ARTIFICIAL INTELLIGENCE-LAB CS304

INTRODUCTION TO PROLOG

By: Nidhi Periwal,

Teaching Assistant, SVNIT, Surat

Reference Books

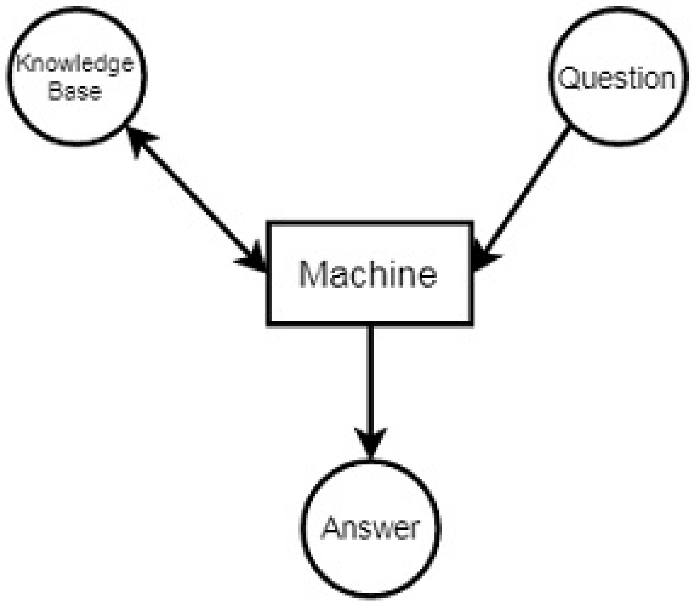
- PROLOG Programming For Artificial Intelligence” -By Ivan Bratko(Addison-Wesley)

Prolog-Introduction

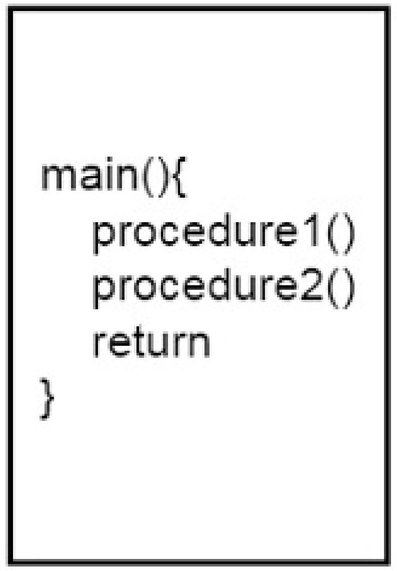
- Prolog is a programming language for symbolic, non-numeric computation.
- It was invented by Alain Colmerauer in 1970.
- Prolog takes only **facts** and **rules** to arrive at **goals**.
- For solving logic and decision problems, Prolog is ideal.
- It is specially well suited for solving problems that involve objects and relations between objects
- Prolog is acronym of **Programming in Logic**

Prolog-Introduction

- The name *Prolog* is taken from the phrase “Programming in logic”
- How is Prolog different from procedural programming ?
    - –Combines logic , data and relationships between data in its program
    - –Suitable for recursion intensive, AI applications
    - –Prolog programs are created by Knowledge Engineers rather than just programmers



Logical Programming



```
main(){
    procedure1()
    procedure2()
    return
}
```

Functional Programming

Prolog

- Facts
- Rules
- Goal

Advantages of Prolog

- It is simple.
- It has built-in list handling, very useful for representing sequences, trees, and so on.
- Procedural language support can be added to a Prolog system through functional interface.
- Both integer and real arithmetic is supported.
- Debugging and compilation is easy.

Limitations of Prolog

- Not very efficient for numerical processing
- All data structures used in conventional procedural programming are not available with Prolog
- It can be very difficult to design a database that accurately represents relationships.
- Prolog is not best suited to **solving complex arithmetical** computations.

Expressing Facts in Prolog

- In prolog, the facts can be represented in symbolic relationship.
- A Prolog Expression is a symbolic extension of an English expressions.
- **Eg:** The right speaker is dead.
  - **is(right_speaker,dead).**
    - This representation in Prolog is called **clause.**
    - **right_speaker** and **dead** are **objects.**
    - **Is** is Relation name in prolog.
    - The Entire expression before period(.) is called **Predicate**
    - The word **before** the parenthesis is the name of **relation**
    - The elements within parentheses are the **arguments** of **predicate**, which may be **objects** or **variables**
    - If we add the **period(.)** to the predicates it becomes a **clause**

Facts in Prolog

- English Statements:
  - Tommy is a dog.
  - John loves to eat Pizza
  - Milly is cat.

- Syntax

  relation(object1,object2...).

Expressing Facts in Prolog

A **fact** is a predicate expression that makes a declarative statement about the problem domain. Note that all Prolog sentences must end with a period.

- likes(john, susie). /* John likes Susie */
- likes(X, susie). /* Everyone likes Susie */
- likes(john, Y). /* John likes everybody */
- likes(john, Y), likes(Y, john). /* John likes everybody and everybody likes John */
- likes(john, susie)**;** likes(john,mary). /* John likes Susie **or** John likes Mary */
- not(likes(john,pizza)). /* John does not like pizza */

Rules in Prolog

- A **rule** is a predicate expression that uses logical implication (:-) to describe a relationship among facts.
- They define implicit relationship between objects.
- Thus a Prolog rule takes the form
  - left_hand_side :- right_hand_side .
- **This sentence is interpreted as: left_hand_side if right_hand_side.**
  - The left_hand_side is restricted to a single, positive, literal, which means it must consist of a positive atomic expression. It cannot be negated and it cannot contain logical connectives.
- This notation is known as a **Horn clause.**
  - In Horn clause logic, the left hand side of the clause is the conclusion, and must be a single positive literal.

Rules in Prolog

- **Examples of valid rules:**
- X and Y are friends if they like each other
  - friends(X,Y) :- likes(X,Y),likes(Y,X).
- X hates Y if X does not like Y.
  - hates(X,Y) :- not(likes(X,Y)).
- X and Y are enemies if they don't like each other
  - enemies(X,Y) :- not(likes(X,Y)),not(likes(Y,X)).

Examples of invalid rules:

- left_of(X,Y) :- right_of(Y,X) (Missing a period)
- likes(X,Y),likes(Y,X) :- friends(X,Y). (LHS is not a single literal)

Rules in Prolog

- Statements about *objects* and their *relationships*
- Expess
  - *If-then conditions*
    - *I use an umbrella if there is a rain*
    - *use(i, umbrella) :- occur(rain).*
  - *Generalizations*
    - *All men are mortal*
    - *mortal(X) :- man(X).*
  - *Definitions*
    - *An animal is a bird if it has feathers*
    - *bird(X) :- animal(X), has_feather(X).*

Prolog Queries

- Based on the Rules and Facts, Prolog can answer questions we ask it
- This is known as querying the system.
- We may want to ask, “What does ali like?”
- In Prolog syntax, we ask: likes(ali,**What**).
  - Note: capital W on what
  - **What** is a variable name

Example

- a_kind_of(aa,ship).
- a_kind_of(bb,ship).
- part_of(aa,jordanian_navy).
- part_of(bb,jordanian_navy).
- part_of(jordanian_navy,jordanian_government).
- a_kind_of(jordanian_government,government).
- color(ship,red).

Example

- **Querying the Facts**
- Answer may be:
- YES or NO
- Example:
  - Goal: a_kind_of(aa,ship).
    - YES
  - Goal: a_kind_of(cc,ship).
    - NO

Example

- **Queries with one variable.**
- Example:
- Goal: a_kind_of(aa,X).
- X=ship
- 1 solution

Multimatching Queries

- Goal: a_kind_of(X,ship).
- X=aa
- X =bb
- 2 solution

Example

- **Multi-Conditional Queries**
- Query has many conditions
- Example
  - Color (aa,X).
  - no solution...........
- BUT can find solution by using the following query:
- a_kind_of(aa,Y),color(Y,X)
- Y=ship
- X=red

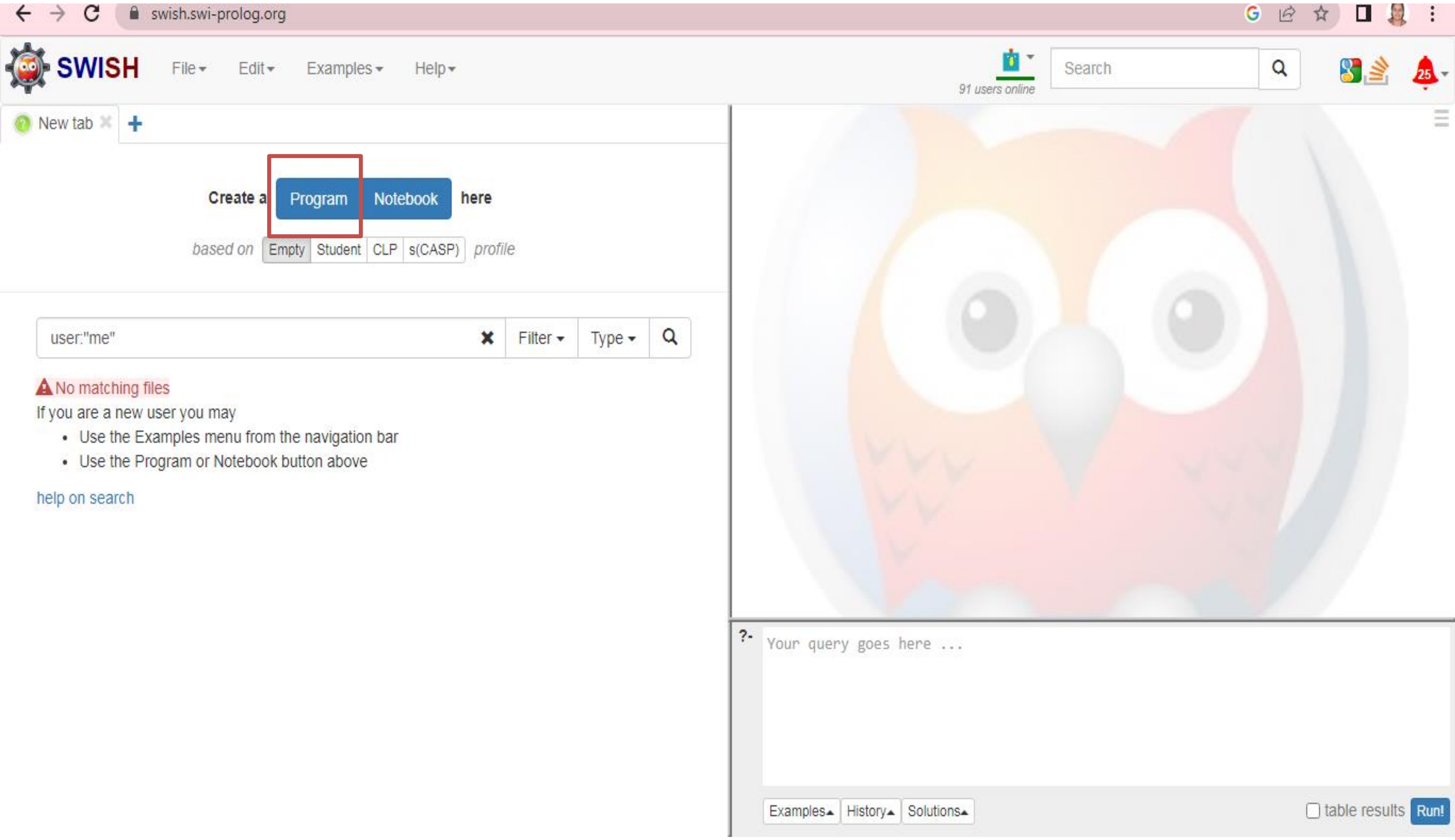
SWI Prolog

- Online Compiler: https://swish.swi-prolog.org/

OR

- Download it from : https://www.swi-prolog.org/download/

- Prolog Files are stored with extension “.pl”.



user:"me"

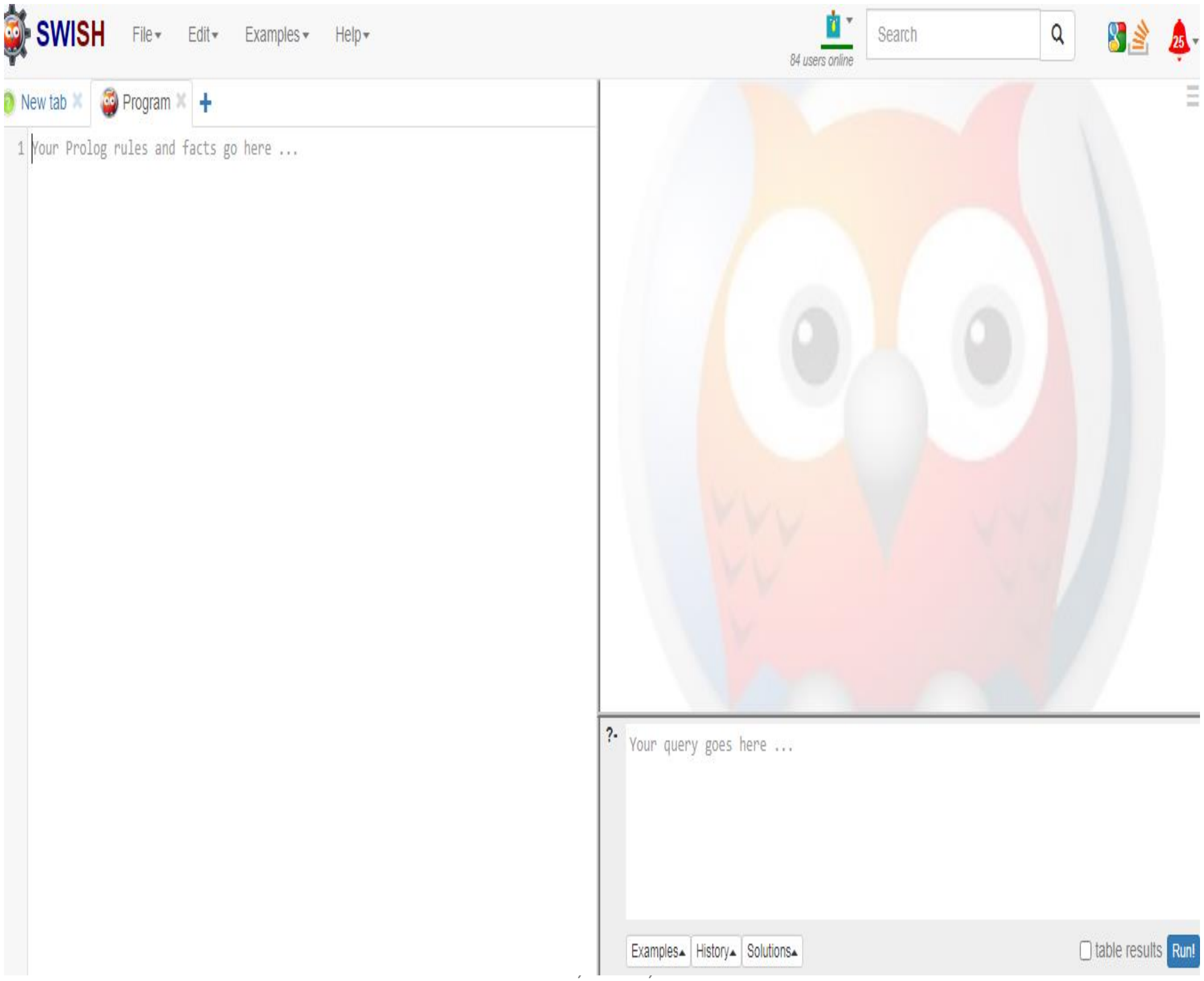
No matching files

If you are a new user you may

- Use the Examples menu from the navigation bar
- Use the Program or Notebook button above

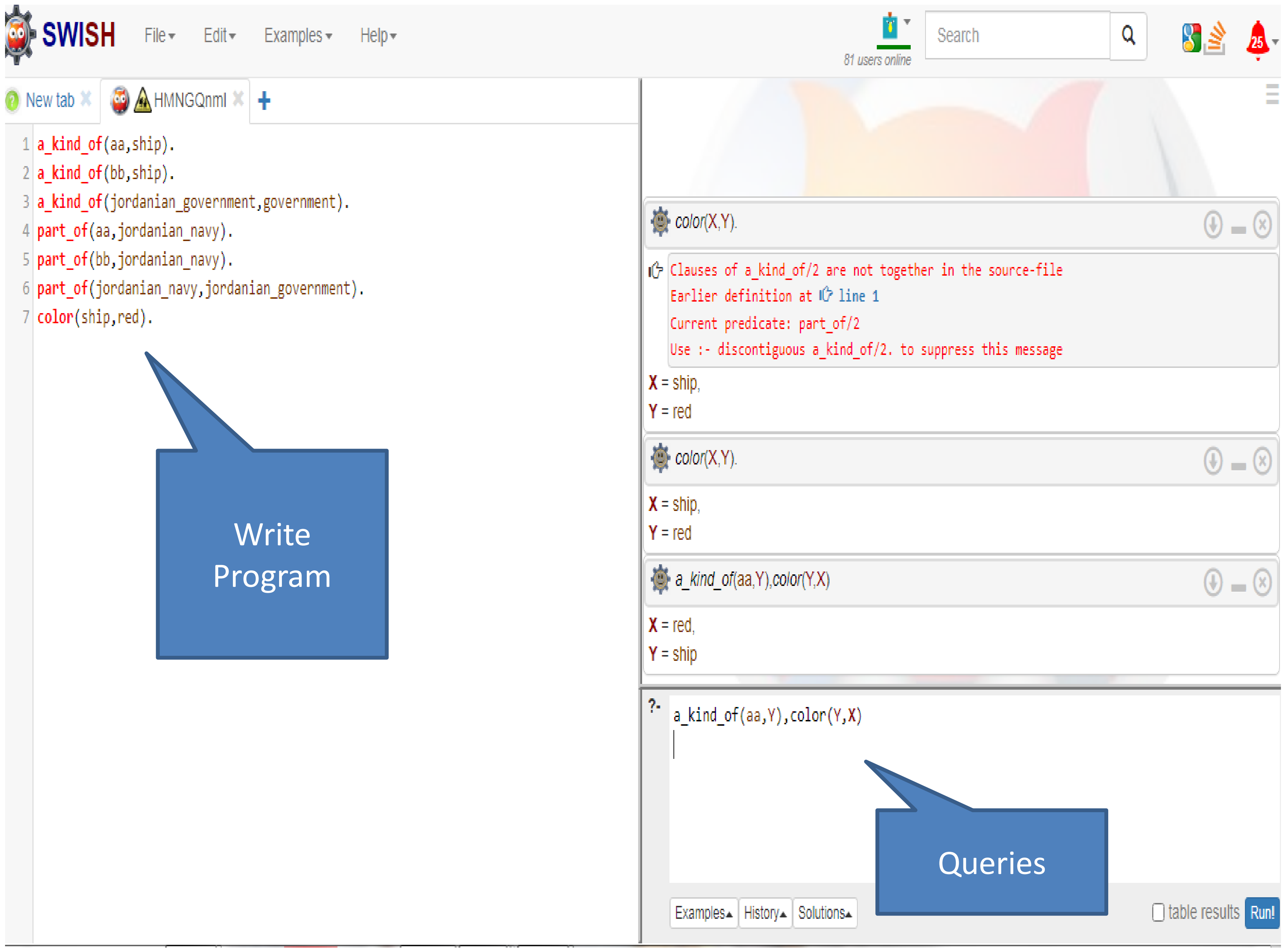
help on search

?- Your query goes here ...



```
Your Prolog rules and facts go here ...
```

```
?- Your query goes here ...
```



```
a_kind_of(aa,ship).
a_kind_of(bb,ship).
a_kind_of(jordanian_government,government).
part_of(aa,jordanian_navy).
part_of(bb,jordanian_navy).
part_of(jordanian_navy,jordanian_government).
color(ship,red).
```

```
color(X,Y).
Clauses of a_kind_of/2 are not together in the source-file
Earlier definition at line 1
Current predicate: part_of/2
Use :- discontiguous a_kind_of/2. to suppress this message
X = ship,
Y = red

color(X,Y).
X = ship,
Y = red

a_kind_of(aa,Y),color(Y,X)
X = red,
Y = ship

?- a_kind_of(aa,Y),color(Y,X)
```