Question 2: "Best Behaviour"

Use structural induction to prove `(mylength x) - (mylength (filterPQ x)) = (countIf x)`. Use the following implementations for `myLength`, `filterPQ`, `beforeP`, `afterQ`, and `countIf` and refer to the individual lines in these implementations using the labels `L1, L2, F1, F2A, F2B, F2C, B1, A1, C1, C2A, C2B`, and `C2C`. You must follow the process of structural induction as it was demonstrated in class and you must show all your work (including the line applied during each step of your equational reasoning).

```
          myLength :: [Char] -> Int
L1        myLength [] = 0
L2        myLength (h:t) = 1 + (myLength t)

          filterPQ :: [Char] -> [Char]
F1        filterPQ [] = []
          filterPQ (h:t)
F2A         | beforeP h = h : (filterPQ t)
F2B         | afterQ h = h : filterPQ t
F2C         | otherwise = filterPQ t

          beforeP :: Char -> Bool
B1        beforeP x = ord x < 80

          afterQ :: Char -> Bool
A1        afterQ x = ord x > 81

          countIf :: [Char] -> Int
C1        countIf [] = 0
          countIf (h:t)
C2A         | h == "P" = 1 + (countIf t)
C2B         | h == "Q" = 1 + (countIf t)
C2C         | otherwise = countIf t
```

Base Case:

Prove `(mylength []) - (mylength (filterPQ [])) = (countIf [])`

LHS `(mylength []) - (mylength (filterPQ []))`

`= (mylength []) - (mylength [])` by F1

`= 0`

RHS `(countIf [])`

`= 0` by C1

Inductive Case:

Prove `(mylength t) - (mylength (filterPQ t)) = (countIf t)`

(via Direct Proof) →

`(mylength (h:t)) - (mylength (filterPQ (h:t))) = (countIf (h:t))`

Inductive Assumption `(mylength t) - (mylength (filterPQ t)) = (countIf t)`

Case 1: the head, h, is a character before 'P' (e.g., 'A')
(and, thus, `beforeP a = ord 'A' < 80 = 65 < 80 = True`)

LHS

```
(mylength (h:t)) - (mylength (filterPQ (h:t)))
= (mylength (h:t)) - (mylength (h : filterPQ t))          by F2A
= (1 + mylength t) - (mylength (h : filterPQ t))          by L2
= (1 + mylength t) - (1 + mylength (filterPQ t))          by L2
= mylength t - mylength (filterPQ t)                      by L2
= countIf t                                               by IA
```

RHS

```
(countIf (h:t))
= countIf t                                               by C2C
```

Case 2: **the head, h, is a character after 'Q' (e.g., 'Z')**
(and, thus, `afterQ a = ord 'Z' > 81 = 90 > 81 = True`)

LHS

```
(mylength (h:t)) - (mylength (filterPQ (h:t)))
= (mylength (h:t)) - (mylength (h : filterPQ t))                    by F2B
= (1 + mylength t) - (mylength (h : filterPQ t))                    by L2
= (1 + mylength t) - (1 + mylength (filterPQ t))                    by L2
= mylength t - mylength (filterPQ t)                                by L2
= countIf t                                                         by IA
```

RHS

```
(countIf (h:t))
= countIf t                                                         by C2C
```

Case 3: **the head, h, is 'P'**
(and, thus, `h == 'P' = True`)

LHS

```
(mylength (h:t)) - (mylength (filterPQ (h:t)))
= (mylength (h:t)) - (mylength (filterPQ t))                        by F2C
= (1 + mylength t) - (mylength (filterPQ t))                        by L2
= 1 + (mylength t - mylength (filterPQ t))                          by L2
= 1 + countIf t                                                     by IA
```

RHS

```
(countIf (h:t))
= 1 + countIf t                                                     by C2A
```

Case 4: the head, h, is 'Q'

(and, thus, `h == 'Q' = True`)

LHS

```
(mylength (h:t)) - (mylength (filterPQ (h:t)))
= (mylength (h:t)) - (mylength (filterPQ t))                    by F2C
= (1 + mylength t) - (mylength (filterPQ t))                    by L2
= 1 + (mylength t - mylength (filterPQ t))                      by L2
= 1 + countIf t                                                 by IA
```

RHS

```
(countIf (h:t))
= 1 + countIf t                                                 by C2B
```