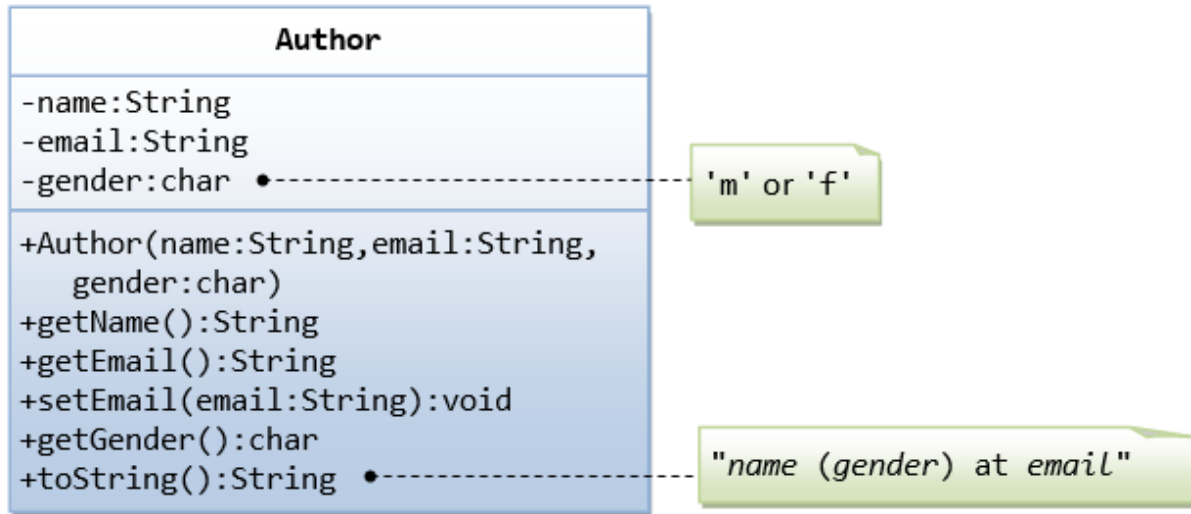


Q1. Convert this UML class diagram to java code and test the java code.



Solution:

```
9 | * @author RAJ
10 | */
11 | public class Author {
12 |     private String name;
13 |     private String email;
14 |     private char gender;
15 |
16 |     public Author(String name, String email, char gender)
17 |     {
18 |         this.name=name;
19 |         this.email=email;
20 |         this.gender=gender;
21 |     }
22 |     public String getName()
23 |     {
24 |         return name;
25 |     }
26 |     public String getEmail()
27 |     {
28 |         return email;
29 |     }
```

```

30 public void setEmail(String Email)
31 {
32     this.email=email;
33 }
34 public char getGender()
35 {
36     return gender;
37 }
38
39 public String toString()
40 {
41     return name + "(" +gender + ") at " + email;
42 }
43 }

```

```

9  * @author RAJ
10 */
11 public class Test {
12     public static void main(String[] args){
13         Author raj=new Author("Raj", "rajlama2007@gmail.com" , 'm');
14         System.out.println(raj);
15         raj.setEmail("rajlama2007@gmail.com");
16         System.out.println("Name:" + raj.getName());
17         System.out.println("Email:"+ raj.getEmail());
18         System.out.println("Gender:"+ raj.getGender());
19     }
20 }

```

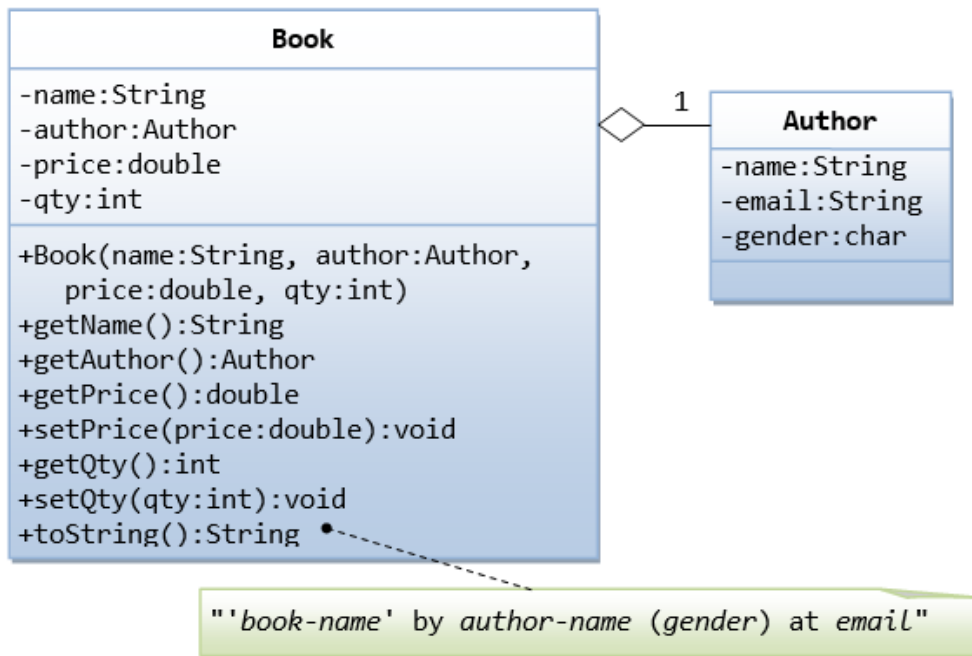
Output - First_Assignment (run) X

```

run:
Raj(m) at rajlama2007@gmail.com
Name:Raj
Email:rajlama2007@gmail.com
Gender:m
BUILD SUCCESSFUL (total time: 0 seconds)

```

Q2. Convert this UML class diagram to java code and test the java code.



Solution:

```
9  | * @author RAJ
10 | */
11 | public class Book {
12 |     private String name;
13 |     private Author author;
14 |     private double price;
15 |     private int qty;
16 |
17 |     public Book(String name, Author author, double price, int qty){
18 |         this.name=name;
19 |         this.author=author;
20 |         this.price=price;
21 |         this.qty=qty;
22 |     }
23 |
24 |     public String getName()
25 |     {
26 |         return name;
27 |     }
28 |     public Author getAuthor()
29 |     {
30 |         return author;
31 |     }
```

```

32 public double getPrice()
33 {
34     return price;
35 }
36 public void setPrice(double price)
37 {
38     this.price=price;
39 }
40 public int getQty()
41 {
42     return qty;
43 }
44 public void setQty(int qty)
45 {
46     this.qty=qty;
47 }
48
49 public String toString()
50 {
51     return "\"" + name + "'by" +author;
52 }
53 }

```

```

9      * @author RAJ
10     */
11     public class TestBook {
12     public static void main(String[] args){
13         Author a= new Author("Raj", "rajlama2007@gmail.com", 'm');
14         System.out.println(a);
15         Book b=new Book("The Lama",a,100,4);
16         b.setPrice(100);
17         b.setQty(4);
18         System.out.println(b);
19         System.out.println("Book name:" + b.getName());
20         System.out.println("Price:" + b.getPrice());
21         System.out.println("Quantity:" + b.getQty());
22         System.out.println("Author is" + b.getAuthor());
23         System.out.println("author's name is: " + b.getAuthor().getName());
24         System.out.println("author's email is: " + b.getAuthor().getEmail());
25         System.out.println("author's gender is: " + b.getAuthor().getGender());
26     }
27 }

```

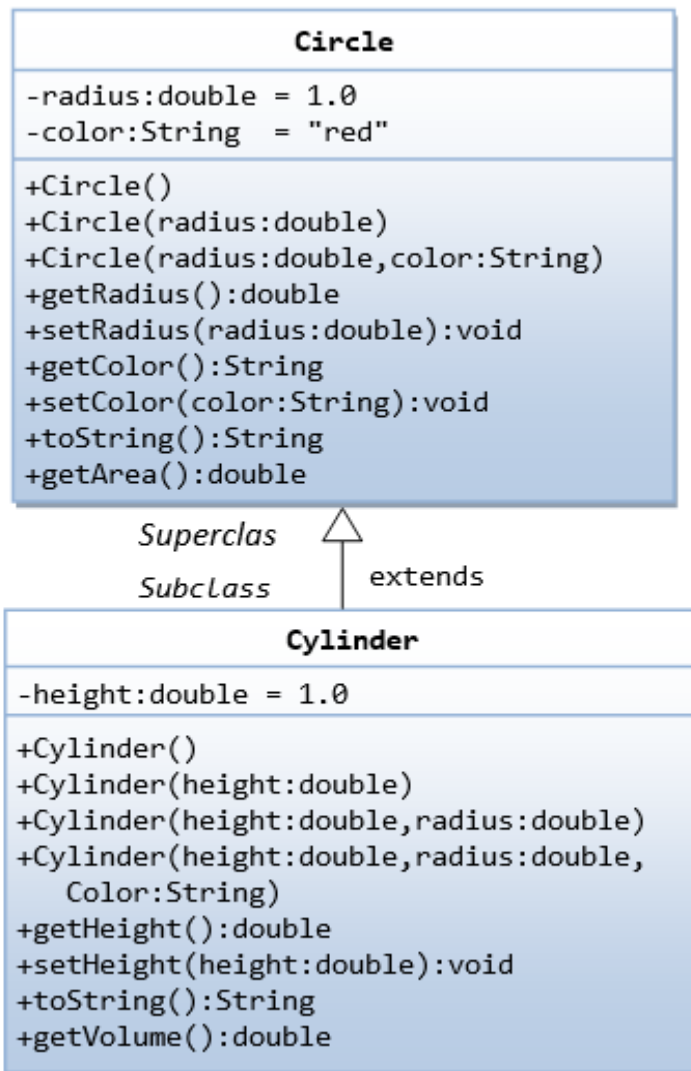
Output - First_Assignment (run) x

```

>> 'The Lama'byRaj(m) at rajlama2007@gmail.com
>> Book name:The Lama
>> Price:100.0
>> Quantity:4
>> Author isRaj(m) at rajlama2007@gmail.com
>> author's name is: Raj
>> author's email is: rajlama2007@gmail.com
>> author's gender is: m
BUILD SUCCESSFUL (total time: 0 seconds)

```

Q3. Using inheritance convert the following UML class diagram to java code and test the java code.



Solution:

```
6   package inherit;
7
8   /**...4 lines */
9   public class Circle {
13      private double radius;
14      private String color;
15
16      public Circle()
17      {
18          this.radius = 1.0;
19          this.color = "red";
20      }
21      public Circle(double radius)
22      {
23          this.radius = radius;
24          this.color = "red";
25      }
26      public Circle(double radius, String color)
27      {
28          this.radius = radius;
29          this.color = color;
30      }
```



```

31
32 // Getters and Setters
33 public double getRadius()
34 {
35     return this.radius;
36 }
37 public String getColor()
38 {
39     return this.color;
40 }
41 public void setRadius(double radius)
42 {
43     this.radius = radius;
44 }
45 public void setColor(String color)
46 {
47     this.color = color;
48 }
49
50 public String toString()
51 {
52     return "Circle[radius=" + radius + ",color=" + color + "];
53 }
54
55 public double getArea()
56 {
57     return radius * radius * Math.PI;
58 }
59 }
60

```

```

6   package inherit;
7
8   /**...4 lines */
12  public class Cylinder extends Circle {
13      private double height;
14
15      public Cylinder()
16      {
17          super();
18          this.height = 1.0;
19      }
20      public Cylinder(double height)
21      {
22          super();
23          this.height = height;
24      }
25      public Cylinder(double height, double radius)
26      {
27          super(radius);
28          this.height = height;
29      }
30
31      public Cylinder(double height, double radius, String color)
32      {
33          super(radius);
34          this.height = height;
35      }
36      public double getHeight()
37      {
38          return this.height;
39      }
40      public void setHeight(double height)
41      {
42          this.height = height;
43      }
44      public double getVolume()
45      {
46          return getArea()*height;
47      }
48      public String toString()
49      {
50          return "This is a Cylinder";
51      }

```

```

12 public class Test {
13     public static void main(String[] args) {
14         Cylinder c = new Cylinder();
15         System.out.println("Radius is " + c.getRadius()
16             + " Height is " + c.getHeight()
17             + " Color is " + c.getColor()
18             + " Base area is " + c.getArea()
19             + " Volume is " + c.getVolume());
20
21         Cylinder c2 = new Cylinder(5.0, 2.0);
22         System.out.println("Radius is " + c2.getRadius()
23             + " Height is " + c2.getHeight()
24             + " Color is " + c2.getColor()
25             + " Base area is " + c2.getArea()
26             + " Volume is " + c2.getVolume());
27     }
28 }

```

Find: Previous Next Select

Output - First_Assignment (run) x run.xml x

```

run:
Radius is 1.0 Height is 1.0 Color is red Base area is 3.141592653589793 Volume is 3.141592653589793
Radius is 2.0 Height is 5.0 Color is red Base area is 12.566370614359172 Volume is 62.83185307179586
BUILD SUCCESSFUL (total time: 0 seconds)

```

Q4. What is encapsulation and advantages of encapsulation? Explain with an example.

Encapsulation is a process of wrapping data and code together into a single unit. For example, a capsule which is mixed of several medicines.

The advantages of encapsulation:

- i. It helps programmer in data hiding: hide the inner classes and give access to specific user to the desired codes which improves security.
- ii. Encapsulate class is easy for testing.
- iii. It allows the user to a programmer to use the program again and again which increases reusability.
- iv. It allows to make the data as read-only or write-only as we require it to be.

Example:

```
12  public class Student {
13
14  private String name;
15  //getter method
16  public String getName() {
17      return name;
18  }
19  //setter method
20  public void setName(String name){
21      this.name=name;
22  }
23  public static void main(String[] args){
24      //creating instance of the encapsulated class
25      Student s=new Student();
26      //setting value
27      s.setName("raj");
28      System.out.println(s.getName());
29  }
30  }
```

Output - First_Assignment (run) x run.xml x

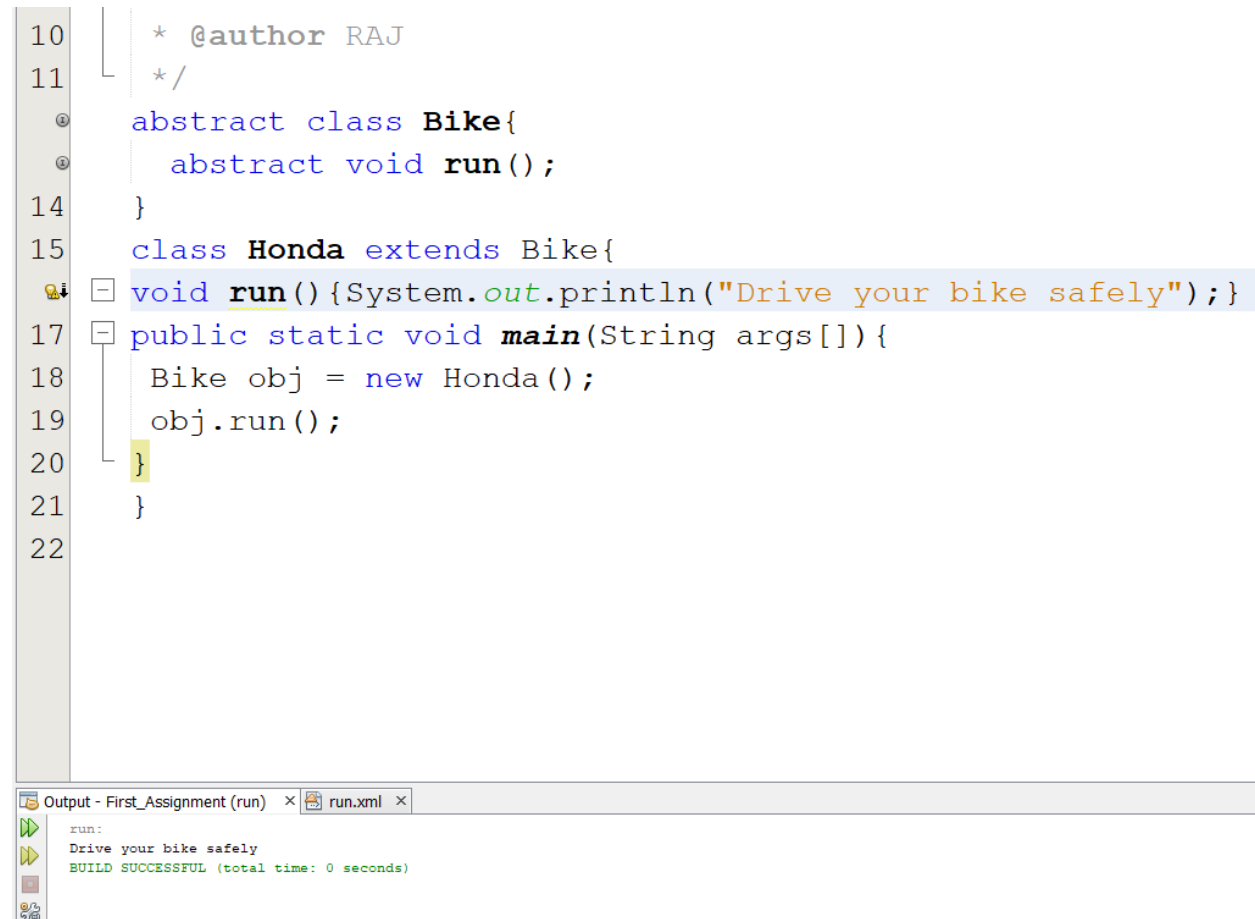
```
run:
raj
BUILD SUCCESSFUL (total time: 0 seconds)
```

Q5. What is abstract method in java? Explain with an example.

Abstract method is a method which does not have implementation and that is declared as abstract in an abstract class in java.

Example:

```
10  * @author RAJ
11  */
12  abstract class Bike{
13      abstract void run();
14  }
15  class Honda extends Bike{
16      void run(){System.out.println("Drive your bike safely");}
17  public static void main(String args[]){
18      Bike obj = new Honda();
19      obj.run();
20  }
21  }
22
```



Output - First_Assignment (run) × run.xml ×

run:
Drive your bike safely
BUILD SUCCESSFUL (total time: 0 seconds)

Q6. What is an interface in java? Explain with an example.

An interface is a blueprint of a class which contains collection of abstract methods that helps to achieve abstraction and multiple inheritance in java.

Example:

```
1  interface Animal {
2      public void animalSound();
3      public void sleep();
15 }
16 class Cow implements Animal {
17     public void animalSound() {
18         System.out.println("The cow says: baaaaa");
19     }
20     public void sleep() {
21         System.out.println("sleping");
22     }
23 }
24 class MainClass {
25     public static void main(String[] args) {
26         Cow myPig = new Cow();
27         myPig.animalSound();
28         myPig.sleep();
29     }
30 }
31 }
```

Output - First_Assignment (run) × run.xml ×

run:
The cow says: baaaaa
sleping
BUILD SUCCESSFUL (total time: 0 seconds)

Q7. Write a Java program to find the maximum and minimum value of an array.

```
6   package first_assignment;
7
8   import java.util.Arrays;
9
10  /**...4 lines */
14  public class Array {
15      static int max;
16      static int min;
17
18      public static void max_min(int my_array[]) {
19          max = my_array[0];
20          min = my_array[0];
21          int len = my_array.length;
22          for (int i = 1; i < len - 1; i = i + 2) {
23              if (i + 1 > len) {
24                  if (my_array[i] > max) max = my_array[i];
25                  if (my_array[i] < min) min = my_array[i];
26              }
27              if (my_array[i] > my_array[i + 1]) {
28                  if (my_array[i] > max) max = my_array[i];
29                  if (my_array[i + 1] < min) min = my_array[i + 1];
30              }

```

```

31         if (my_array[i] < my_array[i + 1]) {
32             if (my_array[i] < min) min = my_array[i];
33             if (my_array[i + 1] > max) max = my_array[i + 1];
34         }
35     }
36 }
37
38 public static void main(String[] args) {
39     int[] my_array = {25, 14, 56, 15, 36, 56, 77, 18, 29, 49};
40     max_min(my_array);
41     System.out.println(" Original Array: "+Arrays.toString(my_array));
42     System.out.println(" Maximum value for the above array = " + max);
43     System.out.println(" Minimum value for the above array = " + min);
44 }
45 }

```

Find:

Previous
 Next
 Select
 Run
 Debug
 Test

Output - First_Assignment (run) × run.xml ×

```

run:
Original Array: [25, 14, 56, 15, 36, 56, 77, 18, 29, 49]
Maximum value for the above array = 77
Minimum value for the above array = 14
BUILD SUCCESSFUL (total time: 0 seconds)
    
```


Q8. Write a Java program to reverse an array of integer values.

```
14 public class Arrayreverse {  
15     public static void main(String[] args){  
16         int[] my_array1 = {  
17             1,2,3,4,5,6,7};  
18         System.out.println("Original array : "+Arrays.toString(my_array1));  
19         for(int i = 0; i < my_array1.length / 2; i++)  
20         {  
21             int temp = my_array1[i];  
22             my_array1[i] = my_array1[my_array1.length - i - 1];  
23             my_array1[my_array1.length - i - 1] = temp;  
24         }  
25         System.out.println("Reverse array : "+Arrays.toString(my_array1));  
26     }  
27 }
```

Find: Previous Next Select

Output - First_Assignment (run) x run.xml x

run:
Original array : [1, 2, 3, 4, 5, 6, 7]
Reverse array : [7, 6, 5, 4, 3, 2, 1]
BUILD SUCCESSFUL (total time: 0 seconds)

Q9. Write a Java program to find the second largest element in an array.

```
12 public class Secondlargestarray {
13     public static void main(String args[]){
14         int temp, size;
15         int array[] = {10, 20, 25, 63, 96, 57};
16         size = array.length;
17
18         for(int i = 0; i<size; i++){
19             for(int j = i+1; j<size; j++){
20
21                 if(array[i]>array[j]){
22                     temp = array[i];
23                     array[i] = array[j];
24                     array[j] = temp;
25                 }
26             }
27         }
28         System.out.println("Second largest number is:: "+array[size-2]);
29     }
30 }
```

Find: Previous Next Select

Output - First_Assignment (run) x run.xml x

run:
Second largest number is:: 63
BUILD SUCCESSFUL (total time: 0 seconds)

Q10. What is an ArrayList in java? Explain with an example.

An ArrayList is a resizable array, also called a dynamic array which can be found in the java.util package that grows its size to accommodate new elements and shrinks the size when the elements are removed.

Example:

```
12 import java.util.ArrayList;
13 import java.util.List;
14
15 public class Arraylist {
16     public static void main(String[] args) {
17         List<String> students = new ArrayList<>();
18
19         students.add("Raj");
20         students.add("Bikal");
21         students.add("Bishes");
22         students.add("Ram");
23
24         System.out.println(students);
25         students.add(2, "Ramesh");
26         System.out.println(students);
27     }
28 }
29
30
```

Output - First_Assignment (run) × run.xml ×

run:
[Raj, Bikal, Bishes, Ram]
[Raj, Bikal, Ramesh, Bishes, Ram]
BUILD SUCCESSFUL (total time: 0 seconds)