universität innsbruck

Functional Programming	WS 2023/2024	LVA 703025
Exercise Sheet 1, 10 points		Deadline: Tuesday, October 17, 2023, 8pm

- Mark your completed exercises in the OLAT course of the PS.
- You can use a template .hs file that is provided on the proseminar page.
- Upload your modified .hs file for Exercise 2 in OLAT.
- Your .hs file should be compilable with ghci.

Exercise 1 *Haskell setup* **5 p.**

Set up a working Haskell environment on your computer and get familiar with `ghci`. To do this follow these steps:

- Install Haskell, e.g., via `ghcup`.[1]
- Run `ghci` in a terminal and evaluate the expression `(5 + 2) * 3`.
- Find and install a suitable text editor for your system to write and edit .hs files.[2] You can try one of the following free editors:
  - Notepad++[3] (Windows)
  - Gedit[4] (Windows, macOS, Linux)
  - Visual Studio Code[5] (Windows, macOS, Linux)
- Copy or enter the following code in your text editor and save it to a file called `myProgram.hs`. Be sure to use standard double quotes (`"`), but neither two single-quotes (`''`) nor fancy-looking double-quotes (“ or ”).

```
hello :: String -> String
hello xs = "Hello " ++ xs
```

- Load the file in `ghci` with the command `ghci myProgram.hs`
- Evaluate the expression `hello "World"`
- Make yourself familiar with `ghci`. In particular, try the following commands:
  - `:?` – help
  - `:load name.hs` or `:l name.hs` – load Haskell script `name.hs`
  - `:reload` or `:r` – reload current Haskell script
  - `:edit` or `:e` – edit current Haskell script
  - `:set editor someEditor` – set `someEditor` as preferred editor

  Further investigate what happens if you type `h` and then the tabulator key, or `hel` and the tabulator key.

You can find links to introductory material about installing Haskell, the command line, etc. on the lecture homepage.[6]

[1] https://www.haskell.org/ghcup/
[2] Word processors like Microsoft Word, Apple Pages, . . . are not text editors.
[3] https://notepad-plus-plus.org/
[4] https://wiki.gnome.org/Apps/Gedit
[5] https://code.visualstudio.com
[6] http://cl-informatik.uibk.ac.at/teaching/ws23/fp/ghc_setup.php

Solution 1

After the proseminar everyone should have access to a working Haskell environment and be able to run `ghci`.

Exercise 2 *Writing simple functions* **5 p.**

1. Define a function `volume r = ...` to compute the volume of a sphere with radius `r`. (1 point)
2. 400 liters of helium cost 50.99 EUR. Define a function `heliumCosts` that computes the costs (in EUR) of filling a balloon with radius `r` (in cm). Note that 1 liter is 1 cubic decimeter.
   Hint: It might be worth to write further auxiliary functions to split up the task. (1 point)
3. Define `balloonRadius` as the inverse function of `heliumCosts`: it takes some amount of money `m` in EUR and computes the radius `r` of the balloon (in cm) that you can fill with `m` EUR.
   Hint: `x ** (1/3)` computes $(x)^{\frac{1}{3}} = \sqrt[3]{x}$. (2 points)
4. Test that `balloonRadius (heliumCosts r)` is `r` and `heliumCosts (balloonRadius m)` is `m` for some test values of `m` and `r`.
   If you did not solve `balloonRadius`, just make sure that your implementation of `heliumCosts` is plausible, e.g., `heliumCosts 20` is 4.27 EUR, and `heliumCosts 10` is 0.53 EUR. (1 point)

Solution 2

```
volume r = 4 / 3 * pi * r^3

literCost v = 50.99 / 400 * v

cmToDm r = r / 10

heliumCosts r = literCost (volume (cmToDm r))

radius v = (3/4 * v / pi) ** (1/3)

helium m = m * 400 / 50.99

dmToCm r = r * 10

balloonRadius m = dmToCm (radius (helium m))
```