MIDDLE EAST TECHNICAL UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

# CENG 111

Fall 2023

Take-Home Exam 4

## REGULATIONS

**Due date:** 23:59, 21 January 2024, Sunday *(Not subject to postpone)*

**Submission:** Electronically. You should save your program source code as a text file named `the4.py`. Check the announcement on the ODTUCLASS course page for the submission procedure.

**Team:** There is **no** teaming up. This is an EXAM.

**Cheating:** Source(s) and Receiver(s) will receive zero and be subject to disciplinary action.

## FAMILY TREE

A family tree is a representation of parent-child relationships. Given such a tree, one can reason about more general kinship than parent-child relationships. Figure 1 depicts an example where the following relationships can be deduced (the list is not comprehensive):

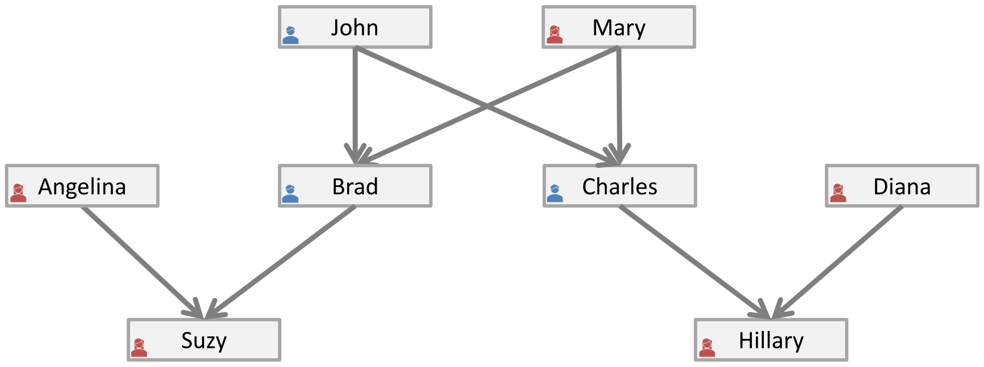

Figure 1: An example family tree.

- Mary is the mother of Brad and Charles.
- John is the father of Brad and Charles.
- Brad is a son of John and Mary; moreover, he is the father of Suzy.
- Suzy is the granddaughter of John and Mary.
- Charles is the uncle of Suzy.

Note that a family tree does not conform to the formal definition of a *tree* in Computer Science since, in the family tree, (i) a node can have more than one parent and (ii) there can be more than one root.

## PROBLEM

In this take-home exam, you will be working with a set of relations on *partial* family trees. We define a partial family tree as a family tree that has at most one parent for each node. In other words, for a person (except for the root), the partial family tree can include that person's mother or father - not both. A partial family tree will be given to you as a nested list where a leaf node is denoted with just a string representing for the name of person. For example, the *partial* family tree example in Figure 2 will be given to you as follows:

```
['Fatma', ['sinan', 'Elif', 'mehmet'], 'veli', ['Ayse', 'Zeynep', 'fikret', 'hikmet']]
```

Note that, in this representation, male names start with a lowercase letter and female names with an uppercase letter.

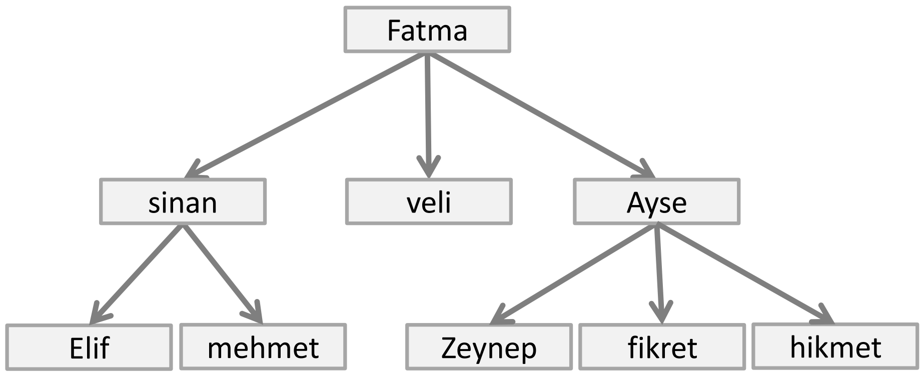

Figure 2: Example partial family tree.

In this THE, you are asked to define the following functions:

- `brothers(T, pname)`: Brothers of `pname`.
- `sisters(T, pname)`: Sisters of `pname`.
- `siblings(T, pname)`: Siblings of `pname`.
- `uncles(T, pname)`: Uncles of `pname`.
- `aunts(T, pname)`: Aunts of `pname`.
- `cousins(T, pname)`: Cousins of `pname`.

Each function takes a partial family tree `T` and a person name `pname` (a string) as two arguments and returns a list (of strings) containing the names of the people conforming to the corresponding relation with `pname`. If no person conforms to the implied relationship or the tree does not contain sufficient information about the relationship (for example, in the case of asking the brothers of the root node), then a function should return an empty list.

## SPECIFICATIONS

- Person names can contain only letters from the English alphabet. Moreover, names are unique; i.e., two people/nodes are different if and only if their names are different.

- The genders of the people will be deducible from their names. If a name starts with a lowercase letter, the person is male; otherwise, the person is female.
- You will not be given erroneous inputs.
- A person can have only one father or one mother represented in the tree, but not both.
- You are not allowed to `import` any module.
- You can use recursion as well as iteration.
- There is no limit on the number of levels in tree.

## EXAMPLE RUN

The following illustrates an example session for the family tree drawn in Figure 2:

```
>>> T = ['Fatma', ['sinan', 'Elif', 'mehmet'], 'veli', ['Ayse', 'Zeynep', 'fikret', 'hikmet']]
>>> cousins(T, 'Elif')
['Zeynep', 'fikret', 'hikmet']
>>> aunts(T, 'Zeynep')
[]
>>> uncles(T, 'Zeynep')
['sinan', 'veli']
```

## RESTRICTIONS and GRADING

- A set of arbitrarily selected students will be subject to oral examination about their solutions. The details will be announced on ODTUclass later.
- Your program will be graded through an automated process and therefore, any violation of the specifications will lead to errors (and reduction of points) in automated evaluation. You should especially avoid printing something on screen.
- Your solutions will not be tested with incorrect/erroneous inputs.
- A program based on randomness will be graded zero.
- Your program will be tested with multiple data (a distinct run for each data). Any program that performs only 30% and below will enter a glass-box test (eye inspection by the grader TA). The TA will judge an overall THE3 grade in the range of [0,30].
- The glass-box test grade is not open to discussion nor explanation.