Homework 1: Dependent Types

Due No Due Date **Points** None

This homework is ungraded. Feel free to ask and answer questions via Canvas, but don't post complete solutions.

Consider again the type `Single` from the slides and the function `sum`. We can rewrite `sum` making the value on which the type depends an implicit parameter. (I call the function `bsum` to avoid naming conflicts with the Idris predefined function `sum`.)

```
bsum : {b : Bool} -> Single b -> Nat
bsum {b = True}  x       = x
bsum {b = False} []      = 0
bsum {b = False} (x::xs) = x + bsum {b = False} xs
```

We can use `sum` as follows.

```
*Idris> bsum {b=True} 4
4 : Nat
*Idris> bsum {b=False} [3,4]
7 : Nat
```

We need to explicitly mention the value `b` in these examples since Idris can't reconstruct automatically the `Single` types from the arguments. However, if we add value definitions to the Idris program with corresponding type declarations, such as this:

```
xs : Single False
xs = [3,4]
```

we can apply `sum` without mentioning the implicit parameter.

```
*Idris> bsum xs
7 : Nat
```

Now consider the following dependent type definition for arbitrary tuple types.

```
Tuple : Nat -> Type
Tuple Z     = Nat
Tuple (S n) = (Nat,Tuple n)
```

Here are a few example values of that type.

```
t0 : Tuple 0
t0 = 2

t3 : Tuple 2
t3 = (2,3,4)

t4 : Tuple 3
t4 = (2,3,4,5)
```

(a) Define the function `first` that extracts the first component of an arbitrary tuple of type `Tuple`.

```
*Idris> first t3
2 : Nat
*Idris> first t4
2 : Nat
```

(b) Define the function `lst` that extracts the last component of an arbitrary tuple of type `Tuple`.

```
*Idris> lst t3
4 : Nat
*Idris> lst t4
5 : Nat
```

(c) Define the function `project` that extracts a specific component from an arbitrary tuple of type `Tuple`. The function should have the following type.

```
project : Nat -> Tuple k -> Nat
```

Here some examples.

```
*Idris> project 1 t3
3 : Nat
*Idris> project 2 t3
4 : Nat
*Idris> project 0 t4
2 : Nat
```

Note: The definition of `project` requires four cases.