Learning to Program with F#
Exercises
Department of Computer Science
University of Copenhagen

Jon Sporring, Martin Elsman, Torben Mogensen, Christina Lioma

October 21, 2022

0.1 Cards

0.1.1 Teacher's guide

0.1.2 Introduction

We will use the following three types to implement various functions relating to cards.

```
type suit = Hearts | Diamonds | Clubs | Spades // The suit of a card

type rank = Two | Three | Four | Five | Six    // The rank of a card
          | Seven | Eight | Nine | Ten
          | Jack | Queen | King | Ace

type card = rank * suit           // Combination of a rank and a suit
```

0.1.3 Exercise(s)

0.1.3.1: Write a function `highCard : card -> card -> card` that takes two cards as arguments and returns the card with the highest rank. In case the cards have the same rank, the function should return the first argument. **Hint:** You can use the >= operator to compare ranks.

0.1.3.2: Using recursion and pattern matching, write a function `initDeck : unit -> card list` that returns a full deck of cards. Check that the call `initDeck()` returns a list of length 52. **Hint:** Implement a recursive helper function that takes a card c as argument and uses pattern matching on the result of calling `succCard` on c. Use the card `(Two,Hearts)` in the initial call to your helper function.

0.1.3.3: Write a function `sameRank : card -> card -> bool` that checks that the two argument cards have the same rank.

0.1.3.4: Write a function `sameSuit : card -> card -> bool` that checks that the two argument cards are of the same suit.

0.1.3.5: Write a function `succCard : card -> card option` that takes a card as argument and returns the next card as an optional value. To implement the function, use a `match` construct and the two functions `succSuit` and `succRank`.

The call `succCard (Ace, Spades)` should return `None`. If `succRank` returns `None` and `succSuit` returns `Some s`, where s is a suit, `succCard` should return the value `Some (Two,`s`)`.

0.1.3.6: Write a function `succRank : rank -> rank option` that takes a rank as argument and returns the next rank as an optional value. The call `succRank Two` should return the value `Some Three`. The call `succRank Ace` should return the value `None`.

0.1.3.7: Write a function `succSuit : suit -> suit option` that takes a suit as argument and returns the next suit as an optional value. The call `succSuit Hearts` should return the value `Some Diamonds`. The call `succSuit Spades` should return `None`.