Haskell list comprehensions - let's make some lists!

```
[ output function | input set* , predicative* ]
```

- output function: applied to members of the resulting list
- input set*: what we're generating the list from
- predicative*: conditions that apply to input set values

*can use multiple input sets and predicatives

A list of squares from 1 to 10: [1,4,9,16,25,36,49,64,81,100]

```
[ x ^ 2 | x <- [1..10] ]
```

2. add a square of each
1. take numbers 1-10

A list of squares of even numbers between 1 and 10: [4,16,36,64,100]

```
[ x ^ 2 | x <- [1..10] , rem x 2 == 0 ]
```

3. if it's even - add a square of it
1. take numbers 1-10
2. for each check if it's even

A list of squares and cubes of even numbers between 1 and 5: [4,8,16,64]

```
[ x ^ y | x <- [1..5] , y <- [2, 3] , rem x 2 == 0 ]
```

3. if it is - add first a square and then a cube of it
1. take numbers 1-5
2. for each check if it's even