Practical Class 0

Prolog and SICStus Basics

Goals:

- Start and use **SICStus Prolog** interactively.
- Load and run Prolog programs from files.
- Write and query **facts** and **rules**.
- Use variables and logical reasoning in queries.

0. Getting Started with SICStus Prolog

a) If you haven't yet done so, install *SICStus Prolog* (or another interpreter, such as *SWI Prolog*, *YAP*, or other). Optionally, you can also install SPIDER (*SICStus Prolog IDE for Eclipse*).

b) After starting SICStus (e.g., using the command **sicstus** in the terminal), you should see a prompt similar to "**| ?-**". This means the interpreter is ready for a query. Some basic commands follow. Create an empty text file named **file.pl** in your favourite code editor and try each command.

Command	Description
`halt.`	Exit Prolog
`[file].`	Load a program from file `file.pl`
`consult('file.pl').`	Same as `[file].`
`reconsult('file.pl').`	Reload file after edits
`listing.`	Show all loaded facts and rules
`trace.`	Start trace mode
`notrace.`	Stop trace mode

1. First Program: Facts and Queries

a) Create a text file **family.pl** with the following content and load it into SICStus.

```
% Simple family facts
parent(alice, bob).
parent(alice, carol).
parent(bob, david).
parent(carol, emma).

female(alice).
female(carol).
female(emma).

male(bob).
```

male(david).

b) Use the interpreter to answer the following questions:

i. *female(alice).*
What answer to you get?

ii. *female(bob).*
What answer to you get?

iii. *parent(alice, bob).*
What answer to you get? Try to press a second time "enter". Run the query again. This time, after the execute the query, try to press ";" or "n". What do you get?

iv. *parent(alice, david).*
What answer to you get?

2. Running Queries in SICStus Prolog

a) After loading a Prolog program, you can ask **queries** about the knowledge it contains. For example, suppose your program includes these facts:

lives_in(lion, savannah).
lives_in(penguin, antarctica).
lives_in(elephant, savannah).
lives_in(panda, forest).
lives_in(snake, forest).

Once the file is loaded (e.g., **?- [animals].**), you can ask Prolog questions about it. Go ahead and the a file with this facts, and load it into SICStus.

b) Some examples of queries you may make:

i. *lives_in(X, savannah).*
This query asks: "Which animals live in the savannah?" Prolog searches through its knowledge base and responds:

```
| ?- lives_in(X, savannah).
X = lion ? ;
X = elephant ? ;
no
| ?-
```

Here's what happens:

- The **first answer** X = lion means Prolog found one fact that satisfies the query.
- Typing **semicolon (;)** tells Prolog: *"Look for another solution."*
- It then finds X = elephant.
- If you press **Return (Enter)** instead, Prolog stops searching and finishes the query.

When Prolog has no more answers, it prints a **full stop (.)**.

ii. Try to find out which animals leave in the forest.

c) If you ask something that isn't true, e.g., *?- lives_in(tiger, desert).,* Prolog will reply **no**. This means Prolog could not find any fact or rule proving the goal. Try other queries that are not true.
d) If the query is written incorrectly, e.g., *?- lives_in(X, savannah*., Prolog detects the mistake and shows an error message, e.g., **ERROR: Syntax error: Unexpected end of term**.

Summary:

Key	Meaning
`?-`	Prompt for a query
`.`	Ends the query
`;`	Ask for the next solution
`no.`	No solutions found
Error message	Problem in query or program syntax