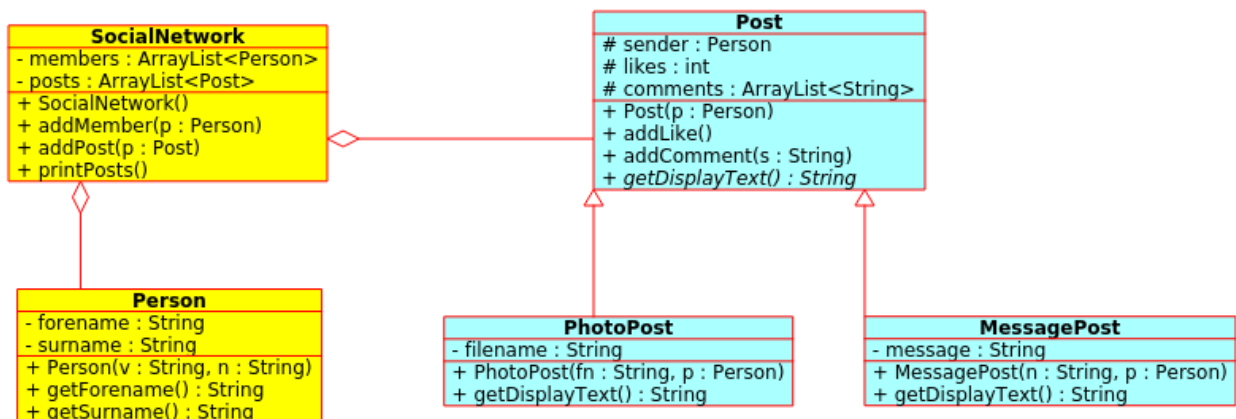


Inhaltsverzeichnis

1. OOP – SocialNetwork1.....	1
2. OOP – SocialNetwork2.....	3

1. OOP – SocialNetwork1

Part 1: Given is the following uml class diagram:



Part 2: Given is a small test program (here java):

```

public static void main(String[] args) {
    SocialNetwork schoolBook = new SocialNetwork();

    Person teacher1 = new Person("Max", "Teacher");
    Person student1 = new Person("Susi", "Student");

    schoolBook.addMember(teacher1);
    schoolBook.addMember(student1);

    MessagePost postTeacher1 = new MessagePost("What is the best coding language?",
teacher1);
    MessagePost postStudent1 = new MessagePost("I like to design software!", student1);
    PhotoPost photoPostTeacher1 = new PhotoPost("teacher1.jpg", teacher1);
    schoolBook.addPost(postTeacher1);
    schoolBook.addPost(postStudent1);
    schoolBook.addPost(photoPostTeacher1);

    postTeacher1.addLike();
    postTeacher1.addLike();
    postTeacher1.addLike();

    postStudent1.addLike();

    postTeacher1.addComment("Java, java, java");
    postTeacher1.addComment("I agree");
    postStudent1.addComment("software testing is cooooool!");
}
  
```

```
schoolBook.printPosts();  
}
```

The above code gives this output:

MESSAGE-POST:

"What is the best coding language?" by Max Teacher, 3 likes
comment: Java, java, java
comment: I agree

MESSAGE-POST:

"I like to design software!" by Susi Student, 1 likes
comment: software testing is coooool?

PHOTO-POST:

"teacher1.jpg" by Max Teacher, 0 likes

Lab: SocialNetwork1

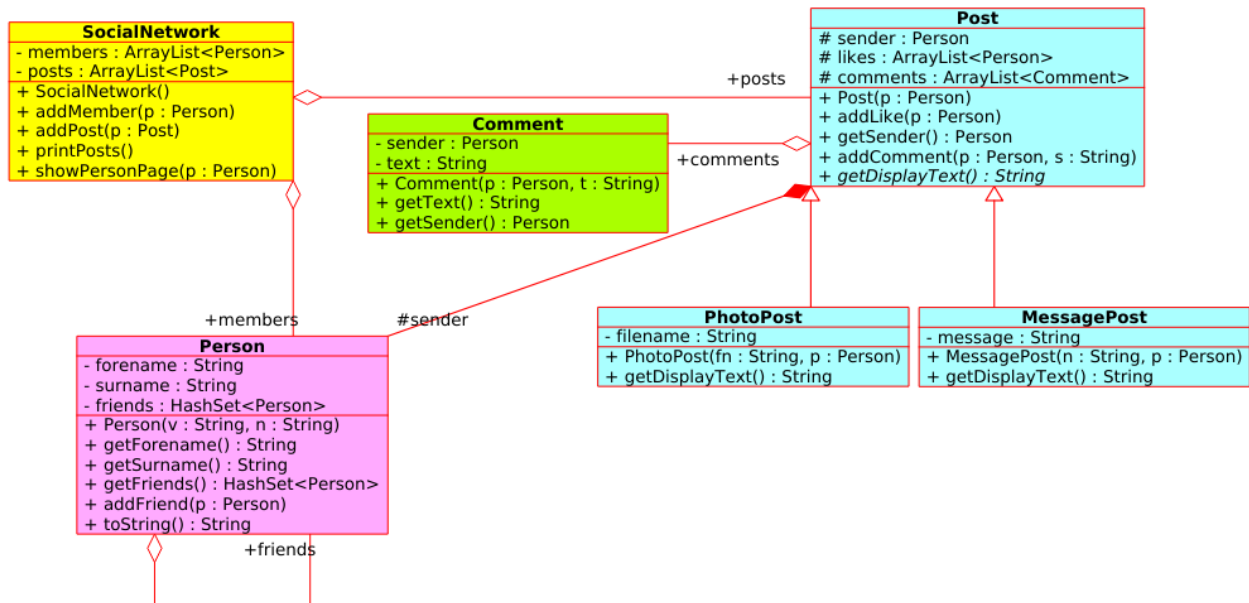
Write the classes shown in the above uml diagram and test your code.

2. OOP – SocialNetwork2

The above code has some constraints:

- members can not have friends
- we dont know who sended a comment
- we dont know who made a like

The uml-diagram:



The main procedure:

```

public static void main(String[] args) {
    // Person and friends
    Person person1 = new Person("Max", "Smith the I");
    Person person2 = new Person("Florin", "Smith the II");
    Person person3 = new Person("Chris", "Smith the III");
    Person person4 = new Person("Alex", "Smith the IV");

    person1.addFriend(person2);
    person1.addFriend(person3);

    // Posts and likes and comments
    MessagePost postPerson1 = new MessagePost("What is the best coding language?",
    person1);
    postPerson1.addLike(person2);
    postPerson1.addLike(person3);

    postPerson1.addComment(person4, "Java,c++,c ...");
    postPerson1.addComment(person2, "I agree");

    MessagePost postPerson2 = new MessagePost("I like to design software!", person2);
    postPerson2.addLike(person1);
    postPerson2.addComment(person1, "Oh no, its hardware. Thats the best!");
}
  
```

```

    PhotoPost photoPostPerson1 = new PhotoPost(person1.getSurname()+".jpg",
person1);
    // ...

    // SocialNetwork
    SocialNetwork schoolBook = new SocialNetwork();
    schoolBook.addMember(person1);
    schoolBook.addMember(person2);
    schoolBook.addPost(postPerson1);
    schoolBook.addPost(postPerson2);
    schoolBook.addPost(photoPostPerson1);

    // output
    schoolBook.showPersonPage(person1);
    System.out.println("\n");
    schoolBook.showPersonPage(person2);
}

```

The output:

PAGE of Max Smith the I

=====

Friends:

Chris Smith the III

Florin Smith the II

Posts:

MESSAGE-POST: "What is the best coding language?" by Max Smith the I, 2 likes

comment: Java,c++,c ... (Alex Smith the IV)

comment: I agree (Florin Smith the II)

PHOTO-POST: "Smith the I.jpg" by Max Smith the I, 0 likes

PAGE of Florin Smith the II

=====

Friends:

Max Smith the I

Posts:

MESSAGE-POST: "I like to design software!" by Florin Smith the II, 1 likes

comment: Oh no, its hardware. Thats the best! (Max Smith the I)

Lab: SocialNetwork2

Write the classes shown in the above uml diagram and test your code intensive.

Tipp:

```

public void addFriend(Person p) {
    friends.add(p);
    p.friends.add(this); // we are friends! (both sides)
}

```

Tipp: create class Comment to hold the comment-text and who made the comment.

Tipp:

class Post: member likes becomes an ArrayList<Person> to store the Person, who likes that post. The size of the ArrayList likes holds the number of likes.