

Quiz Section 5: Reasoning – Solutions

The problems that follow make use of the following inductive type, representing lists of integers

$$\mathbf{type} \text{ List} := \text{ nil} \mid \text{ cons}(\text{hd} : \mathbb{Z}, \text{tl} : \text{List})$$

Below, we will also use the function `sum`, which returns the sum of the integers in the list:

$$\text{sum} : \text{List} \rightarrow \mathbb{Z}$$
$$\text{sum}(\text{nil}) \quad := \quad 0$$
$$\text{sum}(a :: L) \quad := \quad a + \text{sum}(L)$$

the function `twice`, which doubles each number in the list:

$$\text{twice} : \text{List} \rightarrow \text{List}$$
$$\text{twice}(\text{nil}) \quad := \quad \text{nil}$$
$$\text{twice}(a :: L) \quad := \quad 2a :: \text{twice}(L)$$

the functions `twice-evens` and `twice-odds`, which double the integers at even and odd indexes in the list:

$$\text{twice-evens} : \text{List} \rightarrow \text{List}$$
$$\text{twice-evens}(\text{nil}) \quad := \quad \text{nil}$$
$$\text{twice-evens}(a :: L) \quad := \quad 2a :: \text{twice-odds}(L)$$
$$\text{twice-odds} : \text{List} \rightarrow \text{List}$$
$$\text{twice-odds}(\text{nil}) \quad := \quad \text{nil}$$
$$\text{twice-odds}(a :: L) \quad := \quad a :: \text{twice-evens}(L)$$

and the function `swap`, which swaps adjacent integers in the list:

$$\text{swap} : \text{List} \rightarrow \text{List}$$
$$\text{swap}(\text{nil}) \quad := \quad \text{nil}$$
$$\text{swap}(a :: \text{nil}) \quad := \quad a :: \text{nil}$$
$$\text{swap}(a :: b :: L) \quad := \quad b :: a :: \text{swap}(L)$$

and the function `len`, which finds the length of the list:

$$\text{len} : \text{List} \rightarrow \mathbb{Z}$$
$$\text{len}(\text{nil}) \quad := \quad 0$$
$$\text{len}(a :: L) \quad := \quad 1 + \text{len}(L)$$

Task 1 – Twice Things Up

You see the following snippet in some TypeScript code. It uses `cons` and `nil`, which are TypeScript implementations of “cons” and “nil”, and also `equal`, which is a TypeScript implementation of “=” on lists.

```
if (equal(L, cons(1, cons(2, nil)))) {
  const R = cons(2, cons(4, nil)); // = twice(L)
  return cons(0, R);             // = twice(cons(0, L))
}
```

The comments show the definition of what *should* be returned (the specification), but the code is *not* a direct translation of those. Below, we will use reasoning to prove that the code is correct.

- (a) Using the fact that $L = 1::2::\text{nil}$, prove by calculation that $\text{twice}(L) = R$, where R is the constant list defined in the code. I.e., prove that

$$\text{twice}(L) = 2::4::\text{nil}$$

$$\begin{aligned} \text{twice}(L) & \\ &= \text{twice}(1::2::\text{nil}) && \text{Def of } L \\ &= 2::\text{twice}(2::\text{nil}) && \text{Def of twice} \\ &= 2::4::\text{twice}(\text{nil}) && \text{Def of twice} \\ &= 2::4::\text{nil} && \text{Def of twice} \end{aligned}$$

- (b) Using the facts that $L = 1::2::\text{nil}$ and $R = 2::4::\text{nil}$, prove by calculation that the code above returns the correct value, i.e., prove that

$$\text{twice}(0 :: L) = 0 :: R$$

Feel free to cite part (a) in your calculation.

$$\begin{aligned} \text{twice}(0 :: L) & \\ &= 0 :: \text{twice}(L) && \text{Def of twice} \\ &= 0 :: 2 :: 4 :: \text{nil} && \text{Part (a)} \\ &= 0 :: R && \text{Def of } R \end{aligned}$$

Task 2 – It’s Raining Len

You see the following snippet in some TypeScript code. It uses `twice_evens`, which is a TypeScript implementation of twice-evens from the previous problem, as well as `len` from before.

```
return 2 + len(twice_evens(L)); // = len(twice-evens(cons(3, cons(4, L))))
```

The comment shows the definition of what should be returned (the specification), but the code is not a direct translation of that. Below, we will use reasoning to prove that the code is correct.

(a) Let a and b be any integers. Prove by calculation that

$$\text{len}(\text{twice-evens}(a :: b :: L)) = 2 + \text{len}(\text{twice-evens}(L))$$

$$\begin{aligned} \text{len}(\text{twice-evens}(a :: b :: L)) & \\ &= \text{len}(2a :: \text{twice-odds}(b :: L)) && \text{Def of twice-evens} \\ &= \text{len}(2a :: b :: \text{twice-evens}(L)) && \text{Def of twice-odds} \\ &= 1 + \text{len}(b :: \text{twice-evens}(L)) && \text{Def of len} \\ &= 1 + 1 + \text{len}(\text{twice-evens}(L)) && \text{Def of len} \\ &= 2 + \text{len}(\text{twice-evens}(L)) \end{aligned}$$

(b) Explain why the calculation from part (a) shows that the code is correct according to the specification (written in the comment).

Applying part (a) with $a = 3$ and $b = 4$ gives us a proof that

$$\text{len}(\text{twice-evens}(3 :: 4 :: L)) = 2 + \text{len}(\text{twice-evens}(L))$$

which says that the code is correct.

Task 3 – Swapaholic

Prove by cases that $\text{swap}(a :: L) \neq \text{nil}$ for any integer $a : \mathbb{Z}$ and list L .

Let a be any integer and L be any list. We argue by cases on L .

First, suppose that $L = \text{nil}$. Then, we can see that

$$\begin{aligned} \text{swap}(a :: L) & \\ &= \text{swap}(a :: \text{nil}) \quad \text{Def of } L \\ &= a :: \text{nil} \quad \text{Def of swap} \\ &\neq \text{nil} \end{aligned}$$

Next, suppose that $L \neq \text{nil}$. That means that $L = b :: R$ for some $b : \mathbb{Z}$ and $R : \text{List}$. Thus, we have

$$\begin{aligned} \text{swap}(a :: L) & \\ &= \text{swap}(a :: b :: R) \quad \text{Def of } L \\ &= \text{swap}(b :: a :: R) \quad \text{Def of swap} \\ &\neq \text{nil} \end{aligned}$$

We have proven the claim in both cases. Since those cases are exhaustive, we have proven it in general.

Task 4 – Here Comes the Sum

You see following snippet in some TypeScript code:

```
const s = sum(L);  
...  
return 2 * s; // = sum(twice(L))
```

This code claims to calculate the answer $\text{sum}(\text{twice}(L))$, but it actually returns $2 \text{sum}(L)$. Prove this code is correct by showing that $\text{sum}(\text{twice}(S)) = 2 \text{sum}(S)$ holds for any list S by structural induction.

Define $P(S)$ to be the claim that $\text{sum}(\text{twice}(S)) = 2 \text{sum}(S)$. We will prove the claim by structural induction.

Base Case (nil). We can calculate

$$\begin{aligned} \text{sum}(\text{twice}(\text{nil})) & \\ &= \text{sum}(\text{nil}) && \text{Def of twice} \\ &= 0 && \text{Def of sum} \\ &= 2 \cdot 0 \\ &= 2 \cdot \text{sum}(\text{nil}) && \text{Def of sum} \end{aligned}$$

Inductive Hypothesis. Suppose that $P(L)$ holds for a list L . I.e., suppose that $\text{sum}(\text{twice}(L)) = 2 \text{sum}(L)$.

Inductive Step. We need to show $P(a :: L)$ for any integer $a : \mathbb{Z}$.

Let a be any integer. Then, we can calculate

$$\begin{aligned} \text{sum}(\text{twice}(a :: L)) & \\ &= \text{sum}(2a :: \text{twice}(L)) && \text{Def of twice} \\ &= 2a + \text{sum}(\text{twice}(L)) && \text{Def of sum} \\ &= 2a + 2 \text{sum}(L) && \text{Inductive Hypothesis} \\ &= 2(a + \text{sum}(L)) \\ &= 2 \text{sum}(a :: L) && \text{Def of sum} \end{aligned}$$

Conclusion. $P(S)$ holds for any list S by structural induction.

Task 5 – Can You Sum a Few Bars?

Prove that

$$\text{sum}(\text{twice-evens}(L)) + \text{sum}(\text{twice-odds}(L)) = 3 \text{sum}(L)$$

holds for any list S by structural induction.

Define $P(S)$ to be the claim that $\text{sum}(\text{twice-evens}(S)) + \text{sum}(\text{twice-odds}(S)) = 3 \text{sum}(S)$. We will prove the claim by structural induction.

Base Case (nil). We can calculate

$$\begin{aligned} & \text{sum}(\text{twice-evens}(\text{nil})) + \text{sum}(\text{twice-odds}(\text{nil})) \\ &= \text{sum}(\text{nil}) + \text{sum}(\text{twice-odds}(\text{nil})) && \text{Def of twice-evens} \\ &= \text{sum}(\text{nil}) + \text{sum}(\text{nil}) && \text{Def of twice-odds} \\ &= \text{sum}(\text{nil}) + 0 && \text{Def of sum} \\ &= 0 && \text{Def of sum} \\ &= 3 \cdot 0 \\ &= 3 \text{sum}(\text{nil}) && \text{Def of sum} \end{aligned}$$

Inductive Hypothesis. Suppose that $P(L)$ holds for a list L . I.e., suppose that $\text{sum}(\text{twice-evens}(L)) + \text{sum}(\text{twice-odds}(L)) = 3 \text{sum}(L)$

Inductive Step. We need to show $P(a :: L)$ for any integer $a : \mathbb{Z}$.

Let a be any integer. Then, we can calculate

$$\begin{aligned} & \text{sum}(\text{twice-evens}(a :: L)) + \text{sum}(\text{twice-odds}(a :: L)) \\ &= \text{sum}(2a :: \text{twice-odds}(L)) + \text{sum}(\text{twice-odds}(a :: L)) && \text{Def of twice-evens} \\ &= 2a + \text{sum}(\text{twice-odds}(L)) + \text{sum}(\text{twice-odds}(a :: L)) && \text{Def of sum} \\ &= 2a + \text{sum}(\text{twice-odds}(L)) + \text{sum}(a :: \text{twice-evens}(L)) && \text{Def of twice-odds} \\ &= 2a + \text{sum}(\text{twice-odds}(L)) + a + \text{sum}(\text{twice-evens}(L)) && \text{Def of sum} \\ &= 3a + \text{sum}(\text{twice-evens}(L)) + \text{sum}(\text{twice-odds}(L)) \\ &= 3a + 3 \text{sum}(L) && \text{Inductive Hypothesis} \\ &= 3(a + \text{sum}(L)) \\ &= 3 \text{sum}(a :: L) && \text{Def of sum} \end{aligned}$$

Conclusion. $P(S)$ holds for any list S by structural induction.