Practical 11

1. Sets can be thought of as lists that don't contain any repeated elements. For example, `[a,4,6]` is a set, but `[a,4,6,a]` is not (as it contains two occurrences of `a`). Write a Prolog program `subset/2` that is satisfied when the first argument is a subset of the second argument (that is, when every element of the first argument is a member of the second argument). For example:

```
subset([a,b],[a,b,c])

yes


subset([c,b],[a,b,c])

yes


subset([],[a,b,c])

yes.
```

   Your program should be capable of generating all subsets of an input set by backtracking. For example, if you give it as input

```
subset(X,[a,b,c])
```

   it should successively generate all eight subsets of `[a,b,c]`.

```
subset([],[]).

subset([X|Xs],[X|Ys]) :- subset(Xs,Ys).

subset(X,[_|Ys]) :- subset(X,Ys).
```

2. Using the subset predicate you have just written, and `findall`, write a predicate `powerset/2` that takes a set as its first argument, and returns the powerset of this set as the second argument. (The powerset of a set is the set of all its subsets.) For example:

```
powerset([a,b,c],P)
```

   should return

```
P = [[],[a],[b],[c],[a,b],[a,c],[b,c],[a,b,c]]
```

   it doesn't matter if the sets are returned in some other order. For example,

```
P = [[a],[b],[c],[a,b,c],[],[a,b],[a,c],[b,c]]
```

is fine too.

```
powerset(X,Y) :- findall(Z,subset(Z,X),Y).
```