Middle East Technical University Department of Computer Engineering

CENG 242

Programming Language Concepts

Spring 2014-2015

Homework 1 - FamilyTree

Due date: 27.03.2015, Friday, 23:55

1 Objective

This homework aims to help you get familiar with functional programming and concept of haskell programming language.

2 Problem Definition

In this homework you are going to implement several functions on a tree structure called FamilyTree which is a simplified version of a real family tree. FamilyTree has three constructors. Empty is for the empty tree. Individual is for a single member FamilyTree. Lastly, Family constructor represents a family with parents and FamilyTrees of the children and their families. FamilyTree data structure and signature of the functions are given to you in the template file.

```
data Gender = Male | Female deriving (Show, Eq)
data Person = Person Gender String deriving (Show, Eq)
data FamilyTree = Empty | Family Person Person [FamilyTree] | Individual Person
    deriving (Show, Eq)
```

3 Functions

There are 6 functions you are going to implement in this homework and they are explained below.

3.1 addchild - 10pts

```
addchild          :: FamilyTree   -> Person -> FamilyTree
```

This function adds the person as a child to the root of the tree if there is a family at the root and returns the tree. Otherwise it should return the same tree without any change.

```
FamilyTreeHomework> addchild (Family (Person Male "John") (Person Female "Emily") [
    Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Male "Jacob")
```

```
Family (Person Male "John") (Person Female "Emily") [Individual (Person Male "Jacob
    "),Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]
```

3.2 ischild - 10pts

```
ischild         :: FamilyTree   -> Person -> Bool
```

This function checks if the person is the child or children-in-law of the family at the root of the tree.

```
FamilyTreeHomework> ischild (Family (Person Male "John") (Person Female "Emily") [
    Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Male "John")
False
FamilyTreeHomework> ischild (Family (Person Male "John") (Person Female "Emily") [
    Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Female "Tina")
True
```

3.3 childrenOf - 15pts

```
childrenOf      :: FamilyTree   -> Person  -> [Person]
```

This function returns the children and children-in-law of the person given as a parameter of the function in the tree. If the tree is empty or there are no children or the person does not exists in the tree, it should return an empty list.

```
FamilyTreeHomework> childrenOf (Family (Person Male "John") (Person Female "Emily")
     [Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Female "Tina")
[Person Male "Willie",Person Female "Carrie"]
FamilyTreeHomework> childrenOf (Family (Person Male "John") (Person Female "Emily")
     [Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Male "John")
[Person Male "Peter",Person Female "Louis",Person Male "Jackson",Person Female "
    Tina",Person Female "Amelia",Person Female "Emma"]
FamilyTreeFunctions> childrenOf (Family (Person Male "John") (Person Female "Emily"
    ) [Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Male "Willie")
[]
```

3.4 marryTo - 15pts

```
marryTo         :: FamilyTree   -> Person  -> Person -> FamilyTree
```

This function marry the person given in the third parameter to the person given in the second parameter in the tree and returns the resulting tree. Same-sex marriage is forbidden and if the person is already married then he/she immediately divorces and marries the new person and the new person adopts all their children. If the tree is empty or the person does not exists in the tree or it is forbidden, it should return the tree without any change.

```
FamilyTreeHomework> marryTo (Family (Person Male "John") (Person Female "Emily") [
    Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Female "Tina") (Person Male "Cole")
Family (Person Male "John") (Person Female "Emily") [Family (Person Male "Peter") (
    Person Female "Louis") [],Family (Person Male "Cole") (Person Female "Tina") [
    Individual (Person Male "Willie"),Individual (Person Female "Carrie")],
    Individual (Person Female "Amelia"),Individual (Person Female "Emma")]
FamilyTreeHomework> marryTo (Family (Person Male "John") (Person Female "Emily") [
    Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Female "Carrie") (Person Male "Ricardo")
Family (Person Male "John") (Person Female "Emily") [Family (Person Male "Peter") (
    Person Female "Louis") [],Family (Person Male "Jackson") (Person Female "Tina")
    [Individual (Person Male "Willie"),Family (Person Female "Carrie") (Person Male
    "Ricardo") []],Individual (Person Female "Amelia"),Individual (Person Female "
    Emma")]
```

3.5 mother - 20pts

```
mother          :: FamilyTree   -> Person -> Maybe Person
```

This function find and returns mother or mother-in-law of a person in the tree if it exists. If the mother of the person can not be found it should return Nothing.

```
FamilyTreeHomework> mother (Family (Person Male "John") (Person Female "Emily") [
   Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
   Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
   Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
   Female "Emma")]) (Person Female "Tina")
Just (Person Female "Emily")
FamilyTreeHomework> mother (Family (Person Male "John") (Person Female "Emily") [
   Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
   Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
   Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
   Female "Emma")]) (Person Male "Willie")
Just (Person Female "Tina")
FamilyTreeHomework> mother (Family (Person Male "John") (Person Female "Emily") [
   Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
   Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
   Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
   Female "Emma")]) (Person Male "John")
Nothing
FamilyTreeHomework> mother (Family (Person Male "John") (Person Female "Emily") [
   Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
   Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
   Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
   Female "Emma")]) (Person Male "Robert")
Nothing
```

3.6 brothers - 30pts

```
brothers        :: FamilyTree   -> Person -> [Person]
```

This functions returns brothers and brother-in-laws of a person in the tree. If the tree is empty or there are no brothers and brother-in-laws or the person does not exists in the tree, it should return an empty list.

```
FamilyTreeHomework> brothers (Family (Person Male "John") (Person Female "Emily") [
   Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
   Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
   Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
   Female "Emma")]) (Person Female "Tina")
```

```
[Person Male "Peter"]
FamilyTreeHomework> brothers (Family (Person Male "John") (Person Female "Emily") [
    Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Female "Emma")
[Person Male "Peter",Person Male "Jackson"]
FamilyTreeHomework> brothers (Family (Person Male "John") (Person Female "Emily") [
    Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Male "Willie")
[]
FamilyTreeHomework> brothers (Family (Person Male "John") (Person Female "Emily") [
    Family (Person Male "Peter") (Person Female "Louis") [],Family (Person Male "
    Jackson") (Person Female "Tina") [Individual (Person Male "Willie"),Individual (
    Person Female "Carrie")],Individual (Person Female "Amelia"),Individual (Person
    Female "Emma")]) (Person Male "Peter")
[Person Male "Jackson"]
```

4 Specifications

- Every person is unique in the FamilyTree structure. There are no duplicates.
- Deriving from Eq typeclass allow Gender, Person and FamilyTree values to be compared directly using (==) and (/=) functions. This checks for exact match between values.
- Parents and children can be given in any order. In the output, the order of the elements are not important as long as it is the output with different order. This goes for lists as well as the FamilyTree structure.

5 The Template Code

You should implement the functions in this template after the comments.

```haskell
module FamilyTreeHomework where

data Gender = Male | Female deriving (Show, Eq)
data Person = Person Gender String deriving (Show, Eq)
data FamilyTree = Empty | Family Person Person [FamilyTree] | Individual Person
    deriving (Show, Eq)

addchild        :: FamilyTree   -> Person -> FamilyTree
ischild         :: FamilyTree   -> Person -> Bool

childrenOf      :: FamilyTree   -> Person  -> [Person]
marryTo         :: FamilyTree   -> Person  -> Person -> FamilyTree

mother          :: FamilyTree   -> Person -> Maybe Person
brothers        :: FamilyTree   -> Person -> [Person]

-- DO NOT MODIFY ABOVE THIS LINE
-- Start Here
```

6 Regulations

- **Programming Language:** You must code your program in Haskell. Your submission will be tested with `hugs` interpreter on moodle. You are expected make sure your code loads successfully in `hugs` interpreter using the run command on the moodle system.
- **Late Submission:** Late submission is allowed and a penalty of $5 * day * day$ is applied to the final grade.
- **Cheating:** In case of cheating, the university regulations will be applied.
- **Newsgroup:** You must follow the newsgroup (news.ceng.metu.edu.tr) for discussions and possible updates on a daily basis.
- **Grading:** This homework will be graded out of 100.

7 Submission

Submission will be done via moodle system. You can either download the template file, make necessary additions and upload the file to the system or edit using the editor on the moodle and save your changes.