Question 1

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.

Question 2

Give:

```
public static Iterator reverse(List list) {
Collections.reverse(list);
return list.iterator();
}
public static void main(String[] args) {
List list = new ArrayList();
list.add(” 1”); list.add(”2”); list.add(”3”);
for (Object obj: reverse(list))
System.out.print(obj + “,”);


}
```

‘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.

Question 3

Given:

```
public static Collection get() {
Collection sorted = new LinkedList();
sorted.add(’B”); sorted.add(”C”); sorted.add(”A”);
return sorted;
}
public static void main(String[] args) {
for (Object obj: get()) {
System.out.print(obj + “, “);
}
}
```

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.

Question 4

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(l));
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();

Question 5

Given:

```
import java.util.*;
public class PQ {
public static void main(String[] args) {
PriorityQueue<String> pq = new PriorityQueue<String>();
pq.add(”carrot”);
pq.add(”apple”);
pq.add(”banana”);
System.out.println(pq.poll() +”:” + pq.peek());
}
}
```

What is the result?

A. apple:apple

B. carrot:apple

C. apple:banana

D. banana:apple

E. carrot:carrot

F. carrot:banana

Question 6

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.

Question 7

```
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 coil collection, but the order is NOT guaranteed to be preserved.

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

Question 8

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*/}

Question 9

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

Question 10

Given:

```
11. public class Person {
12. private String name, comment;
13. private int age;
14. public Person(String n, int a, String c) {
15. name = n; age = a; comment = c;
16. }
17. public boolean equals(Object o) {
18. if(! (o instanceof Person)) return false;
19, Person p = (Person)o;
20. return age == p.age && name.equals(p.name);
21. }
22. }
```

What is the appropriate definition of the hashCode method in class Person?

A. return super.hashCode();

B. return name.hashCode() + age * 7;

C. return name.hashCode() + comment.hashCode() /2;

D. return name.hashCode() + comment.hashCode() / 2 - age * 3;

Question 11

Given:

11. public class Key {

12. private long id1;

13. private long 1d2;

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)

Question 12

Given:

```
public class Person {
private name;
public Person(String name) {
this.name = name;
}
public boolean equals(Object o) {
if( !o instanceof Person ) return false;
Person p = (Person) o;
return p.name.equals(this.name);
}
}
```

Which is true?

A. Compilation fails because the hashCode method is not overridden.

B. A HashSet could contain multiple Person objects with the same

name.

C. All Person objects will have the same hash code because the

hashCode method is not overridden.

D. If a HashSet contains more than one Person object with

name=”Fred”, then removing another Person, also with name=”Fred”,

will remove them all.

Question 13

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 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.

Question 14

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.

Question 15

```
import java.util.*;
class KeyMaster {
public int i;
public KeyMaster(int i) { this.i = i; }
public boolean equals(Object o) { return i == ((KeyMaster)o).i; }
public int hashCode() { return i; }
}
public class MapIt {
public static void main(String[] args) {
Set<KeyMaster> set = new HashSet<KeyMaster>();
KeyMaster k1 = new KeyMaster(1);
KeyMaster k2 = new KeyMaster(2);
set.add(k1); set.add(k1);
set.add(k2); set.add(k2);
System.out.print(set.size() + “:”);
k2.i = 1;
System.out.print(set.size() + “:”);
set.remove(k1);
System.out.print(set.size() + “:”);
set.remove(k2);
System.out.print(set.size());
}
```

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

Question 16

Consider the following class:

```
public class IntPair

{

   private int a;

   private int b;

   public void setA(int i){ this.a = i; }

   public int getA(){ return this.a; }

   public void setB(int i){ this.b = i; }

   public int getB(int b){ return b; }

   public boolean equals(Object obj)

   {

      return ( obj instanceof IntPair && this.a == ((IntPair) obj).a );

   }

   public int hashCode()
```

```
    {
        //1
    }
}
```

Which of the following options would be valid at //1?

Select 4 correct options

a return 0;

b return a;

c return a+b;

d return a*a;

e return a/2;

Question 17.

Consider the following class:

```
public class IntPair
{
    private int a;
    private int b;
    public void setA(int i){ this.a = i; }
    public int getA(){ return this.a; }
    public void setB(int i){ this.b = i; }
    public int getB(){ return b; }
    public boolean equals(Object obj)
    {
        return ( obj instanceof obj && this.a == ((IntPair) obj).a  && this.b == ((IntPair) obj).b );
```

```
   }
   public int hashCode()
   {
      //1
   }
}
```

Which of the following options would NOT be valid at //1?

Select 1 correct option.

a return a;

b return a*b;

c return a+b;

d return b;

e None of these is invalid.

Question 18.

Which of the following are valid implementation of equals() method of a class TestClass?

1.

```
public boolean equals(TestClass tc)
{
   return this == tc;
}
```

2.

```
public boolean equals(TestClass tc)
{
   return this != tc;
}
```

3.

```
public boolean equals(Object tc)
{
    return this == tc;
}
```

4.

```
public boolean equals(Object tc)
{
    if( tc instanceof TestClass && this.someVar == ( (TestClass)tc).someVar )
    {
        if(this != tc) return true;
        else return false;
    }
    else return false;
}
```

Select 1 correct option.

a 1

b 2

c 3

d 4

e None of these.

Question 19.

Which of the following statments are correct regarding the equals() method?

Select 1 correct option.

a It must be symmetric but need not be transitive.

b It must be reflexive but need not be transitive.

c It must be symmetric and transitive but not reflexive.

d If passed a null, it must return false.

e None of these.

Question 20.

Given:

```
public static Iterator reverse(List list) {
Collections.reverse(list);
return list.iterator();
}
public static void main(String[] args) {
List list = new ArrayList();
list.add("1"); list.add("2"); list.add("3");
for (Object obj: reverse(list))
System.out.print(obj + ", ");
}
```

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.