EDAN40: Functional Programming
Assignment 1F: Verify Sudoku
Assignment 1N: Chatterbots

Jacek Malec
Dept. of Computer Science, Lund University, Sweden
April 6th, 2020

Legend:
EDAF95: 5hp, G2, programming focus
EDAN40: 7.5hp, A, theory as well
Fi: lectures for all
Xi: lectures for EDAN40
Öi: classes for all
A1F, A2F: assignments for EDAF95
A1N, A2N, A3N: assignments for EDAN40

Programming partner search

- Go to canvas
- If you are alone, register into an assignment group as first member, then
- Wait for contact, or
- Contact someone else registered alone in some other group.
- You should be able to move yourself between groups.

folds and infinite arguments

\[
\begin{align*}
\text{foldl} &:: (a -> b -> a) -> a -> [b] -> a \\
\text{foldl} f z [] &= z \\
\text{foldl} f z (x:xs) &= \text{foldl} f (f z x) xs \\
\text{foldr} &:: (a -> b -> b) -> b -> [a] -> b \\
\text{foldr} f z [] &= z \\
\text{foldr} f z (x:xs) &= f x (\text{foldr} f z xs)
\end{align*}
\]
**folds and infinite arguments**

Consider folding a list of \( n \) values \([x_1, x_2, x_3, x_4 \ldots x_n]\) with some function \( f \) and seed \( z \).

**foldl** is:
- Left associative: \( f \ldots (f (f (f (f z x_1) x_2) x_3) x_4) \ldots) x_n \)
- Tail recursive: It iterates through the list, producing the value afterwards
- Lazy: Nothing is evaluated until the result is needed
- Backwards: \( \text{foldl} \ (\text{flip} \ (\_)) \ [] \) reverses a list.

Consider folding a list of \( n \) values \([x_1, x_2, x_3, x_4 \ldots x_n]\) with some function \( f \) and seed \( z \).

**foldr** is:
- Right associative: \( f x_1 \ldots (f x_3 (f x_4 \ldots (f x_n z) \ldots)) \)
- Recursive into an argument: Each iteration applies \( f \) to the next value and the result of folding the rest of the list.
- Lazy: Nothing is evaluated until the result is needed
- Forwards: \( \text{foldr} \ (\_ \ []) \) returns a list unchanged.

**IO and Random**

A fragment of Chatterbot.hs:

```haskell
stateOfMind :: BotBrain -> IO (Phrase -> Phrase)
{- TO BE WRITTEN -}
stateOfMind _ = return id

return is of type:

```
return :: a -> m a
```

while in System.Random you have

```haskell
randomIO :: IO n
```

where \( n \) is an (almost) arbitrary numeric type.

**Sudoku (F1)**

```
2 6 3 7
5 1 4
6 5 2

4 8 1 5
9 6 7
8 1 3

1 2 4 5
3 7 6 1
```

Jacek Malec, http://rss.cs.lth.se
**The task (F1)**

All solutions are easily found on the net. Please don’t use them.

**Learn!**

- Read the task description and make sure you understand it. What do you need to verify?
- Build up your code bottom-up to get the desired functionality.
- Polish your code to get point-free style as much as possible. You will be helped by the LA and TAs;
- Use **HLint** for suggestions how to improve it;
- Work in pairs. Ask others if you need. Ask TAs. Ask me. **Don’t copy!** Do learn.
- Enjoy.

---

**Chatterbots (N1)**

A *chatterbot* is a program that attempts to simulate typed conversation, with the aim of at least temporarily fooling a human into thinking they are talking to another person.

- Eliza (Joseph Weizenbaum, 1966), Analiza:-), M-x doctor Turing test
- Loebner prize (not any longer)
- Elbot (www.elbot.com) Artificial Solutions, SAS, IKEA
- Cleverbot (www.cleverbot.com), fun
- 85% of the customer support nowadays is done without human intervention!

---

**The task (N1)**

All solutions are easily found on the net. Please don’t use them.

**Learn!**

- Read the existing code and make sure you understand it;
- Write your code in places marked by `{ - TO BE WRITTEN - }`
- When you get Eliza running, polish your code to get point-free style as much as possible. You will be helped by the TAs;
- Use **HLint** for suggestions how to improve it;
- Work in pairs. Ask others if you need. Ask TAs. Ask me. **Don’t copy!** Do learn.
- Enjoy.