Question-2
key points about Constructor

* The constructor name must be same as class name.
* Constructor doesnot have any retun type.
* Constructor is genrally of two types Default and Parameterized constructor.
* Constructor can be overloaded.
* Generally constructor is used for initialization of the instance variables.
* Constructor gets called by the JVM during object creation.
* Constructors can have access modifiers like public, private, protected, or no modifier (default access).
* Constructor chaining can be performed using this() method. But constructor regression is not possible.
* Using super() method we can call parent class constructor.
* Super() and this() method should be at first line inside constructor, we cannot place these methods
  anywhere except at the first line or else we will get compile time error.

Example :-

```
Plane.java
==========
public class Plane {

 private String name;
 private String type;
 private Double totalweight;

 public Plane(String name, String type, Double totalweight) {
  super();
  this.name = name;
  this.type = type;
  this.totalweight = totalweight;
 }
}
CargoPlane.java
==============
public class CargoPlane extends Plane {

 private Double totalGoodsWeigth;

 public CargoPlane(String name, String type, Double totalweight, Double totalGoodsWeigth) {
  super(name, type, totalweight);

  this.totalGoodsWeigth = totalGoodsWeigth;
 }
}

TestApp.java
============
public class TestApp {

 public static void main(String[] args) {

  CargoPlane cargePlane = new CargoPlane("Airbus", "Cargo", 80000.05, 750000.22);
  System.out.println(cargePlane);
 }
}
```

* In the above example we have two classes Plane and CargoPlane. Plane is parent class , while cargoplane is child class.
* Inside CargoPlane class(child) constructor we have to call the parent class constructor, since by default parent class constructor is called by child classconstructor through super() method.

Program for this question-2 is in program file.

==

Question-3
Write a Java programme that takes an integer from the user and throws an exception if it is negative.Demonstrate Exception handling of same program as solution.

Exception Handing :-
* Where there is risky code i.e, chances of getting any exception we place that code inside try and catch block.
* The risky code is placed inside try block if there any exception during execution, then the catch block catches the exception and if there is any handling logic that logic will get executed.

Example:-

```
public class TestApp {
 public static void main(String[] args) {

  Scanner sc = new Scanner(System.in);
  System.out.println("Enter Number");

  int number = sc.nextInt();

  try {
   checkNumberIsNegativeOrNot(number);
  } catch (ArithmeticException e) {
   System.out.println(e.getMessage());
   e.printStackTrace();
  } catch (Exception e) {
   e.printStackTrace();
  } finally {
   sc.close();
  }
 }

 private static String checkNumberIsNegativeOrNot(Integer number) throws ArithmeticException {
  if (number < 0) {
   throw new ArithmeticException("Negative Number");
  } else {
   return "Entered number is a positive number";
  }
 }
}
```

* In the above example we have a method(checkNumberIsNegativeOrNot) to check if the number
  negative or not,if the number is negative it throws a ArithmeticException.
* During Execution when the user enters the number , from the main method we call this
  checkNumberIsNegativeOrNot method,since this method will returns string if it is a
  positive number or else it will throw an exception. Since it is a risky code we place
  this code inside a try and catch block.While calling the method if it returns exception
  the ArithmeticException catch block catches the exception and the logic inside this catch
  block gets executed.

Program for this Question-3 is in the programs file
===

Question-5
Demonstrate the difference between abstract class and interface?

Interface
=========
* In interface we have only methods which are by default abstract.But from JDK 1.8 we can have
  static and default methods.
* Variables inside interface are by default public static and final
* Interface supports multiple inheritance A class can implement any number of interfaces.
* Interface can be implemented using implements keyword.
* Members in inheritance are by default public.
* We can achieve 100% abstraction.
* Default methods can be inherted and overridden.
* static methods cannot participate in inheritance ,but can be called directly.
* Interface corresponds to SRS(servic requriment specification) or contract between client
  and service

Abstract class
=============
* In abstract class we can have both abstract and non abstract methods.
* In abstract class we can have final, non-final , static and non-static variables.
* Abstract class does not support multiple inheritance. A class can extend only one abstract class.
* Abstract class can be extended using extends keyword.
* Abstract class members can be public,protected,private.
* We cannot achieve 100% abstraction.

Program for this Question-5 is in the programs file.