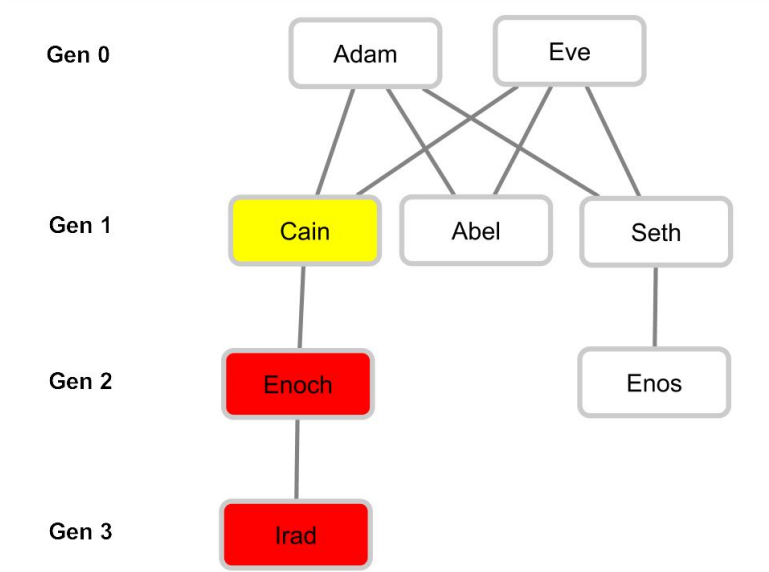
Level 2

Level 2

Descendants

- Good, now that we know who our ancestors are, let's see who the descendants are
- The descendants of a person are that person's children as well as their children's children and so on
- **descendants(x) = What is the list of descendants of x, sorted in ascending alphabetical order?**
- The output should be a list of names separated by commas, or the string "None" in case there are no descendants

- Example:
  - Input: **descendants(Cain)**
  - Output: **Enoch,Irad**




The Social Network
International Coding Contest
10th May 2019

> Intro

December 2537
location: Research facility, somewhere in northern Scandinavia
History Preservation Team (H.P.T)

Many years have passed since the great doom hit Earth eradicating most life.
Not only life was affected, but technology as well - 99.9% of all data centers destroyed.
Countless exabytes of information, lost. Earth's history, almost wiped out.

The last surviving humans formed research groups, clustering up in rehabilitated research facilities and trying to salvage whatever they could from whatever scraps they could find.

One day, they found a relic of old times - a hard-drive containing a social network...

Your task

Decipher the contents of the hard-drive, analyze the data and compare the structure of the society from ancient times to that which we have now.

Level 1

Level 1

The network

- The **network** is defined as a collection of people that are related by mother/father/children relationships.
- You will be provided a CSV (comma separated values) file that contains information about the network that we want to analyze
- Each row describes a person and has the following 3 attributes separated by a comma (“,”)
  - Name
    - string
    - unique among all persons in our network
  - Father name
    - string
    - optional (may be empty in case we don’t have information about the father)
    - used to uniquely identify who the father of this person is
  - Mother name
    - string
    - optional (may be empty in case we don’t have information about the mother)
    - used to uniquely identify who the mother of this person is
- If a name appears in the father/mother name column, it is guaranteed that it will also appear on its own row
- No incest: The child-relationships **always** describe **a DAG** (Directed Acyclic Tree), so no children with mothers, sisters, cousins etc.

> Level 1

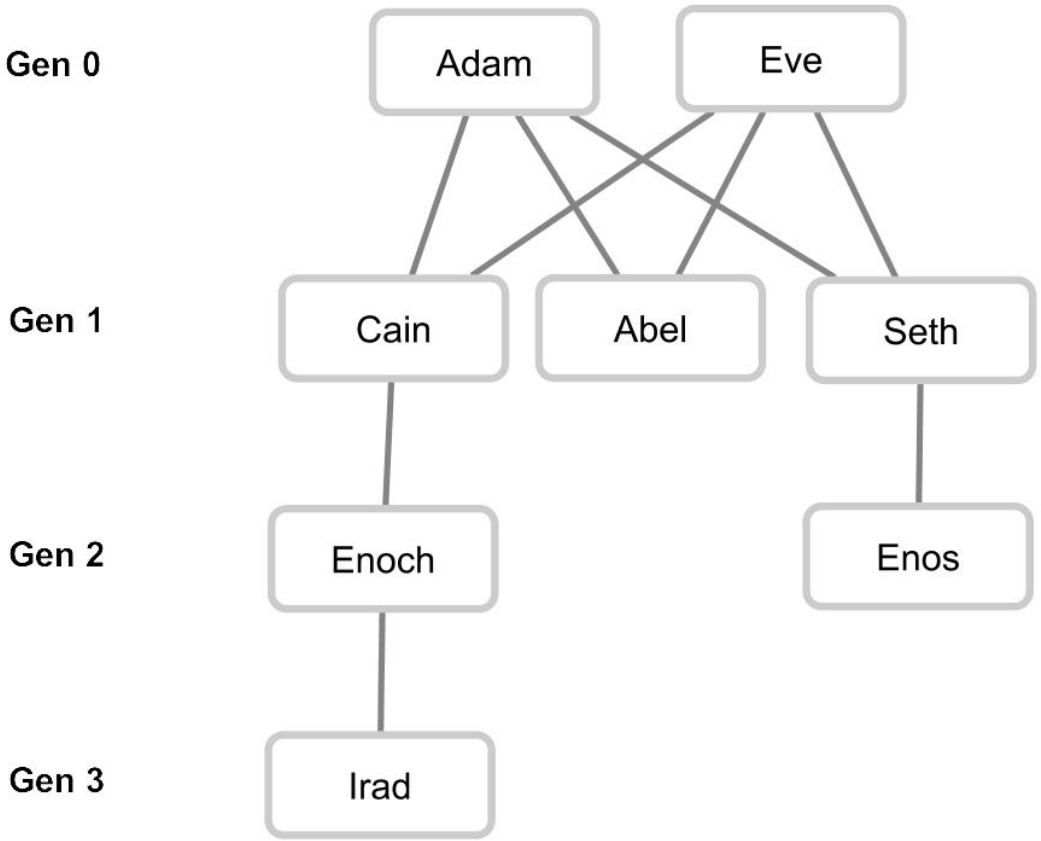
The network - example

```
Adam,,
Eve,,
Cain,Adam,Eve
Abel,Adam,Eve
Seth,Adam,Eve
Enoch,Cain,
Irad,Enoch,
Enos,Seth,
```

- This file describes a network where
  - Enoch is the child of Cain
  - Irad is the child of Enoch
  - Enos is the child of Seth
  - Cain, Abel and Seth are the children of Adam and Eve

  - Adam is the father of Cain, Abel and Seth
  - Eve is the mother of Cain, Abel and Seth

  - Cain, Abel and Seth are siblings



> Level 1

The questions

- A question is defined similar to a function call in most programming languages
- For example, A(x) means who is the father of x =>A(Cain) = Adam
- Those functions can have zero, one, or more arguments
  - B() -> how many people are there in our network?
  - C(x) -> how many children does x have?
  - D(x,y) -> are x and y siblings?
- There will be no whitespace in after or before the parentheses or commas.
- One input file can contain multiple questions, each question will be on its own row
- For each input file you should generate an output file containing the answers to the given questions, one per row, in the same order as the input file.
- Output that file on the contest page in the corresponding input and if you get all the inputs correct, you will pass to the next level
- Each new level will define new type of questions that you need to handle in your solution. The code itself does not need to be submitted, but you can do that after you finish a level in order to get bonus points.

> Level 1

The actual level 1 - Ancestors

- Now that we've cleared those details, let's start
- The first questions we have are about who are the ancestors of some people
- The ancestors of a person are that person's parents as well as their parents' parents and so on
- **ancestors(x) = What is the list of ancestors of x, sorted in ascending alphabetical order?**
- The output should be a list of names separated by commas, or the string "None" in case there are no ancestors

- Example:
  - Input: **ancestors(Enoch)**
  - Output: **Adam,Cain,Eve**