**MCQ- Exception handling**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **1. The exception class is in \_\_\_\_ package** | | | | | | |
| A. java.file | | B. java.io | | C. java.lang | D. java.util | |
| **2. Which keyword is used to monitor statement for exception?** | | | | | | |
| A. try | B.ctach | | C. throw | | | D. throws |
| **3. If an exception  is generated in try block , then it is caught in \_\_\_\_ block** | | | | | | |
| A. finally | B. throw | | C. throws | | | D. catch |
| **4. To explicitly throw an exception , \_\_\_\_\_\_ keyword is used.** | | | | | | |
| A. try | B.ctach | | C. throw | | | D. throwing |
| **5. \_\_\_\_\_\_ is a superclass of all exception classes.** | | | | | | |
| A. Exception | B. Throwable | | C. RuntimeException | | | D. IOException |
| **6. Exception is subclass of \_\_\_\_ class** | | | | | | |
| A. Exception | B. Throwable | | C. RuntimeException | | | D. IOException |
| **7. If a statement tries to divide by zero which exception is thrown?** | | | | | | |
| A.ArithemticException | | | B.NullPointerException | | | |
| C.NumberFormatException | | | D. None of these | | | |
| **8. When a method can throw an exception then it is specified by \_\_\_\_\_ keyword** | | | | | | |
| A. finally | B. throw | | C. throws | | | D. catch |
| **9. Which block gets executed compulsory whether exception is caught or not.** | | | | | | |
| A. finally | B. throw | | C. throws | | | D. catch |
| **10. To create our own exception class , we have to \_\_\_\_\_\_\_** | | | | | | |
| A. Extend exception class | | | B. Create our own try and catch block | | | |
| C. use finally block | | | D. Use throws keyword | | | |
| **11. In case of multiple catch blocks,\_\_\_\_\_\_** | | | | | | |
| A. The superclass exception must be caught first | | | | | | |
| B. A. The superclass exception can not caught first | | | | | | |
| C. Either super or subclass can be caught first. | | | | | | |
| D. None of these | | | | | | |
| **12. When an array element is accessed beyond the array size , \_\_\_\_ exception is thrown** | | | | | | |
| A. ArrayElementOutOfLimit | | | | B. ArrayIndexOutOfBounds Exception | | |
| C. ArrayIndexOutOfBounds | | | | D. ArrayElementOutOfBounds | | |
| **13. Which method is used to print the description of the exception?** | | | | | | |
| A. printStackTrace() | | | | A. printExceptionMessage() | | |
| A. printStackMessage() | | | | A. printExceptionTrace() | | |
| **14. What is the output of following try catch block**  try  {      int i;      return;   }   catch(Exception e)     {         System.out.println(“Hello India”);     }  finally     {         System.out.println(“Hello Morbi”);     } | | | | | | |
| A. Hello India | | | | B. Hello India      Hello Morbi | | |
| C. Hello Morbi | | | | D. the program will return without printing anything | | |
| **15. Which of the following must be true of the object thrown by a throw statement?** | | | | | | |
| A. It must be assignable to the Throwable type | | | | | | |
| B. It must be assignable to the Error type | | | | | | |
| C. It must be assignable to the Exception type | | | | | | |
| D. It must be assignable to the String type | | | | | | |
|  |  |  |  |  |  |  |

**Answers:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | **- C** | 2 | **- A** | 3 | **- D** | 4 | **- C** | 5 | **- B** |
| 6 | **- B** | 7 | **- A** | 8 | **- C** | 9 | **- A** | 10 | **- A** |
| 11 | **- B** | 12 | **- B** | 13 | **- A** | 14 | **- C** | 15 | **- A** |

Question Excerpt From JAVA EXCEPTION HANDILING

|  |  |
| --- | --- |
| **Q.1)** | pick runtime exception?.... |
|  |  |

|  |  |
| --- | --- |
| **A.** | class cast exception |

|  |  |
| --- | --- |
| **B.** | File not found exception |

|  |  |
| --- | --- |
| **C.** | Nullpointer exception |

|  |  |
| --- | --- |
| **D.** | security exception |

|  |  |
| --- | --- |
| **E.** | above all |

|  |  |
| --- | --- |
| **Q.2)** | say true or false.The following program will compile: try { // do risky IO things } catch (IOException ioe) { // handle general IOExceptions } catch (IOException ioe) { // handle general IOExceptions } |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.3)** | Output of the following program  try { x.doStuff(); } int y = 50; } catch(FooException fe) { } |
|  |  |

|  |  |
| --- | --- |
| **A.** | compile without any error |

|  |  |
| --- | --- |
| **B.** | runtime error |

|  |  |
| --- | --- |
| **C.** | compile time error |

|  |  |
| --- | --- |
| **D.** | none of the above |

|  |  |
| --- | --- |
| **Q.4)** | A try with finally without catch can declare the exception |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.5)** | Which one of the following statement is correct? |
|  |  |

|  |  |
| --- | --- |
| **A.** | The ‘try’ block should be followed by a ‘catch’ block. |

|  |  |
| --- | --- |
| **B.** | The ‘try’ block should be followed by a ‘finally’ block. |

|  |  |
| --- | --- |
| **C.** | The ‘try’ block should be followed by either a ‘catch’ block or a ‘finally’ block. |

|  |  |
| --- | --- |
| **D.** | The ‘try’ block should be followed by at least two ‘catch’ blocks.  The ‘try’ block should be followed by at least two ‘catch’ blocks. |

|  |  |
| --- | --- |
| **Q.6)** | Parent of Error is....... |
|  |  |

|  |  |
| --- | --- |
| **A.** | Object |

|  |  |
| --- | --- |
| **B.** | collections |

|  |  |
| --- | --- |
| **C.** | throwable |

|  |  |
| --- | --- |
| **D.** | exception |

|  |  |
| --- | --- |
| **Q.7)** | If you throw an exception in your code, then you must declare it using the throws keyword in your method declaration. |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.8)** | The subclass exception should precede the base class exception when used within the catch clause. |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.9)** | The statements following the throw keyword in a program are not executed. |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.10)** | Exceptions can be caught or rethrown to a calling method. |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.11)** | Checked exceptions include all subtypes of Exception, including classes that extend RuntimeException. |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.12)** | File Not Found Exception class has descendants |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.13)** | In Java, exceptions are divided into two categories, namely checked and unchecked exceptions. |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.14)** | All subclasses of the RuntimeException and Error classes are unchecked exceptions. |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.15)** | when reading or writing a file it throws class not found exception |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.16)** | Question: Match each situation in the first list with an item in the second list. a)int[] A;  A[0] = 0; b)The JVM starts running your program, but the JVM can't find the Java platform classes. (The Java platform classes reside in classes.zip or rt.jar.) c)A program is reading a stream and reaches the end of stream marker. d)Before closing the stream and after reaching the end of stream marker, a program tries to read the stream again. 1\_\_error 2\_\_checked exception 3\_\_compile error 4\_\_no exception |
|  |  |

|  |  |
| --- | --- |
| **A.** | a-2,b-1,c-3,d-4 |

|  |  |
| --- | --- |
| **B.** | a-4,b-3,b-2,c-1 |

|  |  |
| --- | --- |
| **C.** | a-3,b-1,c-4,d-2 |

|  |  |
| --- | --- |
| **D.** | a-1,b-2.c-3.d-4 |

|  |  |
| --- | --- |
| **Q.17)** | what are checked exceptions |
|  |  |

|  |  |
| --- | --- |
| **A.** | checked by java compiler |

|  |  |
| --- | --- |
| **B.** | checked by java virtual machine |

|  |  |
| --- | --- |
| **C.** | above two |

|  |  |
| --- | --- |
| **D.** | none of the above |

|  |  |
| --- | --- |
| **Q.18)** | what are un checked exceptions |
|  |  |

|  |  |
| --- | --- |
| **A.** | checked by java compiler |

|  |  |
| --- | --- |
| **B.** | checked by java virtual machine |

|  |  |
| --- | --- |
| **C.** | above two |

|  |  |
| --- | --- |
| **D.** | none of the above |

|  |  |
| --- | --- |
| **Q.19)** | what is throws in exception |
|  |  |

|  |  |
| --- | --- |
| **A.** | a programmer can handle |

|  |  |
| --- | --- |
| **B.** | a programmer can not handle |

|  |  |
| --- | --- |
| **C.** | it handled by jvm |

|  |  |
| --- | --- |
| **D.** | none of the above |

|  |  |
| --- | --- |
| **Q.20)** | can i use more than one try block |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.21)** | is it possible to re-throw exceptions |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.22)** | creating an exception object and handling it to the run time system is called |
|  |  |

|  |  |
| --- | --- |
| **A.** | exception handler |

|  |  |
| --- | --- |
| **B.** | catch the exception |

|  |  |
| --- | --- |
| **C.** | pass the exception |

|  |  |
| --- | --- |
| **D.** | throwing an exception |

|  |  |
| --- | --- |
| **Q.23)** | Finally block will get invoke whether the exception is thrown or not |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

|  |  |
| --- | --- |
| **Q.24)** | Benefits of java Exception handler(any 3) |
|  |  |

|  |  |
| --- | --- |
| **A.** | skip the errors in program |

|  |  |
| --- | --- |
| **B.** | propagating errors up the call stack |

|  |  |
| --- | --- |
| **C.** | make the program to run |

|  |  |
| --- | --- |
| **D.** | grouping & diff error type |

|  |  |
| --- | --- |
| **E.** | separating error handling code from regular business logic code |

|  |  |
| --- | --- |
| **Q.25)** | exception is available in util package |
|  |  |

|  |  |
| --- | --- |
| **A.** | True |

|  |  |
| --- | --- |
| **B.** | False |

*Section 22.1 Introduction*

***1***  The Collection interface is the base interface for \_\_\_\_\_\_\_\_\_\_\_\_\_.

A. Set

B. List

C. ArrayList

D. LinkedList

E. Map

***2***  The Map is the base interface for \_\_\_\_\_\_\_\_\_\_\_\_\_.

A. TreeMap

B. HashMap

C. LinkedHashMap

D. ArrayList

E. LinkedList

***3***  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

*Section 22.2 The Collection Interface*

***5***  Which of the following methods are in the Collection interface?

A. clear()

B. isEmpty()

C. size()

D. getSize()

***6***  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

***7***  Which of the following methods are in the Collection interface?

A. remove(o: E): boolean

B. removeAll(c: Collection<?>): boolean

C. delete(o: E): boolean

D. deleteAll(c: Collection<?>): boolean

*Section 22.3 Sets*

***8***  Which of the data types below does not allow duplicates?

A. Set

B. List

C. Vector

D. Stack

E. LinkedList

***9***  Which of the following data types does not implement the Collection interface?

A. HashSet

B. TreeSet

C. ArrayList

D. LinkedList

E. Map

***10***  Which of the data types below could be used to store elements in their natural order based on the compareTo method?

A. HashSet

B. TreeSet

C. LinkedHashSet

D. Collection

E. Set

***11***  If two objects o1 and o2 are equal, what are the values for o1.equals(o2) and o1.hashCode() == o2.hashCode()?

A. true true

B. true false

C. false true

D. false false

***12***  What is the output for the following code?  
  
import java.util.\*;  
public class Test {  
    public static void main(String[] args) {  
        Set<A> set = new HashSet<A>();  
        set.add(new A());  
        set.add(new A());  
        set.add(new A());  
        set.add(new A());  
        System.out.println(set);  
    }  
}  
  
class A  {  
    int r = 1;  
  
    public String toString() {  
        return r + "";  
    }  
  
    public boolean equals(Object o) {  
        return this.r == ((A)o).r;  
    }  
  
    public int hashCode() {  
        return r;  
    }  
}

A. [1]

B. [1, 1]

C. [1, 1, 1]

D. [1, 1, 1, 1]

***13***  What is the output for the following code?  
  
import java.util.\*;  
public class Test {  
    public static void main(String[] args) {  
        Set<A> set = new HashSet<A>();  
        set.add(new A());  
        set.add(new A());  
        set.add(new A());  
        set.add(new A());  
        System.out.println(set);  
    }  
}  
  
class A  {  
    int r = 1;  
  
    public String toString() {  
        return r + "";  
    }  
  
    public int hashCode() {  
        return r;  
    }  
}

A. [1]

B. [1, 1]

C. [1, 1, 1]

D. [1, 1, 1, 1]

***14***  What is the output for the following code?  
  
import java.util.\*;  
public class Test {  
    public static void main(String[] args) {  
        Set<A> set = new HashSet<A>();  
        set.add(new A());  
        set.add(new A());  
        set.add(new A());  
        set.add(new A());  
        System.out.println(set);  
    }  
}  
  
class A  {  
    int r = 1;  
  
    public String toString() {  
        return r + "";  
    }  
  
    public boolean equals(Object o) {  
        return this.r == ((A)o).r;  
    }  
}

A. [1]

B. [1, 1]

C. [1, 1, 1]

D. [1, 1, 1, 1]

***15***  Which of the following data types have iterators?

A. HashSet

B. TreeSet

C. ArrayList

D. LinkedList

E. LinkedHashSet

***16***  To get an iterator from a set, you may use the \_\_\_\_\_\_\_\_\_\_ method.

A. getIterator

B. findIterator

C. iterator

D. iterators

***17***  Suppose set s1 is [1, 2, 5] and set s2 is [2, 3, 6]. After s1.addAll(s2), s1 is \_\_\_\_\_\_\_\_\_\_.

A. [1, 2, 2, 3, 5, 6]

B. [1, 2, 3, 5, 6]

C. [1, 5]

D. [2]

***18***  Suppose set s1 is [1, 2, 5] and set s2 is [2, 3, 6]. After s1.addAll(s2), s2 is \_\_\_\_\_\_\_\_\_\_.

A. [1, 2, 2, 3, 5, 6]

B. [1, 2, 3, 5, 6]

C. [1, 5]

D. [2, 3, 6]

E. [2]

***19***  Suppose set s1 is [1, 2, 5] and set s2 is [2, 3, 6]. After s1.removeAll(s2), s1 is \_\_\_\_\_\_\_\_\_\_.

A. [1, 2, 2, 3, 5, 6]

B. [1, 2, 3, 5, 6]

C. [1, 5]

D. [2]

***21***  The printout of the following code is \_\_\_\_\_\_\_\_\_\_\_\_.  
  
    LinkedHashSet<String> set1 = new LinkedHashSet<String>();  
    set1.add("New York");  
    LinkedHashSet<String> set2 = (LinkedHashSet<String>)(set1.clone());  
    set1.add("Atlanta");  
    set2.add("Dallas");  
    System.out.println(set2);

A. [New York]

B. [New York, Atlanta]

C. [New York, Atlanta, Dallas]

D. [New York, Dallas]

***22***  The printout of the following code is \_\_\_\_\_\_\_\_\_\_\_\_.  
  
    LinkedHashSet<String> set1 = new LinkedHashSet<String>();  
    set1.add("New York");  
    LinkedHashSet<String> set2 = set1;  
    set1.add("Atlanta");  
    set2.add("Dallas");  
    System.out.println(set2);

A. [New York]

B. [New York, Atlanta]

C. [New York, Atlanta, Dallas]

D. [New York, Dallas]

***23***  Analyze the following code:  
  
import java.util.\*;  
  
public class Test {  
  public static void main(String[] args) {  
    HashSet<String> set1 = new HashSet<String>();  
    set1.add("red");  
    Set set2 = set1.clone();  
  }  
}

A. Line 5 is wrong because a HashSet object cannot be cloned.

B. Line 5 has a compile error because set1.clone() returns an Object. You have to cast it to Set in order to compile it.

C. The program will be fine if set1.clone() is replaced by (Set)set1.clone()

D. The program will be fine if set1.clone() is replaced by (Set)(set1.clone())

E. The program will be fine if set1.clone() is replaced by (HashSet)(set1.clone())

***24***  Analyze the following code:  
  
import java.util.\*;  
  
public class Test {  
  public static void main(String[] args) {  
    Set<String> set1 = new HashSet<String>();  
    set1.add("red");  
    Set set2 = set1.clone();  
  }  
}

A. Line 5 is wrong because the declared type for set1 is Set and the clone method is not defined Set.

B. The program will be fine if set1.clone() is replaced by (HashSet)set1.clone()

C. The program will be fine if set1.clone() is replaced by (Set)((HashSet)set1).clone()

D. The program will be fine if set1.clone() is replaced by (HashSet)((HashSet)set1).clone()

E. The program will be fine if set1.clone() is replaced by (LinkedHashSet)((HashSet)set1).clone()

***25***  If you want to store non-duplicated objects in the order in which they are inserted, you should use \_\_\_\_\_\_\_\_\_\_\_\_.

A. HashSet

B. LinkedHashSet

C. TreeSet

D. ArrayList

E. LinkedList

***26***  Which of the following statements are true?

A. All the methods in HashSet are inherited from the Collection interface.

B. All the methods in TreeSet are inherited from the Collection interface.

C. All the methods in LinkedHashSet are inherited from the Collection interface.

D. All the methods in Set are inherited from the Collection interface.

E. All the concrete classes of Collection have at least two constructors. One is the no-arg constructor that constructs an empty collection. The other constructs instances from a collection.

***27***  Which of the following is correct to perform the set union of two sets s1 and s2.

A. s1.union(s2)

B. s1 + s2

C. s1.addAll(s2)

D. s1.add(s2)

***28***  Which of the following is correct to perform the set difference of two sets s1 and s2.

A. s1.difference(s2)

B. s1 - s2

C. s1.subtract(s2)

D. s1.removeAll(s2)

***29***  Which of the following is correct to perform the set intersection of two sets s1 and s2.

A. s1.intersect(s2)

B. s1.join(s2)

C. s1.retainAll(s2)

D. s1.intersection(s2)

***30***  Analyze the following code.  
  
import java.util.\*;  
  
public class Test {  
  public static void main(String[] args) throws Exception {  
    Set<String> set = new TreeSet<String>();  
      
    set.add("Red");  
    set.add("Green");  
    set.add("Blue");  
      
    System.out.println(set.first());  
  }  
}

A. The program displays Red

B. The program displays Blue

C. The program displays Green

D. The program may display Red, Blue, or Green.

E. The program cannot compile, because the first() method is not defined in Set.

***31***  Analyze the following code.  
  
import java.util.\*;  
  
public class Test {  
  public static void main(String[] args) throws Exception {  
    TreeSet<String> set = new TreeSet<String>();  
      
    set.add("Red");  
    set.add("Green");  
    set.add("Blue");  
      
    System.out.println(set.last());  
  }  
}

A. The program displays Red

B. The program displays Blue

C. The program displays Green

D. The program may display Red, Blue, or Green.

E. The program cannot compile, because the first() method is not defined in Set.

***32***  Analyze the following code.  
  
import java.util.\*;  
  
public class Test {  
  public static void main(String[] args) throws Exception {  
    TreeSet<String> set = new TreeSet<String>();  
      
    set.add("Red");  
    set.add("Yellow");  
    set.add("Green");  
    set.add("Blue");  
    SortedSet temp = set.headSet("Purple");  
      
    System.out.println(temp.first());  
  }  
}

A. The program displays Red

B. The program displays Blue

C. The program displays Green

D. The program displays Yellow

E. The program displays Purple

***33***  Analyze the following code.  
  
import java.util.\*;  
  
public class Test {  
  public static void main(String[] args) throws Exception {  
    TreeSet<String> set = new TreeSet<String>();  
      
    set.add("Red");  
    set.add("Yellow");  
    set.add("Green");  
    set.add("Blue");  
    SortedSet temp = set.tailSet("Purple");  
      
    System.out.println(temp.first());  
  }  
}

A. The program displays Red

B. The program displays Blue

C. The program displays Green

D. The program displays Yellow

E. The program displays Purple

*Section 22.4 The Comparator Interface*

***34***  Which of the following statements are true?

A. The Comparable interface contains the compareTo method with the signature "public int compareTo(Object)".

B. The Comparator interface contains the compare method with the signature "public int compare(Object, Object)".

C. A Comparable object can compare this object with the other object.

D. A Comparator object contains the compare method that compares two objects.

*Section 22.5 Lists*

***35***  Which of the following statements are true?

A. java.util.List inherits all the methods from java.util.Collection. Additionally, it contains new methods for manipulating a list.

B. The AbstractList class provides a partial implementation for the List interface.

C. ArrayList is a concrete implementation of List using an array.

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.

E. ListIterator is a subinterface of Iterator and it provides the methods to support bi-directional traversal of a list.

***36***  Which of the following statements are true?

A. An ArrayList can grow automatically.

B. An ArrayList can shrink automatically.

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.

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

A. add(int index, E element)

B. get(int index)

C. set(int index, E element)

D. remove(int index)

E. subList(int fromIndex, int toIndex)

***38***  Which of the following are true?

A. You can insert an element anywhere is an arraylist.

B. You can insert an element anywhere is a linked list.

C. You can use a linked list to improve efficiency for adding and removing elements at the beginning of a list.

D. You should use an array list if your application does not require adding and removing elements at the beginning of a list.

***39***  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.

***40***  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.

***41***  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)

***42***  When you create an ArrayList using ArrayList<String> x = new ArrayList<String>(2), \_\_\_\_\_\_\_\_

A. two elements are created in the array list.

B. no elements are currently in the array list.

C. the array list size is currently 2.

D. the array list capacity is currently 2.

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

A. x.get(2)

B. x.set(3, "New York");

C. x.get(3)

D. x.remove(3)

E. x.size()

***44***  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()

***45***  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

***46***  What is the printout of the following code?  
        List<String> list = new ArrayList<String>();  
        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

***47***  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]

***48***  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]

***49***  Which of the following statements are correct.

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.

C. When you create an ArrayList using new ArrayList(), an ArrayList object is created with no elements in the ArrayList object.

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.

***50***  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"}

***51***  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"}

D. {}

***52***  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"}

D. {}

*Section 22.6 Static Methods for Lists and Collections*

***53***  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(new String[]{"red", "green", "blue"})

***54***  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.

***55***  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.

***56***  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.sort(list1) and Collections.sort(list2).

D. If list1 and list2 are identical, the two lists are still identical after invoking Collections.sort(list1, new Random(3)) and Collections.sort(list2, new Random(3)) with the same Random object.

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

A. new List({"red", "green", "blue"})

B. new List(new String[]{"red", "green", "blue"})

C. Arrays.asList(new String[]{"red", "green", "blue"})

D. new ArrayList(new String[]{"red", "green", "blue"})

E. new LinkedList(new String[]{"red", "green", "blue"})

***58***  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"}))

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

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

***59***  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

***60***  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.

C. sort an array.

D. shuffle an array.

E. do a binary search on an array.

*Section 22.7 The Vector and Stack Classes*

***61***  Which data type should you use if you want to store duplicate elements and be able to insert or delete elements anywhere efficiently.

A. ArrayList

B. LinkedList

C. Vector

D. Set

E. Stack

***62***  java.util.Vector is a subtype of \_\_\_\_\_\_\_\_\_\_.

A. java.util.ArrayList

B. java.util.LinkedList

C. java.util.AbstractList

D. java.util.Vector

E. java.util.List

***63***  The methods for modifying element in the \_\_\_\_\_\_\_\_\_\_\_ class are synchronized.

A. ArrayList

B. LinkedList

C. TreeMap

D. Vector

E. HashSet

***64***  java.util.Stack is a subclass of \_\_\_\_\_\_\_\_\_\_.

A. java.util.ArrayList

B. java.util.LinkedList

C. java.util.AbstractList

D. java.util.Vector

E. java.util.List

*Section 22.11 Maps*

***73***  To empty a Collection or a Map, you use the \_\_\_\_\_\_\_\_\_\_ method.

A. empty

B. clear

C. zero

D. setEmpty

***74***  Which of the following are correct methods in Map?

A. put(Object key, Object value)

B. put(Object value, Object key)

C. get(Object key)

D. get(int index)

***75***  Which of the following are correct methods in Map?

A. containsKey(Object key)

B. containsValue(Object value)

C. remove(Object key)

D. remove(int index)

E. isEmpty()

***76***  The elements in \_\_\_\_\_\_\_\_ are sorted.

A. TreeSet

B. List

C. TreeMap

D. HashSet

E. LinkedHashSet

***77***  Suppose your program frequently tests whether a student is in a soccer team, what is the best data structure to store the students in a soccer team?

A. ArrayList

B. HashSet

C. LinkedList

E. Vector

***78***  Suppose your program frequently tests whether a student is in a soccer team and also need to know the student?s information such as phone number, address, and age, what is the best data structure to store the students in a soccer team?

A. ArrayList

B. HashMap

C. TreeMap

D. LinkedList

E. HashSet

***79***  Analyze the following code:  
  
public class Test {  
  public static void main(String[] args) {  
    Map<String, String> map = new HashMap<String, String>();  
    map.put("123", "John Smith");  
    map.put("111", "George Smith");  
    map.put("123", "Steve Yao");  
    map.put("222", "Steve Yao");  
  }  
}

A. After all the four entries are added to the map, "123" is a key that corresponds to the value "John Smith".

B. After all the four entries are added to the map, "123" is a key that corresponds to the value "Steve Yao".

C. After all the four entries are added to the map, "Steve Yao" is a key that corresponds to the value "222".

D. After all the four entries are added to the map, "John Smith" is a key that corresponds to the value "123".

E. A runtime error occurs because two entries with the same key "123" are added to the map.