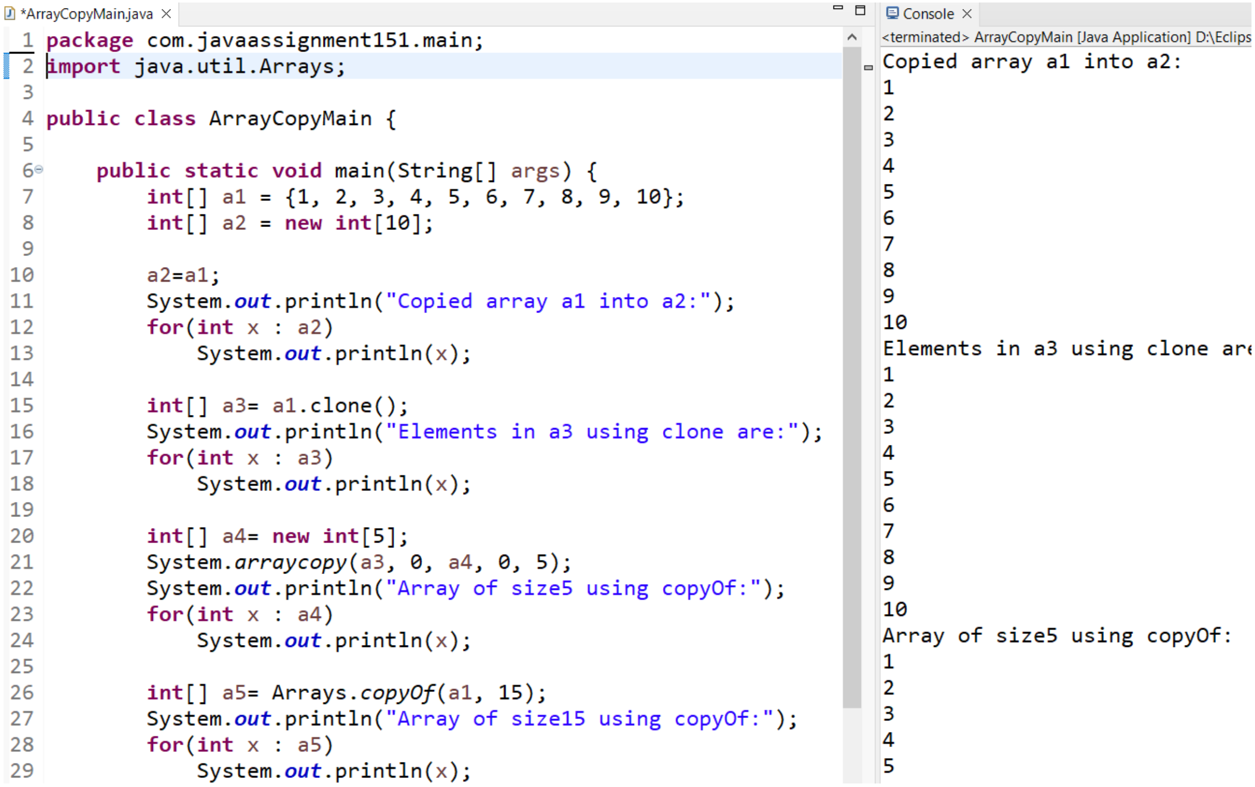
PG DAC–March 2023
C-DAC THIRUVANANTHAPURAM
OOPs WITH JAVA-- LAB 15

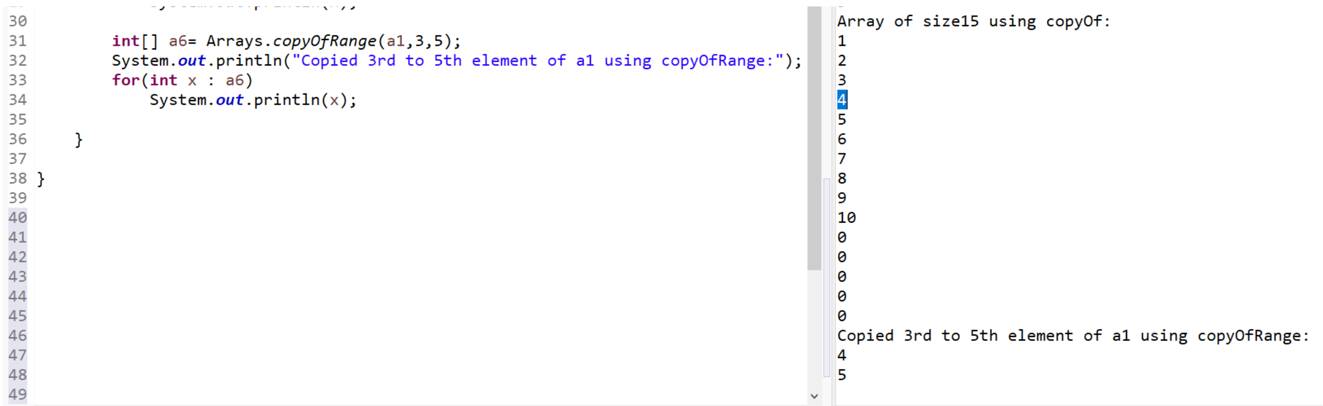
Q1. Create an array a1 of 10 integers. Copy a1 array to another array using

- **Assigning an array to another array**
- **Clone()**
- **using copyOf create a output array of size 5**
- **using copyOf create a output array of size 15**
- **using copyOfRange copy from 3rd to 5th element of a1**



```java
package com.javaassignment151.main;
import java.util.Arrays;

public class ArrayCopyMain {

    public static void main(String[] args) {
        int[] a1 = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
        int[] a2 = new int[10];

        a2=a1;
        System.out.println("Copied array a1 into a2:");
        for(int x : a2)
            System.out.println(x);

        int[] a3= a1.clone();
        System.out.println("Elements in a3 using clone are:");
        for(int x : a3)
            System.out.println(x);

        int[] a4= new int[5];
        System.arraycopy(a3, 0, a4, 0, 5);
        System.out.println("Array of size5 using copyOf:");
        for(int x : a4)
            System.out.println(x);

        int[] a5= Arrays.copyOf(a1, 15);
        System.out.println("Array of size15 using copyOf:");
        for(int x : a5)
            System.out.println(x);
```

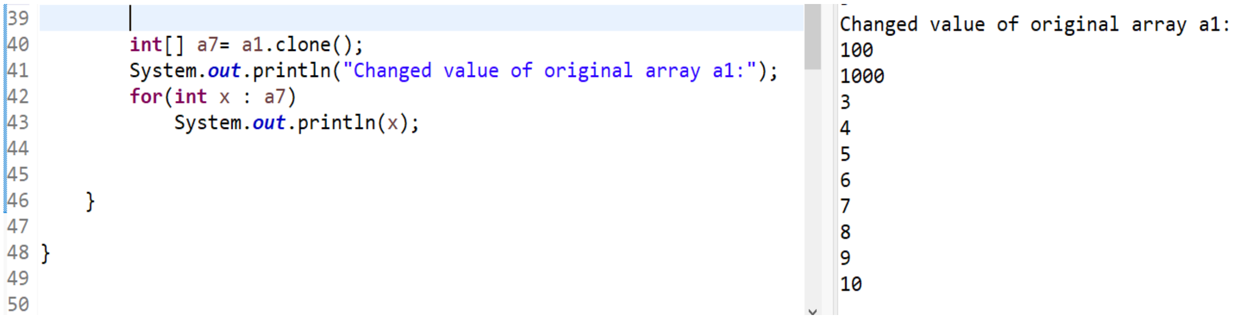
```
Copied array a1 into a2:
1
2
3
4
5
6
7
8
9
10
Elements in a3 using clone ar
1
2
3
4
5
6
7
8
9
10
Array of size5 using copyOf:
1
2
3
4
5
```



```java

        int[] a6= Arrays.copyOfRange(a1,3,5);
        System.out.println("Copied 3rd to 5th element of a1 using copyOfRange:");
        for(int x : a6)
            System.out.println(x);

    }

}
```

```
Array of size15 using copyOf:
1
2
3
4
5
6
7
8
9
10
0
0
0
0
0
Copied 3rd to 5th element of a1 using copyOfRange:
4
5
```

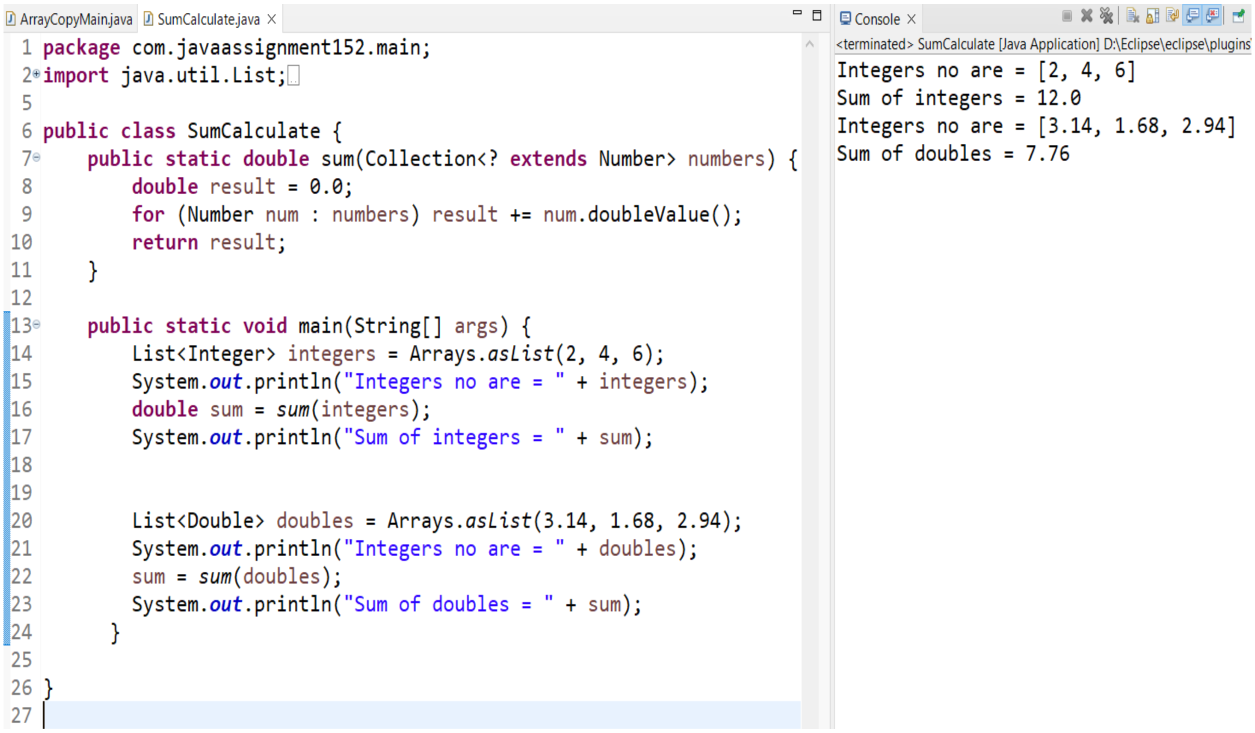
Change the value of original array and display both the arrays for each question



```java

        int[] a7= a1.clone();
        System.out.println("Changed value of original array a1:");
        for(int x : a7)
            System.out.println(x);


    }

}

```

```
Changed value of original array a1:
100
1000
3
4
5
6
7
8
9
10
```

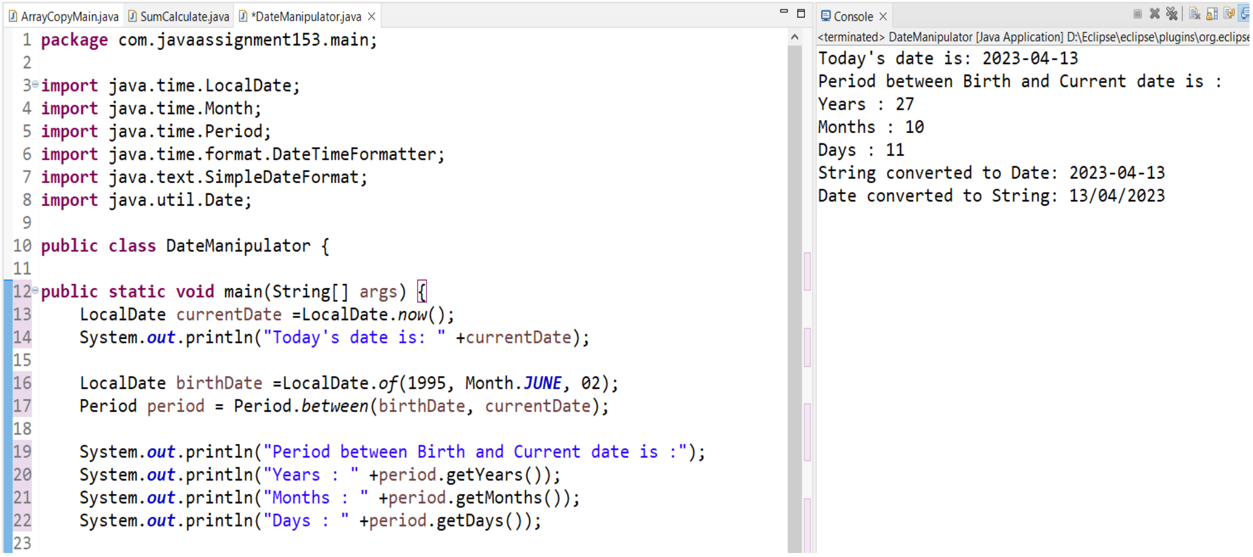
Q2. Write a class with a method that calculates sum of numbers in a collection implemented using wildcards Generics method. Invoke this class from main to find the sum of integers & sum of floating point values.


ArrayCopyMain.java
SumCalculate.java
Console
<terminated> SumCalculate [Java Application] D:\Eclipse\eclipse\plugins

```java
package com.javaassignment152.main;
import java.util.List;

public class SumCalculate {
    public static double sum(Collection<? extends Number> numbers) {
        double result = 0.0;
        for (Number num : numbers) result += num.doubleValue();
        return result;
    }

    public static void main(String[] args) {
        List<Integer> integers = Arrays.asList(2, 4, 6);
        System.out.println("Integers no are = " + integers);
        double sum = sum(integers);
        System.out.println("Sum of integers = " + sum);


        List<Double> doubles = Arrays.asList(3.14, 1.68, 2.94);
        System.out.println("Integers no are = " + doubles);
        sum = sum(doubles);
        System.out.println("Sum of doubles = " + sum);
      }

}

```

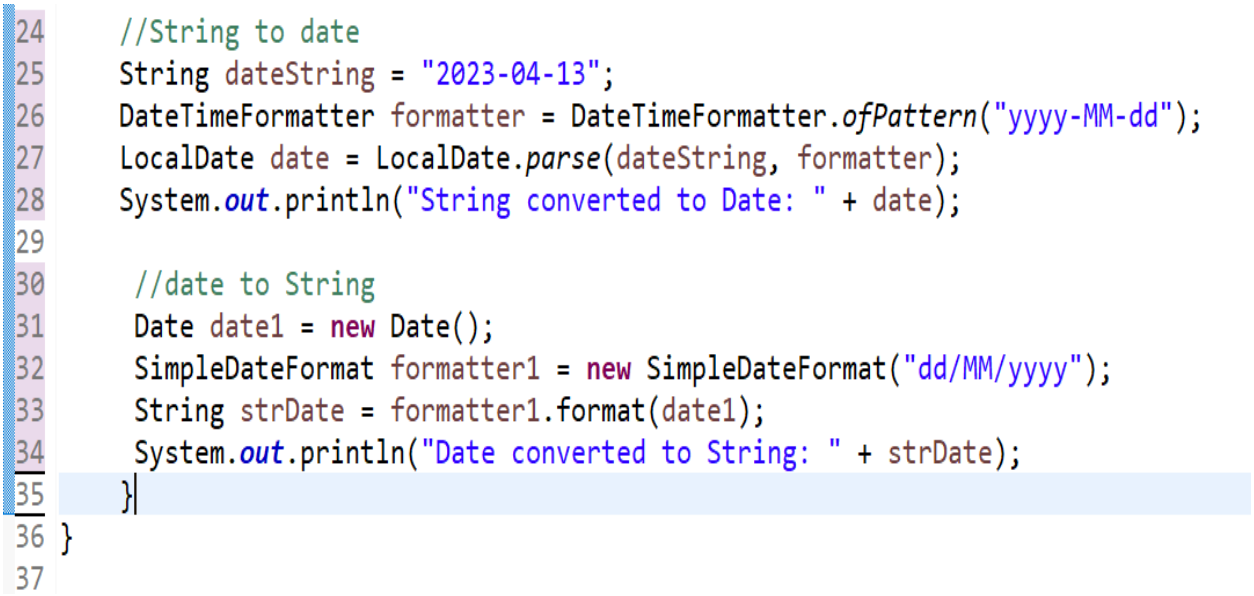
```
Integers no are = [2, 4, 6]
Sum of integers = 12.0
Integers no are = [3.14, 1.68, 2.94]
Sum of doubles = 7.76
```

Q3. Create Date Manipulator class to convert String to date, date to String and to find out number of days between two dates.


ArrayCopyMain.java
SumCalculate.java
*DateManipulator.java
Console
<terminated> DateManipulator [Java Application] D:\Eclipse\eclipse\plugins\org.eclipse

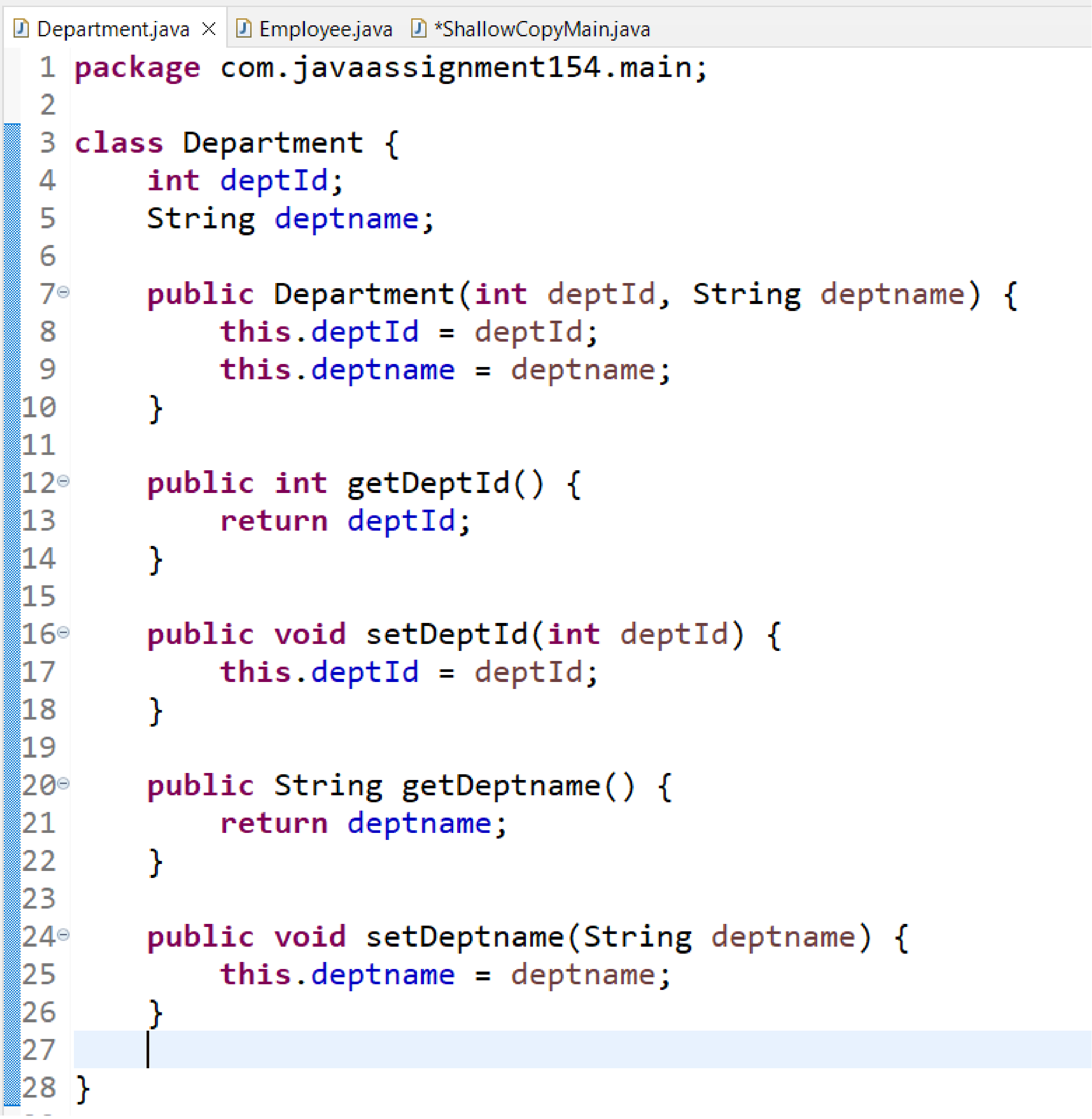
```java
package com.javaassignment153.main;

import java.time.LocalDate;
import java.time.Month;
import java.time.Period;
import java.time.format.DateTimeFormatter;
import java.text.SimpleDateFormat;
import java.util.Date;

public class DateManipulator {

public static void main(String[] args) {
    LocalDate currentDate =LocalDate.now();
    System.out.println("Today's date is: " +currentDate);

    LocalDate birthDate =LocalDate.of(1995, Month.JUNE, 02);
    Period period = Period.between(birthDate, currentDate);

    System.out.println("Period between Birth and Current date is :");
    System.out.println("Years : " +period.getYears());
    System.out.println("Months : " +period.getMonths());
    System.out.println("Days : " +period.getDays());

```

```
Today's date is: 2023-04-13
Period between Birth and Current date is :
Years : 27
Months : 10
Days : 11
String converted to Date: 2023-04-13
Date converted to String: 13/04/2023
```

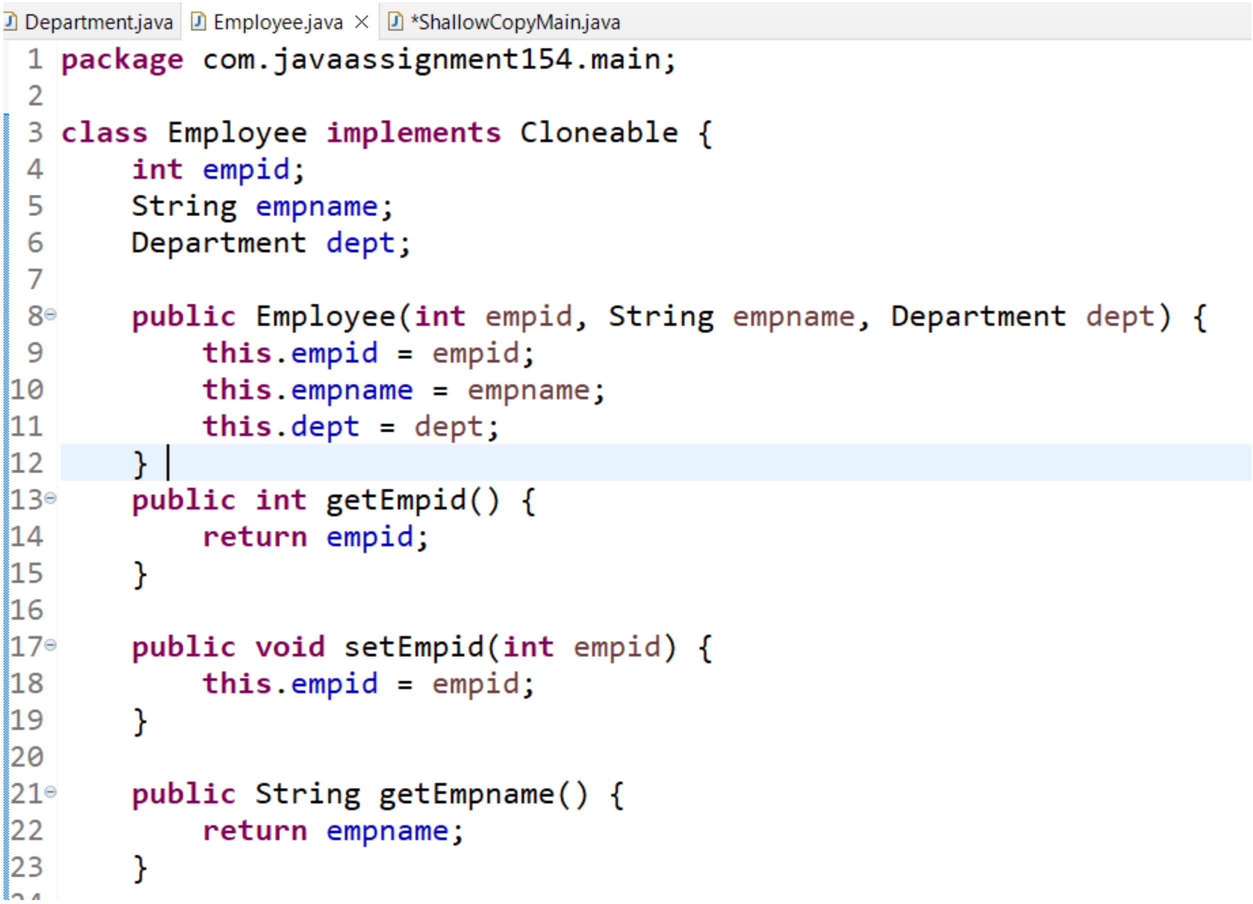


```java
    //String to date
    String dateString = "2023-04-13";
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
    LocalDate date = LocalDate.parse(dateString, formatter);
    System.out.println("String converted to Date: " + date);

     //date to String
     Date date1 = new Date();
     SimpleDateFormat formatter1 = new SimpleDateFormat("dd/MM/yyyy");
     String strDate = formatter1.format(date1);
     System.out.println("Date converted to String: " + strDate);
    }
}

```

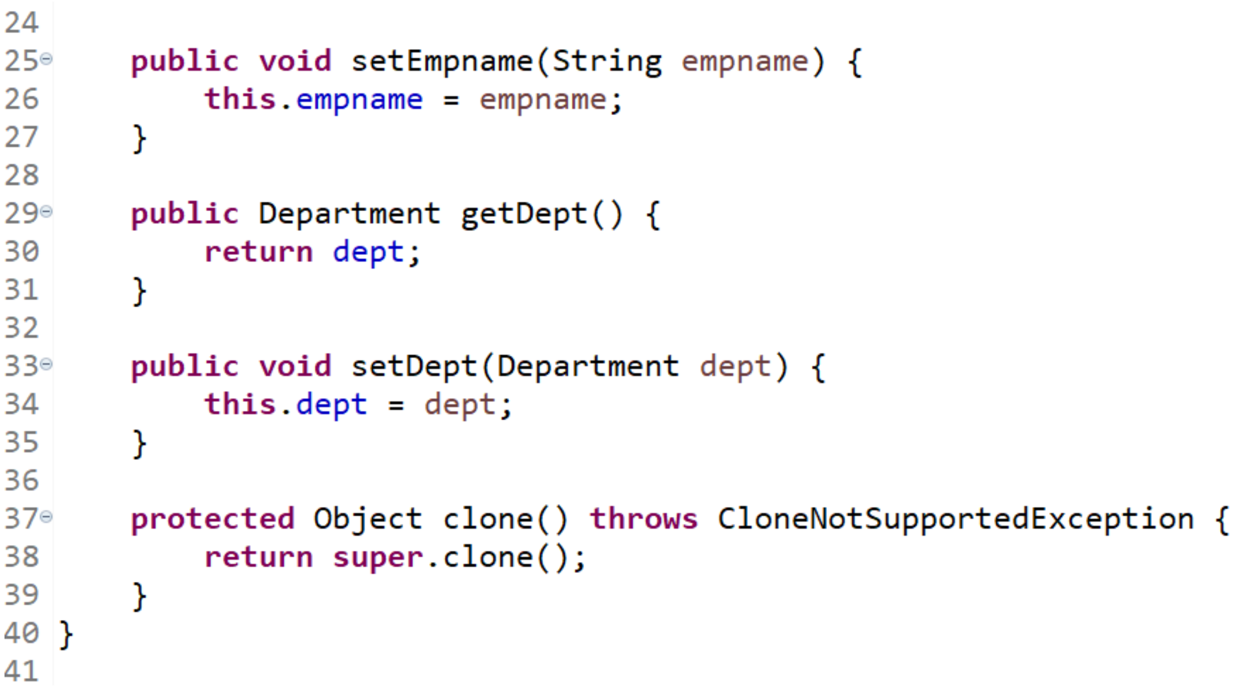
Q4. Create a class named Department with data members deptid and deptname. Create another class Employee with data members empid, empname and reference of Department. Create an object of Employee and make a shallow copy of it.



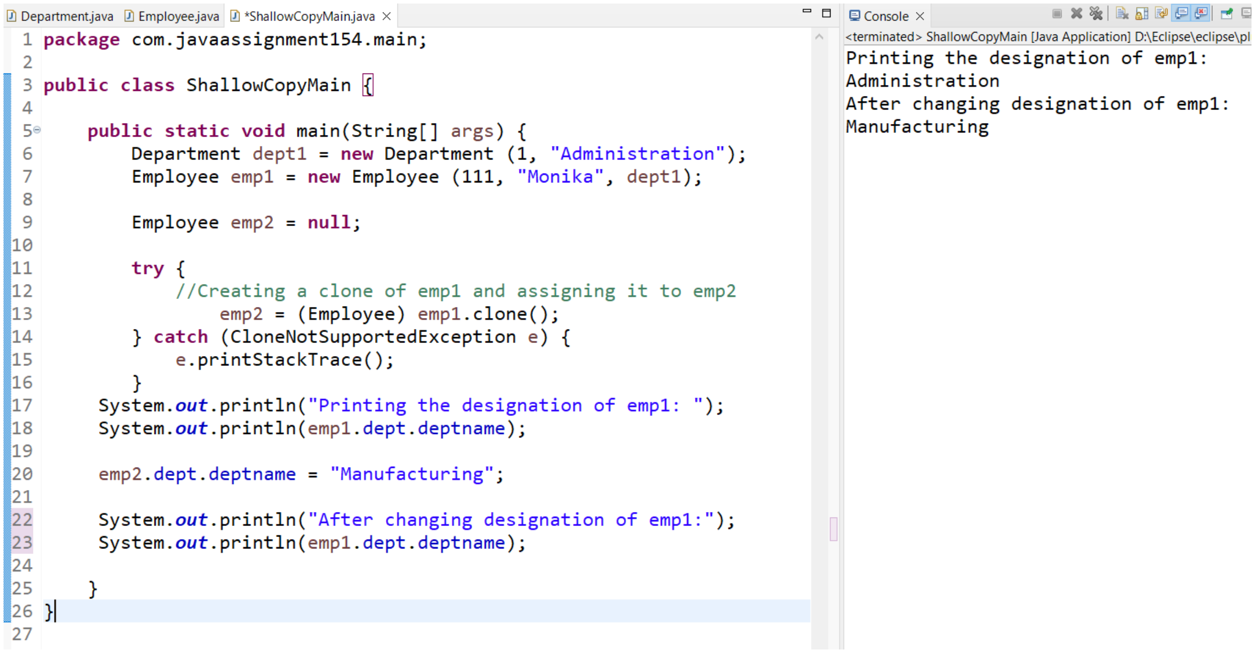
```java
package com.javaassignment154.main;

class Department {
    int deptId;
    String deptname;

    public Department(int deptId, String deptname) {
        this.deptId = deptId;
        this.deptname = deptname;
    }

    public int getDeptId() {
        return deptId;
    }

    public void setDeptId(int deptId) {
        this.deptId = deptId;
    }

    public String getDeptname() {
        return deptname;
    }

    public void setDeptname(String deptname) {
        this.deptname = deptname;
    }

}
```



```java
package com.javaassignment154.main;

class Employee implements Cloneable {
    int empid;
    String empname;
    Department dept;

    public Employee(int empid, String empname, Department dept) {
        this.empid = empid;
        this.empname = empname;
        this.dept = dept;
    }
    public int getEmpid() {
        return empid;
    }

    public void setEmpid(int empid) {
        this.empid = empid;
    }

    public String getEmpname() {
        return empname;
    }
```



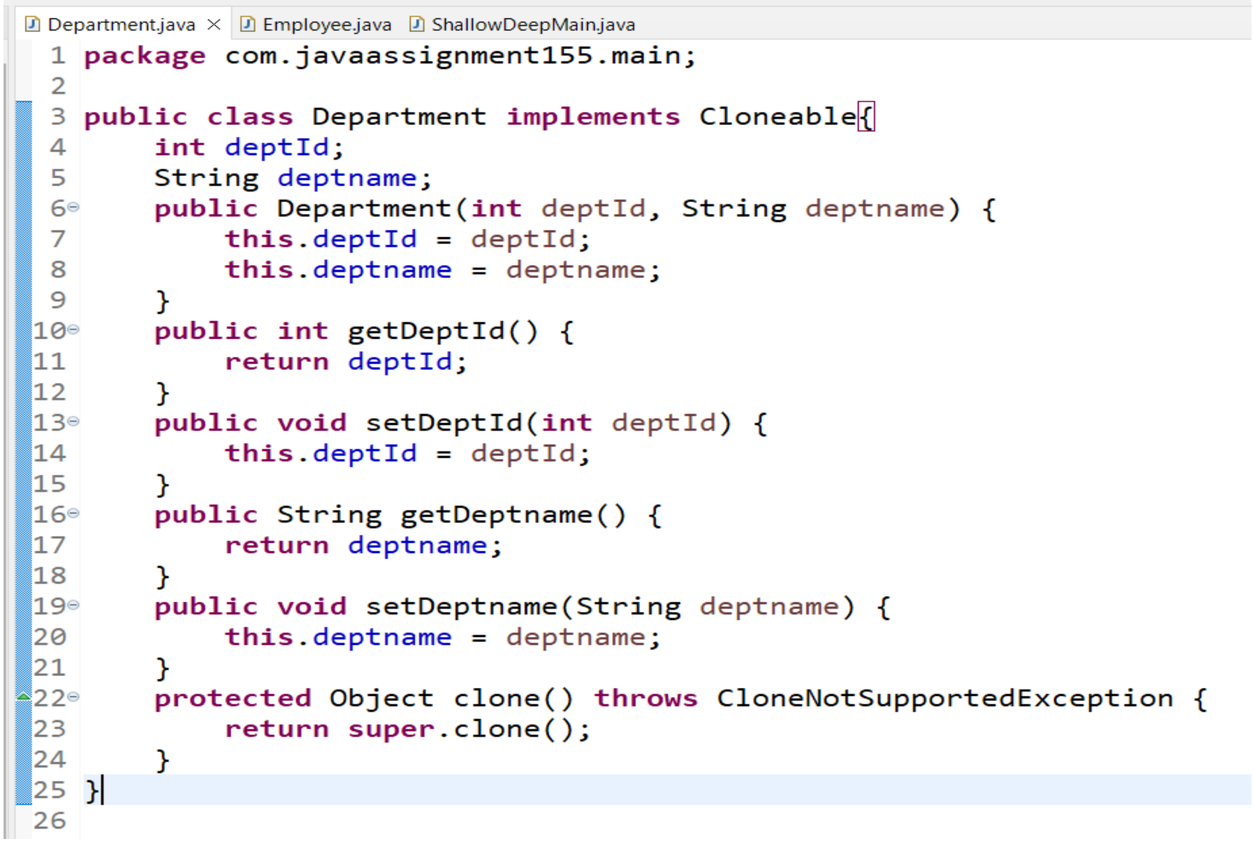
```java

    public void setEmpname(String empname) {
        this.empname = empname;
    }

    public Department getDept() {
        return dept;
    }

    public void setDept(Department dept) {
        this.dept = dept;
    }

    protected Object clone() throws CloneNotSupportedException {
        return super.clone();
    }
}

```


Department.java
Employee.java
*ShallowCopyMain.java
Console
<terminated> ShallowCopyMain [Java Application] D:\Eclipse\eclipse\pl

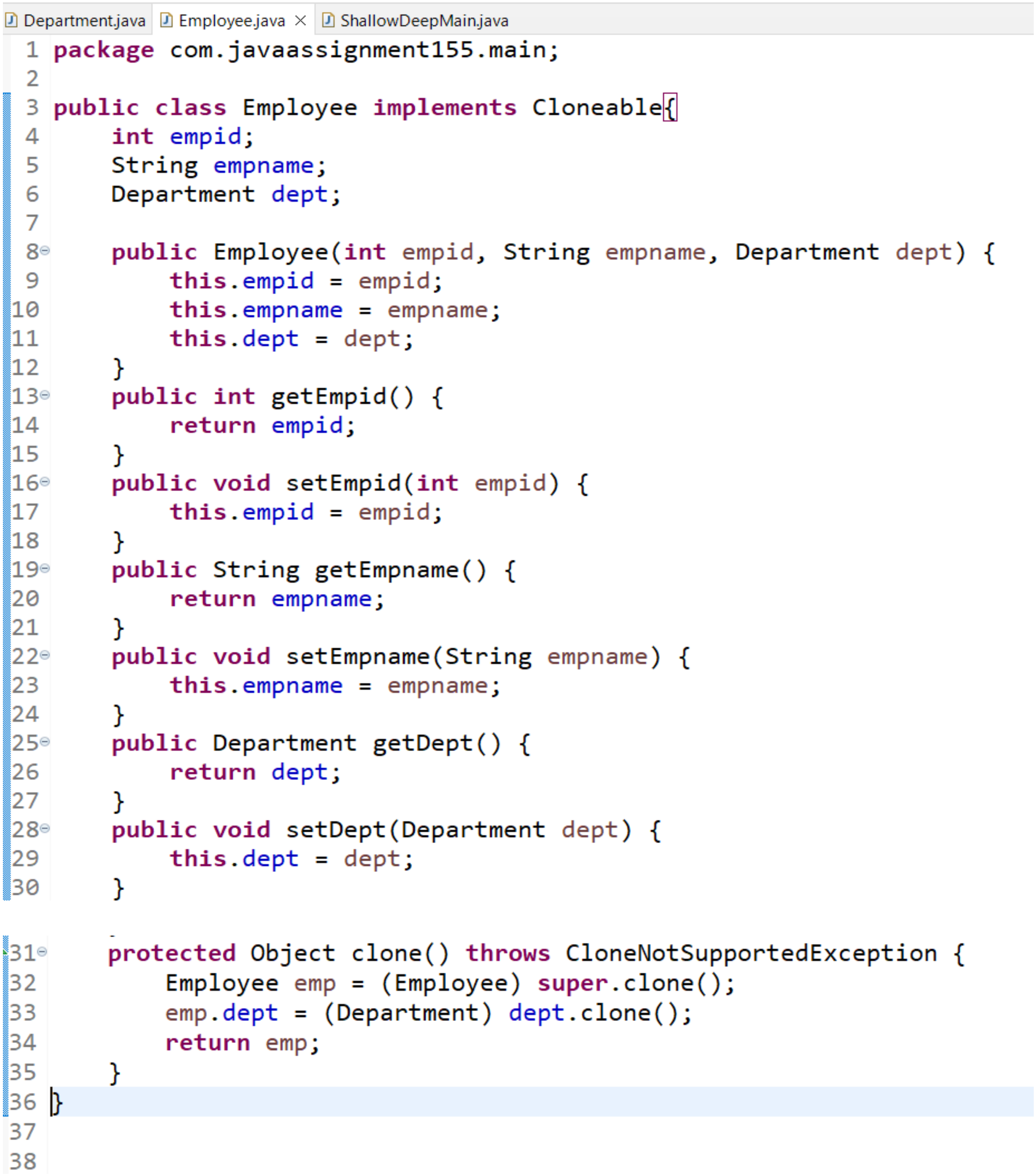
```java
package com.javaassignment154.main;

public class ShallowCopyMain {

    public static void main(String[] args) {
        Department dept1 = new Department (1, "Administration");
        Employee emp1 = new Employee (111, "Monika", dept1);

        Employee emp2 = null;

        try {
            //Creating a clone of emp1 and assigning it to emp2
                emp2 = (Employee) emp1.clone();
        } catch (CloneNotSupportedException e) {
            e.printStackTrace();
        }
     System.out.println("Printing the designation of emp1: ");
     System.out.println(emp1.dept.deptname);

     emp2.dept.deptname = "Manufacturing";

     System.out.println("After changing designation of emp1:");
     System.out.println(emp1.dept.deptname);

    }
}

```

```
Printing the designation of emp1:
Administration
After changing designation of emp1:
Manufacturing
```

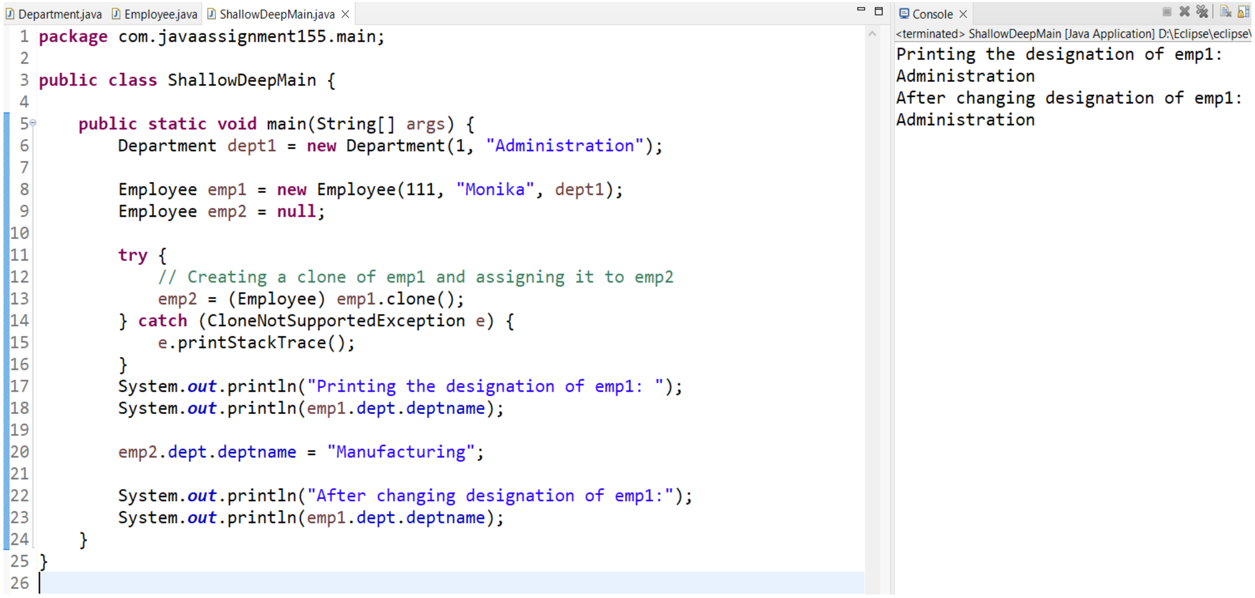
Q5. Create a class named Department with data members , deptid and deptname. Create another class Employee with data members empid, empname and reference of Department. Create an object of Employee and make a deep copy of it.



```java
package com.javaassignment155.main;

public class Department implements Cloneable{
    int deptId;
    String deptname;
    public Department(int deptId, String deptname) {
        this.deptId = deptId;
        this.deptname = deptname;
    }
    public int getDeptId() {
        return deptId;
    }
    public void setDeptId(int deptId) {
        this.deptId = deptId;
    }
    public String getDeptname() {
        return deptname;
    }
    public void setDeptname(String deptname) {
        this.deptname = deptname;
    }
    protected Object clone() throws CloneNotSupportedException {
        return super.clone();
    }
}

```



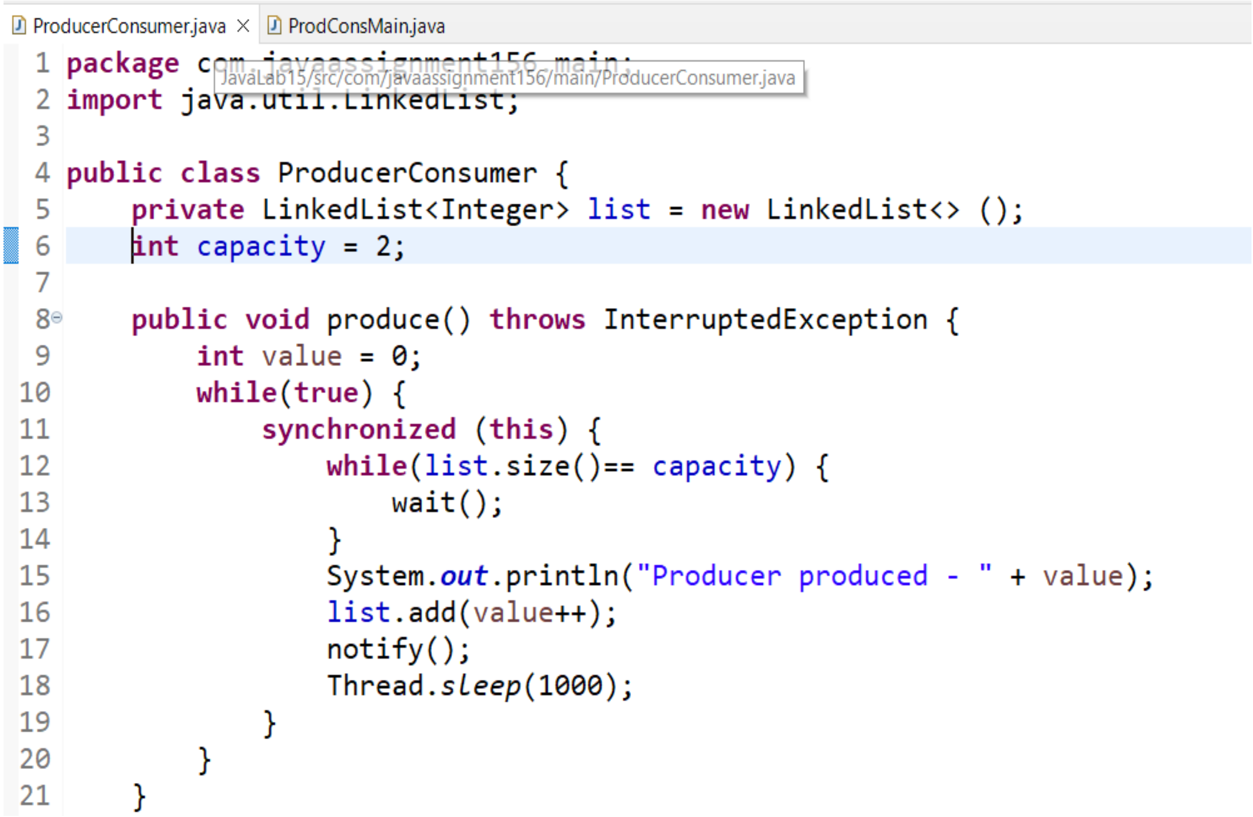
```java
package com.javaassignment155.main;

public class Employee implements Cloneable{
    int empid;
    String empname;
    Department dept;

    public Employee(int empid, String empname, Department dept) {
        this.empid = empid;
        this.empname = empname;
        this.dept = dept;
    }
    public int getEmpid() {
        return empid;
    }
    public void setEmpid(int empid) {
        this.empid = empid;
    }
    public String getEmpname() {
        return empname;
    }
    public void setEmpname(String empname) {
        this.empname = empname;
    }
    public Department getDept() {
        return dept;
    }
    public void setDept(Department dept) {
        this.dept = dept;
    }

    protected Object clone() throws CloneNotSupportedException {
        Employee emp = (Employee) super.clone();
        emp.dept = (Department) dept.clone();
        return emp;
    }
}


```



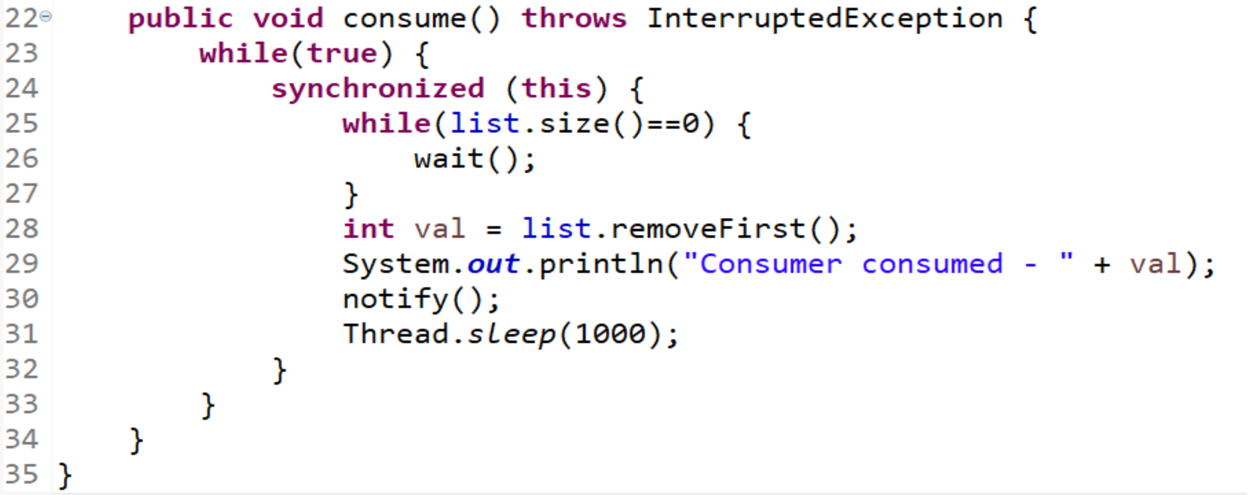
```java
package com.javaassignment155.main;

public class ShallowDeepMain {

    public static void main(String[] args) {
        Department dept1 = new Department(1, "Administration");

        Employee emp1 = new Employee(111, "Monika", dept1);
        Employee emp2 = null;

        try {
            // Creating a clone of emp1 and assigning it to emp2
            emp2 = (Employee) emp1.clone();
        } catch (CloneNotSupportedException e) {
            e.printStackTrace();
        }
        System.out.println("Printing the designation of emp1: ");
        System.out.println(emp1.dept.deptname);

        emp2.dept.deptname = "Manufacturing";

        System.out.println("After changing designation of emp1:");
        System.out.println(emp1.dept.deptname);
    }
}
```

```
<terminated> ShallowDeepMain [Java Application] D:\Eclipse\eclipse\
Printing the designation of emp1:
Administration
After changing designation of emp1:
Administration
```

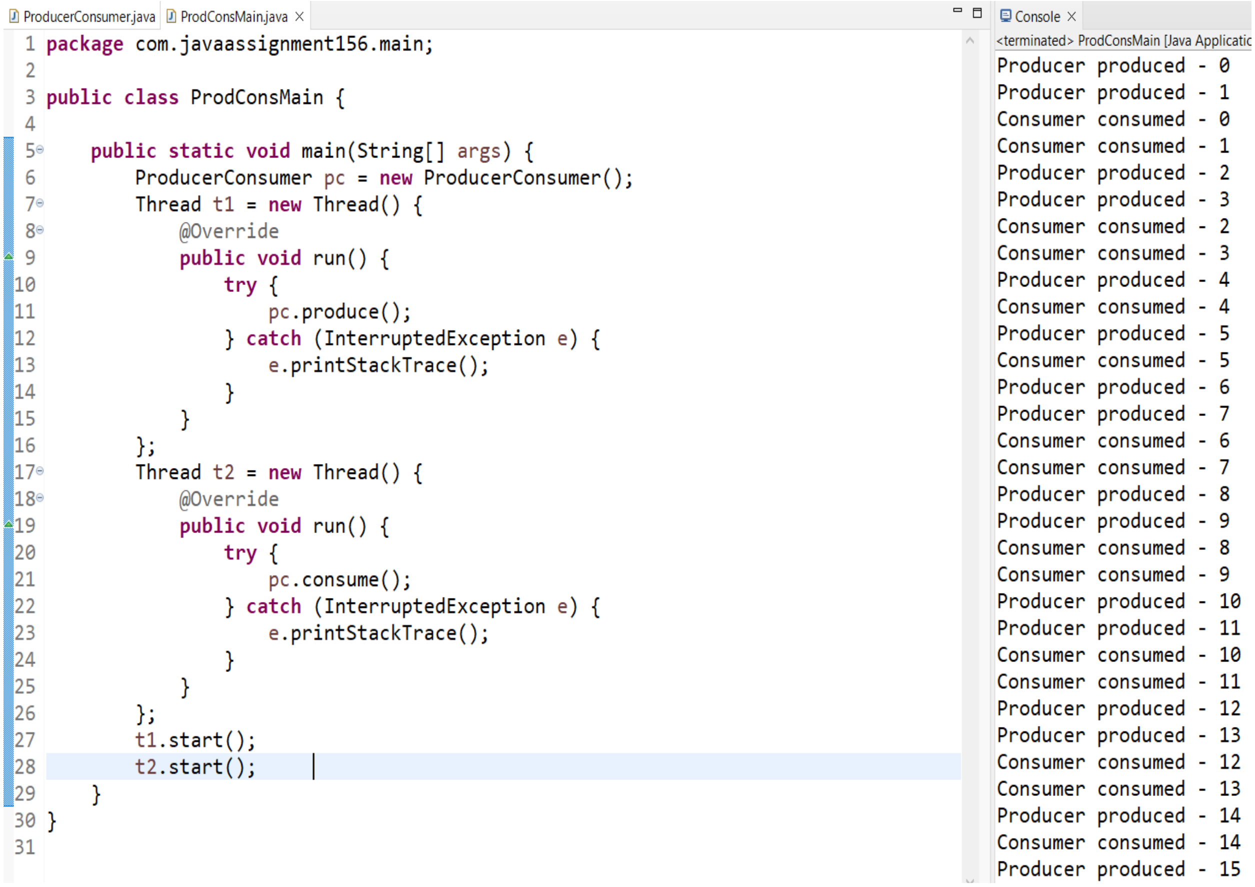
Q6. Try Producer Consumer Problem with an example


ProducerConsumer.java
ProdConsMain.java
JavaLab15/src/com/javaassignment156/main/ProducerConsumer.java

```java
package com.javaassignment156.main;
import java.util.LinkedList;

public class ProducerConsumer {
    private LinkedList<Integer> list = new LinkedList<> ();
    int capacity = 2;

    public void produce() throws InterruptedException {
        int value = 0;
        while(true) {
            synchronized (this) {
                while(list.size()== capacity) {
                    wait();
                }
                System.out.println("Producer produced - " + value);
                list.add(value++);
                notify();
                Thread.sleep(1000);
            }
        }
    }
```



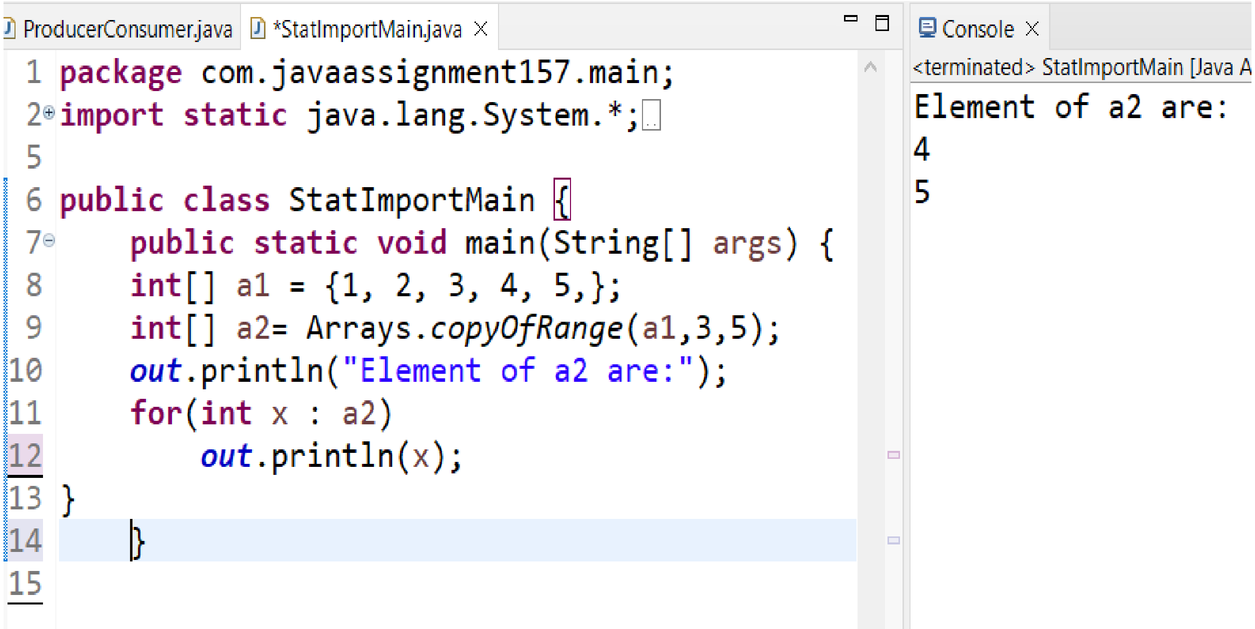
```java
    public void consume() throws InterruptedException {
        while(true) {
            synchronized (this) {
                while(list.size()==0) {
                    wait();
                }
                int val = list.removeFirst();
                System.out.println("Consumer consumed - " + val);
                notify();
                Thread.sleep(1000);
            }
        }
    }
}
```



```java
package com.javaassignment156.main;

public class ProdConsMain {

    public static void main(String[] args) {
        ProducerConsumer pc = new ProducerConsumer();
        Thread t1 = new Thread() {
            @Override
            public void run() {
                try {
                    pc.produce();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        };
        Thread t2 = new Thread() {
            @Override
            public void run() {
                try {
                    pc.consume();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
        };
        t1.start();
        t2.start();
    }
}

```

```
Producer produced - 0
Producer produced - 1
Consumer consumed - 0
Consumer consumed - 1
Producer produced - 2
Producer produced - 3
Consumer consumed - 2
Consumer consumed - 3
Producer produced - 4
Consumer consumed - 4
Producer produced - 5
Consumer consumed - 5
Producer produced - 6
Producer produced - 7
Consumer consumed - 6
Consumer consumed - 7
Producer produced - 8
Producer produced - 9
Consumer consumed - 8
Consumer consumed - 9
Producer produced - 10
Producer produced - 11
Consumer consumed - 10
Consumer consumed - 11
Producer produced - 12
Producer produced - 13
Consumer consumed - 12
Consumer consumed - 13
Producer produced - 14
Consumer consumed - 14
Producer produced - 15
```

Q7. Try out static import with a suitable example.



```java
package com.javaassignment157.main;
import static java.lang.System.*;

public class StatImportMain {
    public static void main(String[] args) {
    int[] a1 = {1, 2, 3, 4, 5,};
    int[] a2= Arrays.copyOfRange(a1,3,5);
    out.println("Element of a2 are:");
    for(int x : a2)
        out.println(x);
}
    }

```

```
<terminated> StatImportMain [Java A
Element of a2 are:
4
5
```