1. Rozważamy klasy:

```
class Osoba {}
class Student extends Osoba {}
```

oraz deklaracje:

```
ArrayList<Osoba> los;
ArrayList<Student> lst;
```

   (a) Czy można przypisać `lst = los`?

   (b) Czy można przypisać `los = lst`?

2. Dodeklarowujemy

```
ArrayList<? extends Osoba> leos.
```

   (a) Czy można zrobić takie przypisanie?

```
leos = new ArrayList<? extends Osoba>()
```

   (b) Czy można zrobić przypisanie `leos = los`?

   (c) Czy można zrobić przypisanie `leos = lst`?

   (d) Czy można wstawiać elementy do `leos`?

   (e) Czy można wyjmować elementy z `leos`?

3. Dodajemy deklarację

```
ArrayList<? super Student> lsst.
```

   (a) Czy można zrobić takie przypisanie?

```
lsst = new ArrayList<? super Student>()
```

   (b) Czy można zrobić przypisanie `lsst = los`?

   (c) Czy można zrobić przypisanie `lsst = lst`?

   (d) Czy można wstawiać elementy do `lsst`?

   (e) Czy można wyjmować elementy z `lsst`?

Lista potęgowa danej listy to lista podlist (podciągów) danej listy. Na przykład lista potęgowa listy `[1,2,3]` to lista

`[ [], [1], [2], [1,2], [3], [1,3], [2,3], [1,2,3] ]`.

Dany jest interfejs listy potęgowej.

```
public interface PowerArrayList<A> {
        public Iterator<ArrayList<A>> iterator();
}
```

Uzupełnij częściową implementację interfejsu `PowerArrayList<A>`.

```
public class PowerArrayListImpl<A> implements PowerArrayList<A> {
        ArrayList<ArrayList<A>> p = new ArrayList<ArrayList<A>>();

        PowerArrayListImpl (ArrayList<A> l) {









        }


        public Iterator<ArrayList<A>> iterator() {
                return p.iterator();
        }
}
```