universität innsbruck

Functional Programming	WS 2023/2024	LVA 703025
Exercise Sheet 2, 10 points		Deadline: Tuesday, October 24, 2023, 8pm

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

Exercise 1 *Parsing expressions* — 5 p.

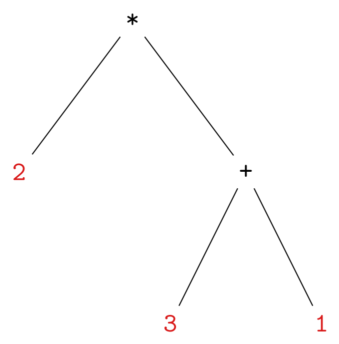
Construct the abstract syntax trees for the given expressions:

1. `2 * (3 + 1)` (1 point)
2. `(x > 3) && (y == (7 - 2)) || (z >= 4)` (2 points)
3. `cube (4 + 1) * (height * width * depth)` (2 points)

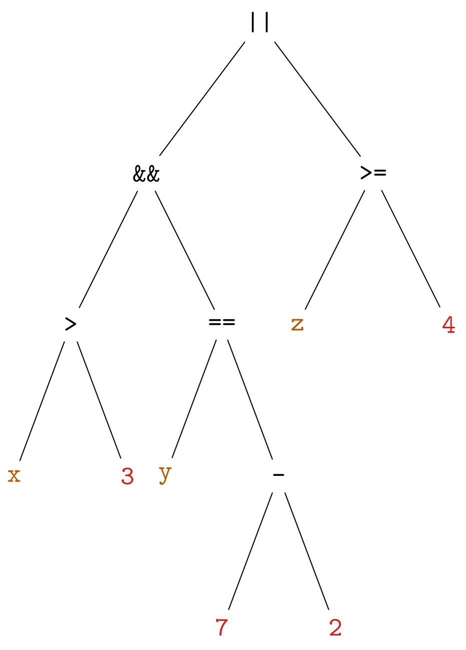
Remark: Function applications (e.g., `cube 4`) bind stronger than operator applications (e.g., `8 * 3`). Also note the precedence rules for logical operators: `&&` has higher precedence than `||`.

Solution 1

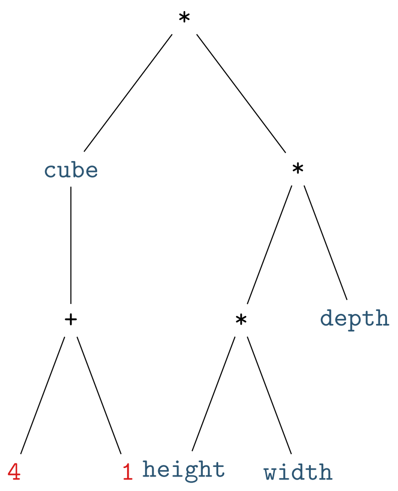
1. `2 * (3 + 1)`



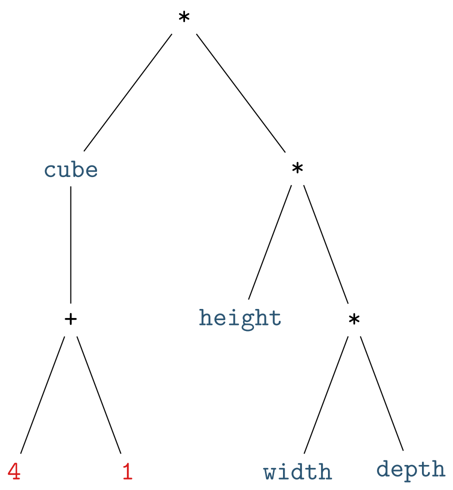
2. `(x > 3) && (y == (7 - 2)) || (z >= 4)`



3. `cube (4 + 1) * (height * width * depth)`



or



Exercise 2 *Datatype definitions* **5 p.**

In this exercise you should design datatypes for listing objects in a fridge. You can use the Haskell template provided on the course website for this exercise.

1. Each object in a fridge has a name and an expiration date. Moreover, each object either has a quantity, e.g., a box of 6 apples, or it is a fluid that has a volume, e.g., 0.5 liters of juice.

   Define a datatype in Haskell called `FridgeObject` to represent such a fridge object. Of course, you may also define auxiliary other datatypes. (1 point)

2. Define the following fridge objects in your Haskell program: (1 point)
   (a) a box of 4 apples with expiration date October 31, 2023
   (b) 2.3 liters of milk with expiration date August 4, 2023
   (c) a net of 6 lemons with expiration date November 3, 2023

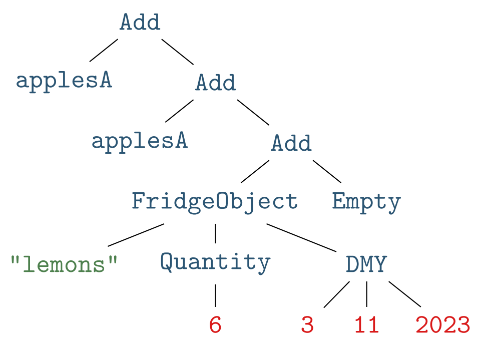
3. Define a datatype `FridgeList` that represents a list of `FridgeObject`s. (1 point)

4. Consider an example fridge that contains two boxes of apples as specified in (a), and the amount of milk as specified in (b). (2 points)
   - Draw the tree that corresponds to the list of objects of the example fridge.
   - Define a constant `exampleFridgeList` in Haskell that represents this tree.
   - Is the representation unique?

Solution 2

```haskell
data Date = DMY
    Int     -- day
    Int     -- month
    Integer -- year
  deriving Show

data Amount =
   Quantity Integer
 | Fluid Double
 deriving Show

data FridgeObject = FridgeObject
    String -- name
    Amount -- kind of object
    Date   -- expiration date
  deriving Show

applesA :: FridgeObject
applesA = FridgeObject "apples" (Quantity 4) (DMY 31 10 2023)

milkB :: FridgeObject
milkB = FridgeObject "milk" (Fluid 2.3) (DMY 4 8 2023)

lemonsC :: FridgeObject
lemonsC = FridgeObject "lemons" (Quantity 6) (DMY 3 11 2023)

data FridgeList =
    Empty
  | Add FridgeObject FridgeList
    deriving Show

exampleFridgeList :: FridgeList
exampleFridgeList = Add applesA (Add applesA (Add lemonsC Empty))

-- the representation is not unique, e.g., one could also have used
-- exampleFridgeList = Add lemonsC (Add applesA (Add applesA Empty))
```

The following tree represents the example fridge list, where here the lemons are expanded, whereas the `applesA` object is not. If you enter `exampleFridgeList` in Haskell, then all definitions will be expanded, so also `applesA` will be replaced by its definition.


Add
applesA
Add
applesA
Add
FridgeObject
Empty
"lemons"
Quantity
DMY
6
3
11
2023