Problem 1

```
package Prob1;
import java.util.*;
public class Admin {
	public static HashMap<Key, Student> processStudents(List<Student>
students)
	{
		HashMap<Key, Student>  result= new HashMap<Key, Student>();

		for (Student student : students) {
			Key key= new Key(student.getFirstName(),
student.getLastName());
			result.put(key, student);


		}
		return result;
	}
}
```

```
package Prob1;

public class Key {
	private String firstName;
	private String lastName;

	public String getFirstName() {
		return firstName;
	}

	public String getLastName() {
		return lastName;
	}

	public Key(String f, String l) {
		this.firstName = f;
		this.lastName = l;
	}



	@Override
	public boolean equals(Object obj) {
		if (this == obj) {
			return true;
		}
		if (obj == null) {
			return false;
		}
		if (getClass() != obj.getClass()) {
			return false;
```

```
			}
			Key other = (Key) obj;
			if (firstName == null) {
				if (other.firstName != null)
					return false;
			}
			if (lastName == null) {
				if (other.lastName != null)
					return false;
			}

			return firstName.equals(other.firstName) &&
lastName.equals(other.lastName);
		}

		@Override
		public int hashCode() {
			final int prime = 31;
			int result = 1;
			result = prime * result + ((firstName == null) ? 0 :
firstName.hashCode());
			result = prime * result + ((lastName == null) ? 0 :
lastName.hashCode());
			return result;
		}
}


package Prob1;

public class Student {
		private String firstName;
		private String lastName;
		private double gpa;
		private Standing standing;
		public Student(String firstName, String lastName, double gpa, Standing
standing) {
			this.firstName = firstName;
			this.lastName = lastName;
			this.gpa=gpa;
			this.standing = standing;
		}
		public Standing getStanding() {
			return standing;
		}
		public double getGpa() {
			return gpa;
		}
		public String getFirstName() {
			return firstName;
		}
		public String getLastName() {
			return lastName;
		}
		@Override
```

```
	public String toString() {
		return "[" + firstName + " " + lastName + "]";
	}

	@Override
	public boolean equals(Object ob) {
		if(ob == null) return false;
		if(ob.getClass() != Student.class) return false;
		Student s = (Student)ob;
		return s.firstName.equals(firstName) &&
s.lastName.equals(lastName);
	}

}
```

```
17
18        HashMap<Key, Student> map = Admin.processStudents(list);
19        boolean[] expectedOutput = {true, false};
20        boolean[] results = new boolean[2];
21        Student s = new Student("Pierre", "Fromage", 2.8, Standing.FRESHMAN);
22        Key pierre = new Key("Pierre", "Fromage");
23        Key richard = new Key("Richard", "Fremling");
24
25        results[0] = (map.get(pierre).equals(s));
26        results[1] = map.containsKey(richard);
27        System.out.println((Arrays.equals(expectedOutput, results) ? "pass" : "fail"));
28
29
30
31
32    }
33
34 }
35
```

```
Javadoc  Declaration  Search  Console
<terminated> Test [Java Application] C:\Program Files\Java\jre1.8.0_60\bin\javaw.exe (16 jun. 2018 9:45:39)
pass
```

Problem 2

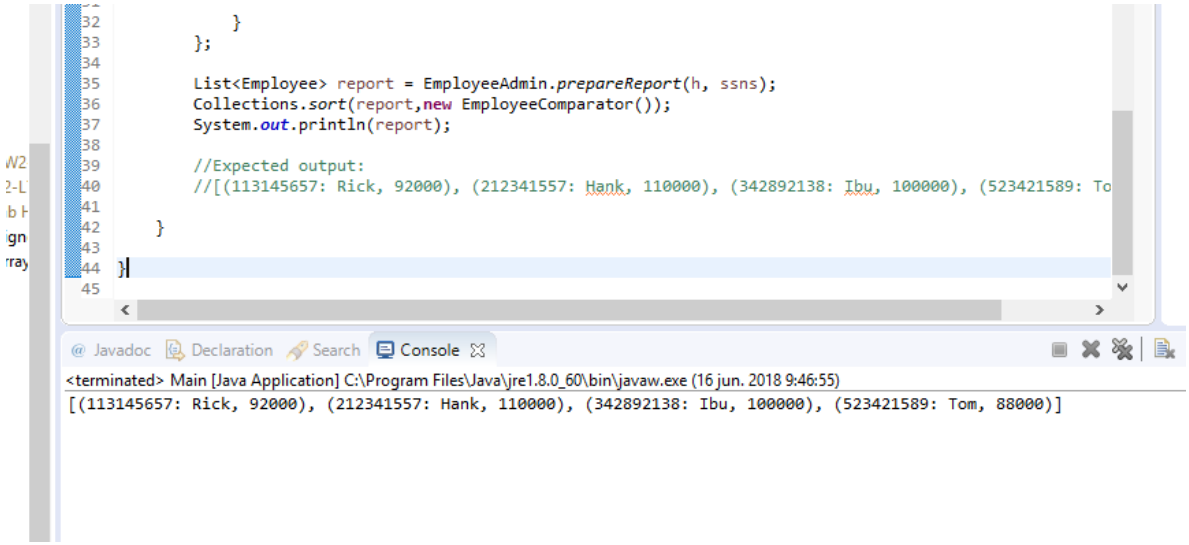
```
package prob2;

//DO NOT MODIFY IN ANY WAY
public class Employee {
	private String name;
	private int salary;
	private String ssn;
	public Employee(String name, int salary, String ssn) {
		this.name = name;
		this.salary = salary;
		this.ssn=ssn;
	}

	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public int getSalary() {
```

```
		return salary;
	}
	public void setSalary(int salary) {
		this.salary = salary;
	}
	public String getSsn() {
		return ssn;
	}

	public void setSsn(String ssn) {
		this.ssn = ssn;
	}

	@Override
	public String toString() { //DO NOT MODIFY
		return "(" + ssn + ": " + name + ", " + salary + ")" ;
	}

	@Override
	public boolean equals(Object ob) {
		if(ob == null) return false;
		if(!(ob instanceof Employee)) return false;
		Employee e = (Employee)ob;
		return e.ssn.equals(ssn);
	}
}
```

```
package prob2;


import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;


public class EmployeeAdmin {


	/**

		Returns a list of Employees whose social security number is on the input list socSecNums

		and whose salary is > 80000. The list must be ordered by social security number,

		from lowest to highest. To sort, you must use a Collections sorting method

		and you must define your own Comparator to be used to compare Employees by ssn.
```

```
	*/
	public static List<Employee> prepareReport(HashMap<String, Employee> table,
List<String> socSecNums)
	{
		int SALARY=80000;
		List<Employee> result= new ArrayList<>();
		for (String  ssn : socSecNums) {

			if (table.containsKey(ssn))
			{
				Employee person= table.get(ssn);
				if (person.getSalary()>SALARY )
				{
					result.add(person);
				}

			}
		}

		return result;

	}


}
package prob2;

import java.util.Comparator;

class EmployeeComparator implements Comparator<Employee>
{
```

```
        @Override
        public int compare(Employee o1, Employee o2) {

                return o1.getSsn().compareTo(o2.getSsn());
        }
}
```



Problem 3

```
package Prob3;

/** NOTE: You must override equals in this class */
public class Employee {
        private String name;
        private int salary;
        public Employee(String name, int salary) {
                this.name = name;
                this.salary = salary;
        }

        public String getName() {
                return name;
        }
        public void setName(String name) {
                this.name = name;
        }
        public int getSalary() {
                return salary;
        }
        public void setSalary(int salary) {
                this.salary = salary;
        }
        @Override
        public String toString() {
                return "(" + name + ", " + salary + ")";
```

```
	}



	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Employee other = (Employee) obj;
		if (name == null)
		{
			if (other.name != null)
			{
				return false;
			}
		}

		return this.salary==other.salary && this.name.equals(other.name);
	}


}
```