Collections and Generics

Q: 01 Given:
34. HashMap props = new HashMap();
35. props.put("key45", "some value");
36. props.put("key12", "some other value");
37. props.put("key39", "yet another value");
38. Set s = props.keySet();
39. // insert code here
What, inserted at line 39, will sort the keys in the props HashMap?
A. Arrays.sort(s);
B. s = new TreeSet(s);
C. Collections.sort(s);
D. s = new SortedSet(s);
Answer: B

Q: 02 Click the Exhibit button.

```
1.  import java.util.*;
2.  public class TestSet {
3.     enum Example { ONE, TWO, THREE }
4.     public static void main(String[] args)
{
5.        Collection coll = new ArrayList();
6.        coll.add(Example.THREE);
7.        coll.add(Example.THREE);
8.        coll.add(Example.THREE);
9.        coll.add(Example.TWO);
10.       coll.add(Example.TWO);
11.       coll.add(Example.ONE);
12.       Set set = new HashSet(coll);
13.    }
14. }
```

Which statement is true about the set variable on line 12?

A. The set variable contains all six elements from the coll collection, and the order is guaranteed to be

preserved.

B. The set variable contains only three elements from the coll collection, and the order is guaranteed to be preserved.

C. The set variable contains all six elements from the coll collection, but the order is NOT guaranteed to be preserved.

D. The set variable contains only three elements from the coll collection, but the order is NOT guaranteed to be preserved.

Answer: D

Q: 03 Given:
23. Object [] myObjects = {
24. new Integer(12),
25. new String("foo"),
26. new Integer(5),
27. new Boolean(true)
28. };
29. Arrays.sort(myObjects);
30. for(int i=0; i<myObjects.length; i++) {
31. System.out.print(myObjects[i].toString());
32. System.out.print(" ");
33. }
What is the result?
A. Compilation fails due to an error in line 23.
B. Compilation fails due to an error in line 29.
C. A ClassCastException occurs in line 29.
D. A ClassCastException occurs in line 31.
E. The value of all four objects prints in natural order.
Answer: C

Q: 04 Click the Task button.

Place code into the class so that it compiles and generates the output
`answer=42`. Note: Code options may be used more than once.

Class

```
public class [Place here] {

    private [Place here] object;

    public [Place here] ([Place here] object) {
        this.object = object;
    }
    public [Place here] getObject() {
        return object;
    }

    public static void main(String[] args) {
        Gen<String> str = new Gen<String>("answer");
        Gen<Integer> intg = new Gen<Integer>(42);
        System.out.println(str.getObject() + "=" +
            intg.getObject());
    }
}
```

Code Options

```
Gen<T>

Gen<?>

Gen

?

T
```

Done

Solution:
1.Gen<T>
2.T
3.Gen
 4.T
 5.T

Q: 05 Click the Task button.

Given:

```
public void takeList(List<? extends String> list) {
    // insert code here
}
```

Place the Compilation Results on each code statement to indicate whether or not that code will compile if inserted into the takeList() method.

Code Statements

```
list.add("Foo");
```

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

```
list = new ArrayList<Object>();
```

```
String s = list.get(0);
```

```
Object o = list;
```

Compilation Results

```
Compilation succeeds
```

```
Compilation fails
```

Done

Solution:
1. list.add("foo");   ----------   Compilation fails
2. list = new ArrayList<String>();   ------Compilation succeeds
3.list=new ArrayList<Object>( ); ---- Compilation fails
4. String s = list.get(0); ------ Compilation succeeds
5. Object o = list; ----- Compilation succeeds

Q: 06 Given:
1. public class Person {
2. private String name;
3. public Person(String name) { this.name = name; }
4. public boolean equals(Person p) {

5. return p.name.equals(this.name);
6. }
7. }
Which statement is true?
A. The equals method does NOT properly override the Object.equals method.
B. Compilation fails because the private attribute p.name cannot be accessed in line 5.
C. To work correctly with hash-based data structures, this class must also implement the hashCode method.
D. When adding Person objects to a java.util.Set collection, the equals method in line 4 will prevent duplicates.
Answer: A

Q: 07 Given:
1. import java.util.*;
2. public class Old {
3. public static Object get0(List list) {
4. return list.get(0);
5. }
6. }
Which three will compile successfully? (Choose three.)
A. Object o = Old.get0(new LinkedList());
B. Object o = Old.get0(new LinkedList<?>());
C. String s = Old.get0(new LinkedList<String>());
D. Object o = Old.get0(new LinkedList<Object>());
E. String s = (String)Old.get0(new LinkedList<String>());
Answer: A, D, E

Q: 08 Given:
1. import java.util.*;
2. public class Example {
3. public static void main(String[] args) {
4. // insert code here
5. set.add(new Integer(2));
6. set.add(new Integer(1));
7. System.out.println(set);
8. }
9. }
Which code, inserted at line 4, guarantees that this program will output [1, 2]?
A. Set set = new TreeSet();
B. Set set = new HashSet();
C. Set set = new SortedSet();
D. List set = new SortedList();
E. Set set = new LinkedHashSet();

Answer: A

Q: 09 Given:
11. public static Collection get() {
12. Collection sorted = new LinkedList();
13. sorted.add("B"); sorted.add("C"); sorted.add("A");
14. return sorted;
15. }
16. public static void main(String[] args) {
17. for (Object obj: get()) {
18. System.out.print(obj + ", ");
19. }
20. }
What is the result?
A. A, B, C,
B. B, C, A,
C. Compilation fails.
D. The code runs with no output.
E. An exception is thrown at runtime.
Answer: B

Q:10 given
Click the Task button.

Place the correct description of the compiler output on the code fragments to be inserted at lines 4 and 5. The same compiler output may be used more than once.

```
1. import java.util.*;
2. public class X {
3.    public static void main(String[] args) {
4.      // insert code here
5.      // insert code here
6.    }
7.    public static void foo(List<Object> list) {
8.  } }
```

Code

```
ArrayList<String> x1 = new ArrayList<String>();
foo(x1);
```

```
ArrayList<Object> x2 = new ArrayList<String>();
foo(x2);
```

```
ArrayList<Object> x3 = new ArrayList<Object>();
foo(x3);
```

```
ArrayList x4 = new ArrayList();
foo(x4);
```

Compiler Output

Compilation succeeds.

Compilation fails due to an error in the first statement.

Compilation of the first statement succeeds, but compilation fails due to an error in the second statement.

Done

Solution:
1. **Compilation of the first statement succeeds ,but compilation fails due to an error in the second statement.**
2. **Compilation fails due to an error in the first statement**
3. **Compilation succeeds**
   4. **Compilation succeeds**

Q: 11 Given:

11. public static Iterator reverse(List list) {

12. Collections.reverse(list);

13. return list.iterator();

14. }

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

16. List list = new ArrayList();

17. list.add("1"); list.add("2"); list.add("3");

18. for (Object obj: reverse(list))

19. System.out.print(obj + ", ");

20. }

What is the result?

A. 3, 2, 1,   B. 1, 2, 3,

C. Compilation fails.   D. The code runs with no output.

E. An exception is thrown at runtime.

Answer: C

Q: 12 Click the Task button.

```
Given:

1.  import java.util.*;
2.  class A { }
3.  class B extends A { }
4.  public class Test {
5.    public static void main(String[] args) {
6.      List<A> listA = new LinkedList<A>();
7.      List<B> listB = new LinkedList<B>();
8.      List<Object> listO = new LinkedList<Object>();
9.      // insert code here
10.   }
11.   public static void m1(List<? extends A> list) { }
12.   public static void m2(List<A> list) { }
13. }
```

Place a result onto each method call to indicate what would happen if the method call were inserted at line 9. Note: Results can be used more than once.

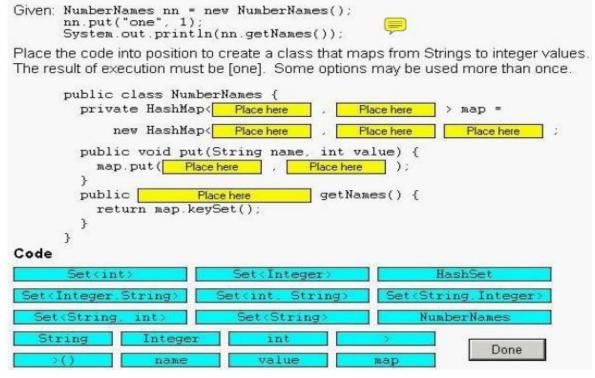| Method Calls		Result
m1(listA);	m2(listA);	Does not compile.
m1(listB);	m2(listB);	Compiles and runs without error.
m1(listO);	m2(listO);	An exception is thrown at runtime.

=============does not compile================
1.m1(listO);
2.m2(listB);
3.m2(listO);
=====compiles and runs with out error=========
1.m1(listA);
2.m1(listB);
3.m2(listA);
Q: 13 Given:
1. import java.util.*;
2. public class PQ {
3. public static void main(String[] args) {
4. PriorityQueue<String> pq = new PriorityQueue<String>();
5. pq.add("carrot");
6. pq.add("apple");
7. pq.add("banana");
8. System.out.println(pq.poll() + ":" + pq.peek());
9. }
10. }
What is the result?
A. apple:apple
B. carrot:apple
C. apple:banana
D. banana:apple
E. carrot:carrot
F. carrot:banana
Answer: C

Q: 14 Given:
1. import java.util.*;
2. public class WrappedString {
3. private String s;
4. public WrappedString(String s) { this.s = s; }
5. public static void main(String[] args) {
6. HashSet<Object> hs = new HashSet<Object>();
7. WrappedString ws1 = new WrappedString("aardvark");
8. WrappedString ws2 = new WrappedString("aardvark");
9. String s1 = new String("aardvark");
10. String s2 = new String("aardvark");
11. hs.add(ws1); hs.add(ws2); hs.add(s1); hs.add(s2);
12. System.out.println(hs.size()); } }
What is the result?
A. 0

B. 1
C. 2
D. 3
E. 4
F. Compilation fails.
G. An exception is thrown at runtime.
Answer: D

Q: 15 Given:
11. public class Key {
12. private long id1;
13. private long id2;
14.
15. // class Key methods
16. }
A programmer is developing a class Key, that will be used as a key in a standard java.util.HashMap.Which two methods should be overridden to assure that Key works correctly as a key? (Choose two.)
A. public int hashCode()
B. public boolean equals(Key k)
C. public int compareTo(Object o)
D. public boolean equals(Object o)
E. public boolean compareTo(Key k)
Answer: A, D

Q: 16 Given a pre-generics implementation of a method:
11. public static int sum(List list) {
12. int sum = 0;
13. for (Iterator iter = list.iterator(); iter.hasNext();) {
14. int i = ((Integer)iter.next()).intValue();
15. sum += i;
16. }
17. return sum;
18. }
Which three changes must be made to the method sum to use generics? (Choose three.)
A. remove line 14
B. replace line 14 with "int i = iter.next();"
C. replace line 13 with "for (int i : intList) {"
D. replace line 13 with "for (Iterator iter : intList) {"
E. replace the method declaration with "sum(List<int> intList)"
F. replace the method declaration with "sum(List<Integer> intList)"
Answer: A, C, F

Q: 17 Given:

11. // insert code here
12. private N min, max;
13. public N getMin() { return min; }
14. public N getMax() { return max; }
15. public void add(N added) {
16. if (min == null || added.doubleValue() < min.doubleValue()) 17. min = added;
18. if (max == null || added.doubleValue() > max.doubleValue()) 19. max = added;
20. }
21. }

Which two, inserted at line 11, will allow the code to compile? (Choose two.)

A. public class MinMax<?> {
B. public class MinMax<? extends Number> {
C. public class MinMax<N extends Object> {
D. public class MinMax<N extends Number> {
E. public class MinMax<? extends Object> {
F. public class MinMax<N extends Integer> {

Answer: D, F

Q: 18 Given:

1. import java.util.*;
2.
3. public class LetterASort{
4. public static void main(String[] args) {
5. ArrayList<String> strings = new ArrayList<String>();
6. strings.add("aAaA");
7. strings.add("AaA");
8. strings.add("aAa");
9. strings.add("AAaa");
10. Collections.sort(strings);
11. for (String s : strings) { System.out.print(s + " "); }
12. }
13. }

What is the result?

A. Compilation fails.   B. aAaA aAa AAaa AaA
C. AAaa AaA aAa aAaA   D. AaA AAaa aAaA aAa
E. aAa AaA aAaA AAaa   F. An exception is thrown at runtime.

Answer: C

Q: 19 Click the Task button.

Given: `NumberNames nn = new NumberNames();`
`nn.put("one", 1);`
`System.out.println(nn.getNames());`

Place the code into position to create a class that maps from Strings to integer values.
The result of execution must be [one]. Some options may be used more than once.

```
public class NumberNames {
    private HashMap< [Place here] , [Place here] > map =
        new HashMap< [Place here] , [Place here] [Place here] ;
    public void put(String name, int value) {
        map.put( [Place here] , [Place here] );
    }
    public [Place here] getNames() {
        return map.keySet();
    }
}
```

Code

Set<int>	Set<Integer>	HashSet
Set<Integer,String>	Set<int, String>	Set<String,Integer>
Set<String, int>	Set<String>	NumberNames
String	Integer	int
>()	name	value
		map

Done

Solution:
```
public class NumberNames{
    private HashMap< String , Integer> map=
    new  HashMap< String , Integer >( );
    public void put(String name . int Value) {
    map.put(name , Value);
    }
    public  Set<String> getNames( ) {
    return map.keySet( );
    }
}
```

Q: 20 Which two statements are true about the hashCode method? (Choose two.)
A. The hashCode method for a given class can be used to test for object equality and object inequality for that class.
B. The hashCode method is used by the java.util.SortedSet collection class to order the elements within that set.
C. The hashCode method for a given class can be used to test for object inequality, but NOT object equality, for that class.
D. The only important characteristic of the values returned by a hashCode method is that the distribution of values must follow a Gaussian distribution.
E. The hashCode method is used by the java.util.HashSet collection class to group the elements within that set into hash buckets for swift retrieval.
Answer: C, E

Q: 22 A programmer has an algorithm that requires a java.util.List that provides an efficient implementation of add(0, object), but does NOT need to support quick random access.
What supports these requirements?
A. java.util.Queue
B. java.util.ArrayList
C. java.util.LinearList
D. java.util.LinkedList
Answer: D

Q: 24 Given:
10. interface A { void x(); }
11. class B implements A { public void x() {} public void y() {} }
12. class C extends B { public void x() {} }
And:
20. java.util.List<A> list = new java.util.ArrayList<A>();
21. list.add(new B());
22. list.add(new C());
23. for (A a : list) {
24. a.x();
25. a.y();
26. }
What is the result?
A. The code runs with no output.
B. An exception is thrown at runtime.
C. Compilation fails because of an error in line 20.
D. Compilation fails because of an error in line 21.
E. Compilation fails because of an error in line 23.
F. Compilation fails because of an error in line 25.
Answer: F

Q: 26 Given:

Place the correct description of the compiler output on the code fragments to be inserted at lines 4 and 5. The same compiler output may be used more than once.

```
1.  import java.util.*;
2.  public class X {
3.    public static void main(String[] args) {
4.      // insert code here
5.      // insert code here
6.    }
7.    public static void foo(List<Object> list) {
8.  } }
```

Code

```
ArrayList<String> x1 = new ArrayList<String>();
foo(x1);
```

```
ArrayList<Object> x2 = new ArrayList<String>();
foo(x2);
```

```
ArrayList<Object> x3 = new ArrayList<Object>();
foo(x3);
```

```
ArrayList x4 = new ArrayList();
foo(x4);
```

Compiler Output

```
Compilation succeeds.
```

```
Compilation fails due to an error in the first statement.
```

```
Compilation of the first statement succeeds, but compilation fails due to an error in the second statement.
```

Done

Solution:

1. **Compilation of the first statement succeeds ,but compilation fails due to an error in the second statement.**
2. **Compilation fails due to an error in the first statement**
3. **Compilation succeeds**
4. **Compilation succeeds**

Q: 26 Click the Task button.

Given:

```
public void takeList(List<? extends String> list) {
    // insert code here
}
```

Place the Compilation Results on each code statement to indicate whether or not that code will compile if inserted into the takeList() method.

Code Statements

```
list.add("Foo");
```

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

```
list = new ArrayList<Object>();
```

```
String s = list.get(0);
```

```
Object o = list;
```

Compilation Results

Compilation succeeds

Compilation fails

Done

Solution:
1. list.add("foo");   -----------   **Compilation fails**
2. list = new ArrayList<String>();   ------Compilation succeeds
3.list=new ArrayList<Object>( ); ---- Compilation fails
4. String s = list.get(0); ------ Compilation succeeds
5. Object o = list; ----- Compilation succeeds

Q: 27 Given:
1. public class Drink implements Comparable {
2. public String name;
3. public int compareTo(Object o) {
4. return 0;
5. }
6. }
and:
20. Drink one = new Drink();
21. Drink two = new Drink();

22. one.name= "Coffee";

23. two.name= "Tea";

23. TreeSet set = new TreeSet();

24. set.add(one);

25. set.add(two);

A programmer iterates over the TreeSet and prints the name of each Drink object. What is the result?

A. Tea

B. Coffee

C. Coffee Tea

D. Compilation fails.

E. The code runs with no output.

F. An exception is thrown at runtime.

Answer: B

Q: 30 Given:

10. abstract public class Employee {

11. protected abstract double getSalesAmount();

12. public double getCommision() {

13. return getSalesAmount() * 0.15;

14. }

15. }

16. class Sales extends Employee {

17. // insert method here

18. }

Which two methods, inserted independently at line 17, correctly complete the Sales class? (Choose two.)

A. double getSalesAmount() { return 1230.45; }

B. public double getSalesAmount() { return 1230.45; }

C. private double getSalesAmount() { return 1230.45; }

D. protected double getSalesAmount() { return 1230.45; }

Answer: B, D

Q: 31 Given:

13. public static void search(List<String> list) {

14. list.clear();

15. list.add("b");

16. list.add("a");

17. list.add("c");

18. System.out.println(Collections.binarySearch(list, "a"));

19. }

What is the result of calling search with a valid List implementation?

A. 0   B. 1

C. 2   D. a
E. b   F. c
G. The result is undefined.
Answer: G

Q: 32 Given:
11. public static void append(List list) { list.add("0042"); }
12. public static void main(String[] args) {
13. List<Integer> intList = new ArrayList<Integer>();
14. append(intList);
15. System.out.println(intList.get(0));
16. }
What is the result?
A. 42
B. 0042
C. An exception is thrown at runtime.
D. Compilation fails because of an error in line 13.
E. Compilation fails because of an error in line 14.
Answer: B

Q: 33 Given:
11. public class Person {
12. private name;
13. public Person(String name) {
14. this.name = name;
15. }
16. public int hashCode() {
17. return 420;
18. }
19. }
Which statement is true?
A. The time to find the value from HashMap with a Person key depends on the size of the map.
B. Deleting a Person key from a HashMap will delete all map entries for all keys of type Person.
C. Inserting a second Person object into a HashSet will cause the first Person object to be removed as a duplicate.
D. The time to determine whether a Person object is contained in a HashSet is constant and does NOT depend on the size of the map.
Answer: A

Q: 34
A programmer must create a generic class MinMax and the type parameter
of MinMax must implement Comparable. Which implementation of MinMax will compile?
A. class MinMax<E extends Comparable<E>> {

E min = null;
E max = null;
public MinMax() {}
public void put(E value) { /* store min or max */ }
B. class MinMax<E implements Comparable<E>> {
E min = null;
E max = null;
public MinMax() {}
public void put(E value) { /* store min or max */ }
C. class MinMax<E extends Comparable<E>> {
<E> E min = null;
<E> E max = null;
public MinMax() {}
public <E> void put(E value) { /* store min or max */ }
D. class MinMax<E implements Comparable<E>> {
<E> E min = null;
<E> E max = null;
public MinMax() {}
public <E> void put(E value) { /* store min or max */ }
Answer: A

Q: 35 Given:
int[] myArray = new int[] {1, 2, 3, 4, 5}; What allows you to create a list from this array?
A. List myList = myArray.asList();
B. List myList = Arrays.asList(myArray);
C. List myList = new ArrayList(myArray);
D. List myList = Collections.fromArray(myArray);
Answer: B

Question: 36
Given:
1. public class Score implements Comparable<Score> {
2. private int wins, losses;
3. public Score(int w, int 1) { wins = w; losses = 1; }
4. public int getWins() { return wins; }
5. **public int getLosses() { return losses; }**
6. public String toString() {
7. return "<" + wins + "," + losses + ">";
8. }
9. // insert code here
10. }
Which method will complete this class?
A. public int compareTo(Object o) {/*mode code here*/}

B. public int compareTo(Score other) {/*more code here*/}
C. public int compare(Score s1,Score s2){/*more code here*/}
D. public int compare(Object o1,Object o2){/*more code here*/}
Answer: B

Question: 37
Click the Exhibit button.
1. import java.util.*;
2. class KeyMaster {
3. public int i;
4. public KeyMaster(int i) { this.i = i; }
5. **public boolean equals(Object o) { return i == ((KeyMaster)o).i; }**
6. public int hashCode() { return i; }
7. }
8. public class MapIt {
9. public static void main(String[] args) {
10. Set<KeyMaster> set = new HashSet<KeyMaster>();
11. KeyMaster k1 = new KeyMaster(1);
12. KeyMaster k2 = new KeyMaster(2);
13. set.add(k1); set.add(k1);
14. set.add(k2); set.add(k2);
15. **System.out.print(set.size() + ":");**
16. k2.i = 1;
17. System.out.print(set.size() + ":");
18. set.remove(k1);
19. System.out.print(set.size() + ":");
20. set.remove(k2);
21. System.out.print(set.size());
22. }
23. }
What is the result?

A. 4:4:2:2   B. 4:4:3:2
C. 2:2:1:0   D. 2:2:0:0
E. 2:1:0:0   F. 2:2:1:1
G. 4:3:2:1

Answer: F

Question: 38
Given:
1. import java.util.*;
2. public class Test {
3. public static void main(String[] args) {
4. List<String> strings = new ArrayList<String>();

5. // **insert code here**
6. }
7. }
Which four, inserted at line *5,* will allow compilation to succeed?
(Choose four.)
A. String s = strings.get(0);
B. Iterator i1 = strings.iterator();
C. String[] array1 = strings.toArray();
D. Iterator<String> i2 = strings.iterator();
E. String[] array2 = strings.toArray(new String[1]);
F. Iterator<String> i3 = strings.iterator<String>();
Answer: ABDE

Question: 39
Given:
classA {}
class B extends A {}
class C extends A {}
class D extends B {}
Which three statements are true? (Choose three.)
A. The type List<A> is assignable to List.
B. The type List<B> is assignable to List<A>.
C. The type List<Object> is assignable to List<?>.
D. The type List<D> is assignable to List<? extends B>.
E. The type List<? extends A> is assignable to List<A>.
F. The type List<Object> is assignable to any List reference.
G. The type List<? extends B> is assignable to List<? extends A>.
Answer: CDG

Question:40
Given:
11. public void addStrings(List list) {
12. list.add("foo");
13. list.add("bar");
14. }
What must you change in this method to compile without warnings?
A. add this code after line 11:
list = (List<String>) list;
B. change lines 12 and 13 to:
list.add<String>("foo");
list.add<String>("bar");
C. change the method signature on line 11 to:
public void addStrings(List<? extends String> list) {

D. change the method signature on line 11 to:
public void addStrings(List<? super String> list) {
E. No changes are necessary. This method compiles without warnings.
Answer: D

Question: 41
 Given:
1. public class Test {
2. public <T extends Comparable> T findLarger(T x, T y) {
3. if(x.compareTo(y) > 0) {
4. return x;
5. **} else {**
6. return y;
7. }
8. }
9. }
and:
22. Test t = new Test();
23. // insert code here
Which two will compile without errors when inserted at line 23?
(Choose two.)
A. Object x = t.findLarger(123, *"456");*
B. int x = t.findLarger(123, new Double(456));
C. int x = t.findLarger(123, new Integer(456));
D. int x = (int) t.findLarger(new Double(123), new Double(456));
Answer: AC

Question: 42
Given:
11. List list = // more code here
12. Collections.sort(list, new MyComparator());
Which code will sort this list in the opposite order of the sort in line
12?
A. Collections.reverseSort(list, new MyComparator());
B. Collections.sort(list, new MyComparator());
list.reverse();
C. Collections.sort(list, new InverseComparator(
new MyComparator()));
D. Collections.sort(list, Collections.reverseOrder(
new MyComparator()));
Answer: D

Question: 43

Given:
ArrayList a = new ArrayList();
containing the values {"1", "2", "3", "4", "5", "6", "7", "8"}
Which code will return 2?
A. Collections. sort(a, a.reverse());
int result = Collections.binarySearch(a, "6");
B. Comparator c = Collections.reverseOrder();
Collections.sort(a, c);
int result = Collections.binarySearch(a, "6");
C. Comparator c = Collections.reverseOrder();
Collections.sort(a, c);
int result = Collections.binarySearch(a, "6",c);
D. Comparator c = Collections.reverseOrder(a);
Collections.sort(a, c);
int result = Collections.binarySearch(a, "6",c);
E. Comparator c = new InverseComparator(new Comparator());
Collections.sort(a);
int result = Collections.binarySearch(a, "6",c);
Answer: C

Question: 44
Given:
11. public class Counter {
12. public static void main(String[] args) {
13. int numArgs = /* insert code here */;
14. }
15. **}**
and the command line:
java Counter one fred 42
Which code, inserted at line 13, captures the number of arguments
passed into the program?
A. args.count
B. args.length
C. args.count()
D. args.length()
E. args.getLength()
Answer: B

45. Given:
import java.util.*;
class Test {
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>();

Answer:

-> D and F are correct. The poll() method is associated with Queues. The LinkedList class implements the Queue interface.

->A is incorrect because the List interface does not implement Queue,

Question: 46
Click the Exhibit button.
Given:
1**. public class TwoThreads {**
2
3. private static Object resource = new Object();
4.
5*.* **private static void delay(long n) {**
6. try { Thread.sleep(n); }
7. catch (Exception e) { System.out.print("Error "); }
8. }
9
10. public static void main(String[] args) {
11. System.out.print("StartMain ");
12. new Thread1().start();
13. delay(1000);
14. Thread t2 = new Thread2();
15*.* **t2.start();**
16. delay(1000);
17. t2.interrupt
18. delay(1000);
19. System.out.print("EndMain ");
20. }
21.
22. static class Thread 1 extends Thread {

23. public void run() {
24. synchronized (resource) {
25. System.out.print("Startl ");
26. delay(6000);
27. System.out.print("End1 ");
28. }
29. }
30. }
31.
32. static class Thread2 extends Thread {
33. public void run() {
34. synchronized (resource) {
35. System.out.print("Start2 ");
36. delay(2000);
37. System.out.print("End2 ");
38. }
39. }
40. }
41. }

Assume that sleep(n) executes in exactly m milliseconds, and all other code executes in an insignificant amount of time. What is the output if the main() method is run?

A. Compilation fails.
B. Deadlock occurs.
C. StartMain Start1 Error EndMain End1
D. StartMain Start1 EndMain End1 Start2 End2
E. StartMain Start1 Error Start2 EndMain End2 End1
F. StartMain Start1 Start2 Error End2 EndMain End1
G. StartMain Start1 EndMain End1 Start2 Error End2

Answer: G

Question: 47
Click the Exhibit button.
10. public class Transfers {
11. public static void main(String[] args) throws Exception {
12. Record r1 = new Record();
13. Record r2 = new Record();
14. doTransfer(r1, r2, 5);
15. doTransfer(r2, r1, 2);
16. doTransfer(r1, r2, 1);
17. // print the result
18. System.out.println("rl = " + r1.get() +", r2=" + r2.get());
19. }

20. private static void doTransfer(
21. final Record a, final Record b, final int amount) {
22. Thread t = new Thread() {
23. public void run() {
24. new Clerk().transfer(a, b, amount);
25. }
26. };
27. t.start();
28. }
29. }
30. class Clerk {
31. public synchronized void transfer(Record a, Record b, int amount){
32. synchronized (a) {
33. synchronized (b) {
34. a.add(-amount);
35. b.add(amount);
36. }
37. }
38. }
39. }
40. class Record {
41.int num=10;
42. public int get() { return num; }
43. public void add(int n) { num = num + n; }
44. }
If Transfers.main() is run, which three are true? (Choose three.)
A. The output may be "r1 = 6, r2 = 14".
B. The output may be "r1 = 5, r2 = 15".
C. The output may be "r1 = 8, r2 = 12".
D. The code may run (and complete) with no output.
E. The code may deadlock (without completing) with no output.
F. M IllegalStateException or InterruptedException may be thrown at runtime.
Answer: ABE

48. Given:
public class Messager implements Runnable {
public static void main(String[] args) {
new Thread(new Messager("Wallace")).start();
new Thread(new Messager("Gromit")).start();
}
private String name;
public Messager(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 Gromit-1 Gromit-2 Wallace-2

D. Gromit-1 Gromit-2

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

F. The code does not compile.

G. An error occurs at run time.

Answer:

-> **C** is correct. Both threads will print two messages each. Wallace-1 must be before Wallace-2, and Gromit-1 must be before Gromit-2. Other than that, the Wallace and Gromit messages can be intermingled in any order.

-> **A, B, D, E, F,** and **G** are incorrect based on the above.

51. Given:

12. public class AccountManager {

13. private Map accountTotals = new HashMap();

14. private int retirementFund;

15.

16. public int getBalance(String accountName) {

17. Integer total = (Integer) accountTotals.get(accountName);

18. if (total == null)

19. total = Integer.valueOf(0);

20. return total.intValue();

21. }

23. public void setBalance(String accountName, int amount) {

24. accountTotals.put(accountName, Integer.valueOf(amount));

25. } }

This class is to be updated to make use of appropriate generic types, with no changes in behavior (for better or worse). Which of these steps could be performed? (Choose three.)

A. Replace line 13 with

private Map<String, int> accountTotals = new HashMap<String, int>();

B. Replace line 13 with

private Map<String, Integer> accountTotals = new HashMap<String, Integer>();

C. Replace line 13 with

private Map<String<Integer>> accountTotals = new HashMap<String<Integer>>();

D. Replace lines 17–20 with
int total = accountTotals.get(accountName);
if (total == null) total = 0;
return total;
E. Replace lines 17–20 with
Integer total = accountTotals.get(accountName);
if (total == null) total = 0;
return total;
F. Replace lines 17–20 with
return accountTotals.get(accountName);
G. Replace line 24 with
accountTotals.put(accountName, amount);
H. Replace line 24 with
accountTotals.put(accountName, amount.intValue());

Answer:
-> B , E, and G are correct.
-> A is wrong because you can't use a primitive type as a type parameter. C is wrong because a Map takes two type parameters separated by a comma. D is wrong because an int can't autobox to a null, and F is wrong because a null can't unbox to 0. H is wrong because you can't autobox a primitive just by trying to invoke a method with it.

52. Given a properly prepared String array containing five elements, which range of results could a proper invocation of Arrays.binarySearch() produce?
A. 0 through 4
B. 0 through 5
C. -1 through 4
D. -1 through 5
E. -5 through 4
F. -5 through 5
G. -6 through 4
H. -6 through 5

Answer:
-> G is correct. If a match is found, binarySearch()will return the index of the element that was matched. If no match is found, binarySearch() will return a negative number that, if inverted and then decremented, gives you the insertion point (array index) at which the value searched on should be inserted into the array to maintain a proper sort.
->A, B, C, D, E, F, and H are incorrect based on the above.

53. Given:
interface Hungry<E> { void munch(E x); }
interface Carnivore<E extends Animal> extends Hungry<E> {}
interface Herbivore<E extends Plant> extends Hungry<E> {}

abstract class Plant {}
class Grass extends Plant {}
abstract class Animal {}
class Sheep extends Animal implements Herbivore<Sheep> {
public void munch(Sheep x) {}
}
class Wolf extends Animal implements Carnivore<Sheep> {
public void munch(Sheep x) {}
}
Which of the following changes (taken separately) would allow this code to compile?
(Choose all that apply.)
A. Change the Carnivore interface to
interface Carnivore<E extends Plant> extends Hungry<E> {}
B. Change the Herbivore interface to
interface Herbivore<E extends Animal> extends Hungry<E> {}
C. Change the Sheep class to
class Sheep extends Animal implements Herbivore<Plant> {
public void munch(Grass x) {}
}
D. Change the Sheep class to
class Sheep extends Plant implements Carnivore<Wolf> {
public void munch(Wolf x) {}
}
E. Change the Wolf class to
class Wolf extends Animal implements Herbivore<Grass> {
public void munch(Grass x) {}
}
F. No changes are necessary.
Answer:
-> B is correct. The problem with the original code is that Sheep tries to implement
Herbivore<Sheep> and Herbivore declares that its type parameter E can be any type that
extends Plant. Since a Sheep is not a Plant, Herbivore<Sheep> makes no sense— the type
Sheep is outside the allowed range of Herbivore's parameter E. Only solutions that either alter
the definition of a Sheep or alter the definition of Herbivore will be able to fix this. So A, E, and F
are eliminated. B works, changing the definition of an Herbivore to allow it to eat Sheep solves
the problem. C doesn't work because an Herbivore<Plant> must have a munch(Plant) method,
not munch(Grass). And D doesn't work, because in D we made Sheep extend Plant, now the
Wolf class breaks because its munch(Sheep) method no longer fulfills the contract of Carnivore.

54. 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.HashSet

B. java.util.LinkedHashSet

C. java.util.List

D. java.util.ArrayList

E. java.util.Vector

F. java.util.PriorityQueue

Answer:

-> D is correct. All of the collection classes allow you to grow or shrink the size of your collection. ArrayList provides an index to its elements. The newer collection classes tend not to have synchronized methods. Vector is an older implementation of ArrayList functionality and has synchronized methods; it is slower than ArrayList.

-> A, B, C, E, and F are incorrect based on the logic described above; Notes: C, List is an interface, and F, PriorityQueue does not offer access by index.

55. Given:

```
import java.util.*;
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.

G. An exception is thrown at runtime.

Answer:

-> F is correct. The problem here is in Group's add() method—it should have been

add(Person), since the class extends HashSet<Person>. So this doesn't compile.
Pop Quiz: What would happen if you fixed this code, changing add(Object) to
add(Person)? Try running the code to see if the results match what you thought.
-> A, B, C, D, E, and G are incorrect based on the above.

56. Given:
import java.util.*;
class AlgaeDiesel {
public static void main(String[] args) {
String[] sa = {"foo", "bar", "baz" };
// insert method invocations here
}
}
What java.util.Arrays and/or java.util.Collections methods could you use to convert sa to
a List and then search the List to find the index of the element whose value is "foo"?
(Choose from one to three methods.)

A. sort()   B. asList()
C. toList()   D. search()
E. sortList()   F. contains()
G. binarySearch()

Answer:
-> A, B, and G are required. The as List() method converts an array to a List. You can find the
index of an element in a List with the binarySearch() method, but before you do that you must
sort the list using sort().
-> F is incorrect because contains() returns a boolean, not an index. C, D, and E are incorrect,
because these methods are not defined in the List interface.

57. Given that String implements java.lang.CharSequence, and:
import java.util.*;
public class LongWordFinder {
public static void main(String[] args) {
String[] array = { "123", "12345678", "1", "12", "1234567890"};
List<String> list = Arrays.asList(array);
Collection<String> resultList = getLongWords(list);
}
// INSERT DECLARATION HERE
{
Collection<E> longWords = new ArrayList<E>();
for (E word : coll)
if (word.length() > 6) longWords.add(word);
return longWords;
}}
Which declarations could be inserted at // INSERT DECLARATION HERE so that the

program will compile and run? (Choose all that apply.)

A. public static <E extends CharSequence> Collection<? extends CharSequence>
getLongWords(Collection<E> coll)

B. public static <E extends CharSequence> List<E>
getLongWords(Collection<E> coll)

C. public static Collection<E extends CharSequence> getLongWords(Collection<E> coll)

D. public static List<CharSequence>
getLongWords(Collection<CharSequence> coll)

E. public static List<? extends CharSequence>
getLongWords(Collection<? extends CharSequence> coll)

F. static public <E extends CharSequence> Collection<E>
getLongWords(Collection<E> coll)

G. static public <E super CharSequence> Collection<E>
getLongWords(Collection<E> coll)

Answer:

-> F is correct.

-> A is close, but it's wrong because the return value is too vague. The last line of the
method expects the return value to be Collection<String>, not
Collection<? extends CharSequence>. B is wrong because longWords has been declared as a
Collection<E>, and that can't be implicitly converted to a List<E> to match the declared return
value. (Even though we know that longWords is really an ArrayList<E>, the compiler only know
what it's been declared as.) C, D, and E are wrong because they do not declare a type variable
E (there's no <> before the return value) so the getLongWords() method body will not compile.
G is wrong because E super CharSequence makes no sense—super could be used in
conjunction with a wildcard but not a type variable like E.

58. Given:
12. TreeSet map = new TreeSet();
13. map.add("one");
14. map.add("two");
15. map.add("three");
16. map.add("four");
17. map.add("one");
18. Iterator it = map.iterator();
19. while (it.hasNext()) {
20. System.out.print(it.next() + " ");
21. }
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.

Answer:

-> D is correct. TreeSet assures no duplicate entries; also, when it is accessed it will return elements in natural order, which for Strings means alphabetical.

-> A, B, C, E, F, G, and H are incorrect based on the logic described above. Note, even though as of Java 5 you don't have to use an Iterator, you still can.

59. 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 this:

// 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 above.

Answer:

->B, E, and F are correct.

-> The return type of process is definitely declared as a List, not an ArrayList, so A and D are wrong. C is wrong because the return type evaluates to List<Integer>, and that can't be assigned to a variable of type List<Number>. Of course all these would probably cause a NullPointerException since the variables are still null—but the question only asked us to get the code to compile.