Practice MCQ

1. What are Generics in Java?

A. A feature that allows multiple methods to have the same name but different signatures.

B. A feature that allows a class or method to operate on objects of various types while providing compile-time type safety.

C. A feature that allows a class or method to be defined with a variable number of arguments.

D. A feature that allows a class or method to be accessed from anywhere in the program.

Answer: B

Explanation: Generics in Java allow a class or method to operate on objects of various types while providing compile-time type safety. They allow you to define classes, interfaces, and methods that work with a specific type without specifying that type until the code is actually used.

2. Which of the following is NOT a benefit of using Generics in Java?

A. Compile-time type safety.

B. Elimination of type casts.

C. Improved performance.

D. Code reusability.

Answer: C

Explanation: While Generics do provide compile-time type safety, elimination of type casts, and code reusability, they do not improve performance. In fact, using Generics may result in a slight performance hit due to the overhead of type erasure.

3. What is Type Erasure in Generics?

A. The process of removing type information at compile-time.

B. The process of removing type information at runtime.

C. The process of adding type information at compile-time.

D. The process of adding type information at runtime.

Answer: A

Explanation: Type Erasure is the process of removing type information at compile-time. This means that the type parameters used in Generics are replaced with their upper bounds or

Object if no bounds are specified. This allows the code to be used with different types while still maintaining type safety at compile-time.

4. Which of the following is a wildcard parameter in Generics?

A. <T>

B. <E>

C. <?>

D. <K, V>

Answer: C

Explanation: The wildcard parameter in Generics is represented by the question mark symbol <?>. It is used as a type argument to indicate that any type can be used in its place.

5. What is the purpose of the extends keyword in Generics?

A. To specify the upper bound of a type parameter.

B. To specify the lower bound of a type parameter.

C. To specify the exact type of a type parameter.

D. To specify the generic type used in a class or method.

Answer: A

Explanation: The extends keyword in Generics is used to specify the upper bound of a type parameter. It is used to restrict the types that can be used in place of the type parameter to only those types that are subtypes of the specified upper bound.

6. Which of the following is NOT a subtype of Object in Java?

A. String

B. Integer

C. Boolean

D. Void

Answer: D

Explanation: Void is not a subtype of Object in Java. It is a keyword that represents the absence of a value.

7. What is the syntax for using generics in Java?

a. <T>

b. <Type>

c. <Object>

d. <Any>

Answer: a

Explanation: The syntax for using generics in Java is <T>, where T can be replaced with any valid identifier.

8. Can you use primitive types with generics in Java?

a. Yes

b. No

Answer: b

Explanation: You cannot use primitive types with generics in Java. Instead, you must use their corresponding wrapper classes (e.g., Integer for int).

9. What is a wildcard in Java generics?

a. A way to specify an unknown type in a generic class or method

b. A way to specify a fixed type in a generic class or method

c. A way to specify a restricted type in a generic class or method

d. A way to specify any type in a generic class or method

Answer: a

Explanation: A wildcard in Java generics is a way to specify an unknown type in a generic class or method. This can be useful when you want to write code that can work with different types of objects, but you don't know what those types are.

10. What is the difference between <?> and <? extends Object> in Java generics?

a. There is no difference

b. <?> allows any type, while <? extends Object> restricts to types that extend Object

c. <? extends Object> allows any type, while <?> restricts to types that extend Object

d. None of the above

Answer: b

Explanation: The difference between <?> and <? extends Object> in Java

11. What is the Java Collection Framework?

a. A set of classes and interfaces that provide the functionality of storing and manipulating groups of objects

b. A set of classes and interfaces that provide the functionality of storing and manipulating primitive data types

c. A set of classes and interfaces that provide the functionality of storing and manipulating strings

d. None of the above

Answer: a

Explanation: The Java Collection Framework is a set of classes and interfaces that provide the functionality of storing and manipulating groups of objects.

12. What is the difference between a Collection and a Map in Java?

a. A Collection is an ordered sequence of elements, while a Map is an unordered set of key-value pairs

b. A Collection is an unordered set of elements, while a Map is an ordered sequence of key-value pairs

c. A Collection is a set of key-value pairs, while a Map is a set of elements

d. None of the above

Answer: a

Explanation: A Collection is an ordered sequence of elements, while a Map is an unordered set of key-value pairs.

13. What is the purpose of the Iterator interface in Java?

a. To provide a way to access the elements of a Collection one at a time

b. To provide a way to add elements to a Collection

c. To provide a way to remove elements from a Collection

d. None of the above

Answer: a

Explanation: The purpose of the Iterator interface in Java is to provide a way to access the elements of a Collection one at a time.

14. What is the difference between an ArrayList and a LinkedList in Java?

a. An ArrayList is a resizable array, while a LinkedList is a linked list

b. An ArrayList is a linked list, while a LinkedList is a resizable array

c. An ArrayList is an ordered sequence of elements, while a LinkedList is an unordered set of elements

d. None of the above

Answer: a

Explanation: An ArrayList is a resizable array, while a LinkedList is a linked list.

15. What is the difference between a HashSet and a TreeSet in Java?

a. A HashSet is an unordered set of elements, while a TreeSet is an ordered set of elements

b. A HashSet is a linked list, while a TreeSet is a resizable array

c. A HashSet is a resizable array, while a TreeSet is a linked list

d. None of the above

Answer: a

Explanation: A HashSet is an unordered set of elements, while a TreeSet is an ordered set of elements.

16. What is the difference between a Set and a List in Java?

a. A Set is an ordered sequence of elements, while a List is an unordered set of elements

b. A Set is an unordered set of elements, while a List is an ordered sequence of elements

c. A Set is a set of key-value pairs, while a List is a set of elements

d. None of the above

Answer: b

Explanation: A Set is an unordered set of elements, while a List is an ordered sequence of elements.

17. What is the difference between a HashMap and a TreeMap in Java?

a. A HashMap is an unordered set of key-value pairs, while a TreeMap is an ordered set of key-value pairs

b. A HashMap is a resizable array, while a TreeMap is a linked list

c. A HashMap is a linked list, while a TreeMap is a resizable array

d. None of the above

Answer: a

Explanation: A HashMap is an unordered set of key-value pairs, while a TreeMap is an ordered set of key-value pairs.

18. What is the purpose of the Comparable interface in Java?

a. To provide a way to compare objects

b. To provide a way to iterate over a collection

c. To provide a way to store key-value pairs

d. None of the above

Answer: a

Explanation: The purpose of the Comparable interface in Java is to provide a way to compare objects.

19. What is the difference between a Queue and a Stack in Java?

a. A Queue is a data structure that follows the First-In-First-Out (FIFO) principle, while a Stack follows the Last-In-First-Out (LIFO) principle

b. A Queue is a data structure that follows the Last-In-First-Out (LIFO) principle, while a Stack follows the First-In-First-Out (FIFO) principle

c. A Queue is an ordered sequence of elements, while a Stack is an unordered set of elements

d. None of the above

Answer: a

Explanation: A Queue is a data structure that follows the First-In-First-Out (FIFO) principle, while a Stack follows the Last-In-First-Out (LIFO) principle.

20. What is the purpose of the Map interface in Java?

a. To provide a way to store key-value pairs

b. To provide a way to access the elements of a Collection one at a time

c. To provide a way to remove elements from a Collection

d. None of the above

Answer: a

Explanation: The purpose of the Map interface in Java is to provide a way to store key-value pairs.

21. What are the two types of IO streams in Java?

a. Byte and Character streams

b. Input and Output streams

c. Text and Binary streams

d. None of the above

Answer: a

Explanation: The two types of IO streams in Java are Byte and Character streams.

22. What is the difference between a Reader and an InputStream in Java?

a. A Reader is used for reading character streams, while an InputStream is used for reading byte streams

b. A Reader is used for reading byte streams, while an InputStream is used for reading character streams

c. A Reader is used for reading text files, while an InputStream is used for reading binary files

d. None of the above

Answer: a

Explanation: A Reader is used for reading character streams, while an InputStream is used for reading byte streams.

23. What is the difference between a Writer and an OutputStream in Java?

a. A Writer is used for writing character streams, while an OutputStream is used for writing byte streams

b. A Writer is used for writing byte streams, while an OutputStream is used for writing character streams

c. A Writer is used for writing text files, while an OutputStream is used for writing binary files

d. None of the above

Answer: a

Explanation: A Writer is used for writing character streams, while an OutputStream is used for writing byte streams.

24. What is the purpose of the File class in Java?

a. To represent a file or directory pathname in a platform-independent manner

b. To provide methods for reading and writing data to files

c. To provide methods for compressing and decompressing files

d. None of the above

Answer: a

Explanation: The purpose of the File class in Java is to represent a file or directory pathname in a platform-independent manner.

25. What is the difference between a FileReader and a FileInputStream in Java?

a. A FileReader is used for reading character streams, while a FileInputStream is used for reading byte streams

b. A FileReader is used for reading byte streams, while a FileInputStream is used for reading character streams

c. A FileReader is used for reading text files, while a FileInputStream is used for reading binary files

d. None of the above

Answer: a

Explanation: A FileReader is used for reading character streams, while a FileInputStream is used for reading byte streams.

26. What is the difference between a FileWriter and a FileOutputStream in Java?

a. A FileWriter is used for writing character streams, while a FileOutputStream is used for writing byte streams

b. A FileWriter is used for writing byte streams, while a FileOutputStream is used for writing character streams

c. A FileWriter is used for writing text files, while a FileOutputStream is used for writing binary files

d. None of the above

Answer: a

Explanation: A FileWriter is used for writing character streams, while a FileOutputStream is used for writing byte streams.

27. What is the purpose of the BufferedReader class in Java?

a. To read text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines

b. To read bytes from a byte-input stream, buffering bytes so as to provide for the efficient reading of bytes, arrays, and lines

c. To write text to a character-output stream, buffering characters so as to provide for the efficient writing of characters, arrays, and lines

d. None of the above

Answer: a

Explanation: The purpose of the BufferedReader class in Java is to read text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines.

28. What is the purpose of the BufferedWriter class in Java?

a. To write text to a character-output stream, buffering characters so as to provide for the efficient writing of characters, arrays, and lines

b. To write bytes to a byte-output stream, buffering bytes so as to provide for the efficient writing of bytes, arrays, and lines

c. To read text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines

d. None of the above

Answer: a

Explanation: The purpose of the BufferedWriter class in Java is to write text to a character-output stream, buffering characters so as to provide for the efficient writing of characters, arrays, and lines.

29. What is the purpose of the ObjectInputStream class in Java?

a. To read Java objects from an input stream

b. To read bytes from an input stream

c. To write Java objects to an output stream

d. None of the above

Answer: a

Explanation: The purpose of the ObjectInputStream class in Java is to read Java objects from an input stream.

30. What is the purpose of the ObjectOutputStream class in Java?

a. To write Java objects to an output stream

b. To read Java objects from an input stream

c. To write bytes to an output stream

d. None of the above

Answer: a

Explanation: The purpose of the ObjectOutputStream class in Java is to write Java objects to an output stream.

31. What does JDBC stand for in Java?

a. Java Data Connection Bridge

b. Java Database Connector

c. Java Database Connectivity

d. Java Data Conversion

Answer: c

Explanation: JDBC stands for Java Database Connectivity. It is a Java API that allows Java programs to interact with databases.

32. Which of the following is not a step in connecting to a database using JDBC?

a. Loading the JDBC driver

b. Creating a connection to the database

c. Performing a SQL query

d. Processing the query results

Answer: c

Explanation: The steps involved in connecting to a database using JDBC are loading the JDBC driver, creating a connection to the database, executing SQL statements, and processing the query results. Performing a SQL query is part of executing SQL statements.

33. Which class in JDBC is responsible for loading the JDBC driver?

a. DriverManager

b. Connection

c. Statement

d. ResultSet

Answer: a

Explanation: The DriverManager class in JDBC is responsible for loading the JDBC driver. This is typically done at the beginning of the program, before any database connections are made.

34. Which method of the Connection class is used to create a Statement object?

a. prepareStatement()

b. createStatement()

c. executeQuery()

d. getConnection()

Answer: b

Explanation: The createStatement() method of the Connection class is used to create a Statement object. The Statement object is used to execute SQL statements against the database.

35. Which method of the Statement class is used to execute a SQL query?

a. executeQuery()

b. executeUpdate()

c. execute()

d. executeBatch()

Answer: a

Explanation: The executeQuery() method of the Statement class is used to execute a SQL query. This method returns a ResultSet object that contains the results of the query.

36. Which method of the ResultSet class is used to move the cursor to the next row?

a. next()

b. previous()

c. first()

d. last()

Answer: a

Explanation: The next() method of the ResultSet class is used to move the cursor to the next row in the ResultSet. This method returns true if there is another row to move to, and false if there are no more rows.

37. Which method of the ResultSet class is used to retrieve data from the current row?

a. getInt()

b. getString()

c. getBoolean()

d. All of the above

Answer: d

Explanation: The ResultSet class provides a number of methods for retrieving data from the current row, depending on the data type of the column. For example, the getInt() method is used to retrieve an integer value, the getString() method is used to retrieve a string value, and the getBoolean() method is used to retrieve a boolean value.

38. Which interface in JDBC is used to represent a SQL statement that may have input parameters?

a. Statement

b. ResultSet

c. PreparedStatement

d. CallableStatement

Answer: c

Explanation: The PreparedStatement interface in JDBC is used to represent a SQL statement that may have input parameters. Input parameters are specified using placeholders, which are replaced with actual values at runtime.

39. Which interface in JDBC is used to represent a SQL statement that returns multiple result sets?

a. Statement

b. ResultSet

c. PreparedStatement

d. CallableStatement

Answer: d

Explanation: The CallableStatement interface in JDBC is used to represent a SQL statement that returns multiple result sets. This is typically used with stored procedures, which may return multiple result sets.

40. Which method of the Connection class is used to commit a transaction?

a. commit()

b. rollback()

c. setAutoCommit()

d. close()

Answer: a

Explanation: The commit() method of the Connection class is used to commit a transaction. A transaction is a set of SQL statements that are executed together as a single unit. If all of the statements in the transaction are successful, the changes are committed to the database. If any of the statements fail, the changes are rolled back. The rollback() method is used to undo all changes made during the current transaction. The setAutoCommit() method is used to turn auto-commit mode on or off. The close() method is used to close the database connection.

1. What will be the output of the following code?

```
import java.util.*;

public class MyClass {
```

```
  public static void main(String[] args) {
    List<String> list = new ArrayList<>();
    list.add("apple");
    list.add("banana");
    list.add("cherry");
    for (Object obj : list) {
      System.out.println((String) obj);
    }
  }
}
```

a. apple, banana, cherry

b. Compilation error

c. ClassCastException

d. NullPointerException

Answer: c. ClassCastException

Explanation: In the for-each loop, the object obj is of type Object, but we are casting it to String. As the List is a List<String>, it cannot be cast to String and it will throw a ClassCastException.

2. What will be the output of the following code?

```
import java.util.*;
public class MyClass {
  public static void main(String[] args) {
    List<Integer> list = new ArrayList<>();
    list.add(1);
    list.add(2);
    List<? extends Number> wildcardList = list;
    for (Number num : wildcardList) {
      System.out.println(num);
    }
```

```
  }
}
```

a. 1, 2

b. Compilation error

c. ClassCastException

d. NullPointerException

Answer: a. 1, 2

Explanation: The List<? extends Number> is a wildcard type that can accept any List whose elements extend Number. As List<Integer> extends Number, it can be assigned to wildcardList. So, we can iterate through the wildcardList and print the elements, which are of type Number.

3. What will be the output of the following code?

```
import java.util.*;
public class MyClass {
  public static void main(String[] args) {
    List<?> list = new ArrayList<String>();
    list.add("apple");
    list.add("banana");
    System.out.println(list.get(0));
  }
}
```

a. apple

b. Compilation error

c. ClassCastException

d. NullPointerException

Answer: b. Compilation error

Explanation: The wildcard type List<?> denotes a list of unknown type. In this case, it is assigned to a List<String>. But, as the type is unknown, we cannot add any elements to it. So, it will throw a compilation error.

4. What will be the output of the following code?

```
import java.util.*;
public class MyClass<T> {
  private T value;
  public void setValue(T value) {
    this.value = value;
  }
  public T getValue() {
    return value;
  }
  public static void main(String[] args) {
    MyClass<Integer> mc1 = new MyClass<>();
    MyClass<String> mc2 = new MyClass<>();
    mc1.setValue(10);
    mc2.setValue("hello");
    System.out.println(mc1.getValue() + ", " + mc2.getValue());
  }
}
```

a. 10, hello

b. Compilation error

c. ClassCastException

d. NullPointerException

Answer: a. 10, hello

Explanation: The generic class MyClass<T> has a type parameter T, which is used for defining the type of instance variables and methods. We create two instances of MyClass, one with type Integer and the other with type String. We can then set and get the values of the instance variables using the setValue and getValue methods. In this case, we set the values as 10 and

5. What will be the output of the following code?

```
public class Test {
```

```
    public static void main(String[] args) {
        List<Integer> integers = new ArrayList<>();
        integers.add(10);
        integers.add(20);
        List<? extends Number> numbers = integers;
        double sum = 0;
        for (Number number : numbers) {
            sum += number.doubleValue();
        }
        System.out.println(sum);
    }
}
```

A. 15.0

B. 30.0

C. Compiler error

D. Runtime error

Answer: B

Explanation:

We first create an ArrayList of Integer objects and add two integers to it. Then, we assign this list to a variable of type List<? extends Number>. This is legal since List<Integer> is a subtype of List<? extends Number>.

In the for-each loop, we iterate over the elements of the numbers list, which is declared as List<? extends Number>. This means that the loop variable number is of type Number, which is a superclass of Integer. We can call the doubleValue() method on a Number object, so we can add up the values of the elements in the loop.

In this case, the sum of the two integers is 30.0, so the program will print 30.0.

6. What will be the output of the following code?

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
```

```
        Set<String> set = new HashSet<>();
        set.add("apple");
        set.add("banana");
        set.add("cherry");
        Iterator<String> iterator = set.iterator();
        while (iterator.hasNext()) {
            System.out.print(iterator.next() + " ");
            set.remove("banana");
        }
    }
}
```

a) An exception is thrown

b) apple banana cherry

c) apple cherry

d) apple

Answer: a) An exception is thrown

Explanation: When using an iterator to iterate over a collection, you should not modify the collection during iteration. In this case, the set.remove("banana") method is called inside the while loop, which modifies the set and causes a ConcurrentModificationException to be thrown.

7. What will be the output of the following code?

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        List<Integer> list = new ArrayList<>();
        list.add(1);
        list.add(2);
        list.add(3);
```

```
        for (int i : list) {
            list.add(i + 1);
            System.out.print(i + " ");
        }
    }
}
```

a) An exception is thrown

b) 1 2 3 4 5 6

c) 1 2 3 4 5

d) 1 2 3

Answer: a) An exception is thrown

Explanation: When using a for-each loop to iterate over a collection, you should not modify the collection during iteration. In this case, the list.add(i + 1) method is called inside the loop, which modifies the list and causes a ConcurrentModificationException to be thrown.

8. What will be the output of the following code?

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Map<String, Integer> map = new HashMap<>();
        map.put("apple", 1);
        map.put("banana", 2);
        map.put("cherry", 3);
        for (String key : map.keySet()) {
            System.out.print(map.get(key) + " ");
        }
    }
}
```

a) 1 2 3

b) 3 2 1

c) The order of output cannot be determined

d) None of the above

Answer: c) The order of output cannot be determined

Explanation: A HashMap does not guarantee the order of its elements, so the order of the output cannot be determined.

9. What will be the output of the following code?

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        List<String> list1 = new ArrayList<>();
        list1.add("apple");
        list1.add("banana");
        List<String> list2 = new ArrayList<>();
        list2.add("cherry");
        list2.add("banana");
        Set<String> set = new HashSet<>(list1);
        set.addAll(list2);
        for (String fruit : set) {
            System.out.print(fruit + " ");
        }
    }
}
```

a) apple banana cherry

b) apple cherry banana

c) banana cherry apple

d) The order of output cannot be determined

Answer: b) apple cherry banana

Explanation: A HashSet does not guarantee the order of its elements, but the `addAll

10. What will be the output of the following code snippet?

```
List<Integer> numbers = Arrays.asList(1, 2, 3, 4, 5);
numbers.replaceAll(n -> n * 2);
System.out.println(numbers);
```

a. [1, 2, 3, 4, 5]

b. [2, 4, 6, 8, 10]

c. [1, 4, 9, 16, 25]

d. [2, 3, 4, 5, 6]

Answer: b

Explanation: The code creates a new List object using the Arrays.asList() method and populates it with five elements. It then calls the replaceAll() method with a lambda expression that doubles the value of each element. As a result, each element is multiplied by 2, producing the output [2, 4, 6, 8, 10].

11. What will be the output of the above code if the "file.txt" contains "Hello world!"?

```
import java.io.*;
public class Test {
  public static void main(String args[]) {
    try {
      FileInputStream input = new FileInputStream("file.txt");
      byte[] buffer = new byte[10];
      input.read(buffer);
      input.close();
      System.out.println(new String(buffer));
    } catch(Exception e) {
      e.getStackTrace();
    }
  }
}
```

a. Hello

b. Hello world!

c. Hel

d. He

Answer: c. Hel

Explanation: In the code, a FileInputStream object is created to read from a file named "file.txt". Then, a byte array of length 10 is created to hold the bytes read from the file. The read() method of the FileInputStream object is called to read 10 bytes into the buffer. Since "file.txt" contains "Hello world!", the buffer will contain the bytes for "Hello worl" (the first 10 characters of the file). Finally, a String is created using the byte array and printed to the console, which will output "Hel".

12. What will be the output of the above code?

```
import java.io.*;
public class Test {
  public static void main(String args[]) {
    try {
      FileOutputStream output = new FileOutputStream("file.txt");
      String str = "Hello world!";
      output.write(str.getBytes());
      output.close();
      FileInputStream input = new FileInputStream("file.txt");
      byte[] buffer = new byte[10];
      input.read(buffer);
      input.close();
      System.out.println(new String(buffer));
    } catch(Exception e) {
      e.getStackTrace();
    }
  }
}
```

a. Hello

b. Hello world!

c. Hel

d. He

Answer: b. Hello world!

Explanation: In the code, a FileOutputStream object is created to write to a file named "file.txt". A String "Hello world!" is created and its bytes are written to the file using the write() method of the FileOutputStream object. Then, a FileInputStream object is created to read from the same file. A byte array of length 10 is created to hold the bytes read from the file. The read() method of the FileInputStream object is called to read 10 bytes into the buffer. Since the file contains "Hello world!", the buffer will contain the bytes for "Hello worl" (the first 10 characters of the file). Finally, a String is created using the byte array and printed to the console, which will output "Hello world!".

13. What will be the output of the following code?

```
import java.io.*;
public class OutputGuessing {
  public static void main(String[] args) throws Exception {
    FileWriter fw = new FileWriter("output.txt");
    BufferedWriter bw = new BufferedWriter(fw);
    PrintWriter pw = new PrintWriter(bw);

    pw.println("Hello");
    pw.println("World");
    pw.close();
    FileReader fr = new FileReader("output.txt");
    BufferedReader br = new BufferedReader(fr);
    String line;
    while ((line = br.readLine()) != null) {
      System.out.println(line);
    }
    br.close();
  }
```

```
}
```

a. Hello World

b. HelloWorld

c. Hello

World

d. None of the above

Answer: c. Hello World

Explanation: The code creates a new text file called "output.txt" and writes "Hello" and "World" to it using a PrintWriter object. Then, the code reads the contents of the file using a BufferedReader object and prints them to the console. Since each call to pw.println() writes a new line to the file, the output will be "Hello" on the first line and "World" on the second line.

14. What will be the output of the following code?

```
import java.io.*;
public class OutputGuessing {
  public static void main(String[] args) throws Exception {
    FileOutputStream fos = new FileOutputStream("output.txt");
    ObjectOutputStream oos = new ObjectOutputStream(fos);
    oos.writeObject("Hello");
    oos.writeObject("World");
    oos.close();
    FileInputStream fis = new FileInputStream("output.txt");
    ObjectInputStream ois = new ObjectInputStream(fis);
    System.out.println(ois.readObject());
    System.out.println(ois.readObject());
    ois.close();
  }
}
```

a. Hello World

b. HelloWorld

c. Hello

World

d. None of the above

Answer: a. Hello World

Explanation: The code creates a new binary file called "output.txt" and writes two String objects to it using an ObjectOutputStream object. Then, the code reads the contents of the file using an ObjectInputStream object and prints them to the console. Since the order of the calls to oos.writeObject() is preserved, the output will be "Hello" followed by "World".

15. What will be the output of the following Java program?

Assume that the file "file.txt" contains the following 3 line text:

This is a test file.

It contains multiple lines.

Each line is separated by a newline character.

```
import java.io.*;
public class Test {
  public static void main(String[] args) throws Exception {
    try {
      FileReader fr = new FileReader("file.txt");
      BufferedReader br = new BufferedReader(fr);
      String line = null;
      while((line = br.readLine()) != null) {
        System.out.println(line);
      }
      br.close();
      fr.close();
    } catch(IOException e) {
      System.out.println(e);
    }
```

```
  }
}
```

a. The contents of the file "file.txt":

This is a test file. It contains multiple lines. Each line is separated by a newline character.

b. An error message

c. The following output:

This is a test file.

It contains multiple lines.

Each line is separated by a newline character.

d. None of the above

Answer: c

Explanation: The program reads the contents of the file "file.txt" using a FileReader and a BufferedReader, which reads the file one line at a time using the readLine() method. Each line is then printed on the console using System.out.println(). If the file is empty, then nothing will be printed on the console. If there is an error while reading the file, then the catch block will handle the exception and print the error message. In this case, the program reads the contents of the file and prints it on the console.

16. What will be the output of the following Java program that connects to a MySQL database using JDBC?

Assume that the "students" table in the "testdb" database contains the following data:

name	age
John	20
Jane	22
James	25

```
import java.sql.*;
public class Test {
  public static void main(String[] args) throws Exception {
    try {
      Class.forName("com.mysql.jdbc.Driver");
```

```
        Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb",
"root", "password");
        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT * FROM students");
        while(rs.next()) {
          System.out.println(rs.getString("name") + "\t" + rs.getInt("age"));
        }
        rs.close();
        stmt.close();
        con.close();
      } catch(Exception e) {
        System.out.println(e);
      }
    }
}
```

a. An error message

b. The following output:

John 20

Jane 22

James 25

c. The SQL query being executed

d. None of the above

Answer: b

Explanation: The program connects to a MySQL database using JDBC and reads the data from the "students" table using a SELECT query. The data is then printed on the console using System.out.println(). The output will be:

John 20

Jane 22

James 25

If there is an error while connecting to the database or executing the query, then the catch block will handle the exception and print the error message. In this case, assuming that the connection and query are successful, the program will print the data from the table.

17. What will be the output of the following code snippet?

```
import java.sql.*;
public class JDBCTest {
  public static void main(String[] args) {
    try {
      Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb", "root", "password");
      Statement stmt = conn.createStatement();
      String sql = "SELECT COUNT(*) FROM employees";
      ResultSet rs = stmt.executeQuery(sql);
      System.out.println(rs.getInt(1));
      conn.close();
    } catch (SQLException e) {
      System.out.println(e.getMessage());
    }
  }
}
```

a. The program will print the number of employees in the database.

b. The program will print an error message if it fails to connect to the database.

c. The program will throw a SQLException.

d. The program will print nothing, as it does not contain a call to System.out.println().

Answer: a.

The program will print the number of employees in the database. The SQL query returns a single value representing the count of rows in the "employees" table, which is retrieved using rs.getInt(1).

18. What will be the output of the following code snippet?

```
import java.sql.*;

public class JDBCTest {

  public static void main(String[] args) {

    try {

      Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb", "root", "password");

      Statement stmt = conn.createStatement();

      String sql = "INSERT INTO employees (id, name, age) VALUES (1, 'Alice', 30)";

      int result = stmt.executeUpdate(sql);

      System.out.println(result);

      conn.close();

    } catch (SQLException e) {

      System.out.println(e.getMessage());

    }

  }

}
```

a. The program will print the number of rows affected by the SQL query.

b. The program will print an error message if it fails to connect to the database.

c. The program will throw a SQLException.

d. The program will print nothing, as it does not contain a call to System.out.println().

Answer: a. The program will print the number of rows affected by the SQL query.

Explanation:

In this code snippet, a connection is established to a MySQL database, and an SQL query is executed to insert a new row into a table called "employees". The query is executed using the Statement object's executeUpdate() method, which returns an integer value indicating the number of rows affected by the query. This value is then printed to the console using System.out.println().

19. What will be the output of the following code snippet?

```
import java.sql.*;

public class JDBCTest {
```

```
  public static void main(String[] args) {
    try {
      Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb", "root", "password");
      Statement stmt = conn.createStatement();
      String sql = "UPDATE employees SET age = age + 1 WHERE id = 1";
      int result = stmt.executeUpdate(sql);
      System.out.println(result);
      conn.close();
    } catch (SQLException e) {
      System.out.println(e.getMessage());
    }
  }
}
```

a. The program will print the number of rows affected by the SQL query.

b. The program will print an error message if it fails to connect to the database.

c. The program will throw a SQLException.

d. The program will print nothing, as it does not contain a call to System.out.println().

Answer: (a) The program will print the number of rows affected by the SQL query, which is the number of rows where the value of the age column was updated to the new value. In this case, since the query only updates one row, the output will be 1.

20. What will be the output of the following code snippet?

```
import java.sql.*;
public class JDBCTest {
  public static void main(String[] args) {
    try {
      Connection conn =
DriverManager.getConnection("jdbc:mysql://localhost:3306/testdb", "root", "password");
```

```
        PreparedStatement pstmt = conn.prepareStatement("SELECT * FROM employees
WHERE age > ?");
        pstmt.setInt(1, 40);
        ResultSet rs = pstmt.executeQuery();
        while (rs.next()) {
          System.out.println(rs.getString("name"));
        }
        conn.close();
      } catch (SQLException e) {
        System.out.println(e.getMessage());
      }
    }
}
```

a. The program will print the names of all employees in the database whose age is greater than 40.

b. The program will print an error message if it fails to connect to the database.

c. The program will throw a SQLException.

d. The program will print nothing, as it does not contain a call to System.out.println().

Answer: a.

Explanation:

The program establishes a connection to a database and prepares a PreparedStatement object to execute a SQL query that selects all employees whose age is greater than a parameter value. The parameter is set to 40 using the setInt() method. The query is executed using the executeQuery() method, which returns a ResultSet object that contains the results of the query. The while loop iterates over the rows in the ResultSet, and for each row, it prints the value of the name column using the getString() method.

Therefore, the program will print the names of all employees in the database whose age is greater than 40.