Module 7-集合和泛型

一、选择题：

Question 1
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 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

Question 2
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

Question 3
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();

Answer: A

Question 4
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

Question 5
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

Question 6

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 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.

Answer: D

Question 7
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

Question 8
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 9
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

Question 10
Given:
int[] myArray=newint[] {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 11
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

Question 12
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;

Answer: B

Question 13
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)

Answer: AD

Question 14
Given:
11. public class Person {
12. private name;
13. public Person(String name) {
14. this.name = name;
15. }
16. public boolean equals(Object o) {
17. if(!o instanceof Person) return false;
18. Person p = (Person) o;
19. return p.name.equals(this.name);
20. }
21. }
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.

Answer: B

Question 15
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.

Answer: A

Question 16
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: CE

Question 17
Given:
enum Example { ONE, TWO, THREE }
Which is true?
A. The expressions (ONE == ONE) and ONE.equals(ONE) are both guaranteed to be true.
B. The expression (ONE < TWO) is guaranteed to be true and ONE.compareTo(TWO) is guaranteed to be less than one.
C. The Example values cannot be used in a raw java.util.HashMap; instead, the programmer must use a java.util.EnumMap.
D. The Example values can be used in a java.util.SortedSet, but the set will NOT be sorted because enumerated types do NOT implement java.lang.Comparable.

Answer: A

Question 18
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 19
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 20
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

Question 21
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 22
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

Question 23
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

Question 24
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 25
Give:
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

Question 26
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: ACF

Question 27
Given:
class A {}
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 28
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 29
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 30
Given:
1. import java.util.*;
2. public class Old {
3. public static Object get()(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.getfl(new LinkedList<String>());
D. Object o = Old.get0(new LinkedList<Object>());
E. String s = (String)Old.get0(new LinkedList<String>());

Answer: ADE

Question 31
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: DF

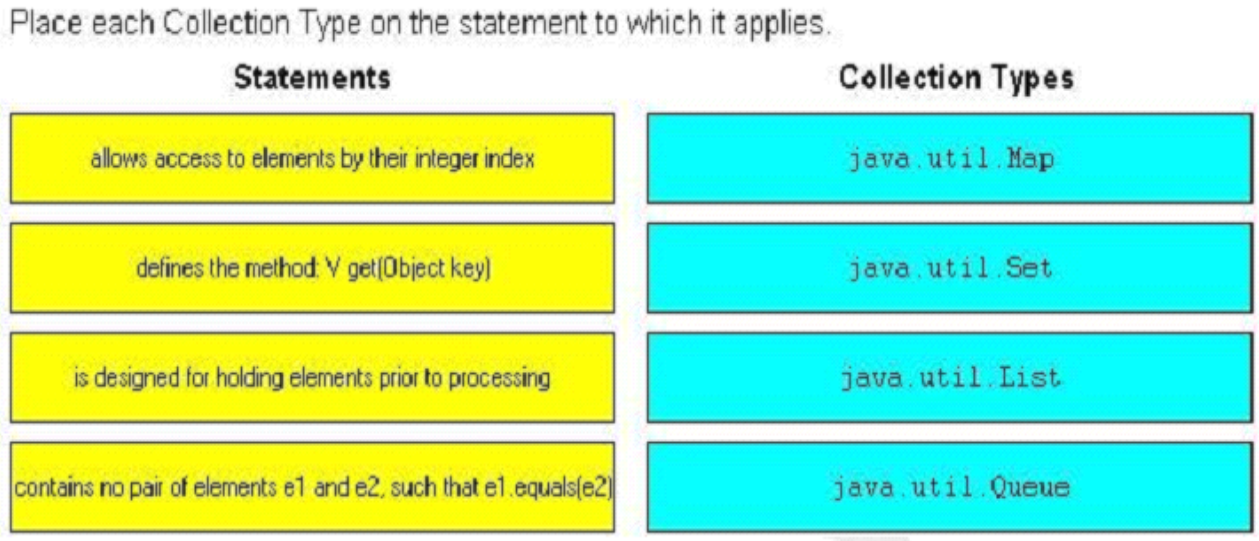
Question 32
A programmer must create a generic class MinMax and the type parameter of MinMax must implement Comparable. Which implementation of MinMax will compile?

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

二、拖拽题：

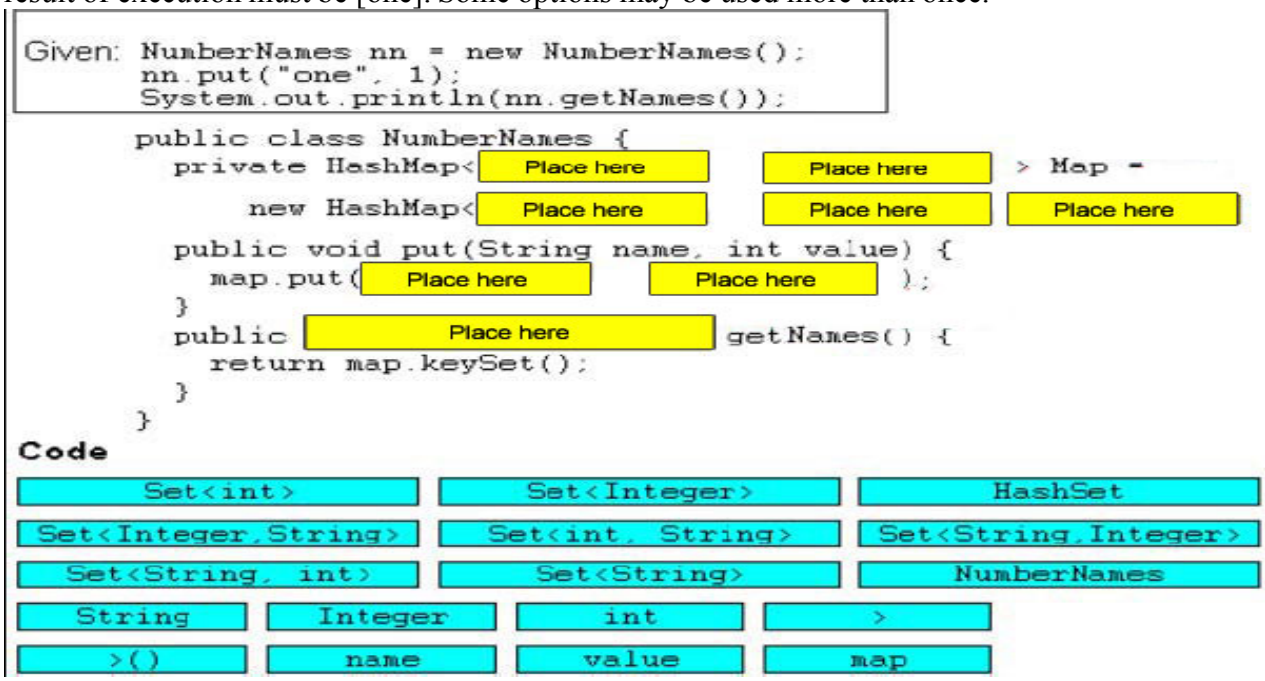
Question 1:



Answer:

allows access to elements by their integer index----------java.util.List
defines the method V get(Object key)----------java.util.Map
is designed for holding elements prior to processing----------java.util.Queue
contains no pair of elements e1 and e2, such that e1.equals(e2)--------java.util.Set

Question 2:
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.



Answer:

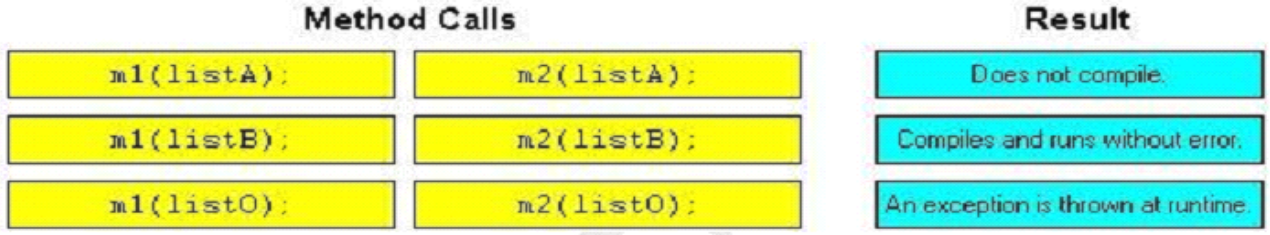
```
public class NumberNames{
    private HashMap<String,Integer>map=new HashMap<String,Integer>;
    public void put(String name,int value){
        map.put(name,value);
    }
    public Set<String> getNames(){
        return map.keySet();
    }
}
```

Question 3:

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. }
```

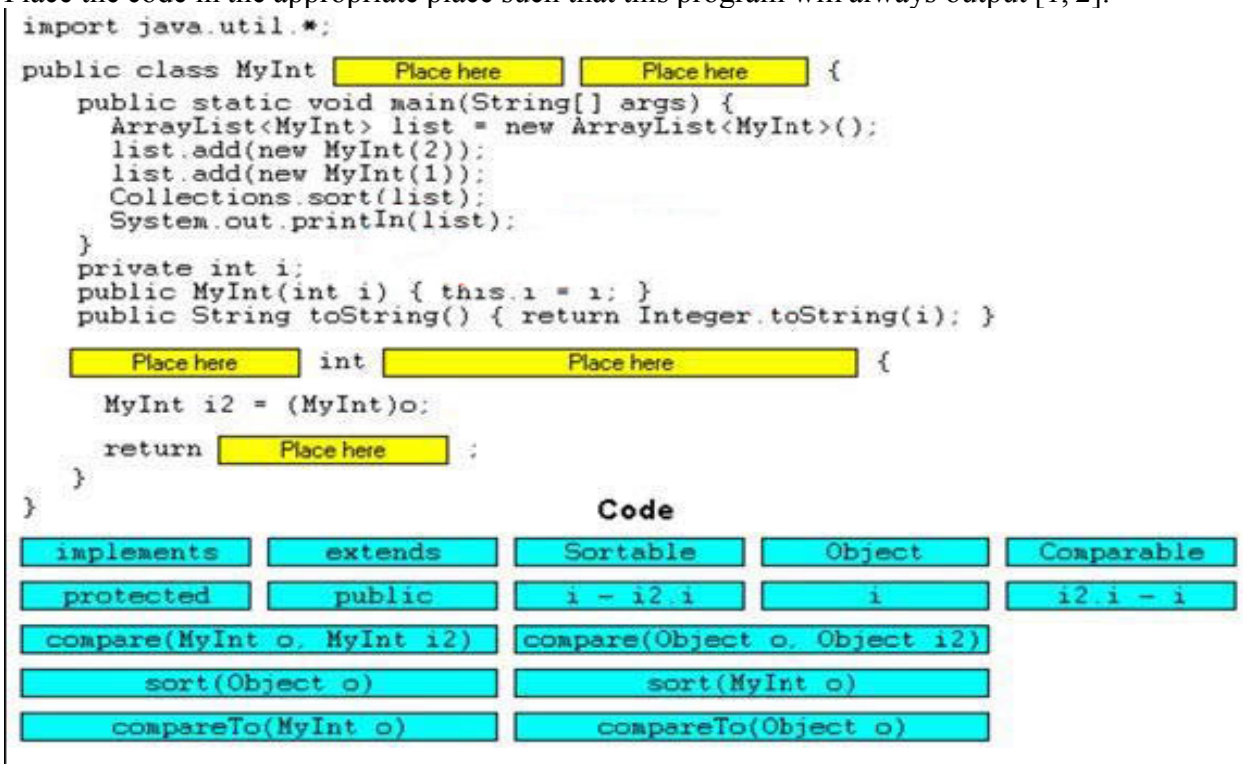
Place a result onto each method call to indicate what would happen if the method call were inserted at line 9. Note: Results can be used more than once.



Answer:

m1(listA);	Compiles and runs without error
m2(listA);	Compiles and runs without error
m1(listB);	Compiles and runs without error
m2(listB);	Compilation fails
m1(listO);	Compilation fails
m2(listO);	Compilation fails

Question 4:
Place the code in the appropriate place such that this program will always output [1, 2].



Answer:

```
import java.util.*;
public class MyInt implements Comparable{
public static void main(String[] args) {
        ArrayList<MyInt> list=new ArrayList<MyInt>();
        list.add(new MyInt(2));
        list.add(new MyInt(1));
        Collections.sort(list);
        System.out.println(list);
}
private int i;
public MyInt(int i){
        this.i=i;
}
public String toString(){
        return Integer.toString(i);
}
public int compareTo(Object o) {
        MyInt i2=(MyInt)o;
        return i-i2.i;
}
}
```

Question 5:

Given the class definitions:

```
class Animal { }
class Dog extends Animal { }
```

and the code:

```
public void go() {
  ArrayList<Dog> aList = new ArrayList<Dog>();
  takeList(aList);
}
// insert defination of the takeList() method here
```

Place the correct Compilation Result on each takeList() method definition to indicate whether or not the go() method would compile given that definition.

takeList() Method Definition

- `public void takeList(ArrayList list) { }`
- `public void takeList(ArrayList<Animal> list) { }`
- `public void takeList(ArrayList<? extends Animal> list) { }`
- `public void takeList(ArrayList<?> list) { }`
- `public void takeList(ArrayList <Object> list) { }`

Compilation Result

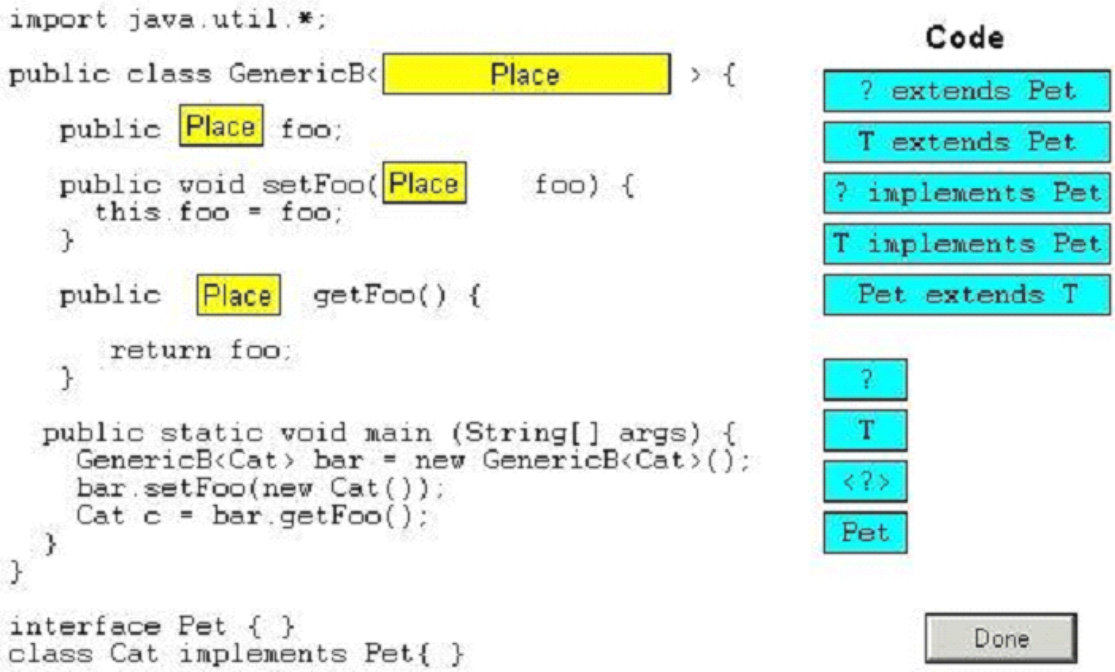
- Compilation succeeds.
- Compilation fails.

Answer:

public void takeList(ArrayList list){} ----------Compilation succeeds
public void takeList(ArrayList<Dog> list) {}----------Compilation succeeds
public void takeList(ArrayList<Animal> list) {}----------Compilation fails
public void takeList(ArrayList<? extends Animal> list){}------Compilation succeeds
public void takeList(ArrayList<?> list){}----------Compilation succeeds
public void takeList(ArrayList<Object> list){} ----------Compilation fails

Question 6:

Place the code into the GenericB class definition to make the class compile successfully.



Answer:

```
public class GenericB<T extends Pet>{
	public T foo;
	public void setFoo(T foo){
		this.foo = foo;
	}
	public T getFoo(){
		return foo;
	}
}
```

Question 7:

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

Answer:

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

foo(x1); //Compilation of the first statement succeeds, but compilation fails due to an error in the second statement

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

foo(x2); //Compilation fails due to an error in the first statement.

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

foo(x3); //Compilation succeeds

ArrayList x4=new ArrayList();

foo(x4); //Compilation succeeds

Question 8:

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

Answer:

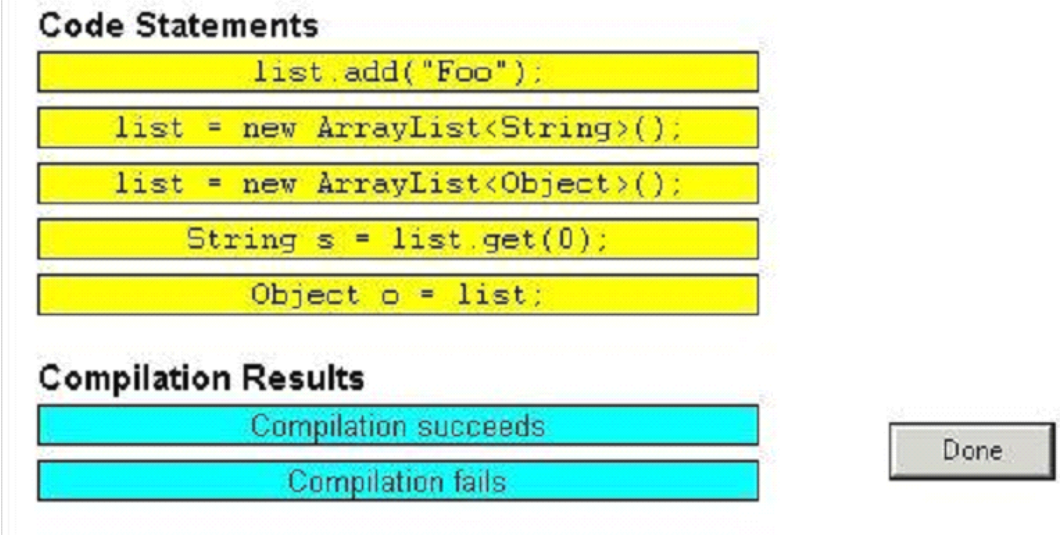
```
public class Gen<T>{
        private T object;
        public Gen(T object){
                this.object = object;
        }
        public T getObject(){
                return object;
        }
 }
```

Question 9:

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.



Answer:

list.add("Foo");	Compilation fails
list = new ArrayList<String>();	Compilation succeeds
list = new ArrayList<Object>();	Compilation fails
String s = list.get(0);	Compilation succeeds
Object o = list;	Compilation succeeds

Question 10:

Given:

```
1. import java.util.*;
2. public class TestGenericConversion {
3.   public static void main(String[] args) {
4.     List list = new LinkedList();
5.     list.add("one");
6.     list.add("two");
7.     System.out.print(((String)list.get(0)).length());
8.   }
9. }
```

Refractor this class to use generics without changing the code's behavior.

```
1. import java.util.*;
2. public class TestGenericConversion {
3.   public static void main(String[] args) {
4.     Place here
5.     list.add("one");
6.     list.add("two");
7.     Place here
8.   }
9. }
```

Code

List list = new LinkedList();	System.out.print(list.get(0).length());
List<String> list = new LinkedList<String>();	System.out.print(list.get<String>(0).length());
List<String> list = new LinkedList();	System.out.print((String)list.get(0).length());
List list = new LinkedList<String>();	System.out.print(((List<String>)list.get(0)).length());

Answer:

```
import java.util.*;
public class _166 {
    public static void main(String[] args) {
        List<String> list=new LinkedList<String>();
        list.add("one");
        list.add("two");
        System.out.println(list.get(0).length());
    }
}
```