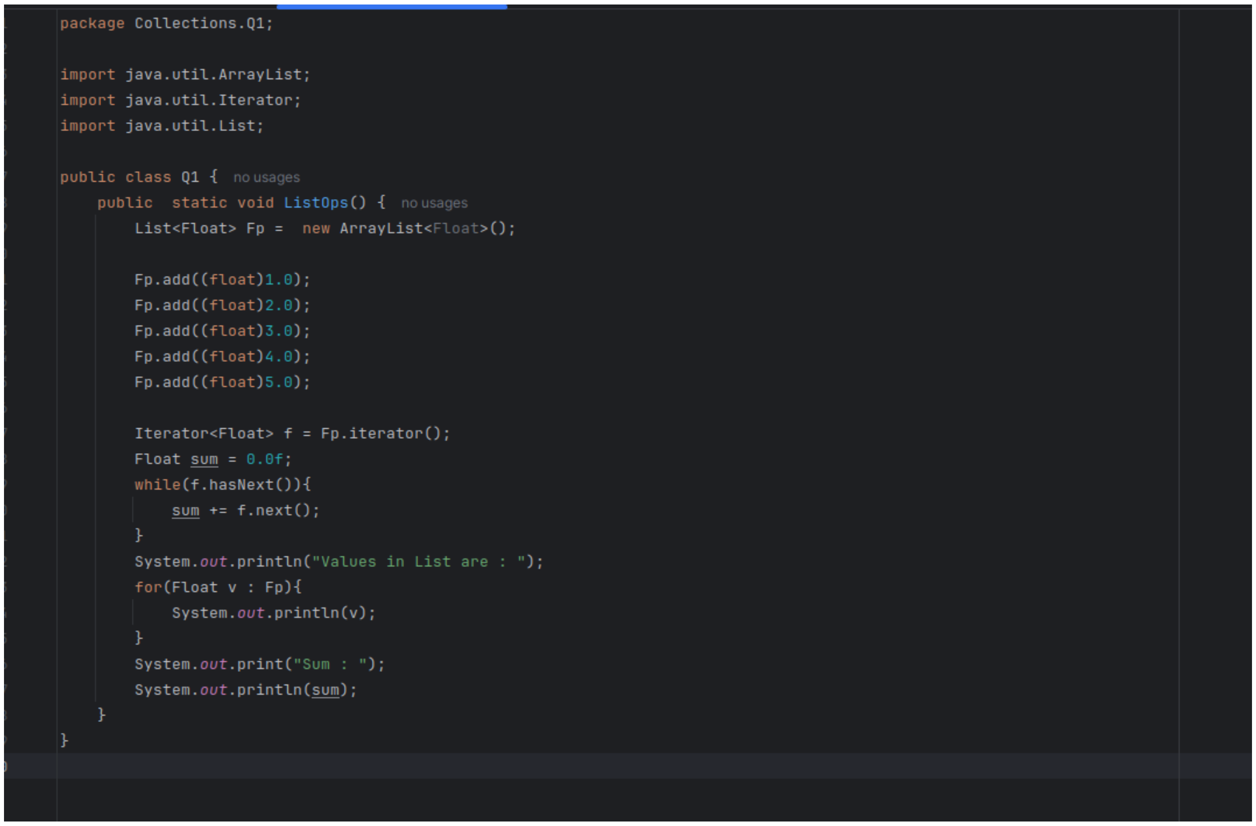
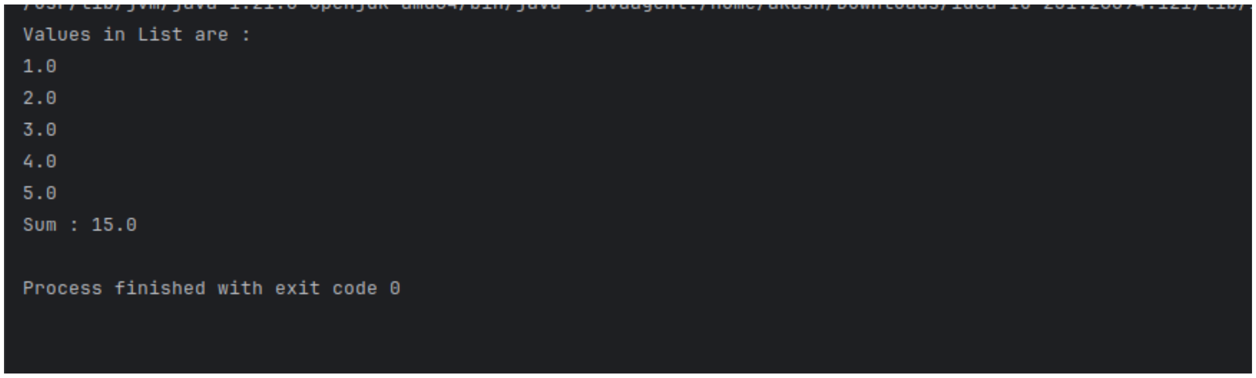
Assignment - Collections

Q1)Write Java code to define List. Insert 5 floating point numbers in List, and using an iterator, find the sum of the numbers in List.

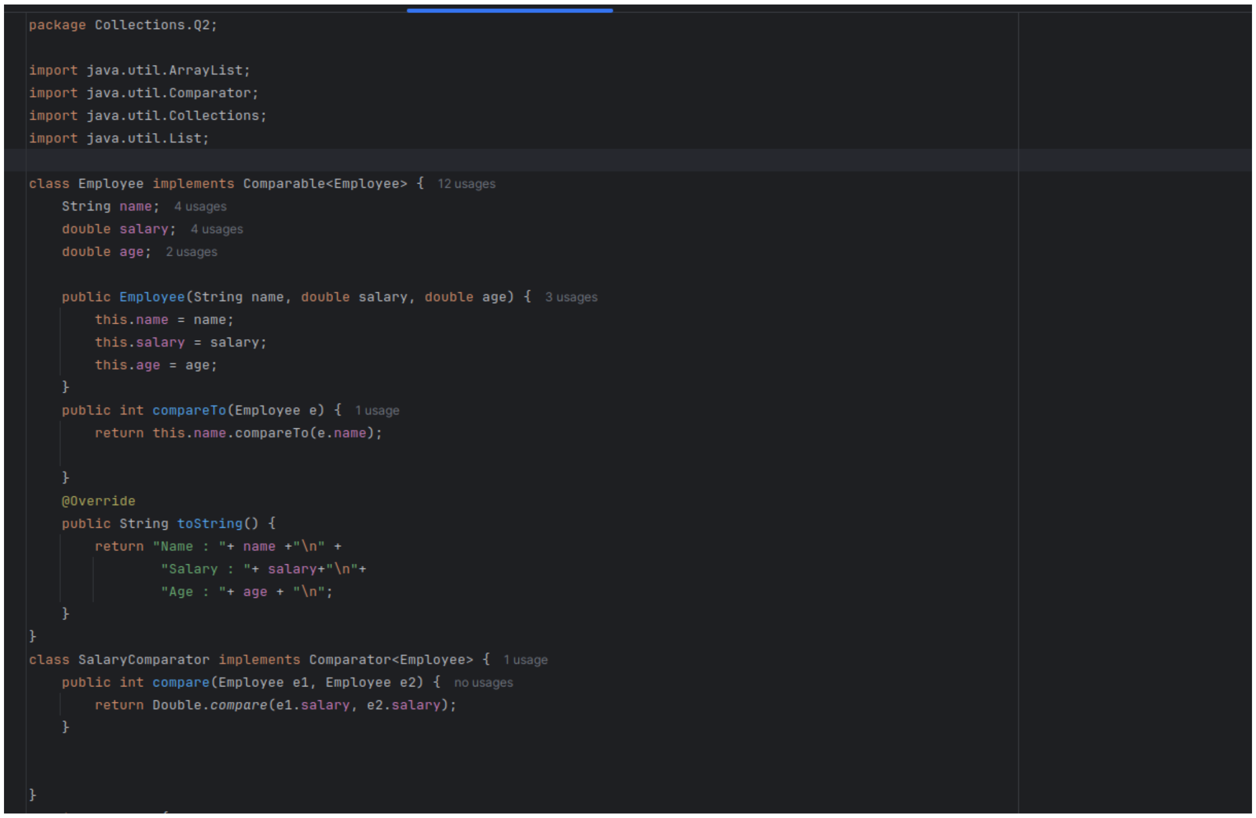


```java
package Collections.Q1;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

public class Q1 {
    public  static void ListOps() {
        List<Float> Fp =  new ArrayList<Float>();

        Fp.add((float)1.0);
        Fp.add((float)2.0);
        Fp.add((float)3.0);
        Fp.add((float)4.0);
        Fp.add((float)5.0);

        Iterator<Float> f = Fp.iterator();
        Float sum = 0.0f;
        while(f.hasNext()){
            sum += f.next();
        }
        System.out.println("Values in List are : ");
        for(Float v : Fp){
            System.out.println(v);
        }
        System.out.print("Sum : ");
        System.out.println(sum);
    }
}
```

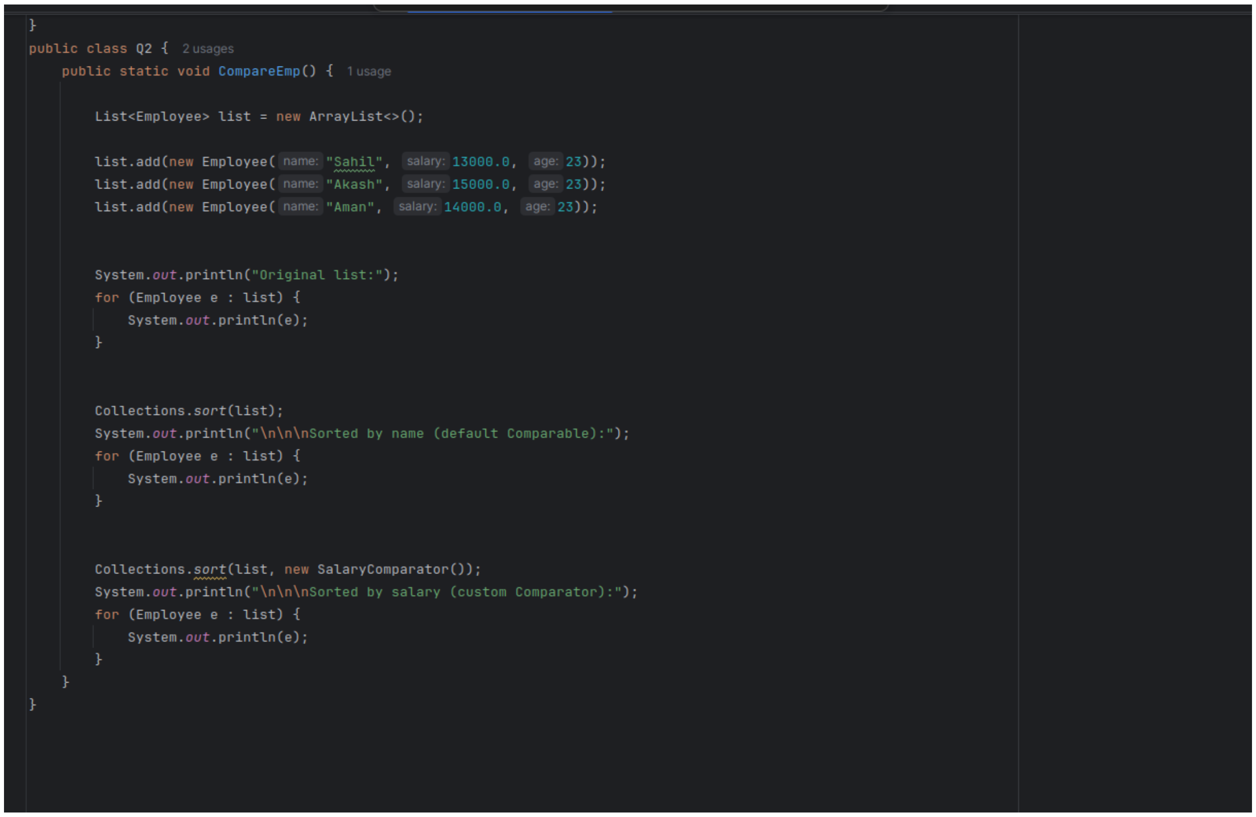


```
Values in List are :
1.0
2.0
3.0
4.0
5.0
Sum : 15.0

Process finished with exit code 0
```

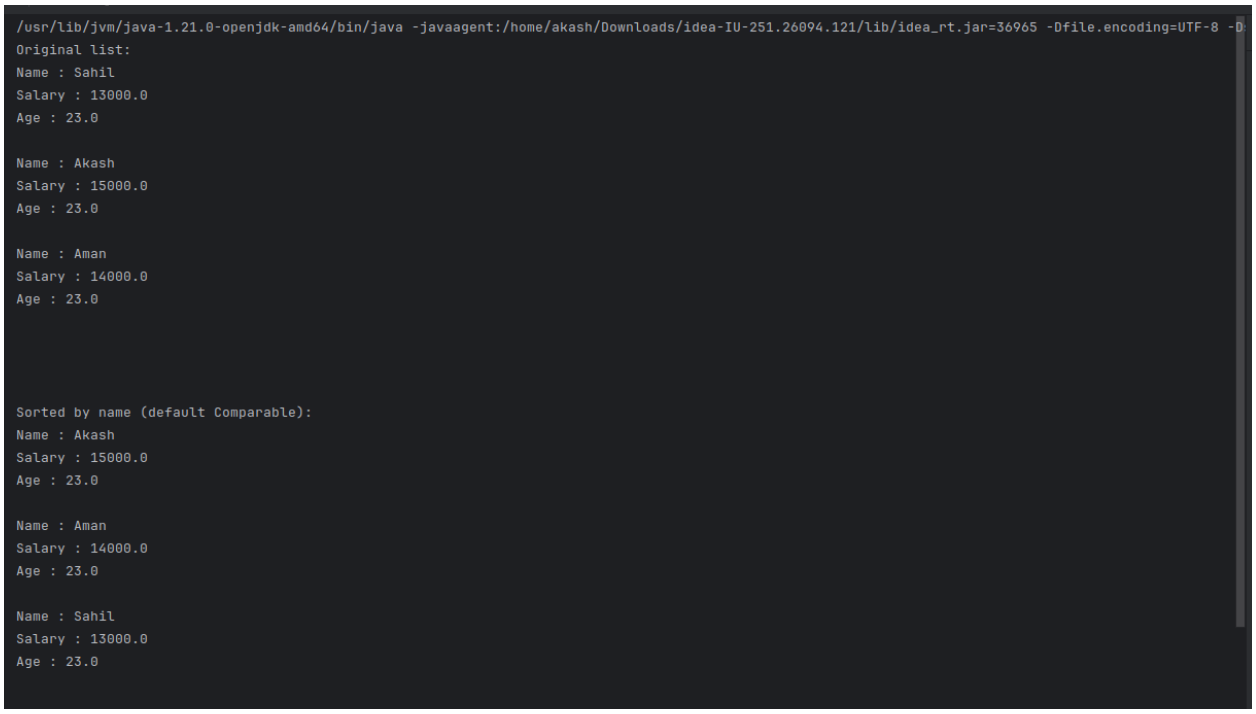
Q2)Given the following class Employee class{ Double Age; Double Salary; String Name} Design the class in such a way that the default sorting should work on firstname and lastname. Also, Write a program to sort Employee objects based on salary using Comparator.



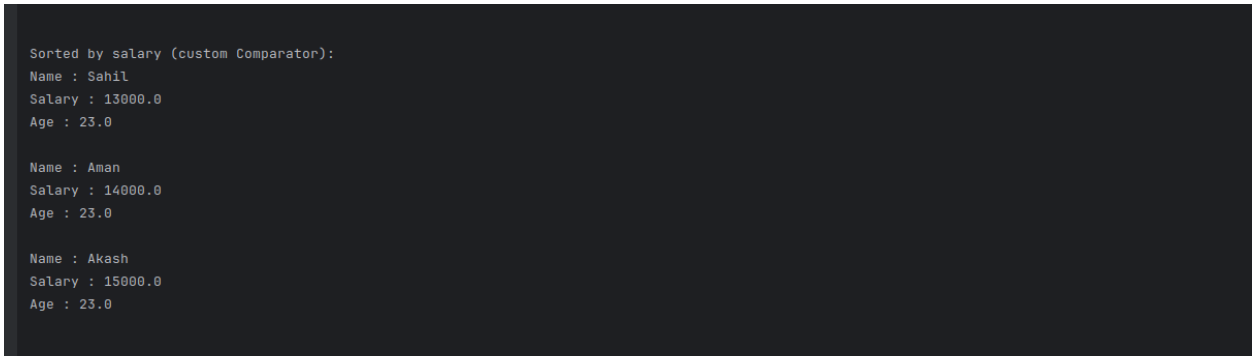
```java
package Collections.Q2;

import java.util.ArrayList;
import java.util.Comparator;
import java.util.Collections;
import java.util.List;

class Employee implements Comparable<Employee> {  12 usages
    String name;  4 usages
    double salary;  4 usages
    double age;  2 usages

    public Employee(String name, double salary, double age) {  3 usages
        this.name = name;
        this.salary = salary;
        this.age = age;
    }
    public int compareTo(Employee e) {  1 usage
        return this.name.compareTo(e.name);

    }
    @Override
    public String toString() {
        return "Name : "+ name +"\n" +
                "Salary : "+ salary+"\n"+
                "Age : "+ age + "\n";
    }
}
class SalaryComparator implements Comparator<Employee> {  1 usage
    public int compare(Employee e1, Employee e2) {  no usages
        return Double.compare(e1.salary, e2.salary);
    }



}
```



```
}
public class Q2 {  2 usages
    public static void CompareEmp() {  1 usage

        List<Employee> list = new ArrayList<>();

        list.add(new Employee( name: "Sahil",  salary: 13000.0,  age: 23));
        list.add(new Employee( name: "Akash",  salary: 15000.0,  age: 23));
        list.add(new Employee( name: "Aman",  salary: 14000.0,  age: 23));


        System.out.println("Original list:");
        for (Employee e : list) {
            System.out.println(e);
        }


        Collections.sort(list);
        System.out.println("\n\n\nSorted by name (default Comparable):");
        for (Employee e : list) {
            System.out.println(e);
        }


        Collections.sort(list, new SalaryComparator());
        System.out.println("\n\n\nSorted by salary (custom Comparator):");
        for (Employee e : list) {
            System.out.println(e);
        }
    }
}
```

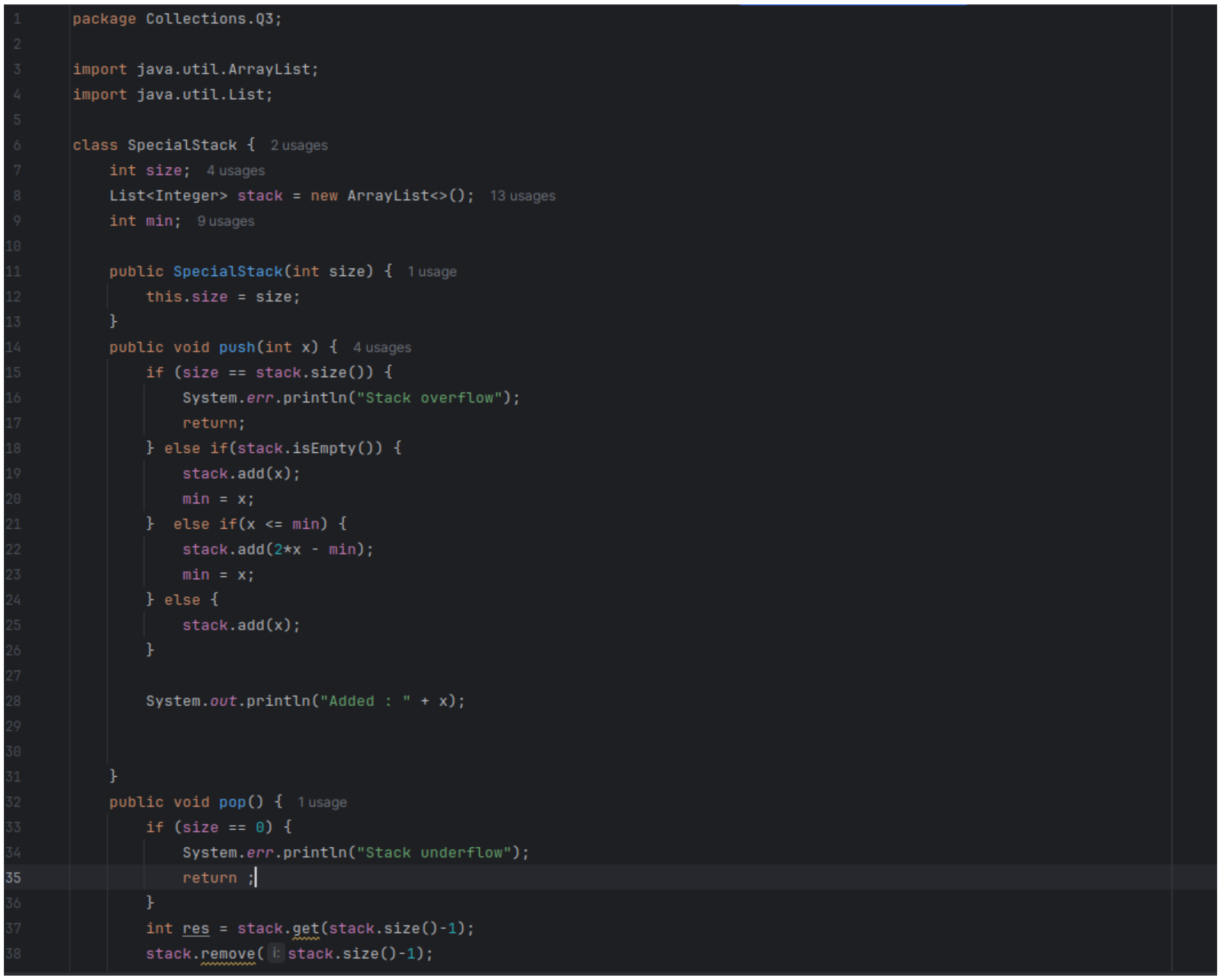


```
/usr/lib/jvm/java-1.21.0-openjdk-amd64/bin/java -javaagent:/home/akash/Downloads/idea-IU-251.26094.121/lib/idea_rt.jar=36965 -Dfile.encoding=UTF-8 -D
Original list:
Name : Sahil
Salary : 13000.0
Age : 23.0

Name : Akash
Salary : 15000.0
Age : 23.0

Name : Aman
Salary : 14000.0
Age : 23.0




Sorted by name (default Comparable):
Name : Akash
Salary : 15000.0
Age : 23.0

Name : Aman
Salary : 14000.0
Age : 23.0

Name : Sahil
Salary : 13000.0
Age : 23.0
```

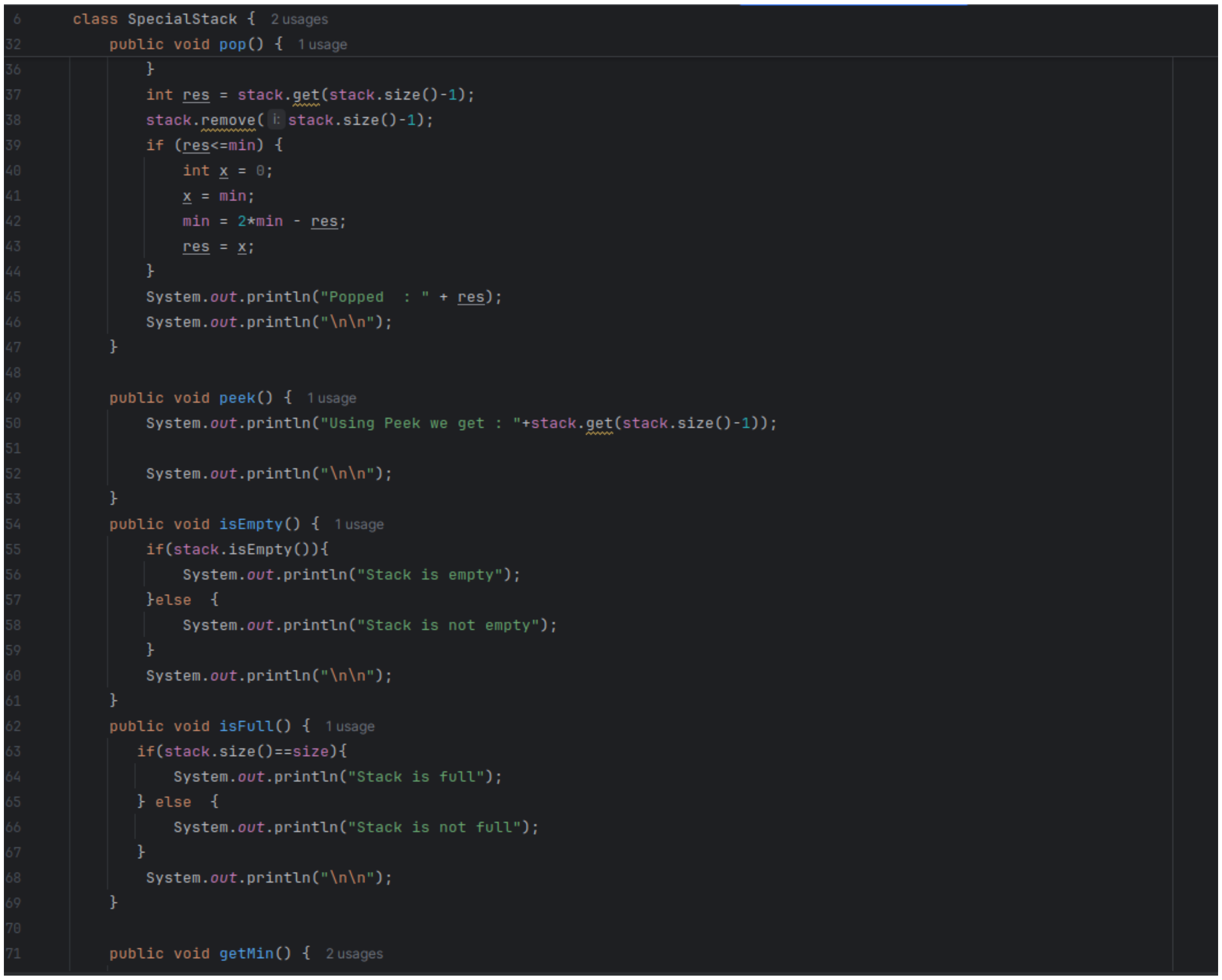


```
Sorted by salary (custom Comparator):
Name : Sahil
Salary : 13000.0
Age : 23.0

Name : Aman
Salary : 14000.0
Age : 23.0

Name : Akash
Salary : 15000.0
Age : 23.0
```

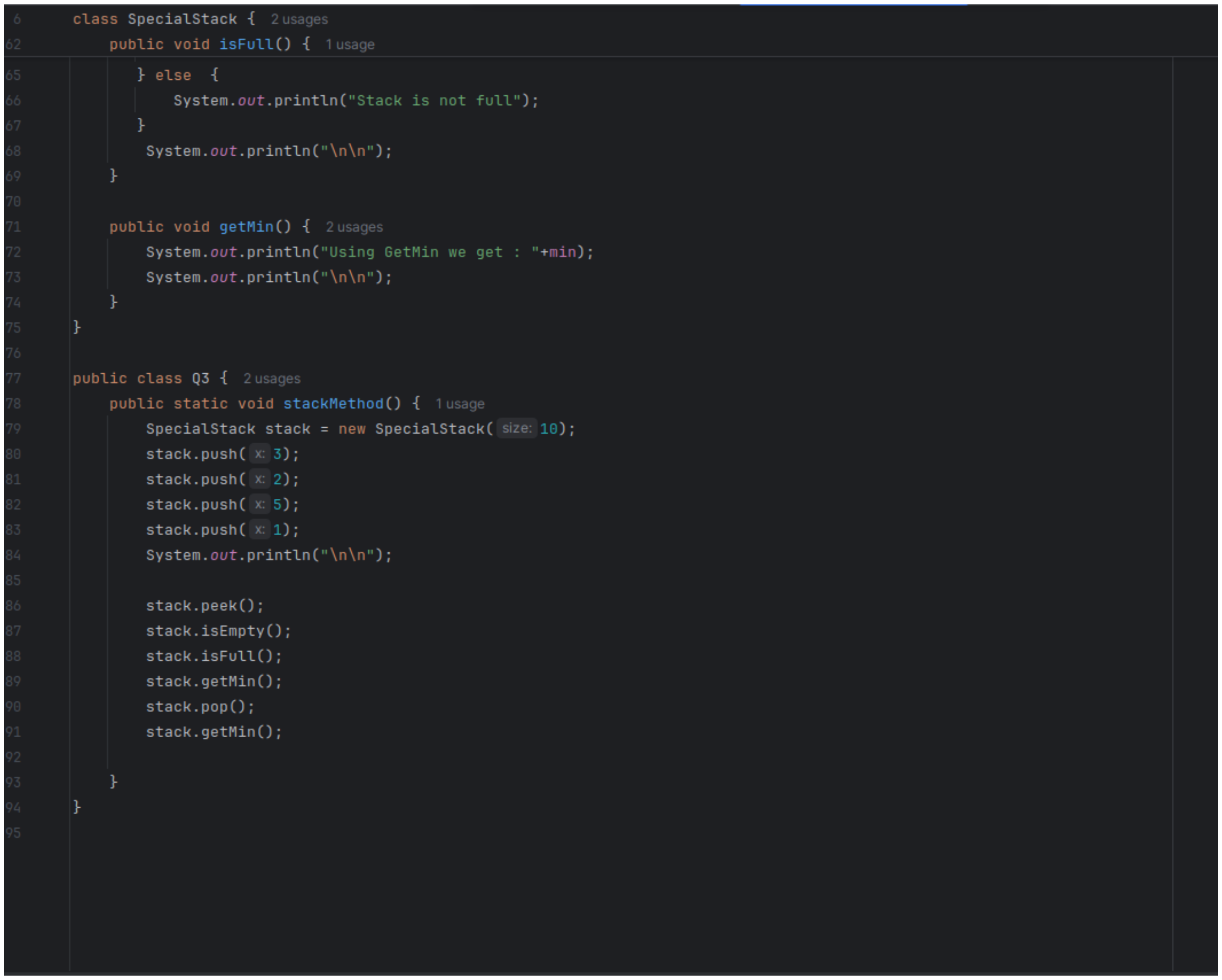
Q3)Design a Data Structure SpecialStack that supports all the stack operations like push(), pop(), isEmpty(), isFull() and an additional operation getMin() which should return minimum element from the SpecialStack. (Expected complexity O(1))



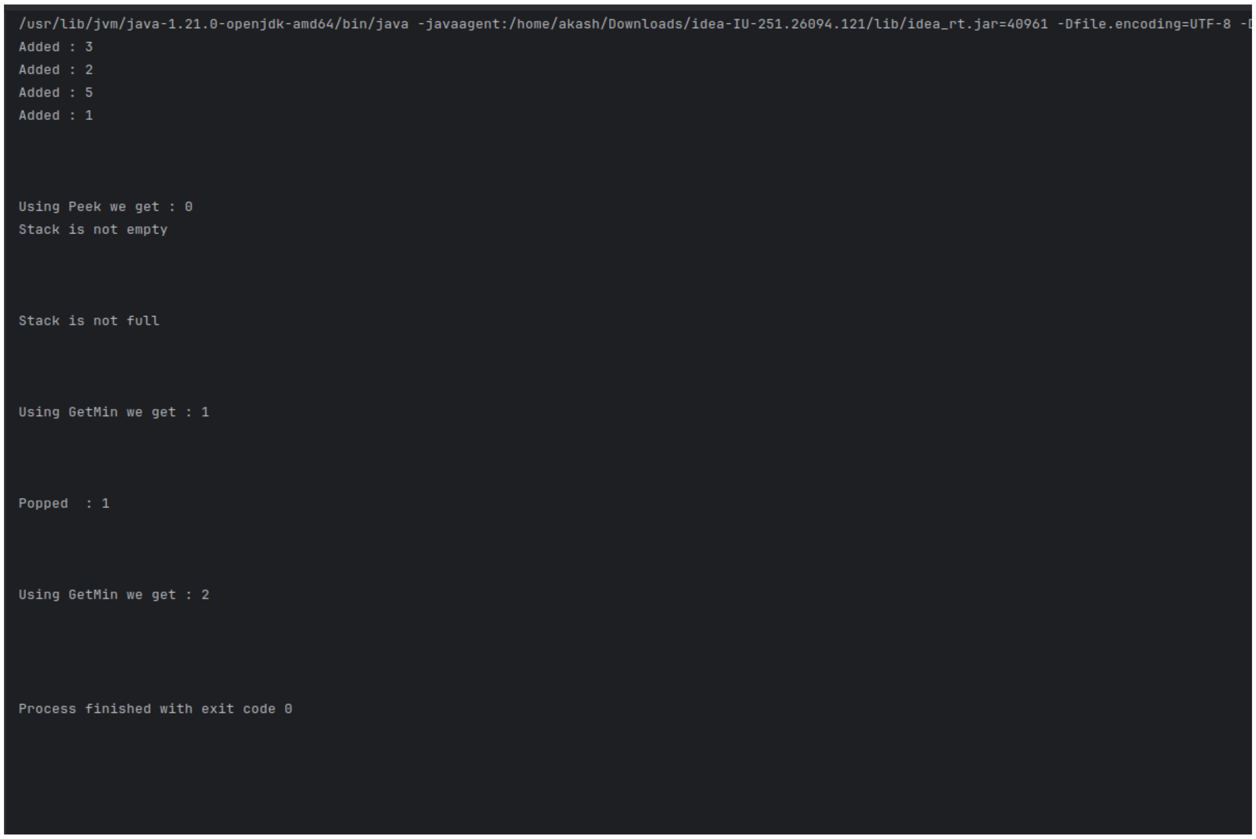
```java
package Collections.Q3;

import java.util.ArrayList;
import java.util.List;

class SpecialStack {  2 usages
    int size;  4 usages
    List<Integer> stack = new ArrayList<>();  13 usages
    int min;  9 usages

    public SpecialStack(int size) {  1 usage
        this.size = size;
    }
    public void push(int x) {  4 usages
        if (size == stack.size()) {
            System.err.println("Stack overflow");
            return;
        } else if(stack.isEmpty()) {
            stack.add(x);
            min = x;
        }  else if(x <= min) {
            stack.add(2*x - min);
            min = x;
        } else {
            stack.add(x);
        }

        System.out.println("Added : " + x);


    }
    public void pop() {  1 usage
        if (size == 0) {
            System.err.println("Stack underflow");
            return ;
        }
        int res = stack.get(stack.size()-1);
        stack.remove( i: stack.size()-1);
```



```
class SpecialStack {  2 usages
    public void pop() {  1 usage
        }
        int res = stack.get(stack.size()-1);
        stack.remove( i: stack.size()-1);
        if (res<=min) {
            int x = 0;
            x = min;
            min = 2*min - res;
            res = x;
        }
        System.out.println("Popped  : " + res);
        System.out.println("\n\n");
    }

    public void peek() {  1 usage
        System.out.println("Using Peek we get : "+stack.get(stack.size()-1));

        System.out.println("\n\n");
    }
    public void isEmpty() {  1 usage
        if(stack.isEmpty()){
            System.out.println("Stack is empty");
        }else  {
            System.out.println("Stack is not empty");
        }
        System.out.println("\n\n");
    }
    public void isFull() {  1 usage
       if(stack.size()==size){
           System.out.println("Stack is full");
       } else  {
           System.out.println("Stack is not full");
       }
        System.out.println("\n\n");
    }

    public void getMin() {  2 usages
```

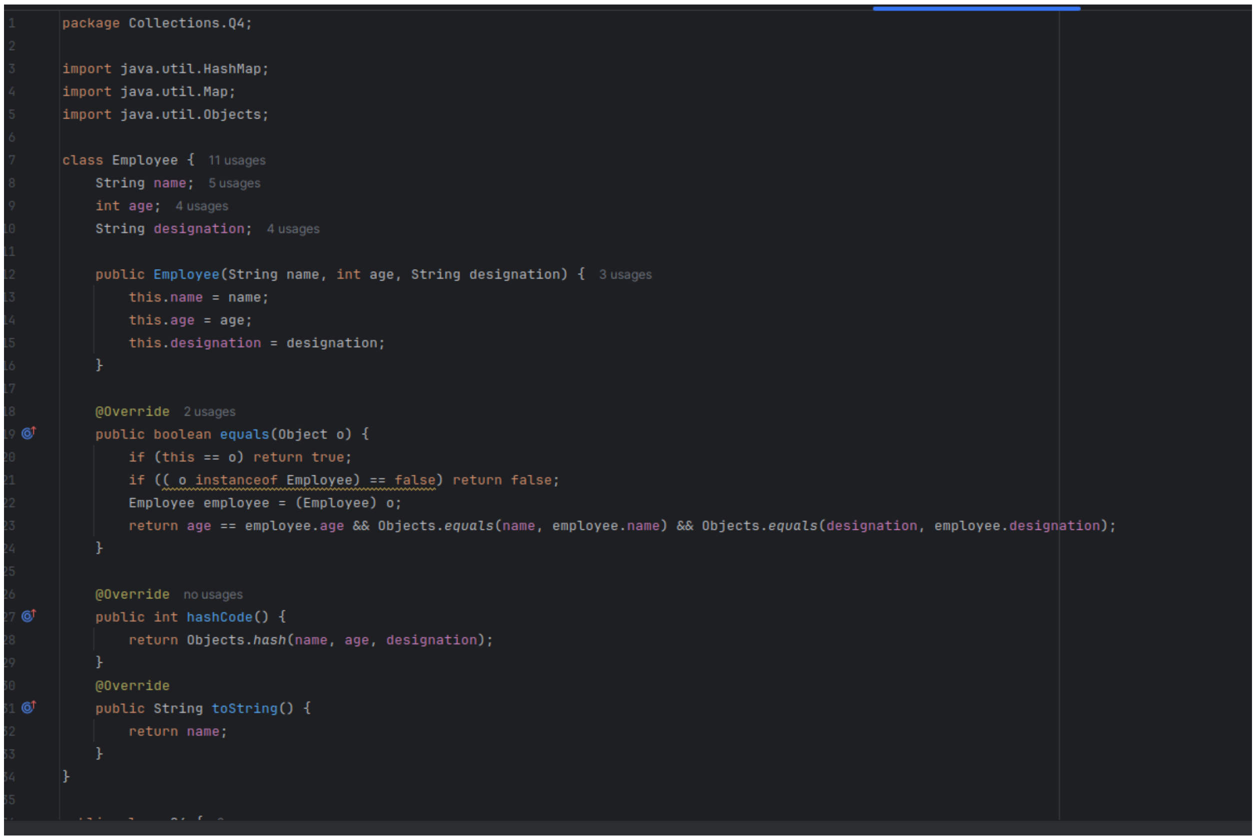


```
class SpecialStack {  2 usages
    public void isFull() {  1 usage
        } else  {
            System.out.println("Stack is not full");
        }
        System.out.println("\n\n");
    }

    public void getMin() {  2 usages
        System.out.println("Using GetMin we get : "+min);
        System.out.println("\n\n");
    }
}

public class Q3 {  2 usages
    public static void stackMethod() {  1 usage
        SpecialStack stack = new SpecialStack( size: 10);
        stack.push( x: 3);
        stack.push( x: 2);
        stack.push( x: 5);
        stack.push( x: 1);
        System.out.println("\n\n");

        stack.peek();
        stack.isEmpty();
        stack.isFull();
        stack.getMin();
        stack.pop();
        stack.getMin();

    }
}
```

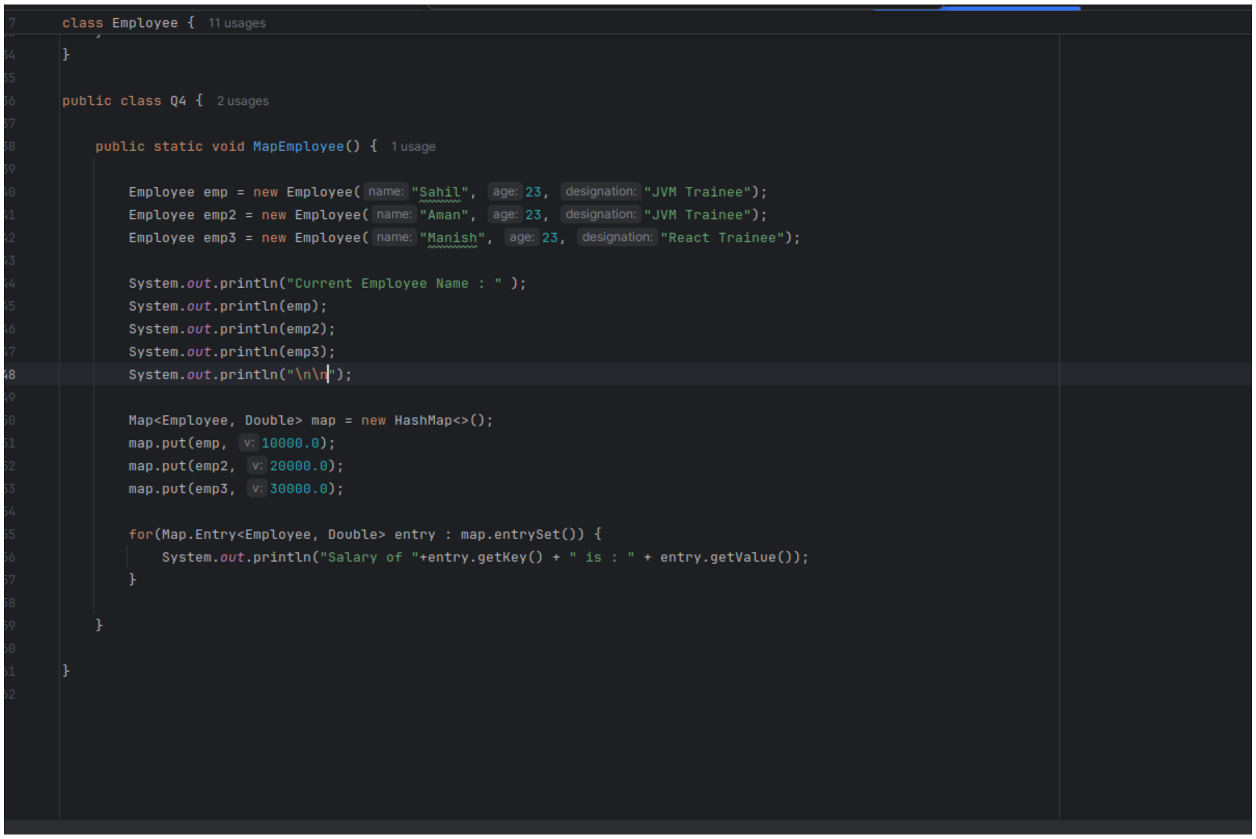


```
/usr/lib/jvm/java-1.21.0-openjdk-amd64/bin/java -javaagent:/home/akash/Downloads/idea-IU-251.26094.121/lib/idea_rt.jar=40961 -Dfile.encoding=UTF-8 -
Added : 3
Added : 2
Added : 5
Added : 1



Using Peek we get : 0
Stack is not empty



Stack is not full



Using GetMin we get : 1



Popped  : 1



Using GetMin we get : 2




Process finished with exit code 0
```

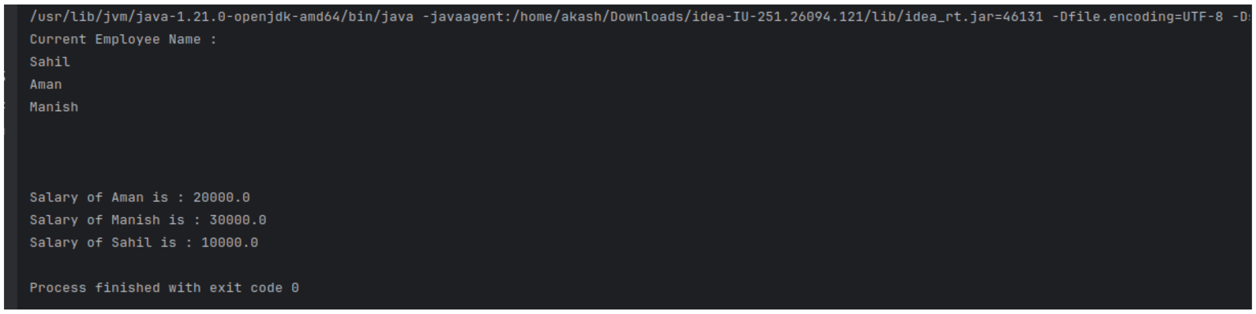
Q4)Create class Employee with attributes name,age,designation and use instances of these class as keys in a Map and their salary as value



```java
package Collections.Q4;

import java.util.HashMap;
import java.util.Map;
import java.util.Objects;

class Employee {  11 usages
    String name;  5 usages
    int age;  4 usages
    String designation;  4 usages

    public Employee(String name, int age, String designation) {  3 usages
        this.name = name;
        this.age = age;
        this.designation = designation;
    }

    @Override  2 usages
    public boolean equals(Object o) {
        if (this == o) return true;
        if (( o instanceof Employee) == false) return false;
        Employee employee = (Employee) o;
        return age == employee.age && Objects.equals(name, employee.name) && Objects.equals(designation, employee.designation);
    }

    @Override  no usages
    public int hashCode() {
        return Objects.hash(name, age, designation);
    }
    @Override
    public String toString() {
        return name;
    }
}
```



```java
class Employee {  11 usages
}

public class Q4 {  2 usages

    public static void MapEmployee() {  1 usage

        Employee emp = new Employee( name: "Sahil",  age: 23,  designation: "JVM Trainee");
        Employee emp2 = new Employee( name: "Aman",  age: 23,  designation: "JVM Trainee");
        Employee emp3 = new Employee( name: "Manish",  age: 23,  designation: "React Trainee");

        System.out.println("Current Employee Name : " );
        System.out.println(emp);
        System.out.println(emp2);
        System.out.println(emp3);
        System.out.println("\n\n");

        Map<Employee, Double> map = new HashMap<>();
        map.put(emp,  v: 10000.0);
        map.put(emp2,  v: 20000.0);
        map.put(emp3,  v: 30000.0);

        for(Map.Entry<Employee, Double> entry : map.entrySet()) {
            System.out.println("Salary of "+entry.getKey() + " is : " + entry.getValue());
        }

    }

}
```



```
/usr/lib/jvm/java-1.21.0-openjdk-amd64/bin/java -javaagent:/home/akash/Downloads/idea-IU-251.26094.121/lib/idea_rt.jar=46131 -Dfile.encoding=UTF-8 -D
Current Employee Name : 
Sahil
Aman
Manish



Salary of Aman is : 20000.0
Salary of Manish is : 30000.0
Salary of Sahil is : 10000.0

Process finished with exit code 0
```