Chapter 1

Library fmcs_a2_1

Q3.1. We will do monkey stuff here!

```
Require Import List.
Inductive binop : Set := Plus | Times.
Inductive exp : Set :=
  | Const : nat → exp
  | Binop : binop → exp → exp → exp.
Definition binopDenote (b:binop) : nat → nat → nat :=
  match b with
  | Plus ⇒ plus
  | Times ⇒ mult
  end.
Fixpoint expDenote (e:exp) : nat :=
  match e with
  | Const n ⇒ n
  | Binop b e1 e2 ⇒ (binopDenote b) (expDenote e1) (expDenote e2)
  end.
Inductive instr : Set :=
  | iConst : nat → instr
  | iBinop : binop → instr.
Definition prog := list instr.
Definition stack := list nat.
Definition instrDenote (i : instr) (s : stack) : option stack :=
  match i with
  | iConst n ⇒ Some (n :: s)
  | iBinop b ⇒ match s with
                | arg1 :: arg2 :: s' ⇒ Some ((binopDenote b) arg1 arg2 :: s')
                | _ ⇒ None
                end
  end.
```

```
Fixpoint progDenote (p : prog) (s : stack) : option stack :=
  match p with
  | nil ⇒ Some s
  | i :: p' ⇒ match instrDenote i s with
                | None ⇒ None
                | Some s' ⇒ progDenote p' s'
                end
  end.
```

1.1 Question 1: Modified compiler and proof.

We would like to port the compiler to another stack machine whose behavior is slightly different from the original one. Although the representation of its structure remains the same (`Definition` *prog* := *list instr* and `Definition` *stack* := *list nat*), the new stack machine's interpretation of instructions is slightly different:

```
Definition instrDenote' (i : instr) (s : stack) : option stack :=
  match i with
  | iConst n ⇒ Some (n :: s)
  | iBinop b ⇒ match s with
                | arg2 :: arg1 :: s' ⇒ Some ((binopDenote b) arg1 arg2 :: s')
                | _ ⇒ None
                end
  end.
```

The instrDenote' function assumes that the second operand at the stack top while instrDenote assumes the first one at the top.

Given this modified instrDenote' function, try to modifiy the implementation of the compiler so that it suit the new definition and prove its correctness.