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 Cat

0.1.1 Teacher's guide

Emne working with files

Sværhedsgrad Easy

0.1.2 Introduction

The program `cat` is a UNIX-program, which concatenates (i.e. joins) files. The program exists on both Linux and macOS. When passing two text files to `cat`, e.g. `a.txt` and `b.txt`, then the program prints the contents of file `a.txt` followed by the contents of `b.txt` to the screen. Consider an inverse version of `cat`, called `tac`, which prints the files in reverse order and prints each file from the last to the first character. For example, if the file `a.txt` contains the characters `abc\ndef\n` and the file `b.txt` contains the characters `123\n456\n` with `\n` being the newline character, then

```
cat a.txt b.txt
```

will output `abc\ndef\n123\n456\n` to the screen. In contrast,

```
tac a.txt b.txt
```

will output `654\n321\nfed\ncba\n` to the screen.

In the following assignments you are to write a (functional) implementation of `cat` and `tac` in F#.

0.1.3 Exercise(s)

0.1.3.1: Make the library `readNWrite.fs` with the function,

```
readFile : filename:string -> string option
```

which takes a filename and returns the contents of the text file as a `string option`. If the file does not exist, the function should return `None`.

0.1.3.2: Make a function,

```
printFile : filenname:string -> bool
```

which prints the content of the file with the name `filename` to the screen. If no error occurs, then the function must return `true`, and otherwise `false`. The function should be placed in the implementation-file `readNWrite.fs`.

0.1.3.3: First extend the library `readNWrite.fs` with a function,

```
cat : filenames:string list -> string option
```

which takes a list of filenames. The function should use `readFile` (Exercise 1) to read the contents of the files. The contents of the files should be merged into a single `string option`, which the function returns. If any of the files do not exist, then the function should return `None`.

Then write an application, `cat`, which takes a list of filenames as command-line arguments, calls the `cat` function with this list and prints the resulting string to the screen. The program must return 1 in case of an error and 0 otherwise.

0.1.3.4: First extend the library `readNWrite.fs` with a function,

```
tac : filenames:string list -> string option
```

which takes a list of files, reads their content with `readFile` (Exercise 1), reverses the order of each file in a line-by-line manner and reverses each line (i.e. the opposite of `cat`) and concatenates the result. If any of the files do not exist, then the function should return `None`.

Then write an application, `tac`, which takes a list of filenames as command-line arguments, calls the `tac` function with this list and prints the resulting string to the screen. The program must return 0 or 1 depending on whether the operation was successful or not.