Jérémie Poisson McGill ID : 260627104
School of Computer Science - McGill University September 24 2015

Question 2.3 : Prove using induction that your functions satisfy the following property:

```
unzip (zip l1 l2) = (l1, l2)
```

Let l1 and l2 be two lists defined in OCaml as follows :

```
l1 = [A0; A1; ...; An] = hl1::tl1
l2 = [B0; B1; ...; Bn] = hl2::tl2
```

... for n elements

In order to consciously prove the statement above, it is useful to prove the *zip* expression at first.

Theorem 1:

```
zip l1 l2 = [(A0, B0); (A1, B1); ...; (An; Bn)]
```

Proof. by structural induction.

Base Case: l1 = [] and l2 = [] where n = 0

```
zip [] []
=> []                (by program)
```

whereas n == 0; which is to be expected

Step Case: l1 = hl1::tl1 and l2 = hl2::tl2
IH1 : let's assume the following:

```
zip tl1 tl2 = [(A1, A2); ...; (An-1, Bn-1)]
```

Where there is *n - 1* elements in *both* lists

```
zip hl1::tl1 hl2::tl2
=> (hl1, hl2) :: zip tl1 tl2                     (by program)
=> (hl1, hl2) :: [(A1, A2); ...; (An-1, Bn-1)]   (by IH1)
=> (A0, B0) :: [(A1, A2); ...; (An-1, Bn-1)]     (by definition)
== [(A0, B0); (A1, B1); ...; (An; Bn)]
```

where there is clearly *n* elements.

■

We have now successfully demonstrated the *zip* expression by induction. It now somewhat easier to demonstrate the *unzip* expression to answer the question as consciously as possible.

We are still considering two lists, l1 and l2, defined in OCaml as follows :

```
l1 = [A0; A1; ...; An] = hl1::tl1
l2 = [B0; B1; ...; Bn] = hl2::tl2
```

... for n elements

Theorem 2:

```
unzip (zip l1 l2) = (l1, l2) = ([A0; A1; ...; An], [B0; B1; ...; Bn])
```

... for n elements

Proof. by structural induction.

Base Case: l1 = [] and l2 = [] where n = 0

```
unzip (zip [] [])
=> unzip ([])                (from base case in theorem 1)
=> ([], [])                  (from program)
```

whereas n ==0; which is to be expected

Step Case: l1 = hl1::tl1 and l2 = hl2::tl2
IH2 : let's assume the following:

```
unzip (zip tl1 tl2)
= unzip [(A1, B1); ...; (An-1; Bn-1)]                (from theorem 1)
= ([A1; ...; An-1], [B1; ...; Bn-1])
```

Where there is *n - 1* elements in *both* lists

```
unzip (zip hl1::tl1 hl2::tl2) = unzip (zip l1 l2)
=> unzip ([(A0, B0); (A1, B1); ...; (An, Bn)])                (from theorem 1)
=> (A0::left_of_pair (unzip [(A1, B1); ...; (An-1; Bn-1)]),
    B0::right_of_pair (unzip [(A1, B1); ...; (An-1; Bn-1)]))    (by program)
=> (A0::left_of_pair ([A1; ...; An-1], [B1; ...; Bn-1]),
    B0::right_of_pair ([A1; ...; An-1], [B1; ...; Bn-1]))       (from IH2)
=> (A0::[A1; ...; An-1], B0::[B1; ...; Bn-1])
== (hl1::tl1, hl2::tl2) = ([A0; A1; ...; An], [B0; B1; ...; Bn]) = (l1, l2)
```

where there is *n* elements in both lists

■