What will be the output of the below code?

```
class ListExample{
    public static void main(String[] args
    {
      List<String> list = new ArrayList<>();
      list.add("I");
      list.add("Love");
      list.add("Java");
      list.add("Language");
      Iterator<Object> iter = list.iterator();
      while (iter.hasNext())
         System.out.print (iter.next().toString() + " ");
      System.out.println();
     }
}
```

Assumption: All classes, interfaces, and necessary methods are available.

A. I Love Java Language

B. Error: Incompatible types: String cannot be converted to Object

C. Error: iterator cannot be created for Object

D. Error: to string() cannot be applied on a string object

Q2. What is the output of the following code snippet?

```
class Bill{
    int itemPrice;
    public Bill (int itemPrice) { this.itemPrice = itemPrice;}
    void display() {
        int itemPrice = 20;
       System.out.println(itemPrice);
    }
}
class Demo {
     public static void main(String[] args) {
        Bill billobj = new Bill (10);
        System.out.println (billobj.itemPrice);
         billobj.display();
     }
}
```

A. 10

0

B. 10

20

C. 10

10

D. Error in the class as there is no default constructor defined

Q3. Consider the Binary Search code given below:

```
public static int search(int arrayofElements[], int low, int high, int
elementToBeSearched)
{
  if (low <= high)
  {
    int mid = (low + high) / 2;
    if (arrayofElements[mid] == elementToBeSearched)
      return mid;
     if (arrayofElements[mid] < elementToBeSearched)
      return search(arrayofElements, mid + 1, high, elementToBeSearched);
    return search(arrayOfElements, low, mid - 1, elementToBesearched);
  }
  return -1;
}
```

Consider the arrayOfElements having 6 elements with low as 0 and high as 5. The elements of the array are as follows.

5 6 9 12 15 29

Find the number of iterations when using binary search if the elementToBeSearched is 6?

A. 1

B. 2

C. 3

D. 4

Q4. Consider the code given below:

```
class Student {
   private int studentId;
   private String studentName;
   Student (int studentId, String studentName) {
       this.studentId = studentId;
       this.studentName = studentName;
   }
}
class College {
    private Student student;
    private int basicFees;
    College (Student student, int basicFees) {
       this.student = student;
       this.basicFees = basicFees;
    }
}
```

Identify the relationship between Student and College classes.

A. Aggregation

B. Association

C. Inheritance

D. The two classes are not related

Q5. Consider the code given below.

Identify the code that needs to be filled in Line 1, 2, and 3 respectively such that:

• The student id is auto-generated starting from 501 in steps of 1

• The method 'getNoOfStudent' returns the total number of students enrolled at any given point.

```
class Student{
   private int studentId;
   private String studentName;
   private int yearofEnrollment;
   public static int counter;
   static {
       //Line 1
   }
   public Student (String name, int yearOfEnrollment) {
      this.studentName=name;
      this.yearOfEnrollment=yearofEnrollment;
                  // Line 2
   }
   public static int getNoOfStudent () {
                  // Line 3
    }
}
```

A. Line 1: Student.counter=501;

Line 2: this.studentid=Student.counter++;

Line 3: return (Student.counter-500);

B. Line 1: Student.counter=501;

Line 2: this.studentid=++Student.counter;

Line 3: return (Student.counter-501);

C. Line 1: Student.counter=500;

Line 2: this.studentId=Student.counter++;

Line 3: return (Student.counter-500);

D. Line 1: Student.counter=500;

Line 2: this.studentid=++Student.counter;

Line 3: return (Student.counter-500);

Q6. Consider the code snippet given below:

```
class Customer {
   public int custId;
   public String custName;
}
public class Tester {
   public static void main(String args[]) {
      Customer obj = new Customer();
      Customer objone = new Customer();
      Customer objTwo;
      Customer objThree = obj;
   }
}
```

How many object and reference variables of class Customer will be created?

A. 3 objects and 1 reference variable

B. 2 objects and 4 reference variables

C. 4 objects and 4 reference variables

D. 2 objects and 3 reference variables

Q7. Consider the code given below:

```
class ClassA{
  void firstMethod() {
      System.out.println("Johnny Johnny..., ");
  }
  void secondMethod() {
     System.out.println("Yes Papa. ");
  }
  void thirdMethod() {
     System.out.println("Eating Sugar..., ");
  }
}
class ClassB extends ClassA{
  void secondMethod(){
          super.firstMethod();
          super. secondMethod();
          super.thirdMethod();
          System.out.println("No Papa. ");
  }
```

```
   void thirdMethod() {
    System.out.println("Telling Lies..., ");
   }
}
class ClassC extends ClassB{
    void firstMethod() {
        System.out.println("Open your mouth..., Ha. Ha. Ha.");}
    void secondMethod() {
        System.out.println("No Papa. ");}
    void thirdMethod() {
        super.secondMethod();
        super. thirdMethod();
        this. secondMethod();
    }
            public static void main(String[] args) {
        ClassA objA= new ClassA();
        ClassB objB=new ClassB();
        ClassC objC=new ClassC();
        //Line 1
   }
}
```

Which among the below options if written at //Line 1, prints the rhyme correctly?

Choose two CORRECT options.

The expected output for your reference:

Johnny Johnny. . . ,

Yes Papa.

Eating Sugar. . .,

No Papa.

Telling Lies . . .,

No Papa.

Open your mouth. . ., Ha. Ha. Ha.

A. objA.firstMethod();

objA.secondMethod();

objA.thirdMethod();

objC.firstMethod();

B. objC.thirdMethod();

objC.firstMethod();

C. objB.secondMethod();

objB.thirdMethod();

objC.secondMethod();

objC.firstMethod();

D. objA.firstMethod();

objB.secondMethod();

objC.thirdMethod();

objC.firstMethod();

Q8. Consider an input queue inQueue of Strings with the following elements:

inQueue(Front-> Rear): "Crib", "Bat", "Crab", "Carl", "Cat", "Row"

What will be the output of the below function if the above inQueue and the String "Par" are passed as input parameters?

```
public static ArrayDeque<String> compareStrings(Queue inQueue, String
inString) {
   ArrayDeque<String> outStack=new ArrayDeque<String>(6);
   String tempString="Empty";
   while (!inQueue.isEmpty()) {
      if (! (inQueue.poll().length()==inString.length())) {
          outStack.push(inQueue. poll());
      }
      else{
             tempString=inQueue. poll();
         outStack.pop();
      }
   }
     outStack.push(tempString);
     return outStack;
}
```

Assumptions:

• Queue class, with the necessary methods, is available

• ArrayDeque class, with the necessary methods, is available

A. outStack(Top->Bottom): [Empty]

B. outStack(Top->Bottom): [Empty, Bat]

C. outStack(Top->Bottom): [Row, Bat]

D. outStack(Top->Bottom): [Row]

Q9. An Employee Management System application is used to maintain information about employees in an organization. In the application, employee details are stored in the ascending order of the employee Ids. Which algorithmic design technique would best fit if an employee needs to be searched based on the employee Id.

A. Greedy Approach

B. Brute Force

C. Divide and Conquer

D. Dynamic Programming

Q10. What is the output of the following code?

```
class Base {
   private int fun() {
      return 0;
   }
    public int run() {
      return 3;
   }
}
class Derived extends Base {
   private int fun() {
      return 1;
   }
   public int run() {
      return fun();
```

```
    }
}
class Derivedl extends Derived {
    public int fun() {
        return 2;
    }
}
class Tester {
    public static void main(String[] args) {
        Base baseRef = new Derivedl();
        System.out.println (baseRef.run());
    }
}
```

A. 1

B. 2

C. 0

D. 3

Question 11:

An Employee Management System application is used to maintain information about employees in an organization. In the application, employee details are stored in the ascending order of the employee Ids. Which algorithmic design technique would best fit if an employee needs to be searched based on the employee ld.

A. Greedy Approach

B. Brute Force

C. Divide and Conquer

D. Dynamic Programming

Question 12:

What is the output of the following code snippet?

```
public class Tester {
    Public static void main (String[ ] args) {
            for(int loop = 0;loop < 5;loop++) {
                    if(loop > 2) {
                        continue;
                    }
                    if(loop>4) {
                        break;
                    }
                    System.out.println(loop) ;
            }
    }
}
```

A:

0

1

2

B:

0

C:

0

1

2

3

4

D:

0

1

Question1 3:

Which of the following statements is TRUE with respect to Java language being platform independent?

A. The code in the java file is platform dependent

B. The JVM is the same across all operating systems

C. A java program written in a machine with Windows operating system cannot be executed on a machine having Linux operating system though Java is installed accordingly

D. A .class file can be run in any operating system where Java is installed

Question 14:

What is the output of the following code snippet?

```
class Demo{
  public static int specialAdd(int num1) {
        if (num1!=0)
            return (num1+2)+specialAdd(num1-1) ;
        elsereturn 3;
  }
  public static int extraordinaryAdd(int num2) {
         if (num2!=0)
             return specialAdd(num2)+extraordinaryAdd(num2-1) ;
          elsereturn 0;
  }
   public static void main (String [ ] args) {
            System.out.println( (extraordinaryAdd(5) ) ) ;
  }
}
```

A. 80

B. 52

C.70

D. 25

Question 15:

What is the output of the code given below when run with the default Junit runner?

```
class Computation {
 public int add(int num1, int num2) {
          return num1 + num2 ;
 }
 public int divide(int num1, int num2) {
          return num1 / num2 ;
 }
}
public class TestComputation {
  Computation comput = new Computation ( ) ;
  @Test
   public void testAdd1 ( ) {
          int expected = 5 ;
              int actual = comput.add(2, 3) ;
          Assert.assertEquals(expected, actual) ;
 }
 @Test
  public void testAdd2 ( ) {
           int expected = 7 ;
```

```
        int actual = comput.add(2, 5) ;
        Assert.assertEquals(expected , actual) ;
 }
}
```

A. Both testAdd1 and testAdd2 fail

B. testAdd1 fails and testAdd2 passes

C. Both testAdd1 and testAdd2 pass

D. testAdd1 passes and testAdd2 fails

Question 16:

Consider the code snippet given below:

```
class Customer {
   public int custId ;
   public String custName ;
}
public class Tester {
    public static void main (String args{ } ) {
          Customer obj = new Customer ( ) ;
          Customer objOne = new Customer ( ) ;
          Customer objTwo ;
           Customer objThree = obj ;
    }
}
```

A. 3 objects and 1 reference variable

B. 2 objects and 4 reference variables

C. 4 objects and 4 reference variables

D. 2 objects and 3 reference variables

Question 17:

Consider the code given below:

```
class Student {
  private int studentId ;
  private String studentName ;
  Student (int studentId,String studentName) {
          this.studentId = studentId ;
           this.studentName = studentName ;
  }
}
class College {
  private Student studentId ;
  private int basicFees ;
  College (Student studentId, int basicFees) {
          this.studentId = student ;
           this.basicFees = basicFees ;
  }
}
```

Identify the relationship between Student and College classes.

A. Aggregation

B. Association

C. Inheritance

D. The two classes are not related

Question 18:

What is the output of the following code snippet?

```
public class ExceptionExample {
   public void checkForExceptions(int num1, int num2) {
          int intArr [ ] = {1,2,3} ;
          String str = null ;
          System.out.println("Before any exception!") ;
          try{
                str.charAt(0) ;
                System.out.println(num1 / num1) ;
                System.out.println("Enjoy no exception!") ;
          }
          catch (ArithmeticException e) {
                     System.out.println("ArithmeticException handler!") ;
           } catch (NullPointerException e) {
                      System.out.println("NullPointException handler!") ;
           } catch (Exception e) {

                     System.out.println("Default exception handler!") ;
         } finally {
                  System.out.println("In finally!");
         }
         System.out.println("After handling exception!") ;
   }
   public static void main(String [ ] args)
```

```
   {
        ExceptionExample exceptionExample = new ExceptionExample( ) ;
        try {
                exceptionExample.checkForExceptions(2, 0) ;
         } catch (ArithmeticException e) {
                 System.out.println("ArithmeticException handler in main!") ;
         }
         System.out.println("End of main") ;
   }
}
```

A:

Before any exception!

Enjoy no exception!

In finally!

After handling exception!

End of main

B:

Before any exception!

Default exception handler!

In finally!

After handling exception!

End of main

C:

Before any exception!

ArithmeticException handler!

In finally!

After handling exception!

ArithmeticException handler in main!

End of main

D:

Before any exception!

NullPointerException handler!

In finally!

After handling exception!

End of main

Question 19:

Consider the problem size as 'n'. Find the worst-case time complexity of the following algorithm.

```
if num1>num2 thenfor (couter1=1;counter1<=n;counter1=counter1*2)
```

```
 print(“num1 is greater than num2”)
  end-forelsefor(counter2=1;counter2<=n;counter2=counter2+1) {
 print(“num2 is greater than num1”)
   end-forend-if
```

A. O(n)

B. O(n2)

C. O(log n)

D. O(n log n)

Question 20:

Consider the code given below which is written in the file ‘Demo.java’.

```
class Book{
  / /Class definition
}
class Demo{
   public static void main(String [ ] args) {
   }
}
```

How many .class files will be generated for the above code and which class out of the two, Demo or Book, will be loaded into the main memory first when executed?

A. 2, Demo

B. 2, Book

C. 1, Demo

D. 1, Book