Lab Sheet 1

Starting with Lists

STS Software Technology Systems

Functional Programming - Winter 2022/2023 - October 26, 2022 - Schupp/Lübke

How to succeed with the labs and exercises?

Labs and exercise sheets are published every week on the course homepage at StudIP. As described in the first lecture, each successfully completed lab and exercise earns you bonus points towards your final score in this semester's exam. Keep in mind that you only get bonus points if you would pass the exam without them. **Cheating does not help you - but we will!**

How to complete a lab successfully?

In the lab, you will solve the tasks on lab sheet. You are encouraged to talk to your neighbors and find solutions together, as well to ask the tutor for help. **Towards the end of the session, the tutor will briefly discuss your solutions with you. To pass the lab, you should complete two thirds of the tasks (rounding half up)**.

How to complete an exercise successfully?

In order to complete an exercise sheet successfully, you must upload your answers using INGInious **before the deadline** printed on the exercise sheet. We will not consider any solutions handed in after the deadline! Furthermore, you must solve and hand in the exercises **individually** and your Haskell code **must compile** and **pass certain amounts of tests** as specified. During the exercise session, we develop possible solutions together. Please participate! We encourage you to ask and answer questions from fellow students.

Technically, Haskell files you submit using INGInious must have the format as specified in the task sheets (usually ".hs", ".lhs", or ".txt"). Furthermore, INGInious will only consider your last submission. Therefore, if you first submit successfully (your code compiles and tests are passed) and afterwards unsuccessfully (your code does not compile or certain tests fail again), your last submission counts, and - if it does not compile - will therefore be ignored. Make sure your last submission was successful!

How to get additional information?

We encourage you to discuss past and present exercise sheets with us. Either approach us during the exercise session, or visit us during the weekly office hours. We are also available via e-mail or on the StudIP forum. We try to reply as quick as possible and in general, you should get a reply the next weekday, but we cannot guarantee this.

Getting started

The lab computers are equipped with all software you need to do the lab. However, especially in the Windows pools, a few setup steps are required. On Linux, there are fewer initial setup steps, but you may be less familiar with that operating system. Therefore, we enclosed the *pool-tools.pdf* which should help you to get started. Please have a look at the file and follow the steps closely. For this lab it is not strictly necessary to write code into files, so for now you may decide to ignore steps referring to that.

Playing with Lists

Lists in Haskell are denoted by brackets and the values in the lists are separated by commas.

Task 1 To solve this exercise you should only use the functions `==`, `head`, `init`, `last`, `tail`, `drop`, `reverse`, and `take`. Consider the following list of characters

```
['n', 'u', 'f', 'F', 'P']
```

and answer the following questions:
Hint: First find out what the functions do[1]!

a) What is the return value of the expression `take 3 ['n', 'u', 'f', 'F', 'P']`?

b) What is the return value of the expression `reverse ['n', 'u', 'f', 'F', 'P']`?

c) Is the list `['n', 'u', 'f', 'F', 'P']` the same as the string `"nufFP"`?
Hint: Use `==` to test for equality!

d) How can you get `'n'` as return value?

e) How can you get `'P'` as return value?

f) How can you get `"ufFP"` as return value?

g) How can you get `"FP"` as return value?

h) How can you get `"fun"` as return value?

Task 2 Given two lists, `xs = [4, 1]` and `ys = [3, 2]`. Provide a Haskell expression that builds the list `[1, 2, 3, 4]` from `xs` and `ys`. Try the following functions and operators: `head`, `tail`, `!!`, `++`, `take`, `drop`, `reverse`, `init`

Task 3 Provide a Haskell expression that returns a list containing all the integers between 1 and 99 that are divisible by 3.
Hint: There exists a short form for creating lists of consecutive numbers.

[1] `http://www.haskell.org/hoogle` is a good place to look that up!

Task 4 *

a) Do both the symbols `[]` and `[[]]` represent the empty list? Motivate your answer.

b) Enter the following expressions in GHCi and observe the result. Why does GHCi return what it does?

   a) `reverse []`

   b) `head []`

   c) `sum []`

* This is an advanced, optional task, but we strongly advise you to give it a try!