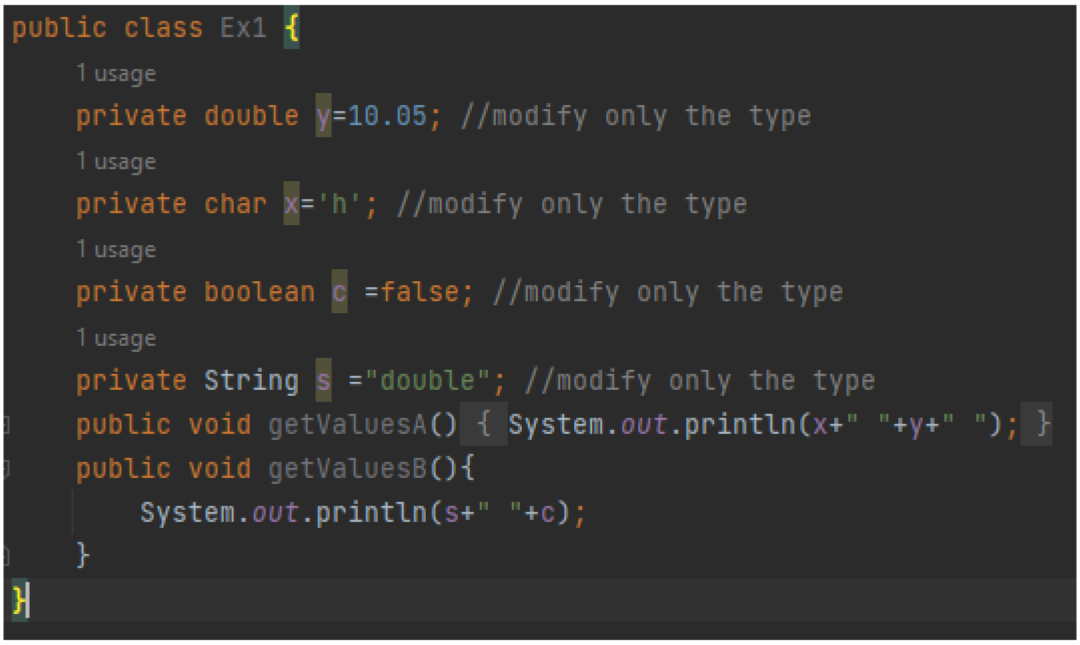
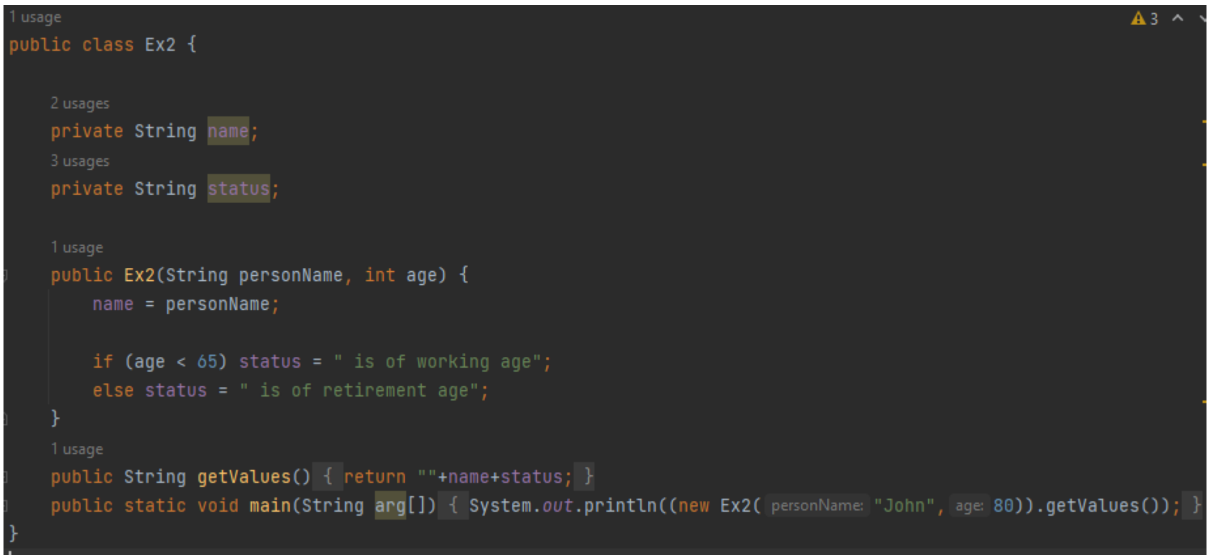
Question 1:



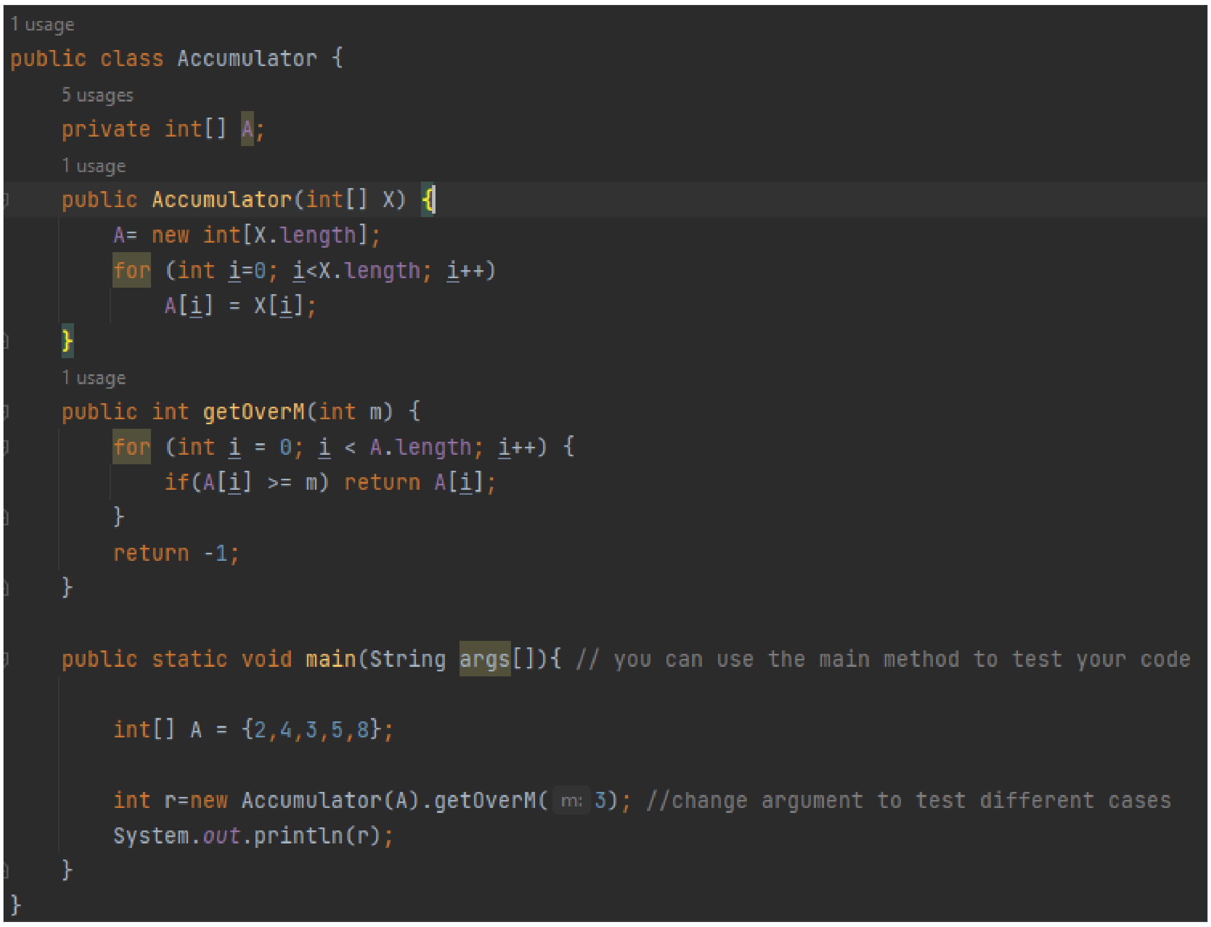
```java
public class Ex1 {
    1 usage
    private double y=10.05; //modify only the type
    1 usage
    private char x='h'; //modify only the type
    1 usage
    private boolean c =false; //modify only the type
    1 usage
    private String s ="double"; //modify only the type
    public void getValuesA() { System.out.println(x+" "+y+" "); }
    public void getValuesB(){
        System.out.println(s+" "+c);
    }
}
```

Question 2:



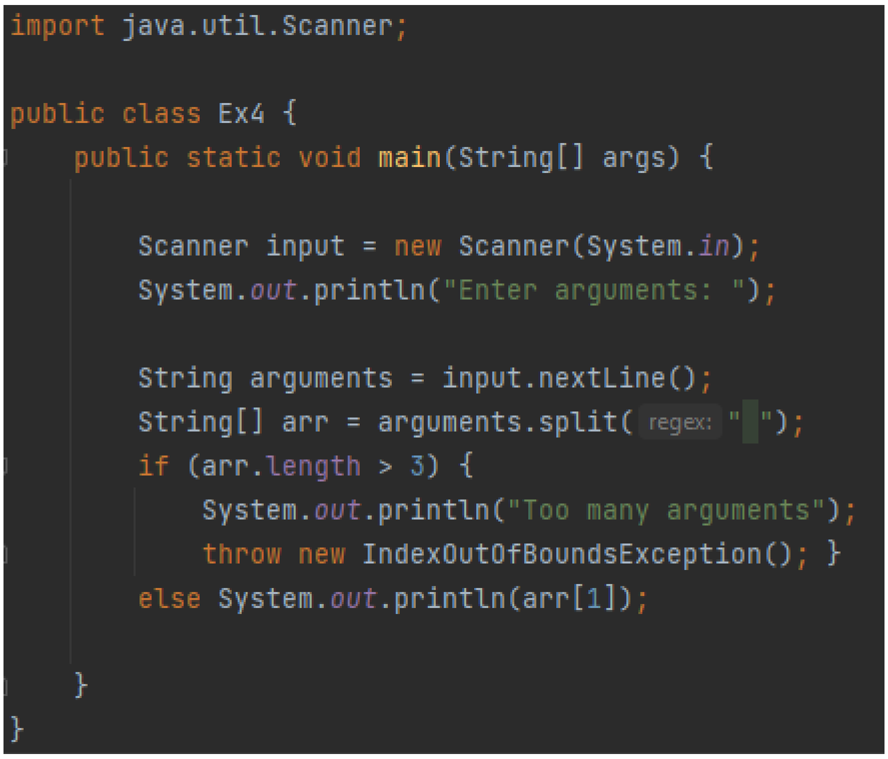
```java
1 usage
public class Ex2 {

    2 usages
    private String name;
    3 usages
    private String status;

    1 usage
    public Ex2(String personName, int age) {
        name = personName;

        if (age < 65) status = " is of working age";
        else status = " is of retirement age";
    }
    1 usage
    public String getValues() { return ""+name+status; }
    public static void main(String arg[]) { System.out.println((new Ex2( personName: "John", age: 80)).getValues()); }
}
```

Question 3:



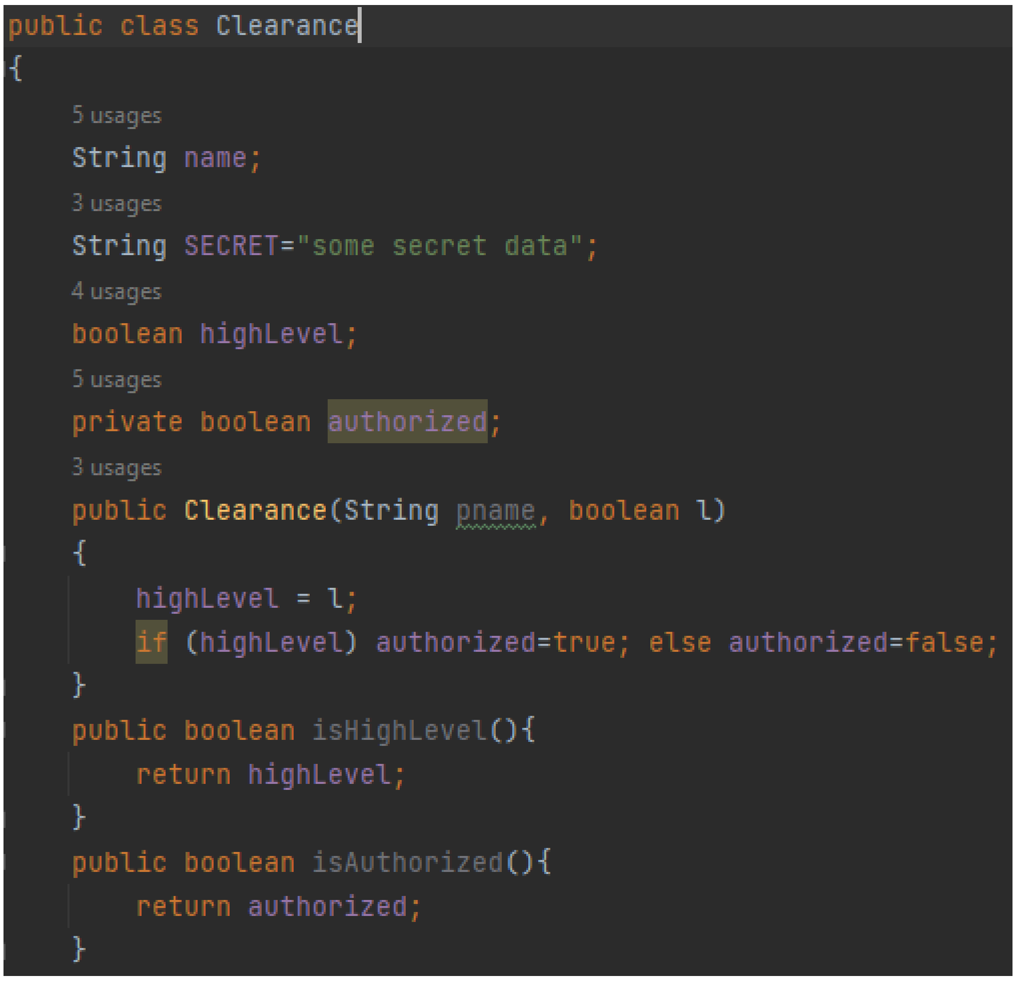
```java
1 usage
public class Accumulator {
    5 usages
    private int[] A;
    1 usage
    public Accumulator(int[] X) {
        A= new int[X.length];
        for (int i=0; i<X.length; i++)
            A[i] = X[i];
    }
    1 usage
    public int getOverM(int m) {
        for (int i = 0; i < A.length; i++) {
            if(A[i] >= m) return A[i];
        }
        return -1;
    }

    public static void main(String args[]){ // you can use the main method to test your code

        int[] A = {2,4,3,5,8};

        int r=new Accumulator(A).getOverM( m: 3); //change argument to test different cases
        System.out.println(r);
    }
}
```

Question 4:

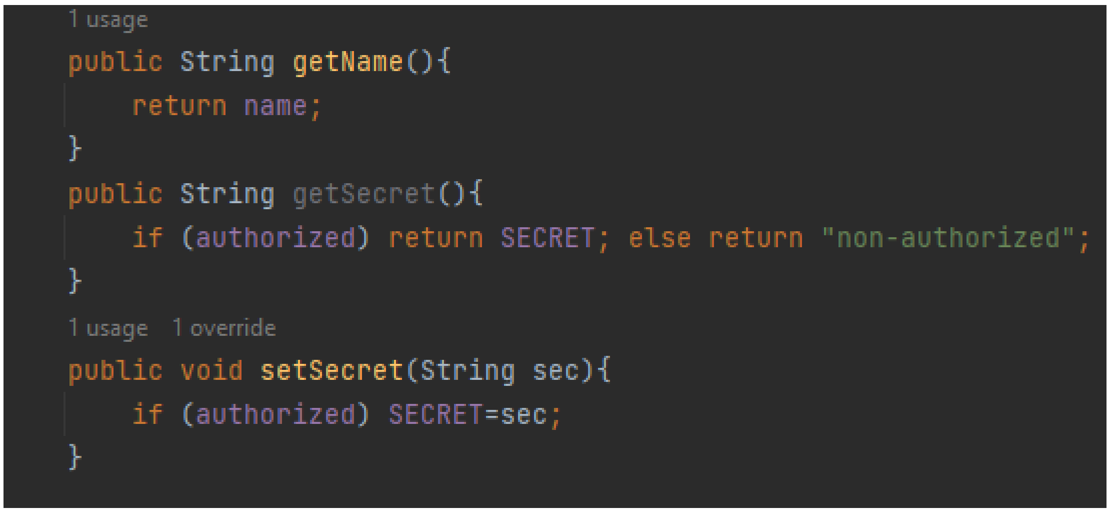


```
import java.util.Scanner;

public class Ex4 {
    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);
        System.out.println("Enter arguments: ");

        String arguments = input.nextLine();
        String[] arr = arguments.split( regex: " ");
        if (arr.length > 3) {
            System.out.println("Too many arguments");
            throw new IndexOutOfBoundsException(); }
        else System.out.println(arr[1]);

    }
}
```

Question 5:

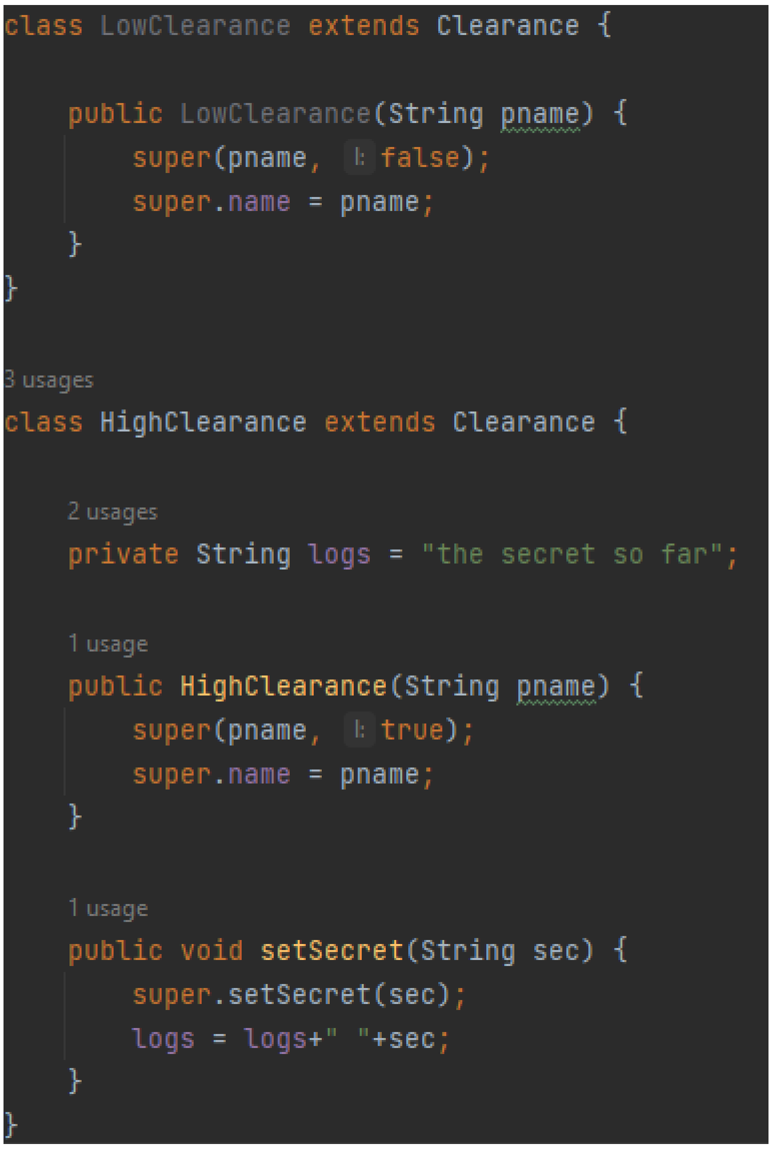


```java
public class Clearance
{
    5 usages
    String name;
    3 usages
    String SECRET="some secret data";
    4 usages
    boolean highLevel;
    5 usages
    private boolean authorized;
    3 usages
    public Clearance(String pname, boolean l)
    {
        highLevel = l;
        if (highLevel) authorized=true; else authorized=false;
    }
    public boolean isHighLevel(){
        return highLevel;
    }
    public boolean isAuthorized(){
        return authorized;
    }
```



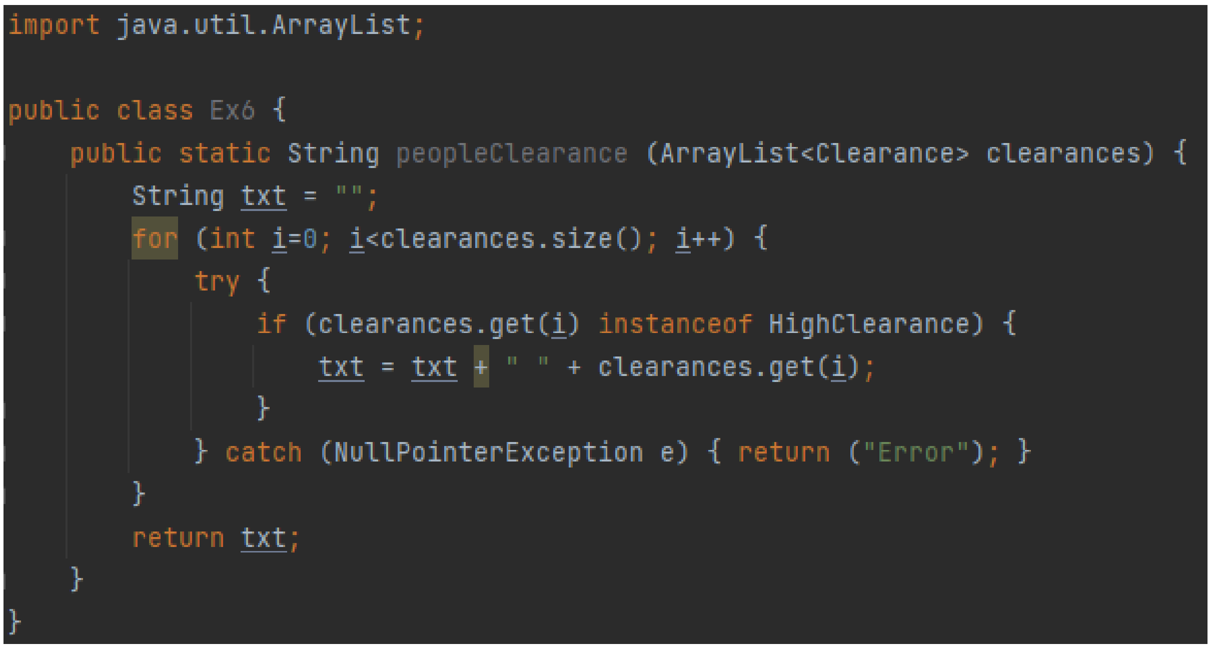
```java
    1 usage
    public String getName(){
        return name;
    }
    public String getSecret(){
        if (authorized) return SECRET; else return "non-authorized";
    }
    1 usage   1 override
    public void setSecret(String sec){
        if (authorized) SECRET=sec;
    }
```

```java
}
```



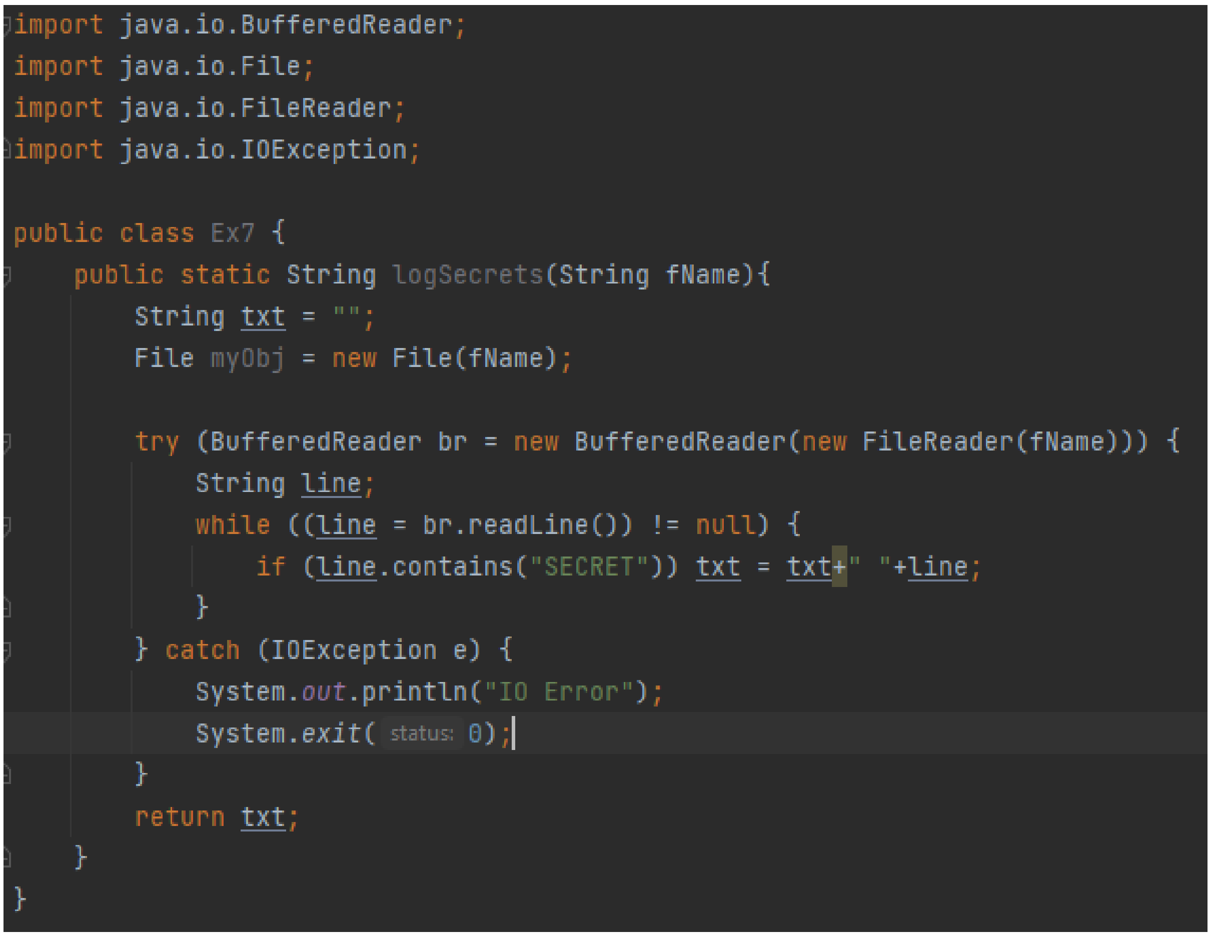
```java
class LowClearance extends Clearance {

    public LowClearance(String pname) {
        super(pname,  false);
        super.name = pname;
    }
}

3 usages
class HighClearance extends Clearance {

    2 usages
    private String logs = "the secret so far";

    1 usage
    public HighClearance(String pname) {
        super(pname,  true);
        super.name = pname;
    }

    1 usage
    public void setSecret(String sec) {
        super.setSecret(sec);
        logs = logs+" "+sec;
    }
}
```

Question 6:



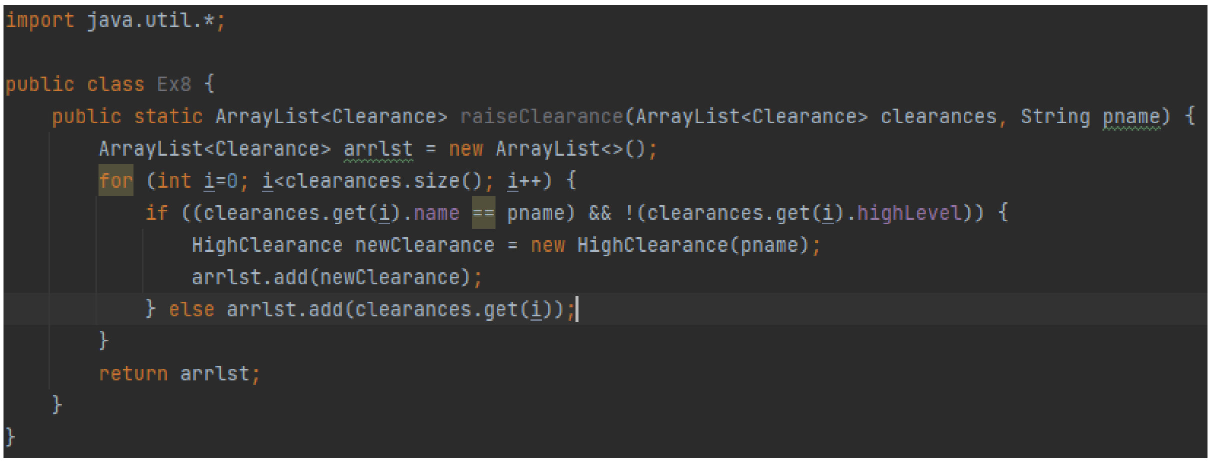
```java
import java.util.ArrayList;

public class Ex6 {
    public static String peopleClearance (ArrayList<Clearance> clearances) {
        String txt = "";
        for (int i=0; i<clearances.size(); i++) {
            try {
                if (clearances.get(i) instanceof HighClearance) {
                    txt = txt + " " + clearances.get(i);
                }
            } catch (NullPointerException e) { return ("Error"); }
        }
        return txt;
    }
}
```

Question 7:



```java
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;

public class Ex7 {
    public static String logSecrets(String fName){
        String txt = "";
        File myObj = new File(fName);

        try (BufferedReader br = new BufferedReader(new FileReader(fName))) {
            String line;
            while ((line = br.readLine()) != null) {
                if (line.contains("SECRET")) txt = txt+" "+line;
            }
        } catch (IOException e) {
            System.out.println("IO Error");
            System.exit( status: 0);
        }
        return txt;
    }
}
```

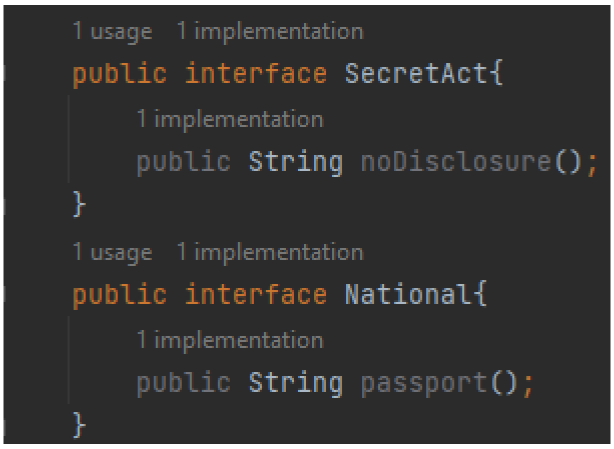
Question 8:



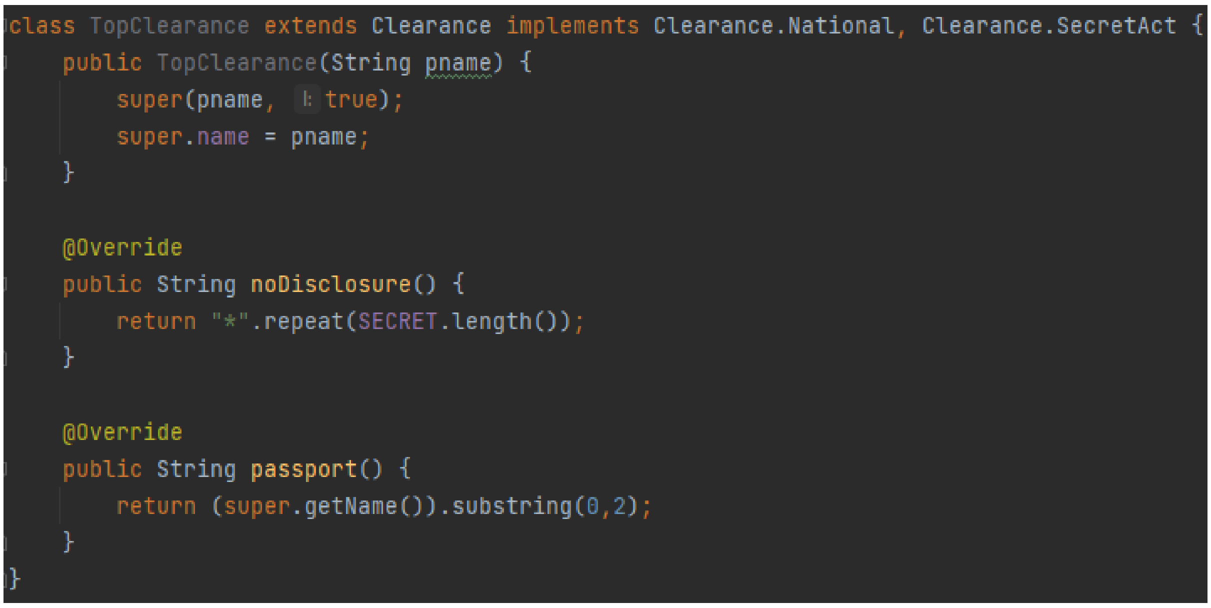
```java
import java.util.*;

public class Ex8 {
    public static ArrayList<Clearance> raiseClearance(ArrayList<Clearance> clearances, String pname) {
        ArrayList<Clearance> arrlst = new ArrayList<>();
        for (int i=0; i<clearances.size(); i++) {
            if ((clearances.get(i).name == pname) && !(clearances.get(i).highLevel)) {
                HighClearance newClearance = new HighClearance(pname);
                arrlst.add(newClearance);
            } else arrlst.add(clearances.get(i));
        }
        return arrlst;
    }
}
```

Question 9:

A:

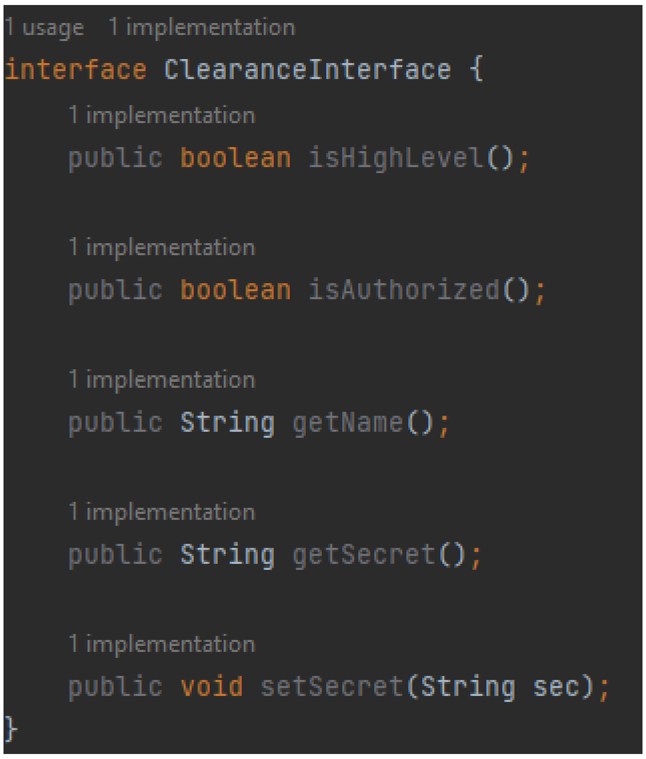


```java
    1 usage   1 implementation
    public interface SecretAct{
        1 implementation
        public String noDisclosure();
    }
    1 usage   1 implementation
    public interface National{
        1 implementation
        public String passport();
    }
```

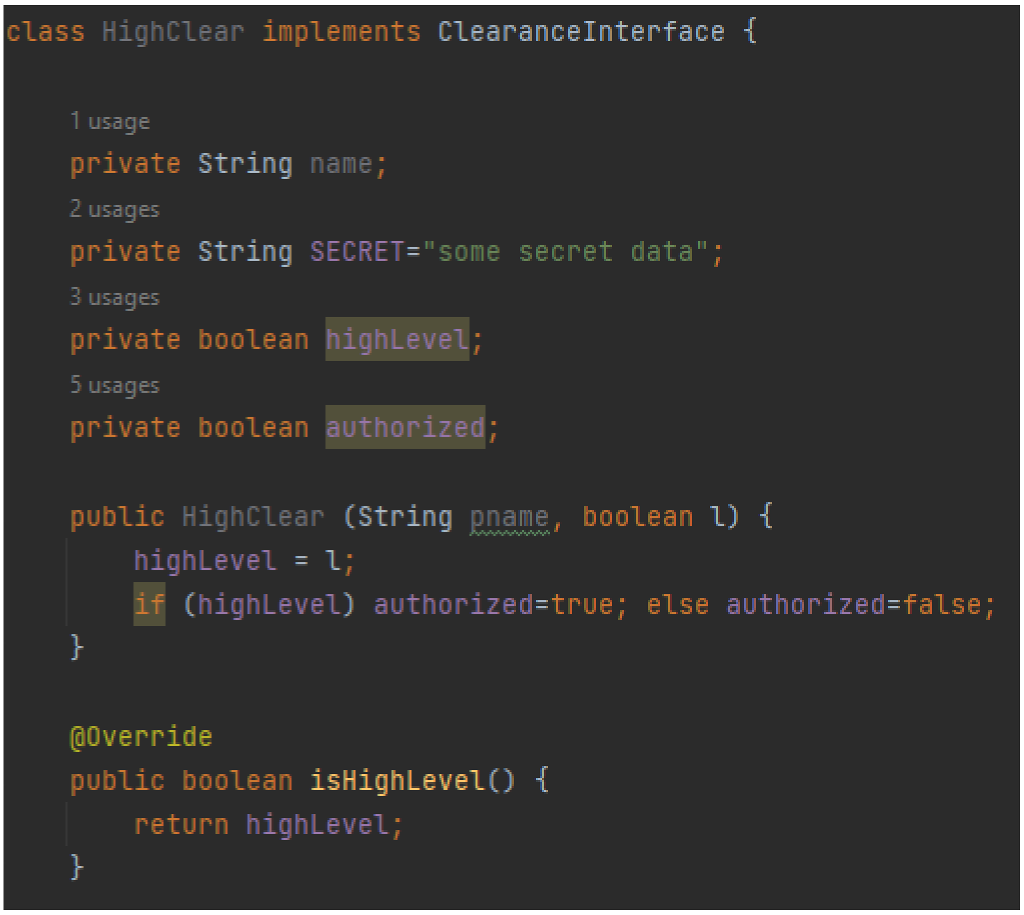


```java
class TopClearance extends Clearance implements Clearance.National, Clearance.SecretAct {
    public TopClearance(String pname) {
        super(pname,  l: true);
        super.name = pname;
    }

    @Override
    public String noDisclosure() {
        return "*".repeat(SECRET.length());
    }

    @Override
    public String passport() {
        return (super.getName()).substring(0,2);
    }
}
```

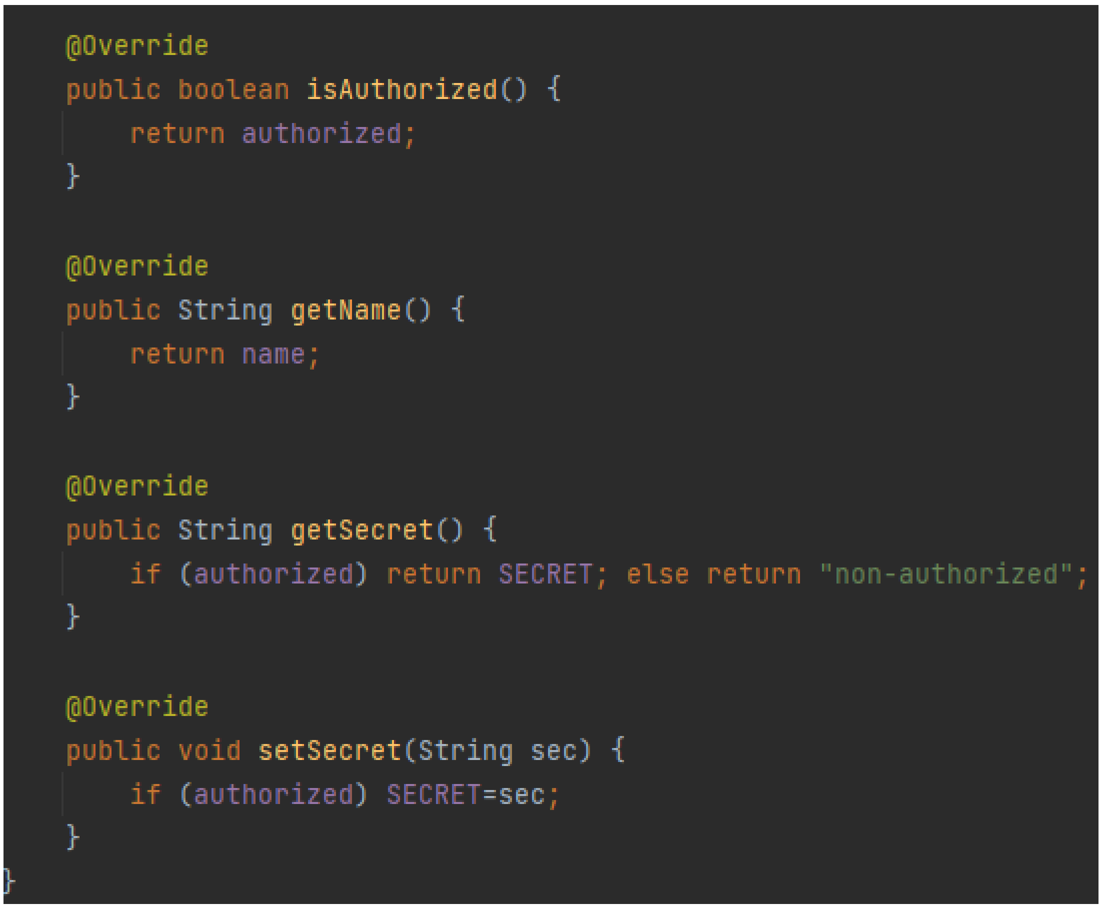
B:



```
1 usage   1 implementation
interface ClearanceInterface {
    1 implementation
    public boolean isHighLevel();

    1 implementation
    public boolean isAuthorized();

    1 implementation
    public String getName();

    1 implementation
    public String getSecret();

    1 implementation
    public void setSecret(String sec);
}
```



```
class HighClear implements ClearanceInterface {

    1 usage
    private String name;
    2 usages
    private String SECRET="some secret data";
    3 usages
    private boolean highLevel;
    5 usages
    private boolean authorized;

    public HighClear (String pname, boolean l) {
        highLevel = l;
        if (highLevel) authorized=true; else authorized=false;
    }

    @Override
    public boolean isHighLevel() {
        return highLevel;
    }
```



```java
    @Override
    public boolean isAuthorized() {
        return authorized;
    }

    @Override
    public String getName() {
        return name;
    }

    @Override
    public String getSecret() {
        if (authorized) return SECRET; else return "non-authorized";
    }

    @Override
    public void setSecret(String sec) {
        if (authorized) SECRET=sec;
    }
}
```