Chapter 1

Library clodomir_poly

Capítulo 4 - Polymorphism and Higher-Order Functions (Poly)

Listas polimórficas

```
Inductive list (X : Type) : Type :=
  | nil : list X
  | cons : X → list X → list X.
```

Função Repetir

```
Fixpoint repeat (X : Type) (x : X) (count : nat) : list X :=
  match count with
  | 0 ⇒ nil X
  | S count' ⇒ cons X x (repeat X x count')
  end.

Example test_repeat1 : repeat nat 4 2 = cons nat 4 (cons nat 4 (nil nat)).

Example test_repeat2 : repeat bool false 1 = cons bool false (nil bool).
```

Exercise: 2 stars (mumble_grumble)

```
Inductive mumble : Type :=
  | a : mumble
  | b : mumble → nat → mumble
  | c : mumble.

Inductive grumble (X : Type) : Type :=
  | d : mumble → grumble X
  | e : X → grumble X.
```

Check d (b a 5).

```
Check d mumble (b a 5).

Check d bool (b a 5).

Check e bool true.
```

```
Check e mumble (b c 0).
```

Check e bool (b c 0).

```
Check c.
```

Função Repetir

```
Fixpoint repeat' X x count : list X :=
  match count with
  | 0 ⇒ nil X
  | S count' ⇒ cons X x (repeat' X x count')
  end.

Example test_repeat'1 : repeat' nat 4 2 = cons nat 4 (cons nat 4 (nil nat)).

Example test_repeat'2 : repeat' bool false 1 = cons bool false (nil bool).

Fixpoint repeat'' X x count : list X :=
  match count with
  | 0 ⇒ nil _
  | S count' ⇒ cons _ x (repeat'' _ x count')
  end.

Example test_repeat''1 : repeat'' nat 4 2 = cons nat 4 (cons nat 4 (nil nat)).

Example test_repeat''2 : repeat'' bool false 1 = cons bool false (nil bool).
```

Definições

```
Definition list123 := cons nat 1 (cons nat 2 (cons nat 3 (nil nat))).

Definition list123' := cons _ 1 (cons _ 2 (cons _ 3 (nil _))).
```

Argumentos

```
Definition list123'' := cons 1 (cons 2 (cons 3 nil)).
```

Função Repetir

```
Fixpoint repeat''' {X : Type} (x : X) (count : nat) : list X :=
  match count with
  | 0 ⇒ nil
  | S count' ⇒ cons x (repeat''' x count')
  end.
```