Fill in

- **FULL NAME**: Raf Engelen
- **STUDENT NUMBER**: r0901812
- **CLASS**: 3APP1

Practical 1: Artificial Intelligence - Logic Programming

Score: ... /4

To solve this question you have to use Kanren, a Python package that enables logic programming in Python. Since Kanren is not supported in the latest version of Python, it is best to solve this question using **Colab**.

Program the following facts. **Don't use the facts function from Kanren, but make use of the fact (singular) function (do this several times).**

- Define a relation **course** and program these facts: ai, react, french and python are courses.
- Define a relation **follows** and program these facts: jane follows ai, react and python. john follows python and cooking. abe follows react, french and yoga. may follows cooking.
- Define a relation **level** and program these facts: The level of ai is medium. The level of react and python is easy. The level of french is hard.

In [ ]:
```
!apt-get install python3.7
!sudo update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.7 1
!sudo update-alternatives --config python3
!apt-get install --reinstall python3.7-distutils
!apt install python3-pip
!pip install kanren
```

```
'apt-get' is not recognized as an internal or external command,
operable program or batch file.
'sudo' is not recognized as an internal or external command,
operable program or batch file.
'sudo' is not recognized as an internal or external command,
operable program or batch file.
'apt-get' is not recognized as an internal or external command,
operable program or batch file.
'apt' is not recognized as an internal or external command,
operable program or batch file.
```

```
Collecting kanren
  Using cached kanren-0.2.3.tar.gz (23 kB)
  Installing build dependencies: started
  Installing build dependencies: finished with status 'done'
  Getting requirements to build wheel: started
  Getting requirements to build wheel: finished with status 'done'
  Preparing metadata (pyproject.toml): started
  Preparing metadata (pyproject.toml): finished with status 'done'
Collecting toolz (from kanren)
  Using cached toolz-0.12.0-py3-none-any.whl (55 kB)
Collecting multipledispatch (from kanren)
  Using cached multipledispatch-1.0.0-py3-none-any.whl.metadata (3.8 kB)
Collecting unification (from kanren)
  Using cached unification-0.2.2-py2.py3-none-any.whl (10 kB)
Downloading multipledispatch-1.0.0-py3-none-any.whl (12 kB)
Building wheels for collected packages: kanren
  Building wheel for kanren (pyproject.toml): started
  Building wheel for kanren (pyproject.toml): finished with status 'done'
  Created wheel for kanren: filename=kanren-0.2.3-py3-none-any.whl size=15892 sha256
=011bd794022de269e5f9083e0c53d37f6346ad45ba9c9c5bfa317aecbb2538a8
  Stored in directory: c:\users\rafen\appdata\local\pip\cache\wheels\69\e7\58\f138e1
a8e1b6a717490d4c69ed313e46b459fb5bde2a100b9b
Successfully built kanren
Installing collected packages: multipledispatch, toolz, unification, kanren
Successfully installed kanren-0.2.3 multipledispatch-1.0.0 toolz-0.12.0 unification-
0.2.2
```

In [ ]:
```python
# A) Imports
from kanren import run, eq, membero, var, conde, lany, lall,fact, Relation # some o
```

In [ ]:
```python
# B) Relations
course = Relation()
follows = Relation()
level = Relation()
```

In [ ]:
```python
# C) Facts

fact(course,'ai')
fact(course,"react")
fact(course, "french")
fact(course, "python")

fact(follows, "Jane", "ai")
fact(follows, "Jane", "react")
fact(follows, "Jane", "python")
fact(follows, "John", "python")
fact(follows, "John", "cooking")
fact(follows, "Abe", "react")
fact(follows, "Abe", "french")
fact(follows, "Abe", "yoga")
fact(follows, "May", "cooking")

# The level of ai is medium. The level of react and python is easy. The level of fr
fact(level, "ai", "medium")
fact(level, "eact", "easy")
fact(level, "python", "easy")
```

```
fact(level, "french", "hard")
```

Program a list of all people who follow the cooking workshop. The output should be:

may john

In [ ]:
```
# D) Who follows cooking?
x=var()
y=var()
run(0, x, follows(x, "cooking"))
```

Out[ ]:
```
('May', 'John')
```

Use the Python `def` statement to declare a rule **follows_course** with one parameter **p**. The rule is true when p follows a course. Use this rule to program a list of all people who follow a course. The output should be:

abe jane john

***Note**: if you run this code on Colab, which handles Kanren slightly differently then a local Kanren install, you might encounter duplicates. You can tackle this by using the set function, but this is not really necessary. This remark also applies to the following questions.*

In [ ]:
```
# E) Who follows a course?
def follows_course(p):
    return conde((follows(p, y), course(y)))


run(0,x, follows_course(x))
```

Out[ ]:
```
('Jane', 'Abe', 'John')
```

Use the Python `def` statement to declare a rule **follows_course_hard** with one parameter **p**. The rule is true when p follows a course with level of difficulty hard. Use this rule to program a list of all people who follow a course with level of difficulty hard. The output should be:

abe

In [ ]:
```
# F) Who follows a course that is hard?

def follows_course_hard(p):
    return conde((follows(p, y), course(y), level(y,"hard")))
run(0,x,follows_course_hard(x))
```

Out[ ]:
```
('Abe',)
```

Use the Python `def` statement to declare a rule **together** with two parameters **p1 and p2**. The rule is true when p1 and p2 are following a course together. Make a list of all the couples taking a course together. The output should be (note that everyone takes course

with themselves and the pairs are double listed. this is not a problem):

{('abe', 'abe'), ('abe', 'jane'), ('jane', 'abe'), ('jane', 'jane'), ('jane', 'john'), ('john', 'jane'), ('john', 'john')}

In [ ]:
```
# G) All couples who are in a course together
def together(p1,p2):
    return conde((follows(p1, y), course(y), follows(p2,y)))
run(0,(x,y),together(x, y))
```

```
(<function lany at 0x00000174A7094F70>, (<function lall at 0x00000174A7094DC0>, <fun
ction Relation.__call__.<locals>.goal at 0x00000174A77CA040>, <function Relation.__c
all__.<locals>.goal at 0x00000174A7510B80>, <function Relation.__call__.<locals>.goa
l at 0x00000174A7510280>))
```