**HW 9** **Prof. Caldwell**
**Due:** 25 September 2008 **COSC 3015**

# 1

**Exercise 1.1.** Write the function `member` exhibiting the following behavior.

```
Lists> :t member
member :: Eq a => a -> [a] -> Bool
Lists> member 1 [2,3,4,1]
True
Lists> member 1 [2,3,4]
False
Lists> member 1 [1,1,1,1]
True
Lists> member 1 []
False
Lists> ['a'..'z']
"abcdefghijklmnopqrstuvwxyz"
Lists> member 'd' ['a'..'z']
True
Lists> member 'D' ['a'..'z']
False
```

**Exercise 1.2.** Write the function `remove` exhibiting the following behavior.

```
Lists> :t remove
remove :: Eq a => a -> [a] -> [a]
Lists> remove 1 []
[]
Lists> remove 1 [2,3,4,5]
[2,3,4,5]
Lists> remove 1 [1,2,3,4]
[2,3,4]
Lists> remove 1 [1,2,3,4,1]
[2,3,4,1]
Lists> remove 1 [1,1,1,1,1]
[1,1,1,1]
Lists>
```

**Exercise 1.3.** Write the function  exhibiting the following behavior.

```
Lists> :t remove_all
remove_all :: Eq a => a -> [a] -> [a]
Lists> remove_all 1 []
[]
Lists> remove_all 1 [1,2,3]
[2,3]
Lists> remove_all 1 [1,1,2,1,1,1,3]
[2,3]
```

```
Lists> remove_all 1 [1,1,1,1,1]
[]
Lists>
```

**Exercise 1.4.** Write the function `unique` exhibiting the following behavior.

```
Lists> :t unique
unique :: Eq a => [a] -> [a]
Lists> unique []
[]
Lists> unique [1,2,3]
[1,2,3]
Lists> unique [1,2,3,1,2,3]
[1,2,3]
Lists> unique [1,2,3,3,2,1]
[3,2,1]
Lists> unique [1,2,3,3,3,3,3,2,2,2,2]
[1,3,2]
Lists>
```

**Exercise 1.5.** Write the function `stable_unique` exhibiting the following behavior.

```
Lists> :t stable_unique
stable_unique :: Eq a => [a] -> [a]
Lists> stable_unique []
[]
Lists> stable_unique [1,2,3]
[1,2,3]
Lists> stable_unique [1,2,3,1,2,3]
[1,2,3]
Lists> stable_unique [1,2,3,3,2,1]
[1,2,3]
Lists> stable_unique [1,2,3,3,3,3,3,2,2,2,2]
[1,2,3]
```

**Exercise 1.6.** Write the function `count` exhibiting the following behavior.

```
Lists> :t count
count :: Eq a => a -> [a] -> Int
Lists> count 1 []
0
Lists> count 1 [1,2,3]
1
Lists> count 1 [2,3]
0
```

```
Lists> count 1 [1,1,1,2,2,2,3,3,3]
3
Lists>
```