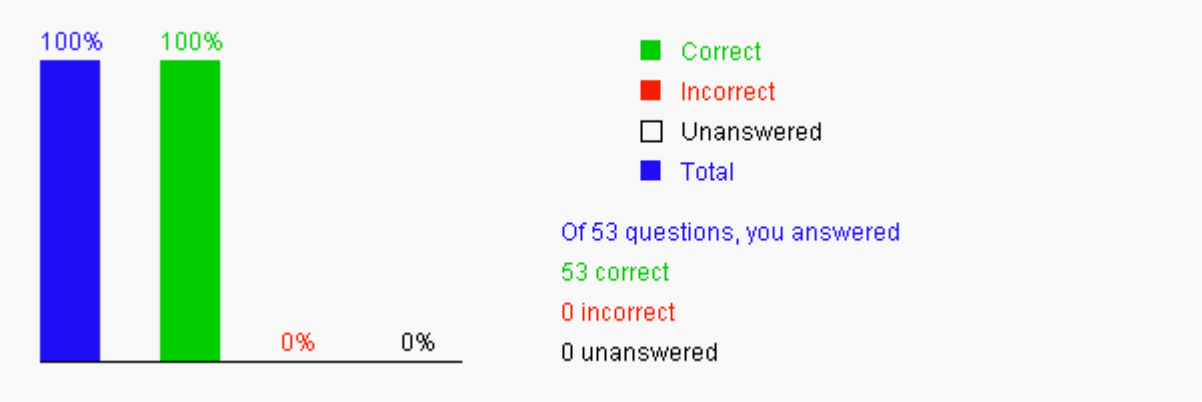
This quiz is for students to practice. A large number of additional quiz is available for instructors using Quiz Generator from the Instructor's Resource Website. Videos for Java, Python, and C++ can be found at https://yongdanielliang.github.io/revelvideos.html.

Chapter 20 Lists, Stacks, Queues, and Priority Queues



Please send suggestions and errata to Dr. Liang at y.daniel.liang@gmail.com. Indicate which book and edition you are using. Thanks!

Section 20.2 Collections

20.1 All the concrete classes in the Java Collections Framework implement _____.

- A. the Cloneable interface
- B. the Serializable interfaces
- C. the Comparable interface
- D. the Comparator interface

Your answer is correct

20.2 Which of the following statements are true?

- A. The Collection interface is the root interface for manipulating a collection of objects.
- B. The Collection interface provides the basic operations for adding and removing elements in a collection.
- C. The AbstractCollection class is a convenience class that provides partial implementation for the Collection interface.
- D. Some of the methods in the Collection interface cannot be implemented in the concrete subclass. In this case, the method would throw java.lang.UnsupportedOperationException, a subclass of RuntimeException.
- E. All interfaces and classes in the Collections framework are declared using generic type since JDK 1.5.

Your answer is correct

20.3 Which of the following methods are in the Collection interface?

- A. clear()
- B. isEmpty()
- C. size()
- D. getSize()

Your answer is correct

20.4 Which of the following methods are in the Collection interface?

- A. add(o: E)
- B. addAll(c: Collection<? extends E>)
- C. contains(o: Object): boolean
- D. containsAll(c: Collection<?>): boolean

Your answer is correct

20.5 Which of the following methods are in the Collection interface?

- A. remove(o: Object): boolean
- B. removeAll(c: Collection<?>): boolean
- C. delete(o: E): boolean
- D. deleteAll(c: Collection<?>): boolean

Your answer is correct

Section 20.3 Iterators

20.6 For an instance of Collection, you can obtain its iterator using _____.

- A. c.getIterator()
- B. c.iterator()
- C. c.iterators()
- D. c.iterable()

Your answer is correct

20.7 The iterator() method is defined in the _____ interface.

- A. Iterator
- B. Collection
- C. Iterable
- D. ArrayList

Your answer is correct

20.8 The iterator() method returns an instance of the _____ interface.

- A. Iterator
- B. Collection
- C. Iterable
- D. ArrayList

Your answer is correct

20.9 You can use a for-each loop to traverse all elements in a container object that implements _____.

- A. Iterator
- B. Collection
- C. Iterable
- D. ArrayList

Your answer is correct

Section 20.5 Lists

20.10 Which of the following statements are true?

- [x] A. java.util.List inherits all the methods from java.util.Collection. Additionally, it contains new methods for manipulating a list.
- [x] B. The AbstractList class provides a partial implementation for the List interface.
- [x] C. ArrayList is a concrete implementation of List using an array.
- [x] D. LinkedList is a concrete implementation of List using a linked list. LinkedList contains all the methods in List and additional new methods for manipulating a linked list.
- [x] E. ListIterator is a subinterface of Iterator and it provides the methods to support bi-directional traversal of a list.

Your answer is correct

20.11 Which of the following statements are true?

- [x] A. An ArrayList can grow automatically.
- [ ] B. An ArrayList can shrink automatically.
- [x] C. You can reduce the capacity of an ArrayList by invoking the trimToSize() method on the list.
- [ ] D. You can reduce the capacity of a LinkedList by invoking the trimToSize() method on the list.

Your answer is correct
Explanation: A LinkedList does not have excess capacity.

20.12 Which of the following methods are in java.util.List?

- [x] A. add(int index, E element)
- [x] B. get(int index)
- [x] C. set(int index, E element)
- [x] D. remove(int index)
- [x] E. subList(int fromIndex, int toIndex)

Your answer is correct

20.13 Which of the following are true?

- [x] A. You can insert an element anywhere in an arraylist.

- [x] B. You can insert an element anywhere in a linked list.
- [x] C. You can use a linked list to improve efficiency for adding and removing elements at the beginning of a list.
- [x] D. You should use an array list if your application does not require adding and removing elements at the beginning of a list.

Your answer is correct

20.14 Suppose list1 is an ArrayList and list2 is a LinkedList. Both contains 1 million double values. Analyze the following code:

```
A:
for (int i = 0; i < list1.size(); i++)
  sum += list1.get(i);

B:
for (int i = 0; i < list2.size(); i++)
  sum += list2.get(i);
```

- (●) A. Code fragment A runs faster than code fragment B.
- ( ) B. Code fragment B runs faster than code fragment A.
- ( ) C. Code fragment A runs as fast as code fragment B.

Your answer is correct

20.15 Suppose list is a LinkedList that contains 1 million int values. Analyze the following code:

```
A:
for (int i = 0; i < list.size(); i++)
  sum += list.get(i);

B:
for (int i: list)
  sum += i;
```

- ( ) A. Code fragment A runs faster than code fragment B.
- (●) B. Code fragment B runs faster than code fragment A.
- ( ) C. Code fragment A runs as fast as code fragment B.

Your answer is correct
Explanation: Becuase code fragment B uses an iterator to traverse the elements in a linked list.

20.16 Which method do you use to test if an element is in a set or list named x?

- ( ) A. (element instanceof List) || (element instanceof Set)
- ( ) B. x.in(element)
- ( ) C. x.contain(element)
- (●) D. x.contains(element)
- ( ) E. x.include(element)

Your answer is correct
Explanation: The contains method defined in the Collection interface checks if an element is in the collection (set or list)

20.17 When you create an ArrayList using ArrayList<String> x = new ArrayList<>(2), _____

- [ ] A. two elements are created in the array list.
- [x] B. no elements are currently in the array list.
- [ ] C. the array list size is currently 2.
- [x] D. the array list capacity is currently 2.

Your answer is correct

20.18 Suppose ArrayList x contains three strings [Beijing, Singapore, Tokyo]. Which of the following methods will cause runtime errors?

- [ ] A. x.get(2)
- [x] B. x.set(3, "New York");
- [x] C. x.get(3)
- [x] D. x.remove(3)
- [ ] E. x.size()

Your answer is correct
Explanation: There is no element at index 3.

20.19 Which method do you use to find the number of elements in a set or list named x?

- ( ) A. x.length()

- B. x.count()
- C. x.counts()
- D. x.size()
- E. x.sizes()

Your answer is correct
Explanation: The size method defined in the Collection interface returns the number of elements in the collection (set or list)

20.20 Which method do you use to remove an element from a set or list named x?

- A. x.delete(element)
- B. x.remove(element)
- C. x.deletes(element)
- D. x.removes(element)
- E. None of the above

Your answer is correct
Explanation: The remove method defined in the Collection interface removes the element from the collection (set or list)

20.21 What is the printout of the following code?

```
List<String> list = new ArrayList<>();
list.add("A");
list.add("B");
list.add("C");
list.add("D");
for (int i = 0; i < list.size(); i++)
  System.out.print(list.remove(i));
```

- A. ABCD
- B. AB
- C. AC
- D. AD
- E. ABC

Your answer is correct
Explanation: Before the loop, the list is [A, B, C, D]. After invoking list.remove(0), the list becomes [B, C, D] and size becomes 3. Invoking remove(1) now deletes C from the list. The list becomes [B, D]. Now the list size is 2 and i is 2. So the loop ends.

20.22 Suppose list list1 is [1, 2, 5] and list list2 is [2, 3, 6]. After list1.addAll(list2), list1 is _____.

- A. [1, 2, 2, 3, 5, 6]
- B. [1, 2, 3, 5, 6]
- C. [1, 5]
- D. [2]

Your answer is correct

20.23 Suppose list list1 is [1, 2, 5] and list list2 is [2, 3, 6]. After list1.addAll(list2), list2 is _____.

- A. [1, 2, 2, 3, 5, 6]
- B. [1, 2, 3, 5, 6]
- C. [1, 5]
- D. [2]
- E. [2, 3, 6]

Your answer is correct

20.24 Which of the following statements are correct?

- [x] A. When you create an array using new int[10], an array object is created with ten integers of value 0.
- [ ] B. When you create an array using new int[10], an array object is created with no values in the array.
- [x] C. When you create an ArrayList using new ArrayList(), an ArrayList object is created with no elements in the ArrayList object.
- [x] D. When you create an array using int[] x = new int[10], x.length() is 10.
- [ ] E. When you create an array using ArrayList x = new ArrayList(10), x.size() is 10.

Your answer is correct

20.25 Suppose a list contains {"red", "green", "red", "green"}. What is the list after the following code?

```
list.remove("red");
```

- A. {"red", "green", "red", "green"}
- B. {"green", "red", "green"}
- C. {"green", "green"}
- D. {"red", "green", "green"}

Your answer is correct

20.26 Suppose a list contains {"red", "green", "red", "green"}. What is the list after the following code?

```
String element = "red";
for (int i = 0; i < list.size(); i++)
  if (list.get(i).equals(element)) {
    list.remove(element);
    i--;
  }
```

- A. {"red", "red", "green"}
- B. {"red", "green"}
- C. {"green", "green"}
- D. {"green"}
- E. {}

Your answer is correct

20.27 Suppose a list contains {"red", "green", "red", "green"}. What is the list after the following code?

```
String element = "red";
for (int i = list.size() - 1; i >= 0; i--)
  if (list.get(i).equals(element))
    list.remove(element);
```

- A. {"red", "red", "green"}
- B. {"red", "green"}
- C. {"green", "green"}
- D. {"green"}
- E. {}

Your answer is correct

20.28 What is the output of the following code?

```
ArrayList<Integer> list = new ArrayList<>();
list.add(1);
list.add(2);
list.add(3);
list.remove(2);
System.out.println(list);
```

- A. [1, 2, 3]
- B. [1, 2]
- C. [1]
- D. [1, 3]
- E. [2, 3]

Your answer is correct

20.29 To remove all the elements in the list in the following code, replace the underlined blank space with _____.

```
import java.util.*;

public class Test {
  public static void main(String[] args) {
    _____<Integer> list = new ArrayList<>();
    list.add(0);
    list.add(1);
    list.add(2);

    for (int i = 0; i < 3; i++) {
      list.remove(i);
    }
    System.out.println(list);
  }
}
```

- A. Collection
- B. List
- C. ArrayList

D. AbstractList

Your answer is correct

Explanation: If list is declared as Collection<Integer>, list.remove(i) is the same as list.remove(new Integer(i)). If list is declared as List<Integer>, ArrayList<Integer>, or AbstractList<Integer>, list.remove(i) is to remove the element at the index i. The list has three elements and the last index is 2. After removing the first element, the last index becomes 1. After removing another element, the last index becomes 0. You will receive an IndexOutOfBoundsException.

Section 20.6 The Comparator Interface

20.30 Which of the following statements are true?

- [x] A. The Comparable interface contains the compareTo method with the signature "public int compareTo(E)".
- [x] B. The Comparator interface contains the compare method with the signature "public int compare(E, E)".
- [x] C. A Comparable object can compare this object with the other object.
- [x] D. A Comparator object contains the compare method that compares two objects.

Your answer is correct

20.31 What is the output of the following code?

```
import java.util.*;

public class Test {
  public static void main(String[] args) {
    List<String> list1 = new ArrayList<>();
    list1.add("Atlanta");
    list1.add("Macon");
    list1.add("Savanna");

    List<String> list2 = new ArrayList<>();
    list2.add("Atlanta");
    list2.add("Macon");
    list2.add("Savanna");

    List<String> list3 = new ArrayList<>();
    list3.add("Macon");
    list3.add("Savanna");
    list3.add("Atlanta");

    System.out.println(list1.equals(list2) + " " + list1.equals(list3));
  }
}
```

- ( ) A. true true
- (•) B. true false
- ( ) C. false false
- ( ) D. false true

Your answer is correct

20.32 What is the output of the following code?

```
import java.util.*;

public class Test {
  public static void main(String[] args) {
    ArrayList<Student> list = new ArrayList<>();
    list.add(new Student("Peter", 65));
    list.add(new Student("Jill", 50));
    list.add(new Student("Sarah", 34));
    Collections.sort(list);
    System.out.print(list + " ");

    Collections.sort(list, new StudentComparator1());
    System.out.println(list);
  }

  static class StudentComparator1 implements Comparator<Student> {
    public int compare(Student s1, Student s2) {
      return s1.name.compareTo(s2.name);
    }
  }

  static class Student implements Comparable<Student> {
    String name;
    int age;
    Student(String name, int age) {
      this.name = name;
      this.age = age;
    }
```

```
    public int compareTo(Student s) {
      return this.age - s.age;
    }

    public String toString() {
      return "[" + name + ", " + age + "]";
    }
  }
}
```

- ○ A. [[Sarah, 34], [Jill, 50], [Peter, 65]] [[Sarah, 34], [Jill, 50], [Peter, 65]]
- ○ B. [[Jill, 50], [Peter, 65], [Sarah, 34]] [[Jill, 50], [Peter, 65], [Sarah, 34]]
- ◉ C. [[Sarah, 34], [Jill, 50], [Peter, 65]] [[Jill, 50], [Peter, 65], [Sarah, 34]]
- ○ D. [[Jill, 50], [Peter, 65], [Sarah, 34]] [[Sarah, 34], [Jill, 50], [Peter, 65]]

Your answer is correct 

Section 20.8 Static Methods for Lists and Collections

20.33 Which of the following is correct to sort the elements in a list lst?

- ○ A. lst.sort()
- ◉ B. Collections.sort(lst)
- ○ C. Arrays.sort(lst)
- ○ D. new LinkedList<String>(new String[]{"red", "green", "blue"})

Your answer is correct 

20.34 You can use the methods in the Collections class to

- ☑ A. find the maximum object in a collection based on the compareTo method.
- ☑ B. find the maximum object in a collection using a Comparator object.
- ☐ C. sort a collection.
- ☐ D. shuffle a collection.
- ☐ E. do a binary search on a collection.

Your answer is correct 
Explanation: The Collections class has a method to sort a list, shuffle a list, and perform binary search on a list, but not on collection. Note that collection includes set.

20.35 Which of the following statements are true?

- ○ A. Collections.shuffle(list) returns a new list while the original list is not changed.
- ○ B. Collections.reverse(list) returns a new list while the original list is not changed.
- ○ C. Collections.sort(list) returns a new list while the original list is not changed.
- ◉ D. Collections.nCopies(int, Object) returns a new list that consists of n copies of the object.

Your answer is correct 

20.36 Which of the following statements are true?

- ☑ A. Collections.shuffle(list) randomly reorders the elements in the list.
- ☑ B. Collections.shuffle(list, Random) randomly reorders the elements in the list with a specified Random object.
- ☑ C. If list1 and list2 are identical, the two lists may be different after invoking Collections.shuffle(list1) and Collections.shuffle(list2).
- ☑ D. If list1 and list2 are identical, the two lists are still identical after invoking Collections.shuffle(list1, new Random(3)) and Collections.shuffle(list2, new Random(3)) with the same Random object.

Your answer is correct 

20.37 Which of the following is correct to create a list from an array?

- ○ A. new List<String>({"red", "green", "blue"})
- ○ B. new List<String>(new String[]{"red", "green", "blue"})
- ◉ C. Arrays.asList<String>(new String[]{"red", "green", "blue"})
- ○ D. new ArrayList<String>(new String[]{"red", "green", "blue"})
- ○ E. new LinkedList(new String[]{"red", "green", "blue"})

Your answer is correct 

20.38 To create a set that consists of string elements "red", "green", and "blue", use

- ☐ A. new HashSet<String>({"red", "green", "blue"})
- ☐ B. new HashSet<String>(new String[]{"red", "green", "blue"})
- ☑ C. new HashSet<String>(Arrays.asList(new String[]{"red", "green", "blue"}))

- [x] D. new LinkedHashSet<String>(Arrays.asList(new String[]{"red", "green", "blue"}))
- [ ] E. new Set<String>(Arrays.asList(new String[]{"red", "green", "blue"}))

Your answer is correct

20.39 To find a maximum object in an array of strings (e.g., String[] names = {"red", "green", "blue"}), use

- ( ) A. Arrays.max(names)
- ( ) B. Arrays.sort(names)
- ( ) C. Collections.max(names)
- (•) D. Collections.max(Arrays.asList(names))
- ( ) E. None of the above

Your answer is correct

20.40 You can use the methods in the Arrays class to

- [ ] A. find the maximum object in an array based on the compareTo method.
- [ ] B. find the maximum object in an array using a Comparator object.
- [x] C. sort an array.
- [ ] D. shuffle an array.
- [x] E. do a binary search on an array.

Your answer is correct

Section 20.9 The Vector and Stack Classes

20.41 Which data type should you use if you want to store duplicate elements and be able to insert or delete elements at the beginning of the list?

- ( ) A. ArrayList
- (•) B. LinkedList
- ( ) C. Vector
- ( ) D. Set
- ( ) E. Stack

Your answer is correct

20.42 java.util.Vector is a subtype of _____.

- [ ] A. java.util.ArrayList
- [ ] B. java.util.LinkedList
- [x] C. java.util.AbstractList
- [ ] D. java.util.Vector
- [x] E. java.util.List

Your answer is correct

20.43 The methods for modifying element in the _____ class are synchronized.

- ( ) A. ArrayList
- ( ) B. LinkedList
- ( ) C. TreeMap
- (•) D. Vector
- ( ) E. HashSet

Your answer is correct

20.44 java.util.Stack is a subclass of _____.

- [ ] A. java.util.ArrayList
- [ ] B. java.util.LinkedList
- [x] C. java.util.AbstractList
- [x] D. java.util.Vector
- [x] E. java.util.List

Your answer is correct
Explanation: E is correct since Vector is a subclass of List.

Section 20.10 Queues and Priority Queues

20.45 The _____ method in the Queue interface retrieves and removes the head of this queue, or null if this queue is empty.

- (●) A. poll()
- ( ) B. remove()
- ( ) C. peek()
- ( ) D. element()

Your answer is correct 

20.46 The _____ method in the Queue interface retrieves and removes the head of this queue and throws an exception if this queue is empty.

- ( ) A. poll()
- (●) B. remove()
- ( ) C. peek()
- ( ) D. element()

Your answer is correct 

20.47 The _____ method in the Queue interface retrieves, but does not remove, the head of this queue, returning null if this queue is empty.

- ( ) A. poll()
- ( ) B. remove()
- (●) C. peek()
- ( ) D. element()

Your answer is correct 

20.48 The _____ method in the Queue interface retrieves, but does not remove, the head of this queue, throwing an exception if this queue is empty.

- ( ) A. poll()
- ( ) B. remove()
- ( ) C. peek()
- (●) D. element()

Your answer is correct 

20.49 Which of the following statements are true?

- [x] A. java.util.LinkedList implements the java.util.Queue interface.
- [ ] B. java.util.ArrayList implements the java.util.Queue interface.
- [ ] C. java.util.HashSet implements the java.util.Queue interface.
- [ ] D. java.util.LinkedHashSet implements the java.util.Queue interface.
- [x] E. java.util.PriorityQueue implements the java.util.Queue interface.

Your answer is correct 

20.50 Which of the following statements are true?

- [x] A. A PriorityQueue orders its elements according to their natural ordering using the Comparable interface if no Comparator is specified.
- [x] B. A PriorityQueue orders its elements according to the Comparator if a Comparator is specified in the constructor.
- [ ] C. The priority of a PriorityQueue cannot be changed once a PriorityQueue is created.
- [ ] D. The priority of a PriorityQueue cannot be reversed once a PriorityQueue is created.

Your answer is correct 

20.51 Analyze the following code:

```
import java.util.*;

public class Test {
  public static void main(String[] args) {
    PriorityQueue<Integer> queue =
      new PriorityQueue<Integer>(
        Arrays.asList(60, 10, 50, 30, 40, 20));

    for (int i: queue)
      System.out.print(i + " ");
  }
}
```

- ( ) A. The program displays 60 10 50 30 40 20
- ( ) B. The program displays 10 20 30 40 50 60
- ( ) C. The program displays 60 50 40 30 20 10
- (●) D. There is no guarantee that the program displays 10 20 30 40 50 60

Your answer is correct 

20.52 Analyze the following code:

```
import java.util.*;

public class Test {
  public static void main(String[] args) {
    PriorityQueue<Integer> queue =
      new PriorityQueue<Integer>(
        Arrays.asList(60, 10, 50, 30, 40, 20));

    while (!queue.isEmpty())
      System.out.print(queue.poll() + " ");
  }
}
```

- A. The program displays 60 10 50 30 40 20
- B. The program displays 10 20 30 40 50 60
- C. The program displays 60 50 40 30 20 10
- D. There is no guarantee that the program displays 10 20 30 40 50 60

Your answer is correct 

20.53 What is list after the following code is executed?

```
ArrayList<Integer> list = new ArrayList<>();
list.add(1);
list.add(2);
list.add(3);
list.add(4);
list.add(5);
list.remove(2);
System.out.println(list);
```

- A. [1, 2, 3, 4, 5]
- B. [2, 3, 4, 5]
- C. [1, 3, 4, 5]
- D. [1, 2, 4, 5]
- E. [1, 2, 3, 4]

Your answer is correct 

Explanation: The ArrayList class has two overloaded remove method remove(Object) and remove(int index). The latter is invoked for list.remove(2) to remove the element in the list at index 2.