Q1) Suppose you are in a team developing an application for a factory: **(20 m)**

A **Factory** is a class with the following attributes:

- name (a string with at least 3 characters)
- location (a string with at least 10 characters)
- List of Workers
- List of Jobs

Each **worker** is an instance of the class '**Worker**' which has the following attributes:

- fullName (a string with at least 5 characters)
- salary (a positive number)

Each **job** is an instance of the class '**Job**' which has the following attributes:

- jobName (a string with at least 5 characters)
- JobLength (a positive number of hours)

1. Code the shells of the three classes **Factory**, **Worker**, and **Job**.
2. Code a two-parameter (name and location) constructor for the class '**Factory**'.
3. Code a two-parameter (fullName and salary) constructor for the class '**Worker**'.
4. Code a two-parameter (jobName and JobLength) constructor for the class '**Job**'.
5. For the class **Factory**, code two methods named '**addWorker**' and '**addJob**' that are responsible for creating new instances of Worker and Job classes and add them to the factory. The new methods should accept as input the required attributes for the new instances (for instance, the method '**addWorker**' should accept as input two values represent the fullName and salary attributes)

Factory class is coded as below:

```
import java.util.ArrayList;

public class Factory {
    //instance variables
    private String name;
    private String location;
    ArrayList<Worker> workers;
    ArrayList<Job>jobs;

    //constructor
    public Factory (String initName, String initLocation){
        if (validateFactory (initName, initLocation)){
            name=initName;
            location=initLocation;
        }
    }
    //helper method
    private boolean validateFactory (String rawName, String rawLocation){
        boolean result=false;
        if (rawName.length()>=3 && rawLocation.length()>=10){
            result=true;
        }
        return result;
    }

    //other methods
    public void addWorker(String fullName, double salary){
        if (Worker.validateWorker(fullName,salary)) {
            Worker newWorker = new Worker(fullName, salary);
            workers.add(newWorker);
        }
    }

    public void addJob (String jobName,double jobLength){
        if (Job.validateJob(jobName,jobLength)){
```

```
        Job newJob=new Job(jobName,jobLength);
        jobs.add(newJob);
        }
    }
}
```

Worker class is coded as below:

```
public class Worker {
    //instance variables
    private String fullName;
    private double salary;

    //constructor
    public Worker(String initFullName,double initSalary){
        if (validateWorker(initFullName,initSalary)){
            fullName=initFullName;
            salary=initSalary;
        }
    }
    //helper method
    public static boolean validateWorker(String rawFullName, double
rawSalary){
        boolean result=false;
        if (rawFullName.length()>=5 && rawSalary>0){
            result=true;
        }
        return result;
    }
}
```

Job class is coded as below:

```
public class Job {
    //instance variables
    private String jobName;
    private double jobLength;

    //constructor
    public Job(String initJobName, double initJobLength){
        if (validateJob(initJobName,initJobLength)) {
            jobName=initJobName;
            jobLength=initJobLength;
        }
    }
    //helper method
    public static boolean validateJob(String rawJobName,double
rawJobLength){
        boolean result=false;
        if (rawJobName.length()>=5 && rawJobLength()>0){
            result=true;
        }
        return result;
    }
}
```

```java
import java.util.ArrayList;

public class Factory {
    private String name;
    private String location;
    private ArrayList<Worker> workers;
    private ArrayList<Job>jobs;


    public Factory (String initName, String initLocation) {
        setLocation(initLocation);
        setName(initName);
        workers = new ArrayList<>();
        jobs = new ArrayList<>();
    }


    public boolean setLocation (String newLocation){
        boolean retVal=false;
        if (newLocation.length()>=10){
            location=newLocation;
            retVal=true;
        }
        return retVal;
    }

    public boolean setName (String newName){
        boolean retVal=false;
        if (newName.length()>3){
            name=newName;
            retVal=true;
        }
        return retVal;
    }


    public void addWorker(String fullName, double salary){
            Worker worker = new Worker(fullName, salary);
            workers.add(worker);
    }

    public void addJob (String jobName,double jobLength){
            Job job = new Job(jobName, jobLength);
            jobs.add(job);
    }
}
```

```java
public class Worker {
    private String fullName;
    private double salary;


    public Worker(String initFullName,double initSalary){
        setFullName(String initFullName);
        setSalary (double initSalary);
    }


    public boolean setFullName(String newFullName){
        boolean retVal=false;
        if (newFullName.length()>=5){
            fullName=newFullName;
            retVal=true;
        }
        return retVal;
    }

    public boolean setSalary(double newSalary){
        boolean retVal=false;
        if (newSalary>0.0){
            salary=newSalary;
            retVal=true;
        }
        return retVal;
    }
}
```

```
public class Job {
    private String jobName;
    private double jobLength;


    public Job(String initJobName, double initJobLength){
        setJobName(initJobName);
        setJobLength(initJobLength);
    }


    public boolean setJobName (String newJobName){
        boolean retVal=false;
        if (newJobName.length()>=5){
            jobName=newJobName;
            retVal=true;
        }
        return retVal;
    }

    public boolean setJobLength(double newJobLength){
        boolean retVal=false;
        if (newJobLength>0.0){
            jobLength=newJobLength;
            retVal=true;
        }
        return retVal;
    }
}
```

Q2) Code a method that takes a string as input and returns a string as output. The input string is a collection of letters, digits, and special characters without white spaces. The method should group (and return) the characters of the input string into the following categories and order: **(10m)**

- uppercase letters
- lowercase letters
- special characters
- digits

For example:

Input: "#FIT1051ws10@Sem1"

Output:"FITSwsem#@1051101"

Note: You must use the appropriate loop pattern and avoid using multiple return statements.

```
public class Test {
    public static void main(String[] args) {
        String s="145@*RaqToL/?";
        System.out.println(reorder(s));
    }

    public static String reorder(String str){
        String upper="";
        String lower="";
        String special="";
        String digit="";
        for (int i=0;i<str.length();i++){
            char character=str.charAt(i);
            //check uppercase
            if (character>='A'&&character<='Z'){
                upper+=character;
            }
            //check lowercase
            else if (character>='a'&& character<='z'){
                lower+=character;
            }
            //check digit
            else if (character>='0'&& character<='9'){
                digit+=character;
            }
            else{
                special+=character;
            }
        }
        String result=upper+lower+special+digit;
        return result;
    }
}

The output is: RTLaqo@*/?145
```