Problem 1: length(elements) == size

base case: elements := Nil[A] =>
=> length(elements) == 0
size := 0
=>
=> length(elements) == size

step case:

We consider length(elements) == size

-> for push: length(elements) increases by one
size := size + 1
=>
=> length(elements) == size

-> for pop: • If the list is empty nothing changes
• If the list is not empty:
length(elements) decreases by one
size := size - 1

=> length(elements) == size

Scanned by CamScanner

Problem 2:

a) Weak class invariant: today.yesterday().tomorrow() = today

```
fun yesterday(): Date = {
    if (day > 1)
        return new Date(year, month, day-1);
    if (month > 1)
        if (month % 2)
            if (month == 3)
                if (year % 4)
                    return new Date(year, 2, 28);
                return new Date(year, 2, 29);
            return new Date(year, month-1, 30);
        return new Date(year, month-1, 31);
    return new Date(year-1, 12, 31);
}
```

```
fun tomorrow(): Date = {
    if (month == 2 && day == 29)
        return new Date(year, 3, 1);
    if (month == 2 && day == 28 && year % 4)
        return new Date(year, 3, 1);
    if (day < 30)
        return new Date(year, month, day+1);
    if (day == 31)
        if (month == 12)
            return new Date(year+1, 1, 1);
        return new Date(year, month+1, 1);
    if (month % 2)
        return new Date(year, month, 31);
    return new Date(year, month+1, 1);
}
```

Scanned by CamScanner

b) X : Priority Queue

x. enqueue(n). dequeue() = n (strong class invariant)

```
fun enqueue (x: int): unit = {
    data = data :: x;
}
```

Scanned by CamScanner

Problem 5.3:

Base Case:

$0 + n = n$ - zero_left
$n + 0 = n$ - zero_right

$\Rightarrow 0 + n = n + 0$

Step Case:

Let's assume that $m + n = n + m$

then:

$S(m) + n = S(m+n)$ - succ_left
$n + S(m) = S(n+m)$ - succ_right
$m + n = n + m \Rightarrow S(m+n) = S(n+m)$

$\Rightarrow S(m) + n = n + S(m)$

$\Rightarrow m + n = n + m$, $\forall\ n, m \in \mathbb{N}$

Scanned by CamScanner