Functional Languages 8th practice

1. Redefine function `take`.

```
take' 4 "hello world"     == "hell"
take' 4 ""                == ""
take' (- 2) "hello world" == ""
take' 8 [True, False]     == [True, False]
take' 0 [True, False]     == []
```

2. Redefine function `drop`.

```
drop' 4 "hello world"     == "o world"
drop' 4 ""                == ""
drop' (- 2) "hello world" == "hello world"
drop' 8 [True, False]     == []
drop' 0 [True, False]     == [True, False]
```

3. A language identifier such as `en-US` and `fr-CA` consists of two parts, language and region. Split a language identifier into two parts. The input is assumed to be well formed. Each part consists of two characters.

```
langAndRegion "en-US" == ("en", "US")
langAndRegion "en-GB" == ("en", "GB")
langAndRegion "fr-CA" == ("fr", "CA")
```

4. Redefine function `zip`.

```
zip' [] [True, False] == []
zip' [1,2,3] [] == []
zip' [1,2,3] [True, False] == [(1, True), (2, False)]
zip' "abc" [1..] == [('a', 1), ('b', 2), ('c', 3)]
```

5. Redefine function `unzip`.

```
unzip' [('a', 1), ('b', 2), ('c', 3)] == ("abc", [1,2,3])
unzip' [(1, True), (2, False)] == ([1,2], [True, False])
unzip' [] == ([],[])
```

6. Given contents of a text file, which lines are empty? Start numbering from 1. Last line does not count as empty.

   Function `lines` is helpful in separating lines.

```
empty "first line\nsecond\n\nfourth\n" == [3]
empty "theme=dark\n\ntoolbar=0\n\nicons=gnome" == [2,4]
empty "" == []
```

7. Split a list at a specific index.

```
splitAt' 0 [True, False] == ([], [True, False])
splitAt' 1 [True, False] == ([True], [False])
splitAt' 2 [] == ([], [])
splitAt' 2 "hello" == ("he", "llo")
splitAt' (- 2) "hello" == ("", "hello")
```

8. Redefine function `nub`, which retains first occurence of each element in a list.

```
nub' :: Eq a => [a] -> [ä]
nub' [1,2,2,3,2,4] == [1,2,3,4]
nub' "hello world" == "helo wrd"
nub' [] == []
```

9. Redefine function `concat`.

```
concat' [[1,2], [3,4,5]] == [1,2,3,4,5]
concat' [[1,2]] == [1,2]
concat' [] == []
concat' ["hello", "haskell"] == "hellohaskell"
```