Introduction to Artificial Intelligence

# Exercise 7: Logic and Inference

---

Teacher: Muhammad Farhan

## 1. Exercise details

Solve one of the following problems using Prolog.

### Variant 1

Perform simple addition and subtraction of dates by a number of N days. Assumptions:

- The year is 2024
- N<=366.

Example:

```
?- add_date("2205", 14)   % add 14 days from 22nd of May
"0506"  % 5th of June

?- sub_date("2205", 10)   % subtract 10 days from 22nd of May
"1205"  % 12th of May
```

### Variant 2

Return the number of N days between 2 dates. Assumptions:

- The year is 2024
- N<=366.

Example:

```
% how many days between 22nd of May and 5th of June
?- interval("2205", "0506")
14

% how many days between 1st of Feb and 11th of Feb
?- interval("0102", "1102")
10
```

### Variant 3

Convert an input number N to English words. N<=1000

Example:

```
?- to_words(45)
"forty five"

?- to_words(394)
"three hundred and ninety four"
```

### Variant 4

This variant is the opposite of variant 3. Convert English words of a written number (up to a thousand) into numerical digits N, where N<=1000.

Example:

```
?- to_num("ninety nine")
99

?- to_num("one hundred and one")
101
```

### Variant 5

Given a day and a date, what will be the day and date after N days of working day. Assumptions:

- There are 5 working days in a week
- The year is 2024
- N<=366

Example:

```
% what will be the day and date after 6 working days from 22nd of May
?- n_work_days("2205", 6)
"Tuesday, 3005"

% what will be the day and date after 10 working days from 1st of June
?- n_work_days("0106", 10)
"Thursday, 1506"
```

## 2. Technical details

1. Write your solution using Prolog. You can use the online swi-prolog environment for quick experiments with no installation.
2. Provide comments to help readability.
3. There are many ways to solve the problem. Treat the examples as a test for your solution. Also test it on a few different example to make sure there are no bugs for some edge cases.

4. A correct solution would ideally perform just like in the examples. However, a less ideal solution will also be accepted (rather than no solutions at all), albeit with less points. Explain your difficulties in the report and what actions you have tried, and you might be rewarded with fewer penalties.
5. Questions related to the exercise and report should be asked in the Lab7 channel

## 3. Submission

You will have 2 weeks to submit a solution. Please send me a GitHub repository link (**do not send code files directly**) where you should commit the following files:

1. Prolog code
2. Instructions (write a readme.md file) on how to run the code (what to query to obtain the intended answer).
3. Short report:
    - Describe the logic flow of your solution
    - Briefly explain the main components of the code
    - Mention challenges you found during the exercise (if any)

## 4. Deadline

To schedule for the online oral assessment, you must commit your work (in GitHub) **before** the submission deadline as noted below:

- Group 101 & 104, **deadline**: Wednesday, 21-05-2025
- Group 102 & 105, **deadline**: Friday, 23-05-2025
- Group 103 & 106, **deadline**: Monday, 26-05-2025

## 5. Oral assessment

To complete this lab exercise, both members of your lab group should **actively** participate in the oral assessment, which takes place during the lab meetings, 2 weeks after this lab exercise was introduced.

The assessment will be approximately 15m where you will:

- Share your screen,
- Briefly explain your solution from report,
- Demonstrate your code,
- Be asked about your work

## 6. Grading

You can get 0-5 points for this lab. The following criteria will be used to evaluate your work:

- **2 points** for submission:
    - Correctness of solution
    - A well documented code and its instructions
    - Completeness of report (refer to section 3.3) **3 points** for oral assessment, which will be about:

Conceptualization, how you approached the solution

- Knowledge and experience from the Lab, about basic concepts in prolog and how it differs from other programming languages

# 7. Additional information

1. References
    - An Introduction to Prolog Programming by Ulle Endriss