) will compile.
(b) The call to the put() method in statements (3) - (4) will compile.
(c) The assignment statements (5), (6), and (7) will compile.
(d) The assignment statement (8) will compile.

14.16 What will be the result of attempting to compile the following code?

```
class Vehicle { }
class Car extends Vehicle { }
class Sedan extends Car { }
class Garage<V> {
  private V v;
  public V get() { return this.v; }
  public void put(V v) { this.v = v; }
}
public class GarageAdmin {
  private Object object = new Object();
  private Vehicle vehicle = new Vehicle();
  private Car car = new Car();
  private Sedan sedan = new Sedan();
  public void doE(Garage<? super Car> g) {
    g.put(object);        // (1)
    g.put(vehicle);       // (2)
    g.put(car);         // (3)
    g.put(sedan);        // (4)
    object  = g.get();        // (5)
    vehicle = g.get();        // (6)
    car     = g.get();        // (7)
    sedan   = g.get();        // (8)
  }
}
```

Select the two correct answers.

(a) The call to the put() method in statements (1) - (2) will compile.
(b) The call to the put() method in statements (3) - (4) will compile.
(c) The assignment statement (5) will compile.
(d) The assignment statements (6), (7), and (8) will compile.

14.17 Given the following class declaration:

```
class ClassA<U> implements Comparable<U> {
  public int compareTo(U a) { return 0; }
}
```

Which class declarations below will compile without errors?

Select the three correct answers.

```
(a) class ClassB<U,V> extends ClassA<R> {}
(b) class ClassC<U,V> extends ClassA<U> {}
(c) class ClassD<U,V> extends ClassA<V, U> {}
(d) class ClassE<U> extends ClassA<Comparable<Number>> {}
(e) class ClassF<U extends Comparable<U> & Serializable> extends
                                                          ClassA<Number> {}
(f) class ClassG<U implements Comparable<U>> extends ClassA<Number> {}
(g) class ClassH<U extends Comparable<U>> extends ClassA<? extends Number> {}
(h) class ClassI<U extends String & Comparable<U>> extends ClassA<U> {}
(i) class ClassJ<U> extends ClassA<Integer> implements Comparable<Number>{}
```

14.18 Which statements can be inserted at (1) without the compiler reporting any errors?

```
public class RQ100_05 {
  public static void main(String[] args) {
    List<?> lst = new ArrayList<String>();
    // (1) INSERT HERE
  }
}
```

Select the two correct answers.
(a) lst.add(null);
(b) lst.add("OK");
(c) lst.add(2007);
(d) String v1 = lst.get(0);
(e) Object v2 = lst.get(0);

14.19 What will the program print when compiled and run?

```
public class RQ100_00 {
  public static void main(String[] args) {
    List<String> lst1 = new ArrayList<String>();
    List<Integer> lst2 = new ArrayList<Integer>();
    List<List<Integer>> lst3 = new ArrayList<List<Integer>>();
    System.out.print(lst1.getClass() + " ");
    System.out.print(lst2.getClass() + " ");
    System.out.println(lst3.getClass());
  }
}
```

Select the one correct answer.
(a) class java.util.ArrayList<String> class java.util.ArrayList<Integer> class java.util.ArrayList<List<Integer>>
(b) class java.util.ArrayList class java.util.ArrayList class java.util.ArrayList
(c) class java.util.List class java.util.List class java.util.List
(d) class java.util.List<String> class java.util.List<Integer> class java.util.List<List<Integer>>
(e) The program will not compile.
(f) The program compiles, but throws an exception when run.

14.20 Which declarations can be inserted at (1) without the compiler reporting any errors?

```
public class RQ100_01 {
  public static void main(String[] args) {
  //  (1) INSERT DECLARATIONS HERE
  }
  public static <E extends Number> List<E> justDoIt(List<? super E> nums) {
return null;
  }
}
```

Select the three correct answers.
(a) ArrayList<Integer> inParam = new ArrayList<Integer>(); ArrayList<Integer> returnValue = justDoIt(inParam);
(b) ArrayList<Integer> inParam = new ArrayList<Integer>(); List<Integer> returnValue = justDoIt(inParam);
(c) ArrayList<Integer> inParam = new ArrayList<Integer>(); List<Number> returnValue = justDoIt(inParam);
(d) List<Number> inParam = new ArrayList<Number>(); ArrayList<Integer> returnValue = justDoIt(inParam);
(e) List<Number> inParam = new ArrayList<Number>(); List<Number> returnValue = justDoIt(inParam);
(f) List<Integer> inParam = new ArrayList<Integer>(); List<Integer> returnValue = justDoIt(inParam);

14.21 The class java.lang.String implements the interface java.lang.CharSequence. Given the following code:

```
public class RQ100_02 {
  public static void main(String[] args) {
    List<String> lst = Arrays.asList("Java", "only", "promotes", "fun");
    Collection<String> resultList = delete4LetterWords(lst);
  }
  // (1) INSERT METHOD HEADER HERE
  {
    Collection<E> permittedWords = new ArrayList<E>();
    for (E word : words) {
      if (word.length() != 4) permittedWords.add(word);
    }
    return permittedWords;
  }
}
```

Which method header can be inserted at (1) so that the program compiles and runs without errors?
Select the one correct answer.
(a) static <E extends CharSequence> Collection<? extends CharSequence> delete4LetterWords(Collection<E> words)
(b) static <E extends CharSequence> List<E> delete4LetterWords(Collection<E> words)
(c) static <E extends CharSequence> Collection<E> delete4LetterWords(Collection<? extends CharSequence> words)
(d) static <E extends CharSequence> List<E> delete4LetterWords(Collection<? extends CharSequence> words)
(e) static <E extends CharSequence> Collection<E> delete4LetterWords(Collection<E> words)
(f) public <E super CharSequence> Collection<E> delete4LetterWords(Collection<E> words)

14.22 Which declaration can be inserted at (1) so that the program compiles and runs without errors?

```
public class RQ100_06 {
  public static void main(String[] args) {
    //  (1) INSERT DECLARATION HERE
    for (int i = 0; i <= 5; i++) {
      List<Integer> row = new ArrayList<Integer>();
      for (int j = 0; j <= i; j++)
        row.add(i * j);
      ds.add(row);
    }
    for (List<Integer> row : ds)
System.out.println(row);
  }
}
```

Select the one correct answer.

```
(a) List<List<Integer>> ds = new List<List<Integer>>();
(b) List<ArrayList<Integer>> ds = new ArrayList<ArrayList<Integer>>();
(c) List<List<Integer>> ds = new ArrayList<List<Integer>>();
(d) ArrayList<ArrayList<Integer>> ds = new ArrayList<ArrayList<Integer>>();
(e) List<List<Integer>> ds = new ArrayList<ArrayList<Integer>>();
(f) List<List, Integer> ds = new List<List, Integer>();
(g) List<List, Integer> ds = new ArrayList<List, Integer>();
(h) List<List, Integer> ds = new ArrayList<ArrayList, Integer>();
```

14.23 Which method declarations cannot be inserted independently at (2) to overload the method at (1)?
Select the two correct answers.
(a) static <T> void overloadMe(Collection<T> s1, List<T> s2) { }
(b) static <T> void overloadMe(List<T> s1, List<? extends T> s2) { }
(c) static <T> void overloadMe(List<T> s1, Collection<? super T> s2) { }
(d) static <T> void overloadMe(Collection<T> s1, Collection<? super T> s2) { }
(e) static <T> void overloadMe(Collection<T> s1, List<? super T> s2) { }
(f) static <T> void overloadMe(List<? extends T> s1, List<? super T> s2) { }

14.24 Which declarations can be inserted at (1) so that the program compiles and runs without errors?

```
public class RQ100_07 {
  public static void main(String[] args) {
    // (1) INSERT DECLARATION HERE
    appendAndPrint(lst, "hello");
  }
  static <T> void appendAndPrint(Collection<T> ds, T t) {
    ds.add(t);
    System.out.println(ds);
  }
}
```

Select the two correct answers.
(a) List<?> lst = new LinkedList<Object>();
(b) List<? extends Object> lst = new LinkedList<Object>();
(c) List<? super Object> lst = new LinkedList<Object>();
(d) List<Object> lst = new LinkedList<Object>();

14.25 Which method declaration can be inserted at (1) so that the program compiles without warnings?

```
public class RQ100_87 {
  public static void main(String[] args) {
    List raw = new ArrayList();
    raw.add("2007");
    raw.add(2008);
    raw.add("2009");
    justDoIt(raw);
  }
  // (1) INSERT METHOD DECLARATION HERE.
}
```

Select the one correct answer.

(a) static void justDoIt(List<Integer> lst) { }
(b) static void justDoIt(List<?> lst) { }
(c) static <T> void justDoIt(List<T> lst) { }
(d) None of the above.

14.26 Which method calls can be inserted at (1) so that the program compiles without warnings?

```
public class GenParam {
  public static void main(String[] args) {
    List<Number> numList = new ArrayList<Number>();
    List<Integer> intList = new ArrayList<Integer>();
    // (1) INSERT CODE HERE
  }
  static <T> void move(List<? extends T> lst1, List<? super T> lst2) { }
}
```

Select the three correct answers.

(a) GenParam.move(numList, intList);
(b) GenParam.<Number>move(numList, intList);
(c) GenParam.<Integer>move(numList, intList);
(d) GenParam.move(intList, numList);
(e) GenParam.<Number>move(intList, numList);
(f) GenParam.<Integer>move(intList, numList);

14.27 Which statement is true about the following code?

```
public class RQ100_86 {
  static void print1(List<String> lst) {     // (1)
    for(String element : lst) {
      System.out.print(element + " ");
    }   }
  static void print2(List<String> lst) {     // (2)
    for(Object element : lst) {
      System.out.print(element + " ");
    }   }
  static void print3(List<?> lst) {          // (3)
    for(Object element : lst) {
      System.out.print(element + " ");
    }   }
  static <T> void print4(List<T> lst) {    // (4)
    for(Object element : lst) {
      System.out.print(element + " ");
    }
  }
  static <T> void print5(List<T> lst) {     // (5)
    for(T element : lst) {
      System.out.print(element + " ");
    }
  }
}
```

Select the one correct answer.

(a) The formal type parameter specification for the methods in (1), (2), and (3) is missing.
(b) The generic methods in (4) and (5) should be declared in a generic class.
(c) The element type Object for the local variable element in the for(:) loop header of the method in (3) is inconsistent with the element type of the list.
(d) The element type Object for the local variable element in the for(:) loop header of the method in (4) is inconsistent with the element type of the list.
(e) The program will compile without warnings.
(f) None of the above.

14.28 Which statements are true about the following code?

```
class MyClass<V> {
      MyClass()    {System.out.println(this);}            // (1)
      MyClass(V v)  {System.out.println(v);}             // (2)
  <T> MyClass(T t) {System.out.println(t);}             // (3)
  <T> MyClass(T t, V v){System.out.println(t + ", " + v);}    // (4)
}
```

Select the two correct answers.

(a) The class attempts to declare four constructors.
(b) Only one of the two constructors in (2) and (3) can be declared in the class.
(c) A generic class cannot declare generic constructors.
(d) The compiler reports an error in (3), since the type parameter V is not used.
(e) The class compiles without problems.

14.29 Which declaration statement is not valid in the code below?

```
class AClass<V> {
      AClass()   {System.out.println(this);}               // (1)
  <T> AClass(T t) {System.out.println(t);}              // (2)
  <T> AClass(T t, V v) {System.out.println(t + ", " + v);}       // (3)
}
```

Select the one correct answer.

(a) AClass<String> ref1 = new AClass<String>();
(b) AClass<String> ref2 = new AClass<String>("one");
(c) AClass<String> ref3 = new AClass<String>(2007);
(d) AClass<String> ref4 = new <Integer>AClass<String>(2007);
(e) AClass<String> ref5 = new <String>AClass<String>("one");
(f) AClass<String> ref6 = new AClass<String>(2007, "one");
(g) AClass<String> ref7 = new <Integer>AClass<String>(2007, "one");
(h) AClass<String> ref8 = new <Integer>AClass<String>("one", 2007);

14.30 Which statements are true about the following code?

```
class SupX {
  public void set(Collection<?> c) {/*...*/} // (1)
}
class SubX extends SupX {
  public void set(List<?> l) {/*...*/}       // (2)
  public void set(Collection c) {/*...*/}    // (3)
}
//---------------------------------------
class SupY {
  public void set(Collection c) {/*...*/}    // (4)
}
class SubY extends SupY {
  public void set(Collection<?> c) {/*...*/} // (5)
}
```

Select the three correct answers.

(a) The method at (2) overloads the method at (1).
(b) The method at (2) overrides the method at (1).
(c) The method at (2) results in a compile-time error.
(d) The method at (3) overloads the method at (1).
(e) The method at (3) overrides the method at (1).
(f) The method at (3) results in a compile-time error.
(g) The method at (5) overloads the method at (4).
(h) The method at (5) overrides the method at (4).
(i) The method at (5) results in a compile-time error.

14.31 Which statements are true about the following code?

```
class SupC<T> {
  public void set(T t) {/*...*/}       // (1)
  public T get() {return null;}        // (2)
}
class SubC1<M,N> extends SupC<M> {
  public void set(N n) {/*...*/}       // (3)
  public N get() {return null;}        // (4)
}
class SubC2<M,N extends M> extends SupC<M> {
  public void set(N n) {/*...*/}       // (5)
  public N get() {return null;}        // (6)
}
```

Select the four correct answers.

(a) The method at (3) overloads the method at (1).
(b) The method at (3) overrides the method at (1).
(c) The method at (3) results in a compile-time error.
(d) The method at (4) overloads the method at (2).
(e) The method at (4) overrides the method at (2).
(f) The method at (4) results in a compile-time error.
(g) The method at (5) overloads the method at (1).
(h) The method at (5) overrides the method at (1).
(i) The method at (5) results in a compile-time error.
(j) The method at (6) overloads the method at (2).
(k) The method at (6) overrides the method at (2).
(l) The method at (6) results in a compile-time error.

14.32 Which types cannot be declared as generic types?

Select the three correct answers.

(a) Enum types
(b) Static member classes
(c) Any subclass of Throwable, i.e., exception classes
(d) Nested interfaces
(e) Anonymous classes
(f) Non-static member classes
(g) Local classes

14.33 What will be printed when the program is compiled and run?

```
class Tantrum<E extends Exception> {
  public void throwOne(E e) throws E {
    throw e;
  }
}
class TantrumException extends Exception {
  TantrumException(String str) {
    super(str);
  }
}
public class TakeException {
  public static void main(String[] args) {
    Tantrum<TantrumException> tantrum = new Tantrum<TantrumException>();
    try {
      tantrum.throwOne(new TantrumException("Tantrum thrown."));
    } catch (TantrumException te) {
      System.out.println(te.getMessage());
    }
  }
}
```

Select the one correct answer.

(a) The class Tantrum will not compile.
(b) The class TakeException will not compile.
(c) The program will compile, print "Tantrum thrown.", and terminate normally when run.
(d) The program will compile and will throw an exception and abort the execution when run.

14.34 What will be printed when the program is compiled and run?

```
public class CastAway {
  public static void main(String[] args) {
    Object obj = new ArrayList<Integer>();      // (1)
    List<?> list1 = (List<?>) obj;        // (2)
    List<?> list2 = (List) obj;         // (3)
    List          list3 = (List<?>) obj;          // (4)
    List<Integer> list4 = (List) obj;             // (5)
    List<Integer> list5 = (List<Integer>) obj;    // (6)
  }
}
```

Select the one correct answer.

(a) The program will not compile.
(b) The program will compile without any unchecked warnings. It will run with no output and terminate normally
(c) The program will compile without any unchecked warnings. When run, it will throw an exception.
(d) The program will compile, but issue unchecked warnings. It will run with no output and terminate normally.
(e) The program will compile, but issue unchecked warnings. When run, it will throw an exception.

14.35 What will be printed when the program is compiled and run?

```
public class InstanceTest2 {
  public static void main(String[] args) {
    List<Integer> intList = new ArrayList<Integer>();
    Set<Double> doubleSet = new HashSet<Double>();
    List<?>     list = intList;
    Set<?>      set = doubleSet;
    scuddle(intList);
    scuddle(doubleSet);
    scuddle(list);
    scuddle(set);
  }
  private static void scuddle(Collection<?> col) {
    if (col instanceof List<?>) {
      System.out.println("I am a list.");
    } else if (col instanceof Set<?>) {
      System.out.println("I am a set.");
    }
  }
}
```

Select the one correct answer.

(a) The method scuddle() will not compile.

(b) The method main() will not compile.

(c) The program will compile, but issue an unchecked warning in method scuddle(). It will run and terminate normally with the following output:

I am a list.
I am a set.
I am a list.
I am a set.

(d) The program will compile, but issue an unchecked warning in the method main(). When run, it will throw an exception.

(e) The program will compile without any unchecked warnings. It will run and terminate normally, with the following output:

I am a list.
I am a set.
I am a list.
I am a set.

(f) The program will compile without any unchecked warnings. It will run and terminate normally, with the following output:

I am a list.
I am a set.

(g) None of the above.

14.36 Which statements will compile without errors and unchecked warnings when inserted at (1)?

```
public class Restrictions<T> {
  public void test() {
    // (1) INSERT ASSIGNMENT HERE.
  }
}
```

Select the four correct answers.

```
(a) T ref = new T();
(b) T[] arrayRef = new T[10];
(c) List<T>[] arrayOfLists0 = { new List<T>(), new List<T>() };
(d) List<T>[] arrayOfLists1 = new List<T>[10];
(e) List<?>[] arrayOfLists2 = new List<?>[10];
(f) List   [] arrayOfLists3 = new List<?>[10];
(g) List<?>[] arrayOfLists4 = new List[10];
(h) List   [] arrayOfLists5 = new List[10];
(i) List<String>[] arrayOfLists6 = new List[10];
```

14.37 What will be printed when the program is compiled and run?

```
public class GenArrays {
  public static <E> E[] copy(E[] srcArray) {
    E[] destArray = (E[]) new Object[srcArray.length];
    int i = 0;
    for (E element : srcArray) {
      destArray[i++] = element;
    }
    return destArray;
  }
  public static void main(String[] args) {
    String[] sa = {"9", "1", "1" };
    String[] da = GenArrays.copy(sa);
    System.out.println(da[0]);
  }
}
```

Select the one correct answer.

(a) The program will not compile.

(b) The program will compile, but issue an unchecked warning. When run, it will print "9".

(c) The program will compile, but issue an unchecked warning. When run, it will throw an exception.

(d) The program will compile without any unchecked warnings. When run, it will print "9".

(e) The program will compile without any unchecked warnings. When run, it will throw an exception.

14.38 What is the result of compiling and running the following program?

```
import java.util.Arrays;
import java.util.List;
public class GenVarArgs {
  public static <T> void doIt(List<T>... aols) {      // (1)
    for(int i = 0; i < aols.length; i++) {
      System.out.print(aols[i] + " ");
    }
  }
  public static void main(String... args) {         // (2)
    List<String> ls1 = Arrays.asList("one", "two");
    List<String> ls2 = Arrays.asList("three", "four");
    List<String>[] aols = new List[] {ls1, ls2};       // (3)
    doIt(aols);              // (4)
  }
}
```

Select the one correct answer.

(a) The program does not compile because of errors in (1).

(b) The program does not compile because of errors in (2).

(c) The program does not compile because of errors in (3).

(d) The program does not compile because of errors in (4).

(e) The program compiles and prints: [one, two] [three, four]