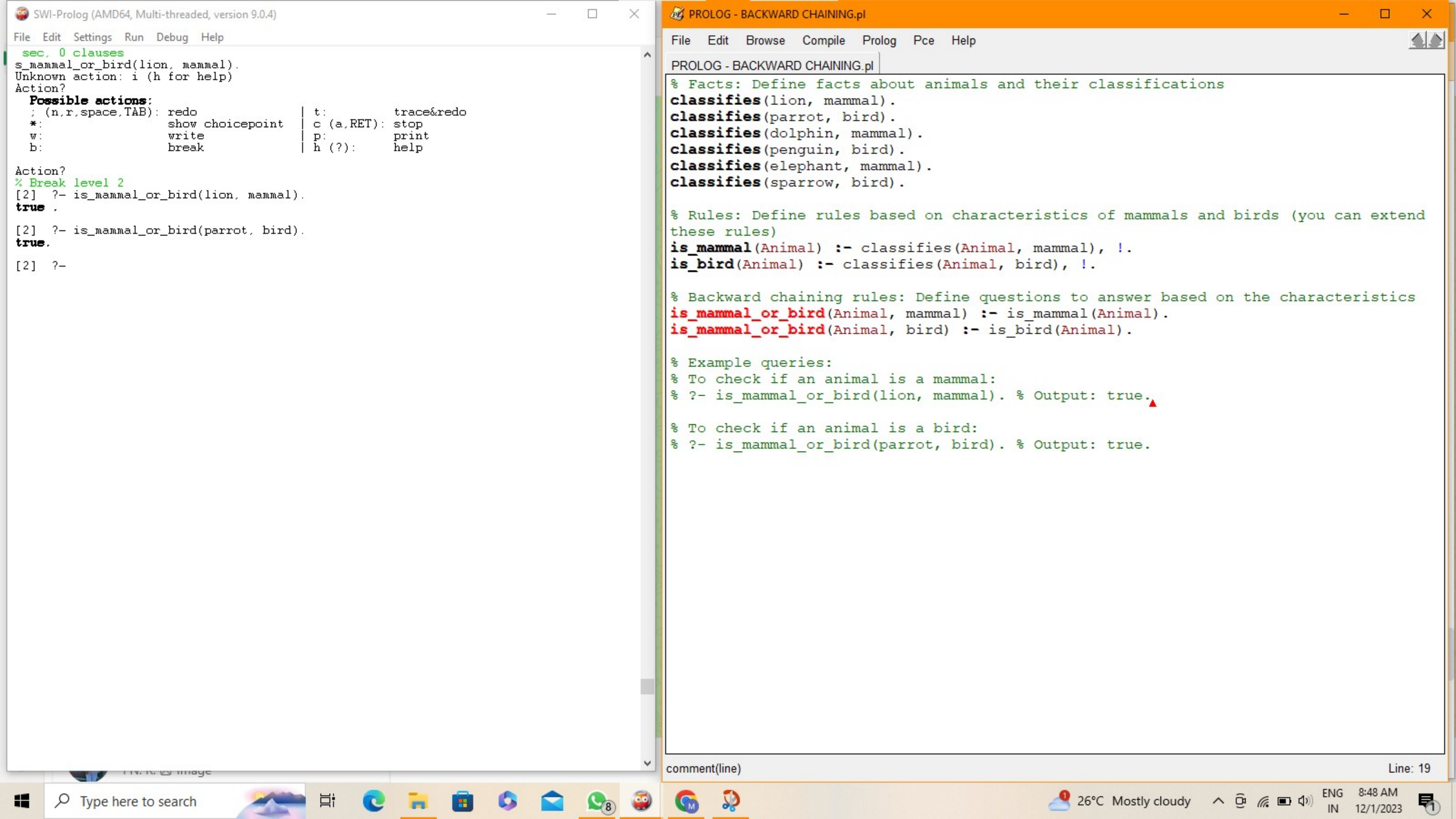
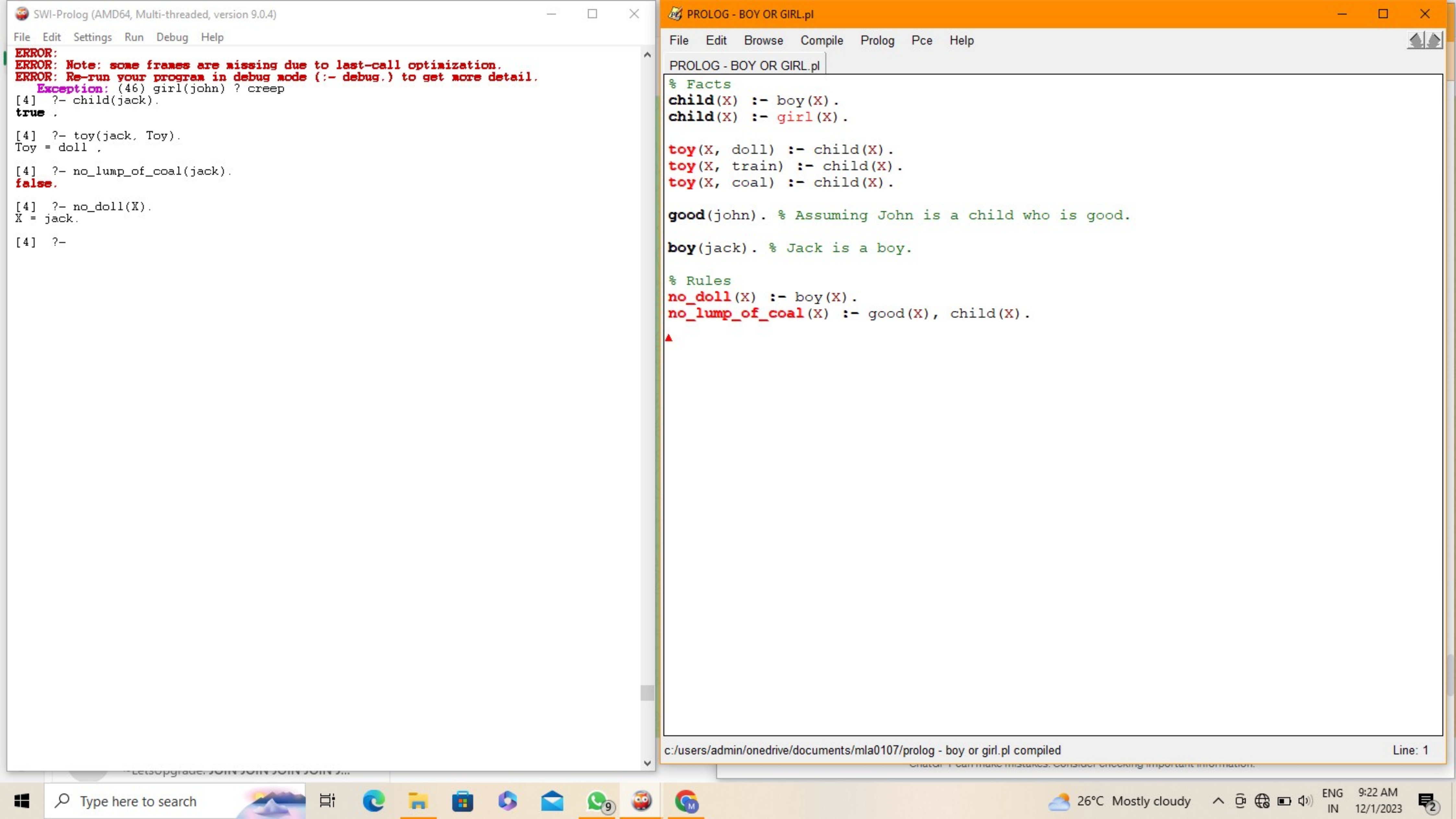

SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - BACKWARD CHAINING.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - BACKWARD CHAINING.pl
comment(line)
Line: 19
Type here to search
26°C Mostly cloudy
ENG IN
8:48 AM 12/1/2023

```
 sec, 0 clauses
s_mammal_or_bird(lion, mammal).
Unknown action: i (h for help)
Action?
  Possible actions:
  ; (n,r,space,TAB): redo              | t:         trace&redo
  *:                 show choicepoint  | c (a,RET): stop
  w:                 write             | p:         print
  b:                 break             | h (?):     help

Action?
% Break level 2
[2]  ?- is_mammal_or_bird(lion, mammal).
true .

[2]  ?- is_mammal_or_bird(parrot, bird).
true.

[2]  ?-
```

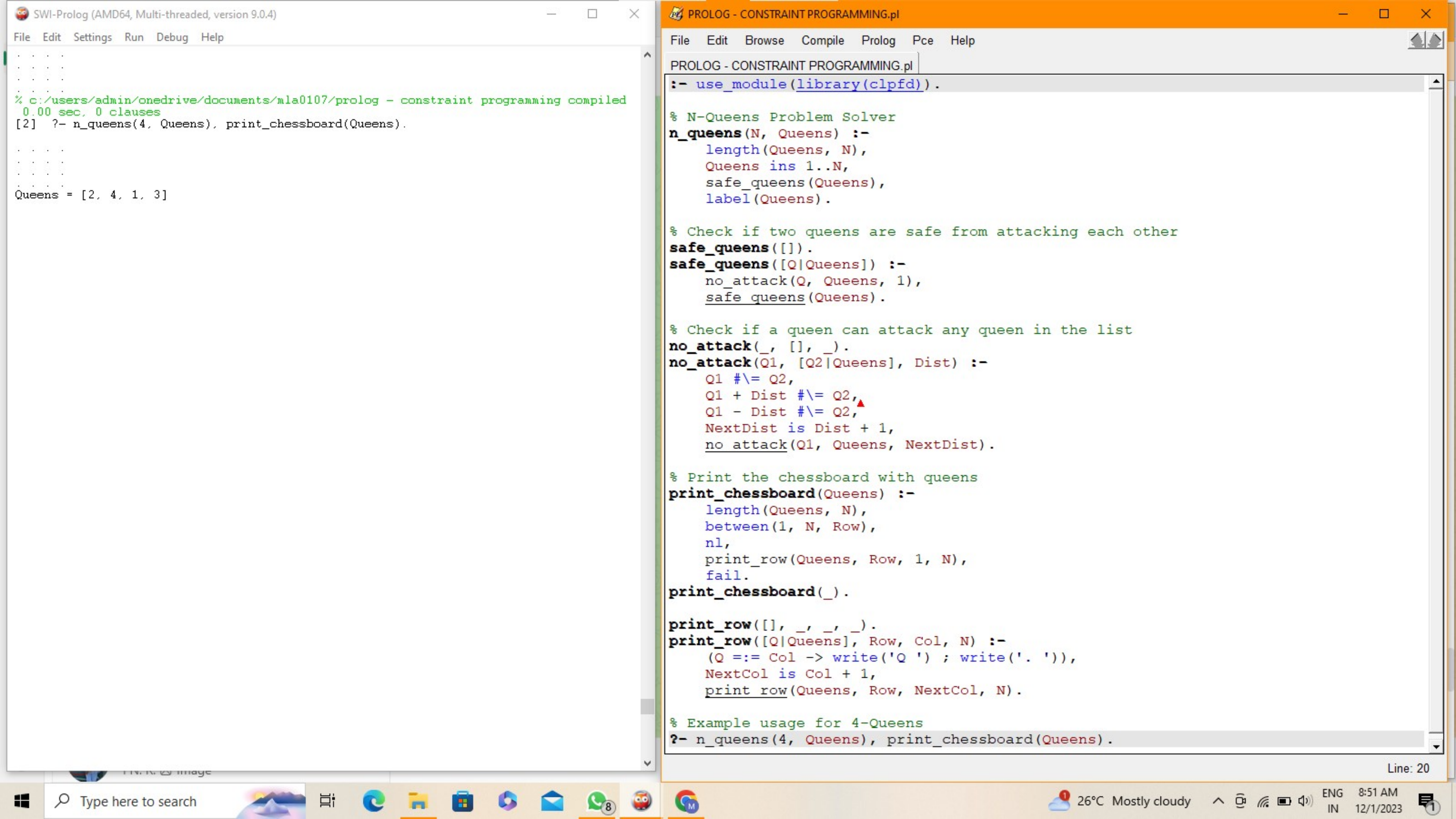
```
% Facts: Define facts about animals and their classifications
classifies(lion, mammal).
classifies(parrot, bird).
classifies(dolphin, mammal).
classifies(penguin, bird).
classifies(elephant, mammal).
classifies(sparrow, bird).

% Rules: Define rules based on characteristics of mammals and birds (you can extend
these rules)
is_mammal(Animal) :- classifies(Animal, mammal), !.
is_bird(Animal) :- classifies(Animal, bird), !.

% Backward chaining rules: Define questions to answer based on the characteristics
is_mammal_or_bird(Animal, mammal) :- is_mammal(Animal).
is_mammal_or_bird(Animal, bird) :- is_bird(Animal).

% Example queries:
% To check if an animal is a mammal:
% ?- is_mammal_or_bird(lion, mammal). % Output: true.

% To check if an animal is a bird:
% ?- is_mammal_or_bird(parrot, bird). % Output: true.
```


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - BOY OR GIRL.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - BOY OR GIRL.pl
Line: 1
Type here to search
26°C Mostly cloudy
ENG
IN
9:22 AM
12/1/2023

```
ERROR:
ERROR: Note: some frames are missing due to last-call optimization.
ERROR: Re-run your program in debug mode (:- debug.) to get more detail.
   Exception: (46) girl(john) ? creep
[4]  ?- child(jack).
true .

[4]  ?- toy(jack, Toy).
Toy = doll ,

[4]  ?- no_lump_of_coal(jack).
false.

[4]  ?- no_doll(X).
X = jack.

[4]  ?-
```

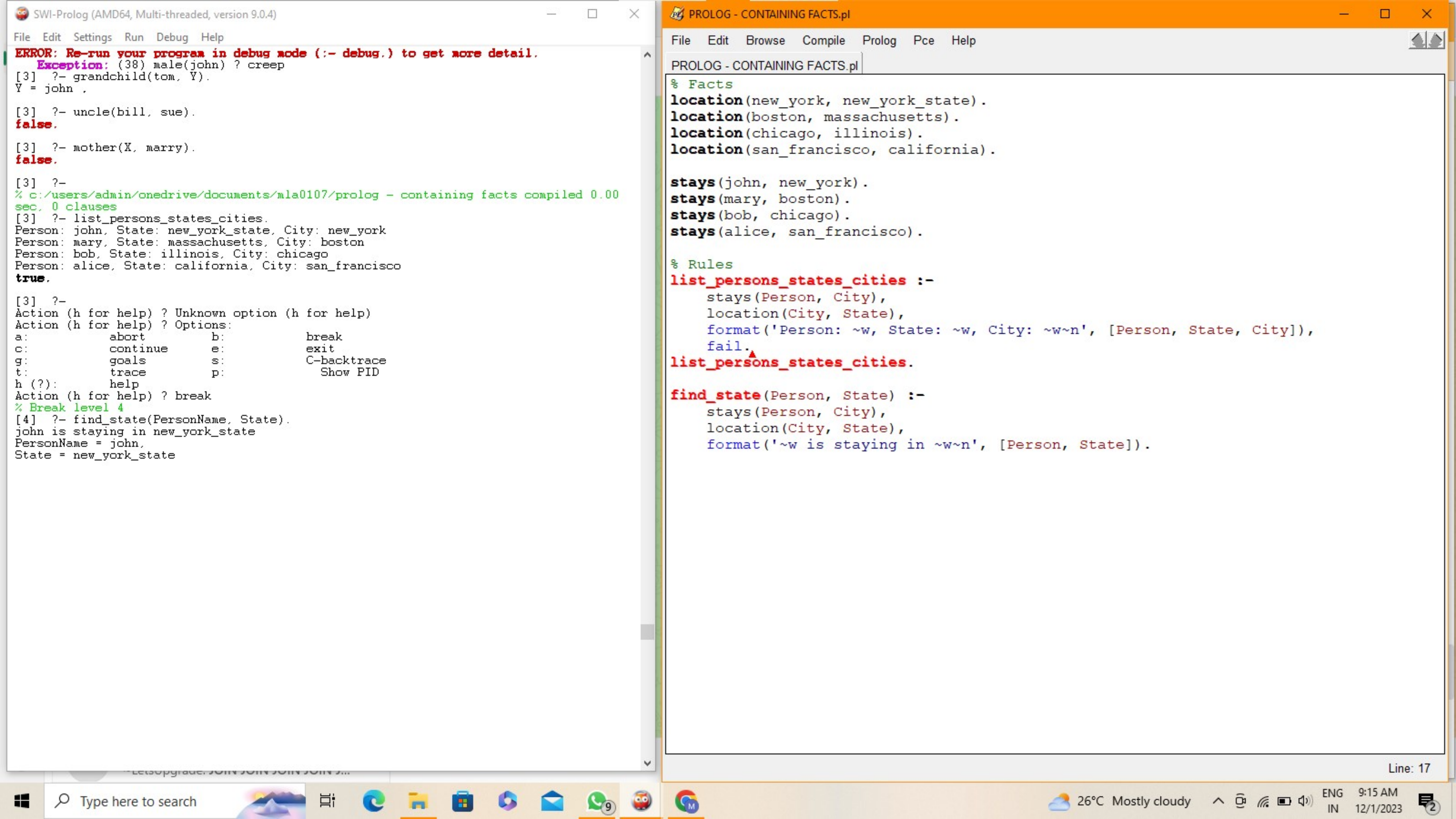
```
% Facts
child(X) :- boy(X).
child(X) :- girl(X).

toy(X, doll) :- child(X).
toy(X, train) :- child(X).
toy(X, coal) :- child(X).

good(john). % Assuming John is a child who is good.

boy(jack). % Jack is a boy.

% Rules
no_doll(X) :- boy(X).
no_lump_of_coal(X) :- good(X), child(X).
```

c:/users/admin/onedrive/documents/mla0107/prolog - boy or girl.pl compiled


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - CONSTRAINT PROGRAMMING.pl
File Edit Browse Compile Prolog Pce Help
Line: 20
Type here to search
26°C Mostly cloudy
ENG IN
8:51 AM 12/1/2023

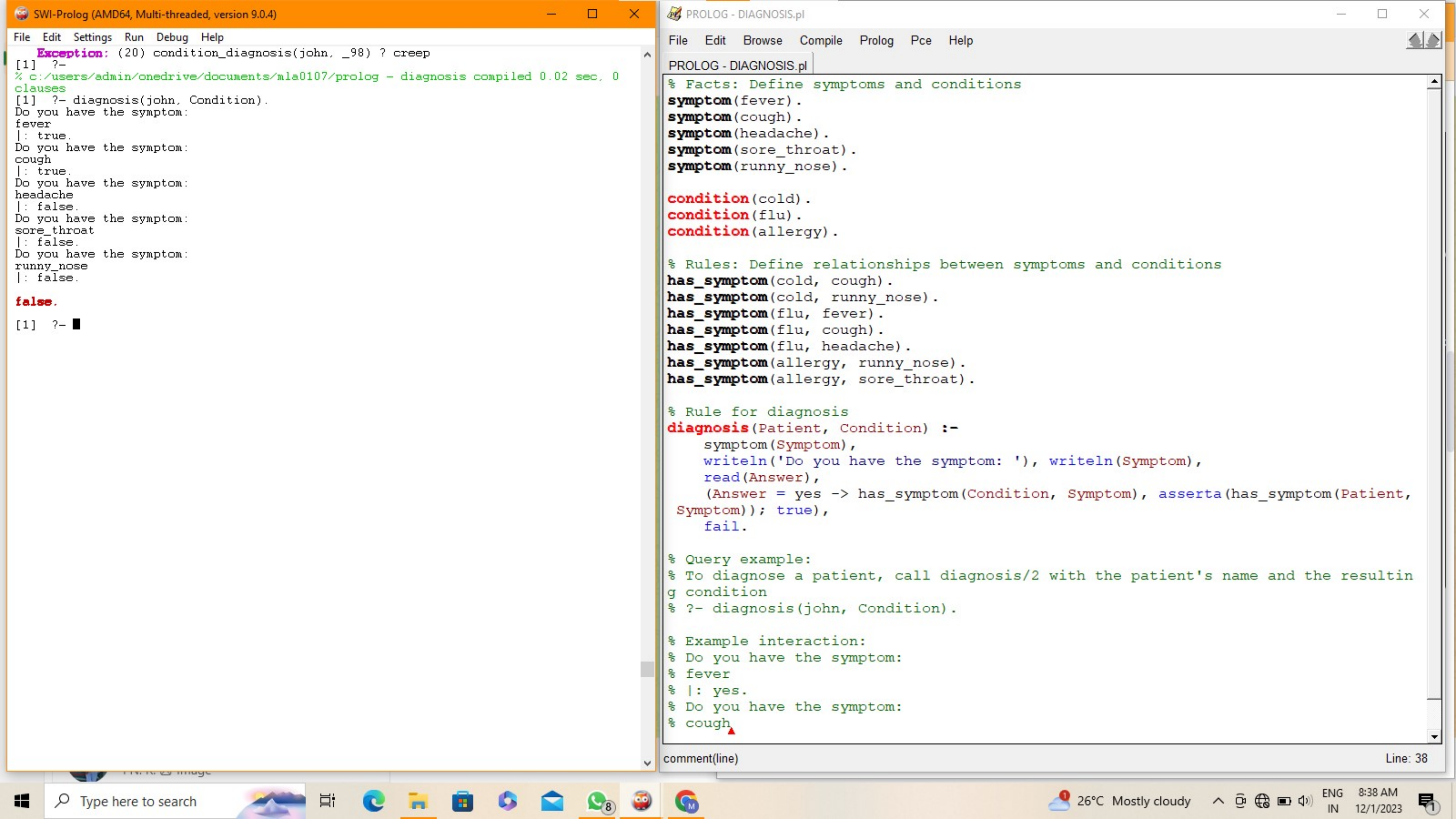
```
. . . .
. . . .
. . . .

. . . .
% c:/users/admin/onedrive/documents/mla0107/prolog - constraint programming compiled
 0.00 sec, 0 clauses
[2]  ?- n_queens(4, Queens), print_chessboard(Queens).

. . . .
. . . .
. . . .

. . . .
Queens = [2, 4, 1, 3]
```

```
:- use_module(library(clpfd)).

% N-Queens Problem Solver
n_queens(N, Queens) :-
    length(Queens, N),
    Queens ins 1..N,
    safe_queens(Queens),
    label(Queens).

% Check if two queens are safe from attacking each other
safe_queens([]).
safe_queens([Q|Queens]) :-
    no_attack(Q, Queens, 1),
    safe_queens(Queens).

% Check if a queen can attack any queen in the list
no_attack(_, [], _).
no_attack(Q1, [Q2|Queens], Dist) :-
    Q1 #\= Q2,
    Q1 + Dist #\= Q2,
    Q1 - Dist #\= Q2,
    NextDist is Dist + 1,
    no_attack(Q1, Queens, NextDist).

% Print the chessboard with queens
print_chessboard(Queens) :-
    length(Queens, N),
    between(1, N, Row),
    nl,
    print_row(Queens, Row, 1, N),
    fail.
print_chessboard(_).

print_row([], _, _, _).
print_row([Q|Queens], Row, Col, N) :-
    (Q =:= Col -> write('Q ') ; write('. ')),
    NextCol is Col + 1,
    print_row(Queens, Row, NextCol, N).

% Example usage for 4-Queens
?- n_queens(4, Queens), print_chessboard(Queens).
```


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - CONTAINING FACTS.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - CONTAINING FACTS.pl
Line: 17
Type here to search
26°C Mostly cloudy
ENG IN
9:15 AM 12/1/2023

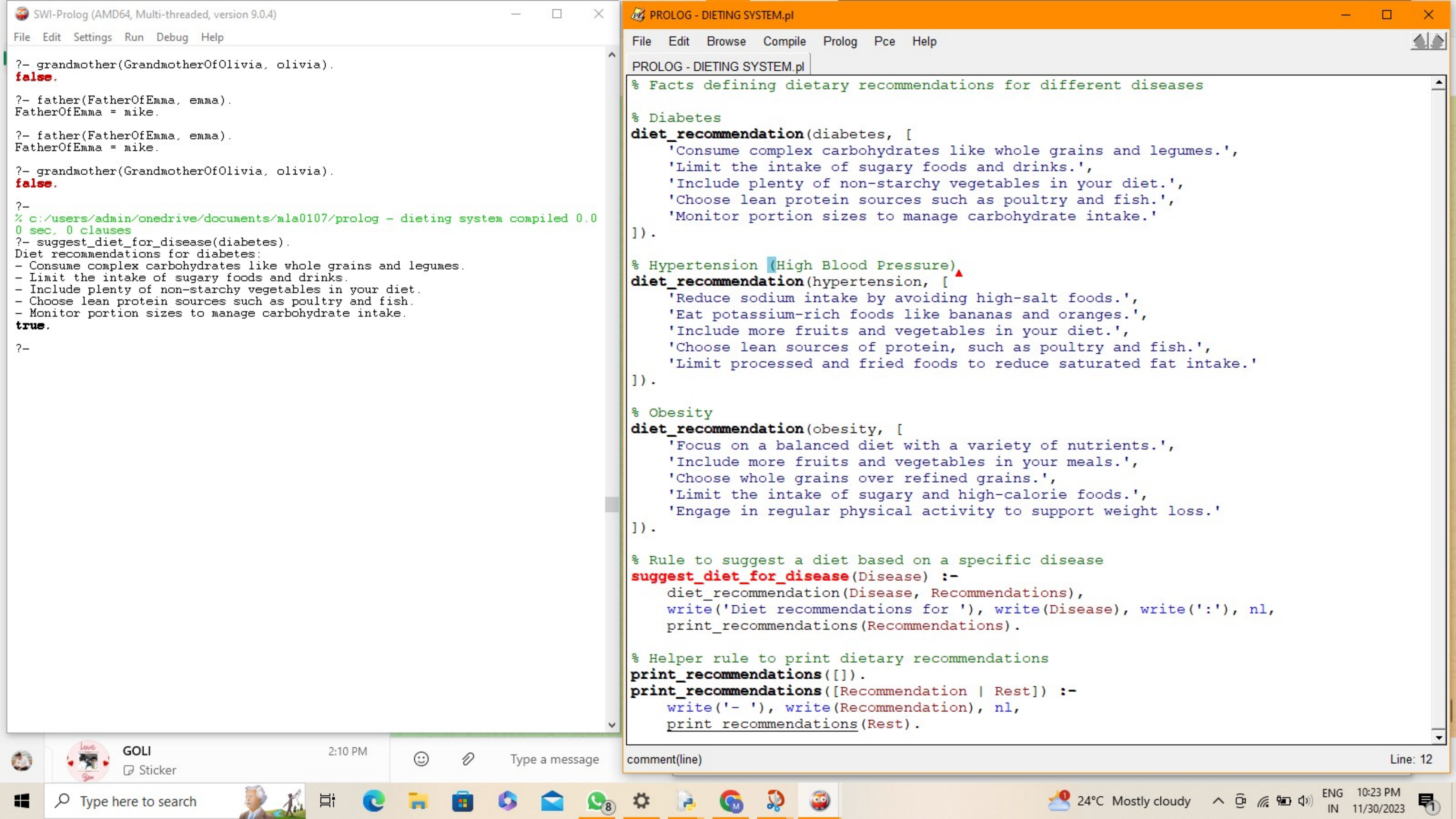
```
ERROR: Re-run your program in debug mode (:- debug.) to get more detail.
   Exception: (38) male(john) ? creep
[3]  ?- grandchild(tom, Y).
Y = john .

[3]  ?- uncle(bill, sue).
false.

[3]  ?- mother(X, marry).
false.

[3]  ?-
% c:/users/admin/onedrive/documents/mla0107/prolog - containing facts compiled 0.00
sec, 0 clauses
[3]  ?- list_persons_states_cities.
Person: john, State: new_york_state, City: new_york
Person: mary, State: massachusetts, City: boston
Person: bob, State: illinois, City: chicago
Person: alice, State: california, City: san_francisco
true.

[3]  ?-
Action (h for help) ? Unknown option (h for help)
Action (h for help) ? Options:
a:            abort          b:           break
c:            continue       e:           exit
g:            goals          s:           C-backtrace
t:            trace          p:             Show PID
h (?):        help
Action (h for help) ? break
% Break level 4
[4]  ?- find_state(PersonName, State).
john is staying in new_york_state
PersonName = john,
State = new_york_state
```

```prolog
% Facts
location(new_york, new_york_state).
location(boston, massachusetts).
location(chicago, illinois).
location(san_francisco, california).

stays(john, new_york).
stays(mary, boston).
stays(bob, chicago).
stays(alice, san_francisco).

% Rules
list_persons_states_cities :-
    stays(Person, City),
    location(City, State),
    format('Person: ~w, State: ~w, City: ~w~n', [Person, State, City]),
    fail.
list_persons_states_cities.

find_state(Person, State) :-
    stays(Person, City),
    location(City, State),
    format('~w is staying in ~w~n', [Person, State]).
```



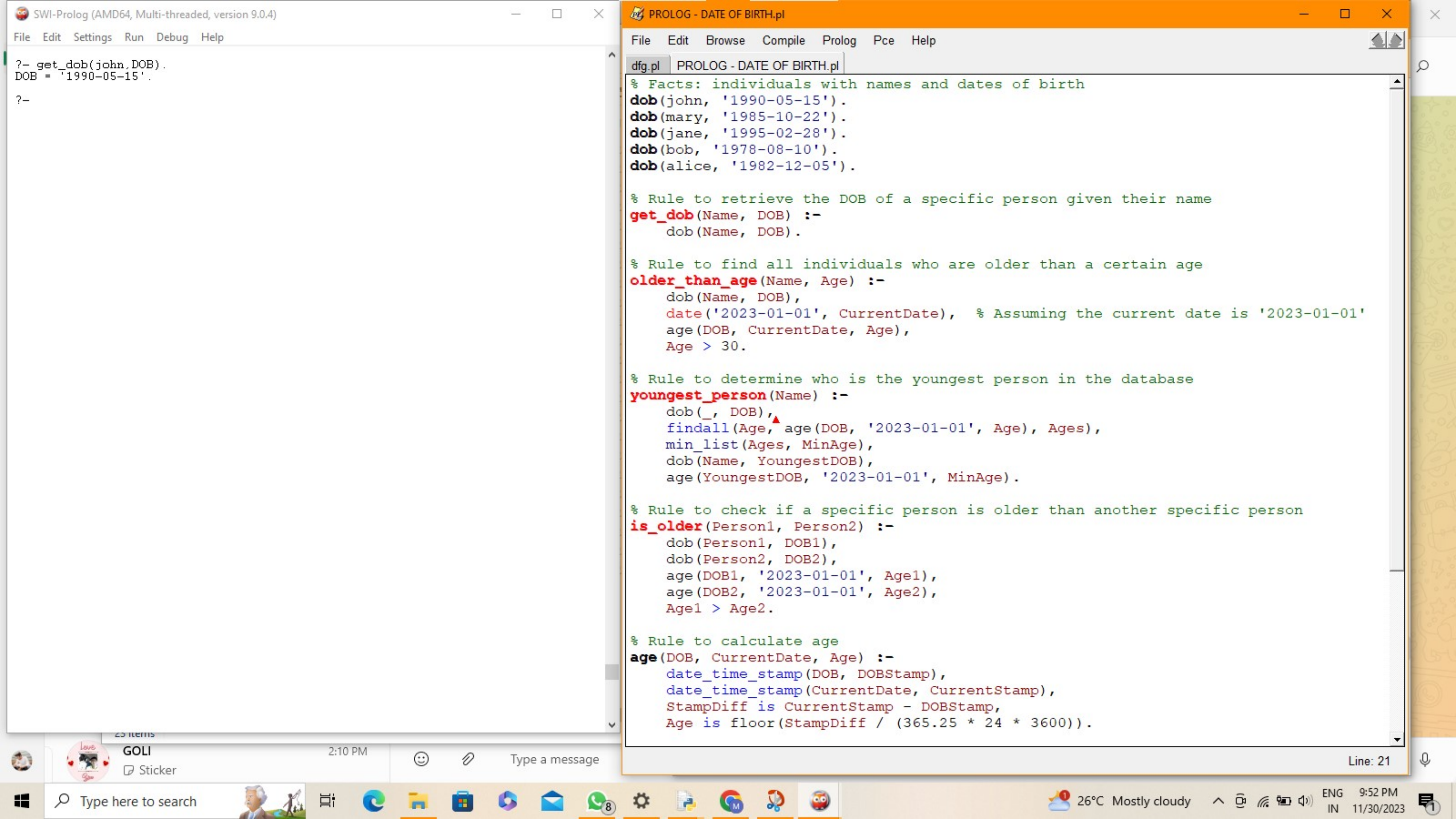
```
   Exception: (20) condition_diagnosis(john, _98) ? creep
[1]  ?-
% c:/users/admin/onedrive/documents/mla0107/prolog - diagnosis compiled 0.02 sec, 0
clauses
[1]  ?- diagnosis(john, Condition).
Do you have the symptom:
fever
|: true.
Do you have the symptom:
cough
|: true.
Do you have the symptom:
headache
|: false.
Do you have the symptom:
sore_throat
|: false.
Do you have the symptom:
runny_nose
|: false.

false.

[1]  ?-
```

```
% Facts: Define symptoms and conditions
symptom(fever).
symptom(cough).
symptom(headache).
symptom(sore_throat).
symptom(runny_nose).

condition(cold).
condition(flu).
condition(allergy).

% Rules: Define relationships between symptoms and conditions
has_symptom(cold, cough).
has_symptom(cold, runny_nose).
has_symptom(flu, fever).
has_symptom(flu, cough).
has_symptom(flu, headache).
has_symptom(allergy, runny_nose).
has_symptom(allergy, sore_throat).

% Rule for diagnosis
diagnosis(Patient, Condition) :-
    symptom(Symptom),
    writeln('Do you have the symptom: '), writeln(Symptom),
    read(Answer),
    (Answer = yes -> has_symptom(Condition, Symptom), asserta(has_symptom(Patient,
 Symptom)); true),
    fail.

% Query example:
% To diagnose a patient, call diagnosis/2 with the patient's name and the resultin
g condition
% ?- diagnosis(john, Condition).

% Example interaction:
% Do you have the symptom:
% fever
% |: yes.
% Do you have the symptom:
% cough
```


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - DIETING SYSTEM.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - DIETING SYSTEM.pl
comment(line)
Line: 12
GOLI
Sticker
2:10 PM
Type a message
Type here to search
24°C Mostly cloudy
ENG
IN
10:23 PM
11/30/2023

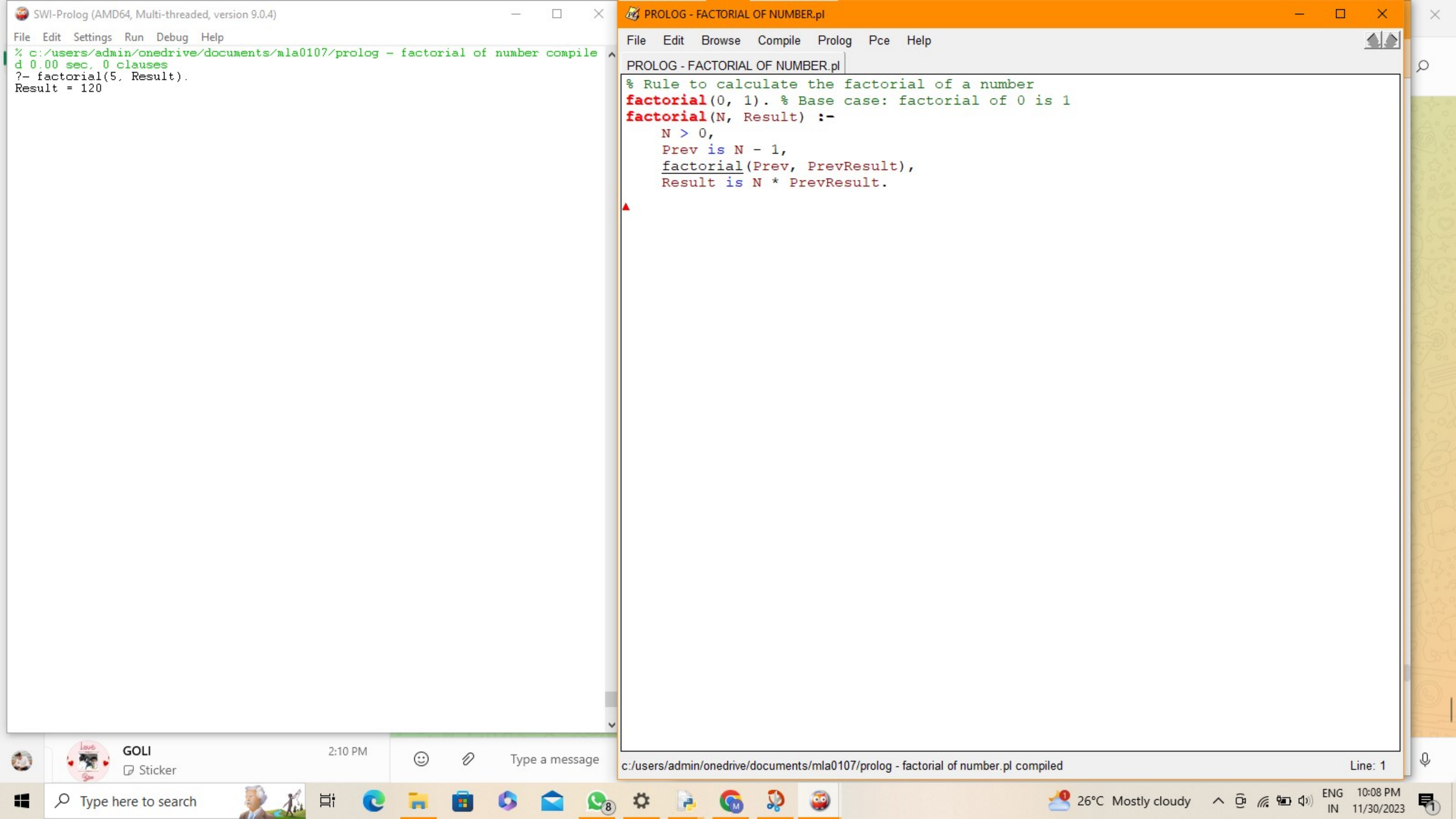
```
?- grandmother(GrandmotherOfOlivia, olivia).
false.

?- father(FatherOfEmma, emma).
FatherOfEmma = mike.

?- father(FatherOfEmma, emma).
FatherOfEmma = mike.

?- grandmother(GrandmotherOfOlivia, olivia).
false.

?-
% c:/users/admin/onedrive/documents/mla0107/prolog - dieting system compiled 0.0
0 sec, 0 clauses
?- suggest_diet_for_disease(diabetes).
Diet recommendations for diabetes:
- Consume complex carbohydrates like whole grains and legumes.
- Limit the intake of sugary foods and drinks.
- Include plenty of non-starchy vegetables in your diet.
- Choose lean protein sources such as poultry and fish.
- Monitor portion sizes to manage carbohydrate intake.
true.

?-
```

```
% Facts defining dietary recommendations for different diseases

% Diabetes
diet_recommendation(diabetes, [
    'Consume complex carbohydrates like whole grains and legumes.',
    'Limit the intake of sugary foods and drinks.',
    'Include plenty of non-starchy vegetables in your diet.',
    'Choose lean protein sources such as poultry and fish.',
    'Monitor portion sizes to manage carbohydrate intake.'
]).

% Hypertension (High Blood Pressure)
diet_recommendation(hypertension, [
    'Reduce sodium intake by avoiding high-salt foods.',
    'Eat potassium-rich foods like bananas and oranges.',
    'Include more fruits and vegetables in your diet.',
    'Choose lean sources of protein, such as poultry and fish.',
    'Limit processed and fried foods to reduce saturated fat intake.'
]).

% Obesity
diet_recommendation(obesity, [
    'Focus on a balanced diet with a variety of nutrients.',
    'Include more fruits and vegetables in your meals.',
    'Choose whole grains over refined grains.',
    'Limit the intake of sugary and high-calorie foods.',
    'Engage in regular physical activity to support weight loss.'
]).

% Rule to suggest a diet based on a specific disease
suggest_diet_for_disease(Disease) :-
    diet_recommendation(Disease, Recommendations),
    write('Diet recommendations for '), write(Disease), write(':'), nl,
    print_recommendations(Recommendations).

% Helper rule to print dietary recommendations
print_recommendations([]).
print_recommendations([Recommendation | Rest]) :-
    write('- '), write(Recommendation), nl,
    print_recommendations(Rest).
```


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - DATE OF BIRTH.pl
File Edit Browse Compile Prolog Pce Help
dfg.pl
PROLOG - DATE OF BIRTH.pl
Line: 21
GOLI
Sticker
2:10 PM
Type a message
Type here to search
26°C Mostly cloudy
ENG
IN
9:52 PM
11/30/2023

```
?- get_dob(john,DOB).
DOB = '1990-05-15'.

?-
```

```
% Facts: individuals with names and dates of birth
dob(john, '1990-05-15').
dob(mary, '1985-10-22').
dob(jane, '1995-02-28').
dob(bob, '1978-08-10').
dob(alice, '1982-12-05').

% Rule to retrieve the DOB of a specific person given their name
get_dob(Name, DOB) :-
    dob(Name, DOB).

% Rule to find all individuals who are older than a certain age
older_than_age(Name, Age) :-
    dob(Name, DOB),
    date('2023-01-01', CurrentDate),  % Assuming the current date is '2023-01-01'
    age(DOB, CurrentDate, Age),
    Age > 30.

% Rule to determine who is the youngest person in the database
youngest_person(Name) :-
    dob(_, DOB),
    findall(Age, age(DOB, '2023-01-01', Age), Ages),
    min_list(Ages, MinAge),
    dob(Name, YoungestDOB),
    age(YoungestDOB, '2023-01-01', MinAge).

% Rule to check if a specific person is older than another specific person
is_older(Person1, Person2) :-
    dob(Person1, DOB1),
    dob(Person2, DOB2),
    age(DOB1, '2023-01-01', Age1),
    age(DOB2, '2023-01-01', Age2),
    Age1 > Age2.

% Rule to calculate age
age(DOB, CurrentDate, Age) :-
    date_time_stamp(DOB, DOBStamp),
    date_time_stamp(CurrentDate, CurrentStamp),
    StampDiff is CurrentStamp - DOBStamp,
    Age is floor(StampDiff / (365.25 * 24 * 3600)).
```

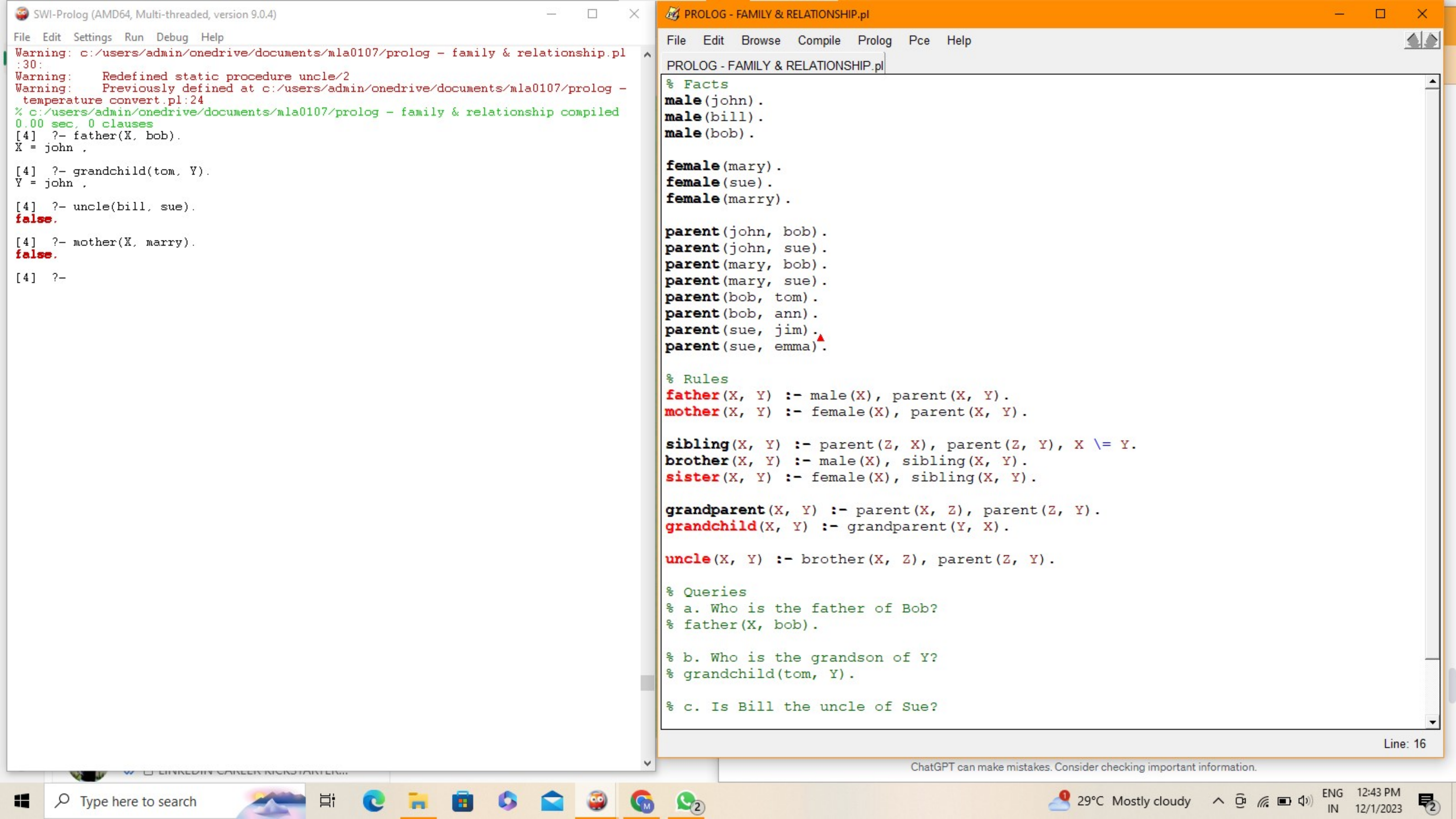

SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - FACTORIAL OF NUMBER.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - FACTORIAL OF NUMBER.pl
GOLI
Sticker
2:10 PM
Type a message
Type here to search
26°C Mostly cloudy
ENG
IN
10:08 PM
11/30/2023

```
% c:/users/admin/onedrive/documents/mla0107/prolog - factorial of number compile
d 0.00 sec, 0 clauses
?- factorial(5, Result).
Result = 120
```

```
% Rule to calculate the factorial of a number
factorial(0, 1). % Base case: factorial of 0 is 1
factorial(N, Result) :-
    N > 0,
    Prev is N - 1,
    factorial(Prev, PrevResult),
    Result is N * PrevResult.
```

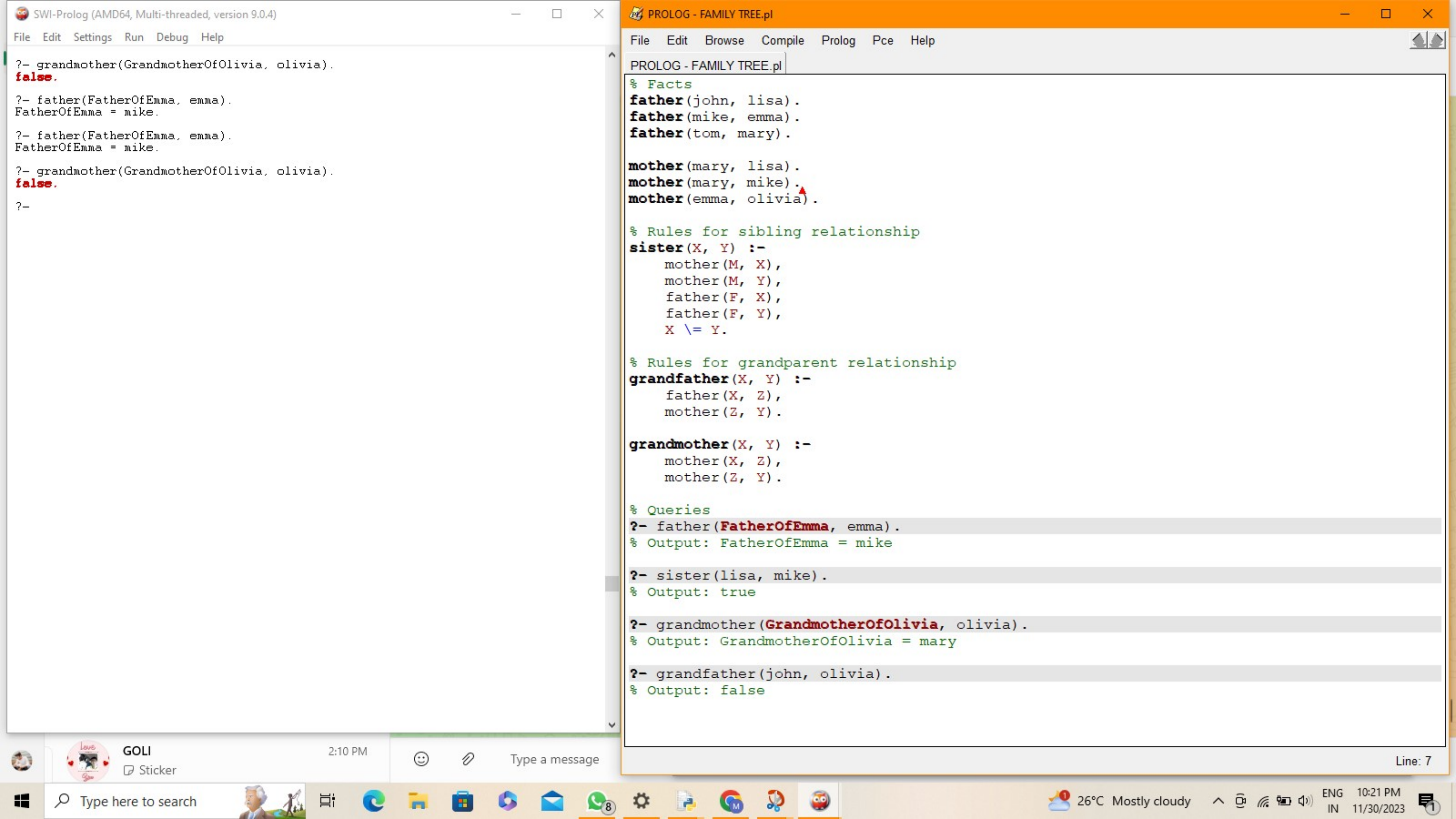
c:/users/admin/onedrive/documents/mla0107/prolog - factorial of number.pl compiled

Line: 1


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - FAMILY & RELATIONSHIP.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - FAMILY & RELATIONSHIP.pl
Line: 16
ChatGPT can make mistakes. Consider checking important information.
Type here to search
29°C Mostly cloudy
ENG
IN
12:43 PM
12/1/2023

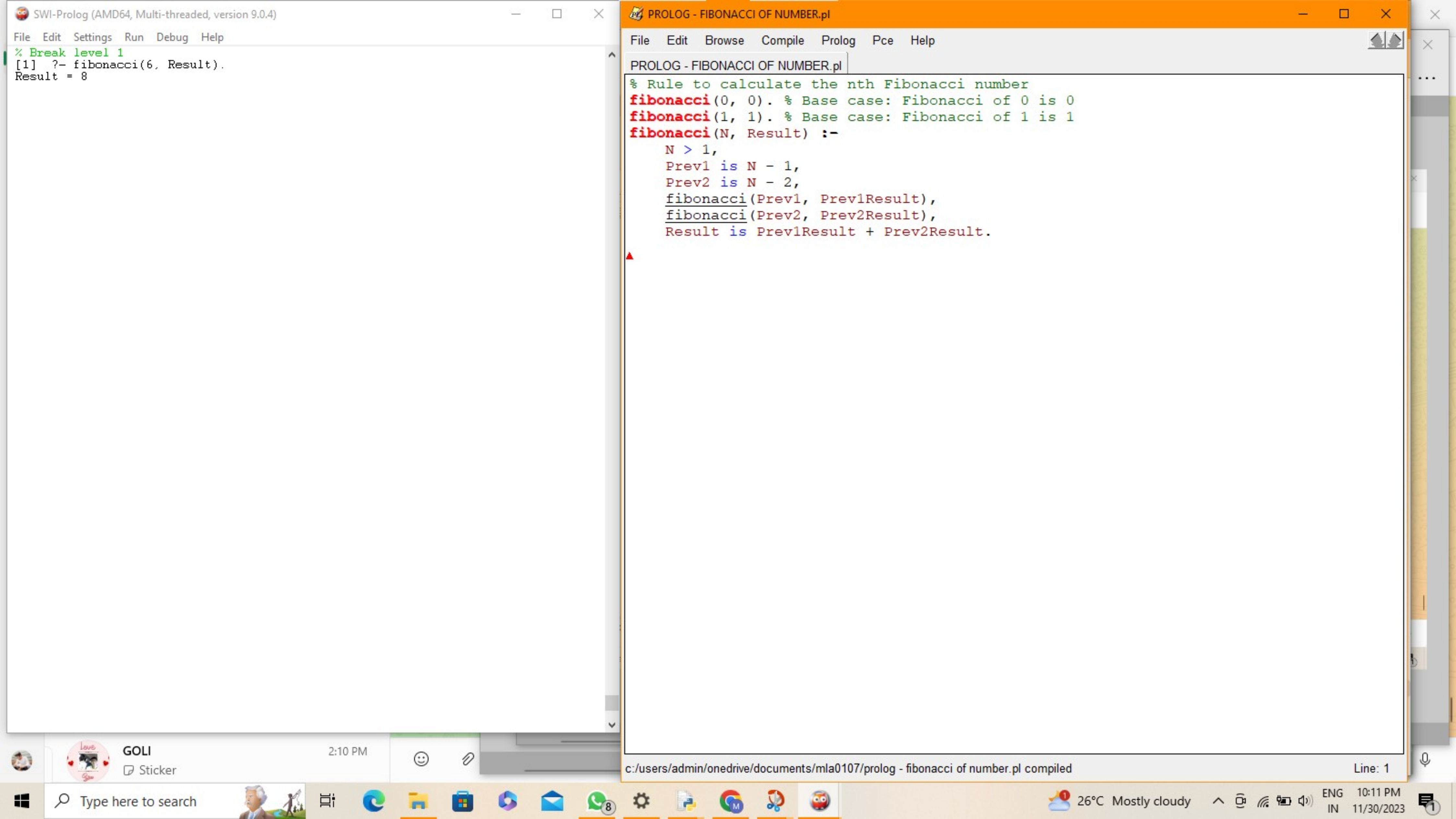
```
Warning: c:/users/admin/onedrive/documents/mla0107/prolog - family & relationship.pl
:30:
Warning:    Redefined static procedure uncle/2
Warning:    Previously defined at c:/users/admin/onedrive/documents/mla0107/prolog -
 temperature convert.pl:24
% c:/users/admin/onedrive/documents/mla0107/prolog - family & relationship compiled
0.00 sec, 0 clauses
[4]  ?- father(X, bob).
X = john ,

[4]  ?- grandchild(tom, Y).
Y = john ,

[4]  ?- uncle(bill, sue).
false.

[4]  ?- mother(X, marry).
false.

[4]  ?-
```

```
% Facts
male(john).
male(bill).
male(bob).

female(mary).
female(sue).
female(marry).

parent(john, bob).
parent(john, sue).
parent(mary, bob).
parent(mary, sue).
parent(bob, tom).
parent(bob, ann).
parent(sue, jim).
parent(sue, emma).

% Rules
father(X, Y) :- male(X), parent(X, Y).
mother(X, Y) :- female(X), parent(X, Y).

sibling(X, Y) :- parent(Z, X), parent(Z, Y), X \= Y.
brother(X, Y) :- male(X), sibling(X, Y).
sister(X, Y) :- female(X), sibling(X, Y).

grandparent(X, Y) :- parent(X, Z), parent(Z, Y).
grandchild(X, Y) :- grandparent(Y, X).

uncle(X, Y) :- brother(X, Z), parent(Z, Y).

% Queries
% a. Who is the father of Bob?
% father(X, bob).

% b. Who is the grandson of Y?
% grandchild(tom, Y).

% c. Is Bill the uncle of Sue?
```



```
?- grandmother(GrandmotherOfOlivia, olivia).
false.

?- father(FatherOfEmma, emma).
FatherOfEmma = mike.

?- father(FatherOfEmma, emma).
FatherOfEmma = mike.

?- grandmother(GrandmotherOfOlivia, olivia).
false.

?-
```

```
% Facts
father(john, lisa).
father(mike, emma).
father(tom, mary).

mother(mary, lisa).
mother(mary, mike).
mother(emma, olivia).

% Rules for sibling relationship
sister(X, Y) :-
    mother(M, X),
    mother(M, Y),
    father(F, X),
    father(F, Y),
    X \= Y.

% Rules for grandparent relationship
grandfather(X, Y) :-
    father(X, Z),
    mother(Z, Y).

grandmother(X, Y) :-
    mother(X, Z),
    mother(Z, Y).

% Queries
?- father(FatherOfEmma, emma).
% Output: FatherOfEmma = mike

?- sister(lisa, mike).
% Output: true

?- grandmother(GrandmotherOfOlivia, olivia).
% Output: GrandmotherOfOlivia = mary

?- grandfather(john, olivia).
% Output: false
```



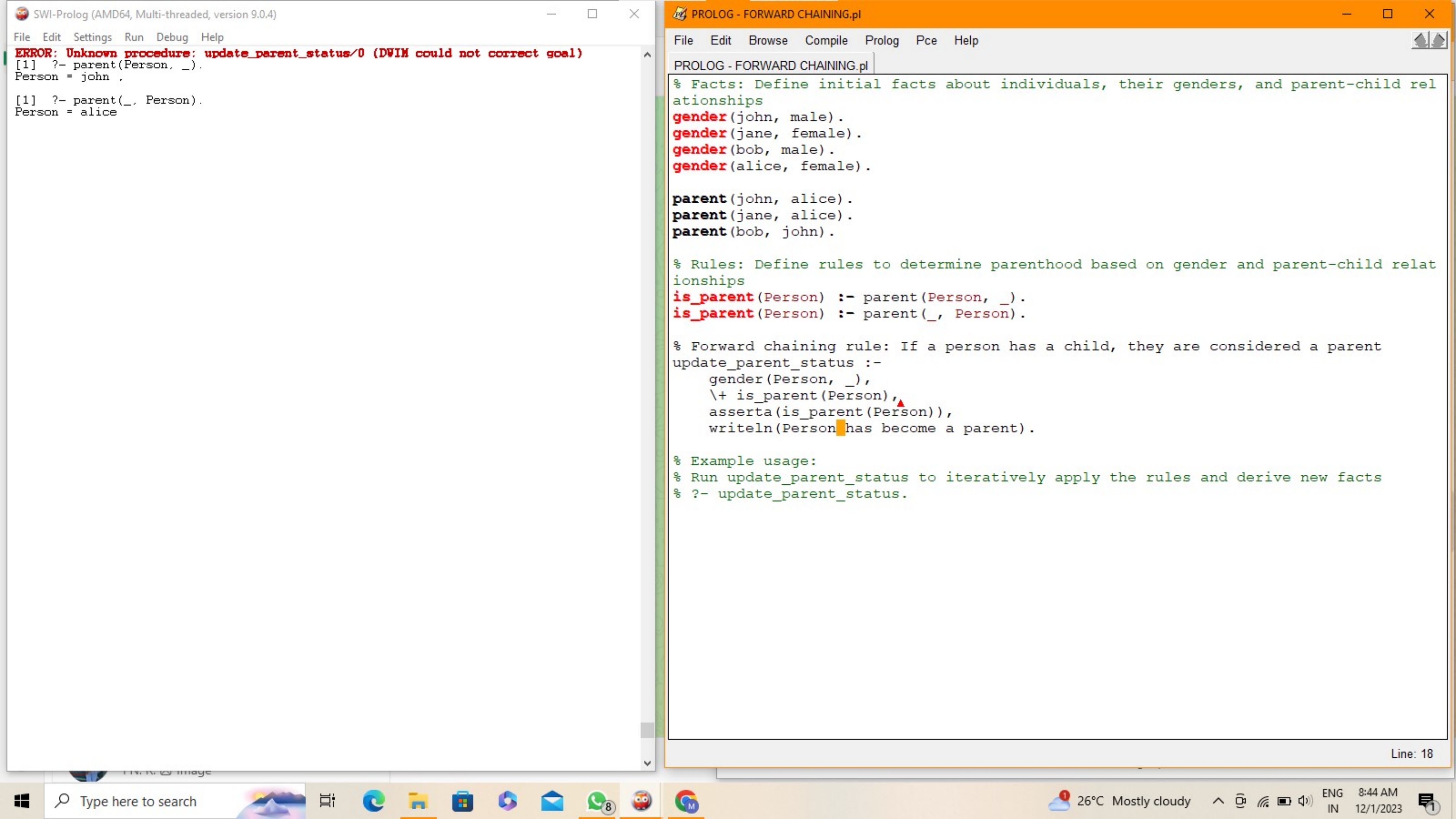
SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help

```
% Break level 1
[1]  ?- fibonacci(6, Result).
Result = 8
```

PROLOG - FIBONACCI OF NUMBER.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - FIBONACCI OF NUMBER.pl

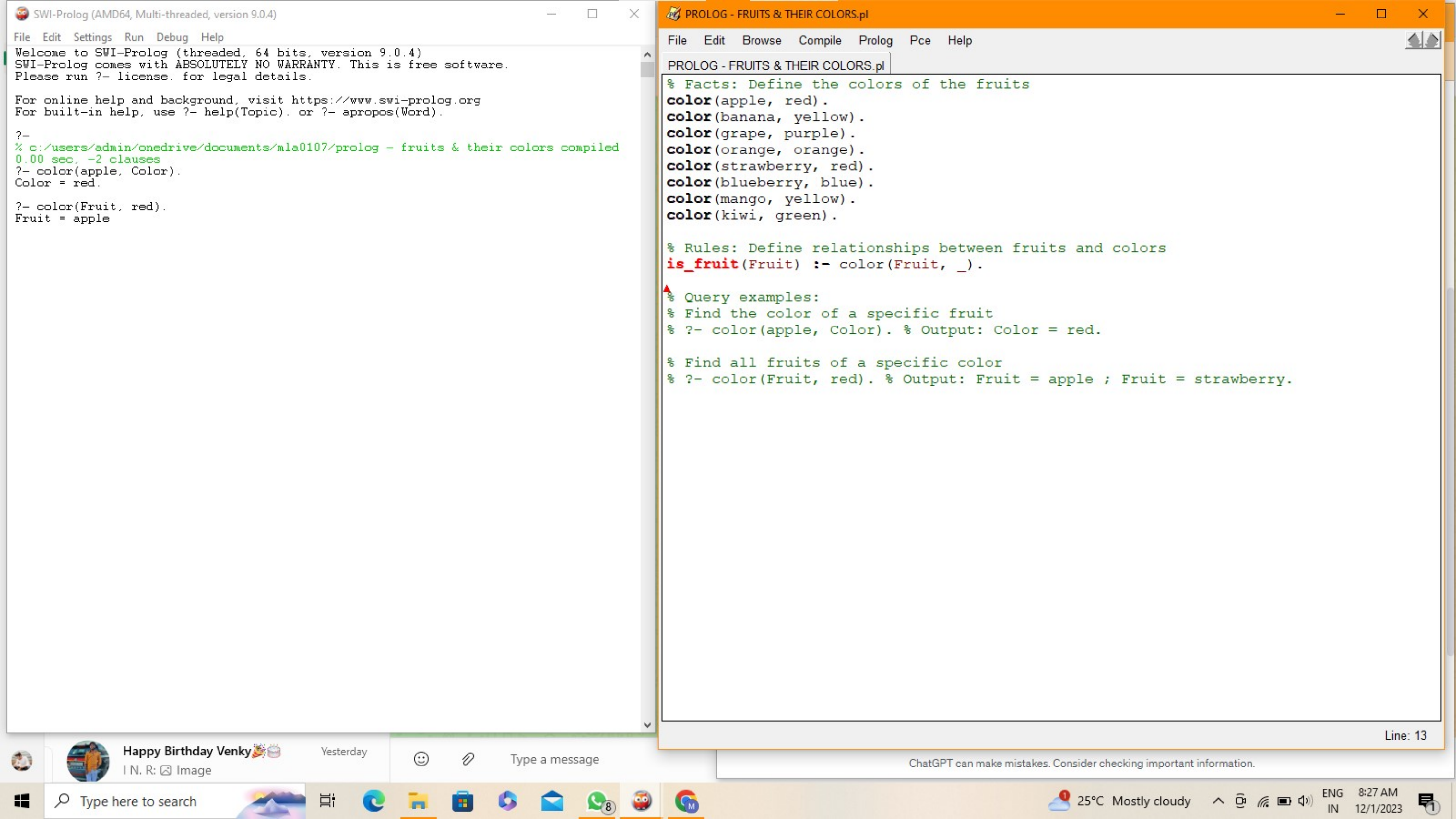
```
% Rule to calculate the nth Fibonacci number
fibonacci(0, 0). % Base case: Fibonacci of 0 is 0
fibonacci(1, 1). % Base case: Fibonacci of 1 is 1
fibonacci(N, Result) :-
    N > 1,
    Prev1 is N - 1,
    Prev2 is N - 2,
    fibonacci(Prev1, Prev1Result),
    fibonacci(Prev2, Prev2Result),
    Result is Prev1Result + Prev2Result.
```

c:/users/admin/onedrive/documents/mla0107/prolog - fibonacci of number.pl compiled



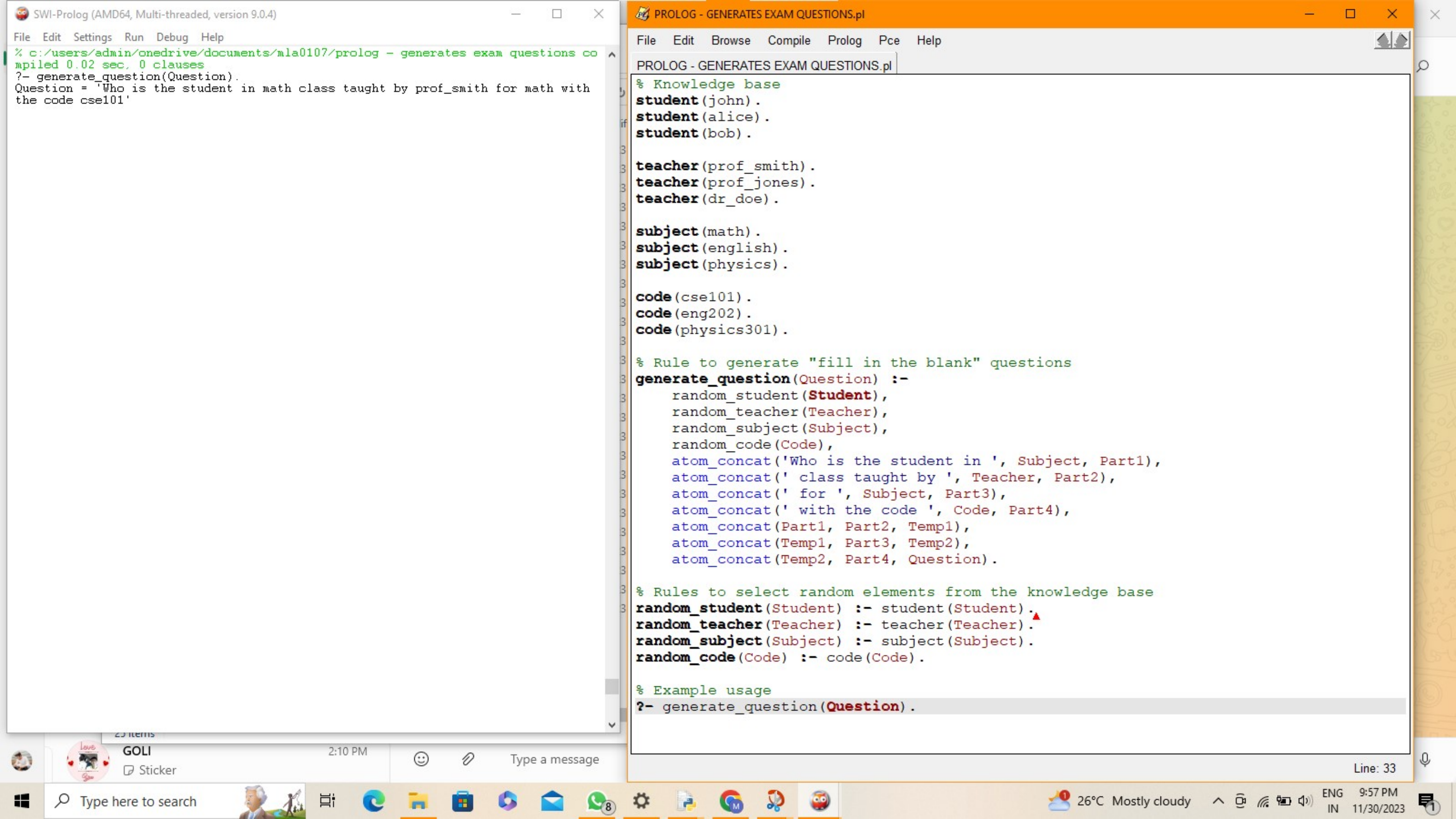
```
ERROR: Unknown procedure: update_parent_status/0 (DWIM could not correct goal)
[1]  ?- parent(Person, _).
Person = john ,

[1]  ?- parent(_, Person).
Person = alice
```

```
% Facts: Define initial facts about individuals, their genders, and parent-child relationships
gender(john, male).
gender(jane, female).
gender(bob, male).
gender(alice, female).

parent(john, alice).
parent(jane, alice).
parent(bob, john).

% Rules: Define rules to determine parenthood based on gender and parent-child relationships
is_parent(Person) :- parent(Person, _).
is_parent(Person) :- parent(_, Person).

% Forward chaining rule: If a person has a child, they are considered a parent
update_parent_status :-
    gender(Person, _),
    \+ is_parent(Person),
    asserta(is_parent(Person)),
    writeln(Person has become a parent).

% Example usage:
% Run update_parent_status to iteratively apply the rules and derive new facts
% ?- update_parent_status.
```


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - FRUITS & THEIR COLORS.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - FRUITS & THEIR COLORS.pl
Line: 13
Happy Birthday Venky
I N. R: Image
Yesterday
Type a message
ChatGPT can make mistakes. Consider checking important information.
Type here to search
25°C Mostly cloudy
ENG
IN
8:27 AM
12/1/2023

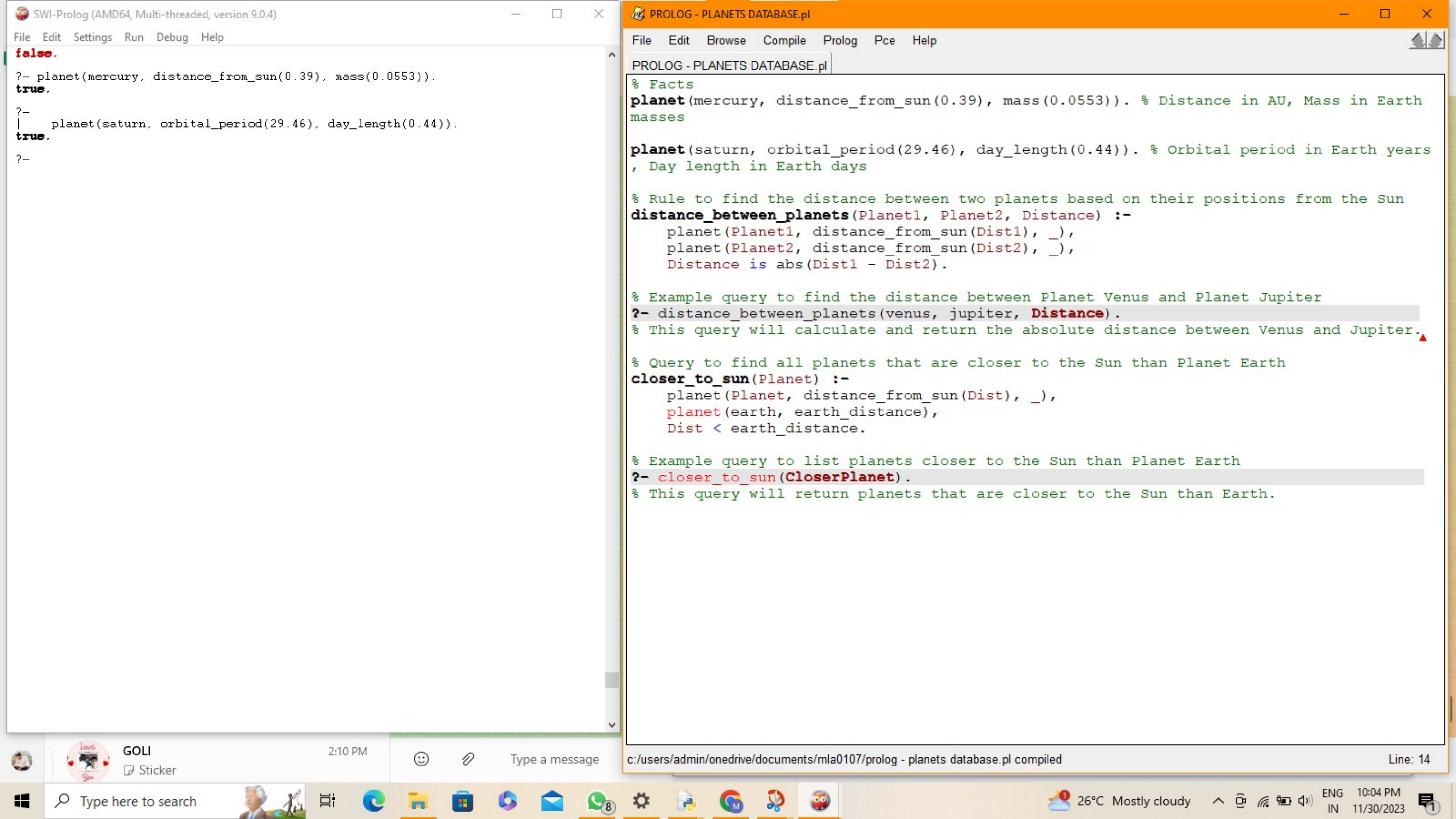
```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
% c:/users/admin/onedrive/documents/mla0107/prolog - fruits & their colors compiled
0.00 sec, -2 clauses
?- color(apple, Color).
Color = red.

?- color(Fruit, red).
Fruit = apple
```

```
% Facts: Define the colors of the fruits
color(apple, red).
color(banana, yellow).
color(grape, purple).
color(orange, orange).
color(strawberry, red).
color(blueberry, blue).
color(mango, yellow).
color(kiwi, green).

% Rules: Define relationships between fruits and colors
is_fruit(Fruit) :- color(Fruit, _).

% Query examples:
% Find the color of a specific fruit
% ?- color(apple, Color). % Output: Color = red.

% Find all fruits of a specific color
% ?- color(Fruit, red). % Output: Fruit = apple ; Fruit = strawberry.
```



```
% c:/users/admin/onedrive/documents/mla0107/prolog - generates exam questions co
mpiled 0.02 sec, 0 clauses
?- generate_question(Question).
Question = 'Who is the student in math class taught by prof_smith for math with
the code cse101'
```

```
% Knowledge base
student(john).
student(alice).
student(bob).

teacher(prof_smith).
teacher(prof_jones).
teacher(dr_doe).

subject(math).
subject(english).
subject(physics).

code(cse101).
code(eng202).
code(physics301).

% Rule to generate "fill in the blank" questions
generate_question(Question) :-
    random_student(Student),
    random_teacher(Teacher),
    random_subject(Subject),
    random_code(Code),
    atom_concat('Who is the student in ', Subject, Part1),
    atom_concat(' class taught by ', Teacher, Part2),
    atom_concat(' for ', Subject, Part3),
    atom_concat(' with the code ', Code, Part4),
    atom_concat(Part1, Part2, Temp1),
    atom_concat(Temp1, Part3, Temp2),
    atom_concat(Temp2, Part4, Question).

% Rules to select random elements from the knowledge base
random_student(Student) :- student(Student).
random_teacher(Teacher) :- teacher(Teacher).
random_subject(Subject) :- subject(Subject).
random_code(Code) :- code(Code).

% Example usage
?- generate_question(Question).
```

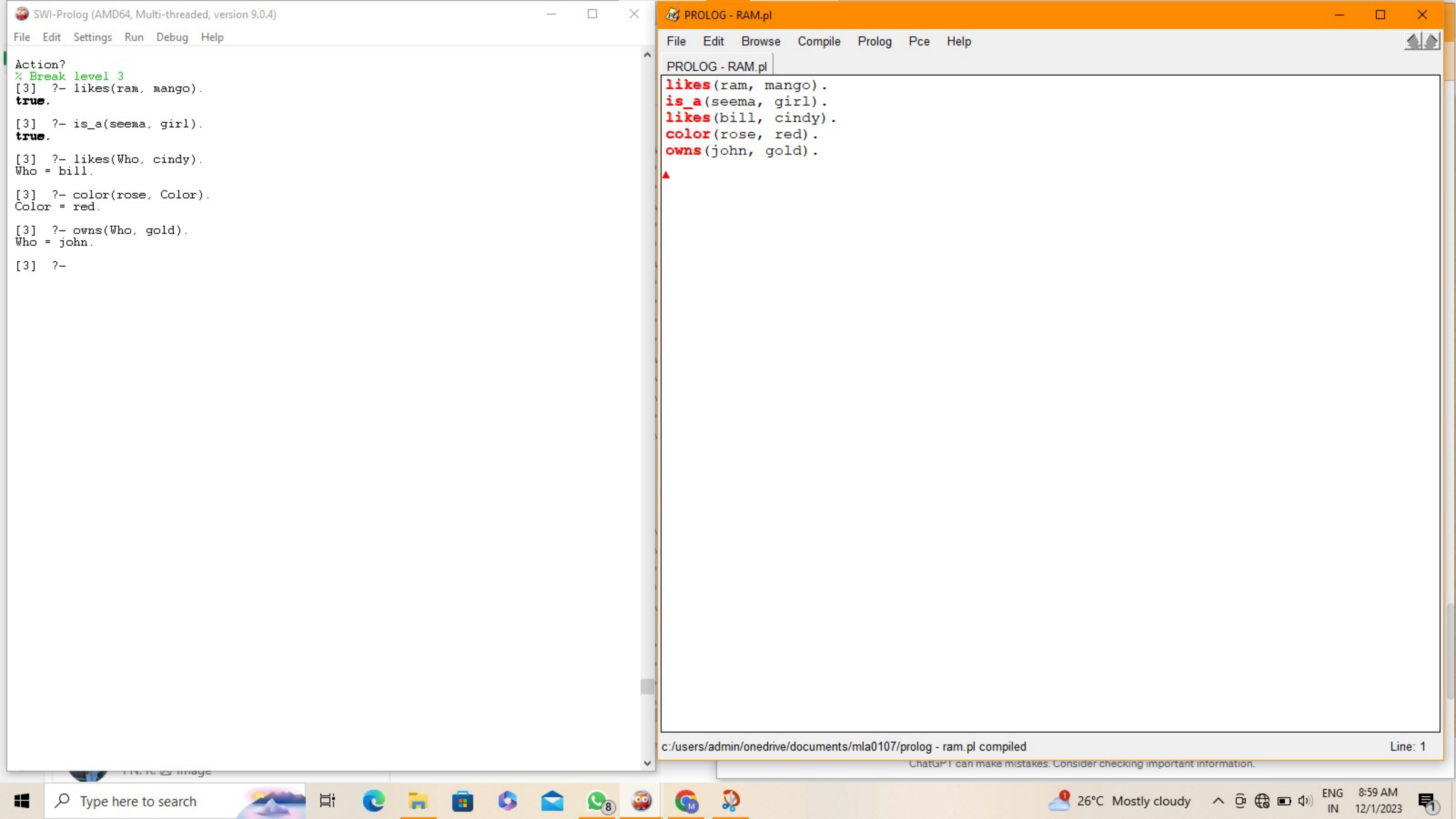

SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - PLANETS DATABASE.pl
File Edit Browse Compile Prolog Pce Help
GOLI
Sticker
2:10 PM
Type a message
Type here to search
26°C Mostly cloudy
ENG
IN
10:04 PM
11/30/2023

```
false.

?- planet(mercury, distance_from_sun(0.39), mass(0.0553)).
true.

?-
|    planet(saturn, orbital_period(29.46), day_length(0.44)).
true.

?-
```

```
% Facts
planet(mercury, distance_from_sun(0.39), mass(0.0553)). % Distance in AU, Mass in Earth masses

planet(saturn, orbital_period(29.46), day_length(0.44)). % Orbital period in Earth years, Day length in Earth days

% Rule to find the distance between two planets based on their positions from the Sun
distance_between_planets(Planet1, Planet2, Distance) :-
    planet(Planet1, distance_from_sun(Dist1), _),
    planet(Planet2, distance_from_sun(Dist2), _),
    Distance is abs(Dist1 - Dist2).

% Example query to find the distance between Planet Venus and Planet Jupiter
?- distance_between_planets(venus, jupiter, Distance).
% This query will calculate and return the absolute distance between Venus and Jupiter.

% Query to find all planets that are closer to the Sun than Planet Earth
closer_to_sun(Planet) :-
    planet(Planet, distance_from_sun(Dist), _),
    planet(earth, earth_distance),
    Dist < earth_distance.

% Example query to list planets closer to the Sun than Planet Earth
?- closer_to_sun(CloserPlanet).
% This query will return planets that are closer to the Sun than Earth.
```

c:/users/admin/onedrive/documents/mla0107/prolog - planets database.pl compiled

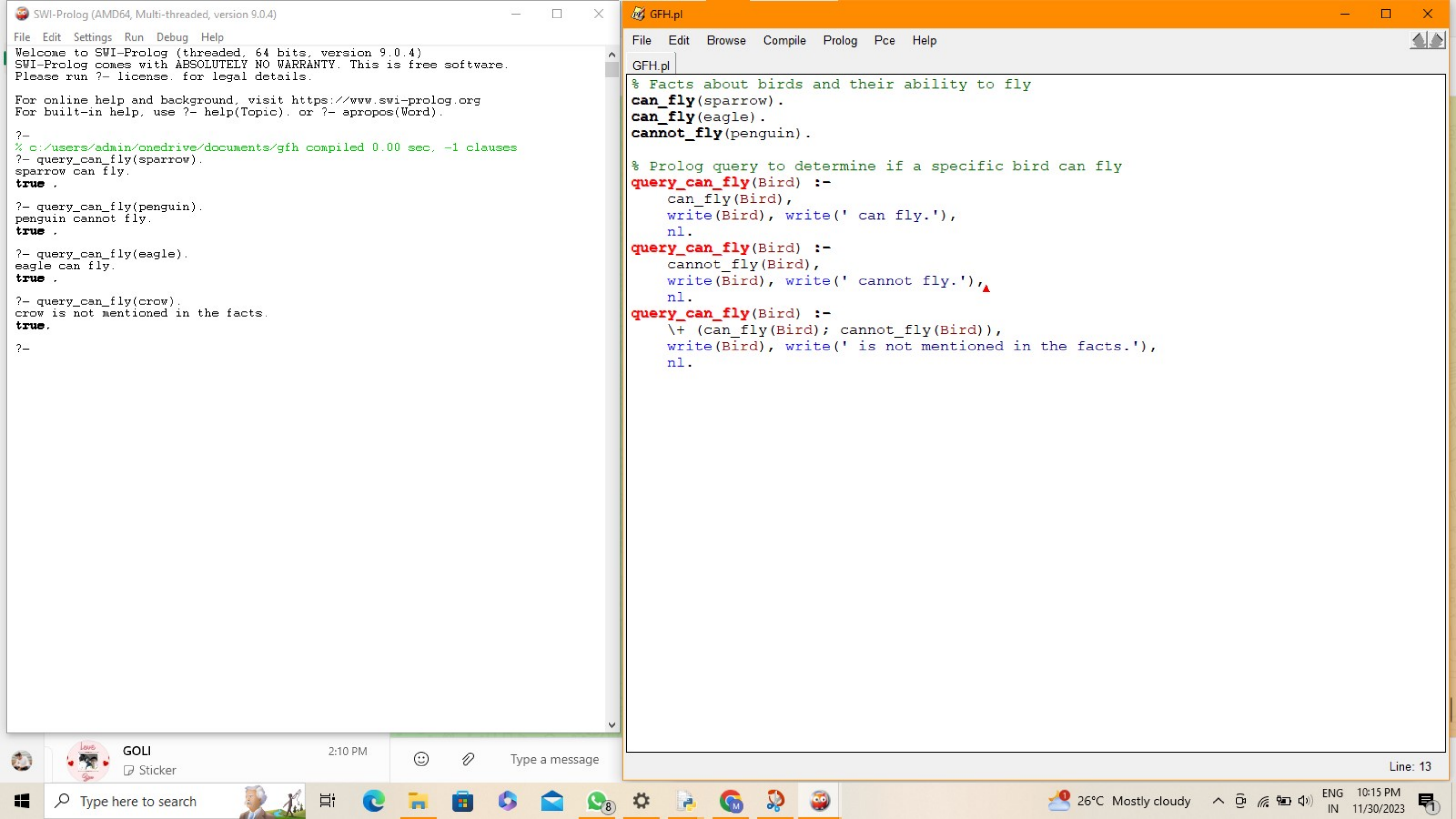
Line: 14


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - RAM.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - RAM.pl
ChatGPT can make mistakes. Consider checking important information.
Type here to search
26°C Mostly cloudy
ENG IN
8:59 AM
12/1/2023

```
Action?
% Break level 3
[3]  ?- likes(ram, mango).
true.

[3]  ?- is_a(seema, girl).
true.

[3]  ?- likes(Who, cindy).
Who = bill.

[3]  ?- color(rose, Color).
Color = red.

[3]  ?- owns(Who, gold).
Who = john.

[3]  ?-
```

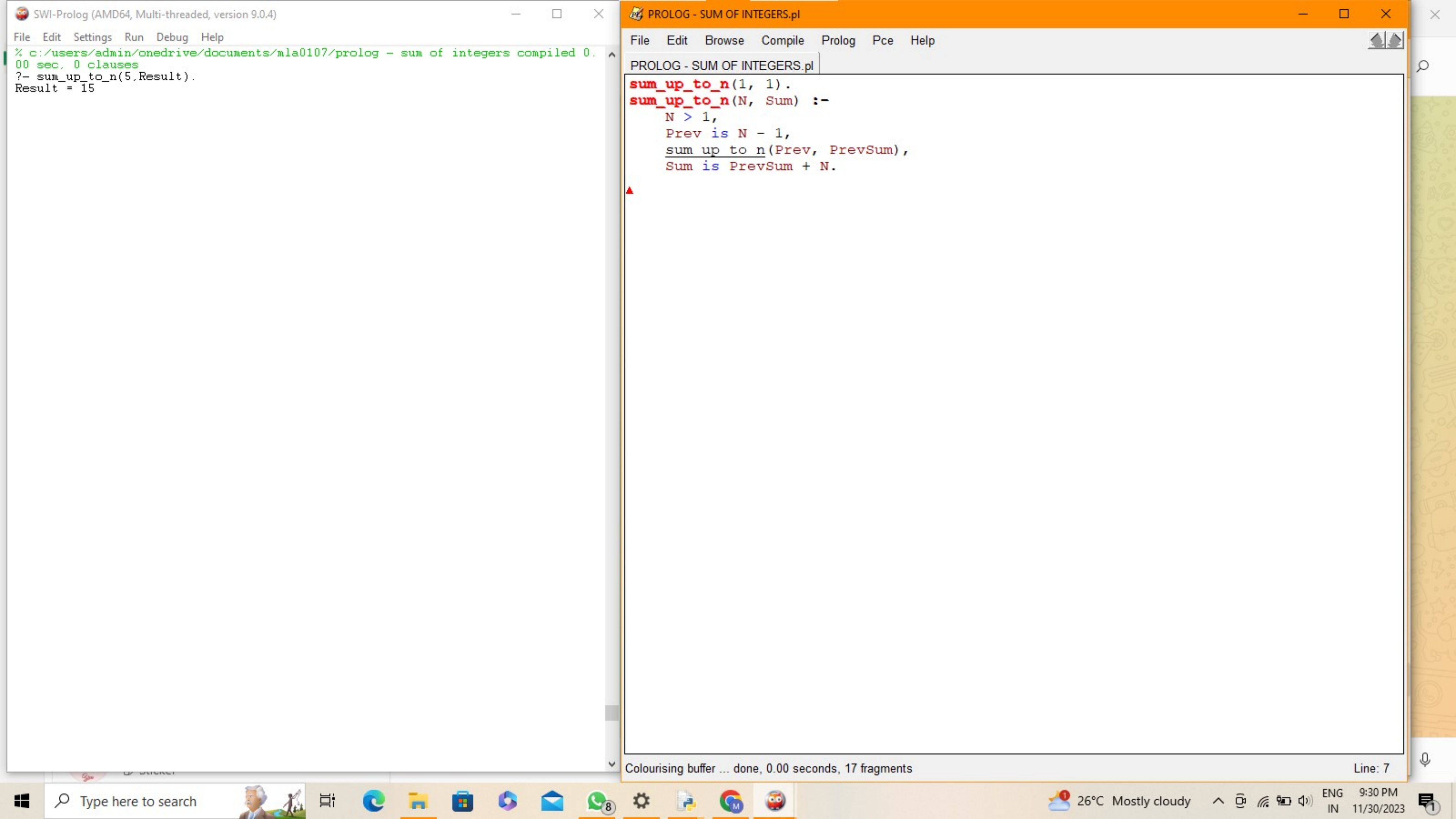
```
likes(ram, mango).
is_a(seema, girl).
likes(bill, cindy).
color(rose, red).
owns(john, gold).
```

c:/users/admin/onedrive/documents/mla0107/prolog - ram.pl compiled   Line: 1


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
GFH.pl
File Edit Browse Compile Prolog Pce Help
GFH.pl
Line: 13
GOLI
2:10 PM
Sticker
Type a message
Type here to search
26°C Mostly cloudy
ENG
IN
10:15 PM
11/30/2023

```
Welcome to SWI-Prolog (threaded, 64 bits, version 9.0.4)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

?-
% c:/users/admin/onedrive/documents/gfh compiled 0.00 sec, -1 clauses
?- query_can_fly(sparrow).
sparrow can fly.
true .

?- query_can_fly(penguin).
penguin cannot fly.
true .

?- query_can_fly(eagle).
eagle can fly.
true .

?- query_can_fly(crow).
crow is not mentioned in the facts.
true.

?-
```

```
% Facts about birds and their ability to fly
can_fly(sparrow).
can_fly(eagle).
cannot_fly(penguin).

% Prolog query to determine if a specific bird can fly
query_can_fly(Bird) :-
    can_fly(Bird),
    write(Bird), write(' can fly.'),
    nl.
query_can_fly(Bird) :-
    cannot_fly(Bird),
    write(Bird), write(' cannot fly.'),
    nl.
query_can_fly(Bird) :-
    \+ (can_fly(Bird); cannot_fly(Bird)),
    write(Bird), write(' is not mentioned in the facts.'),
    nl.
```



SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help

```
% c:/users/admin/onedrive/documents/mla0107/prolog - sum of integers compiled 0.
00 sec, 0 clauses
?- sum_up_to_n(5,Result).
Result = 15
```

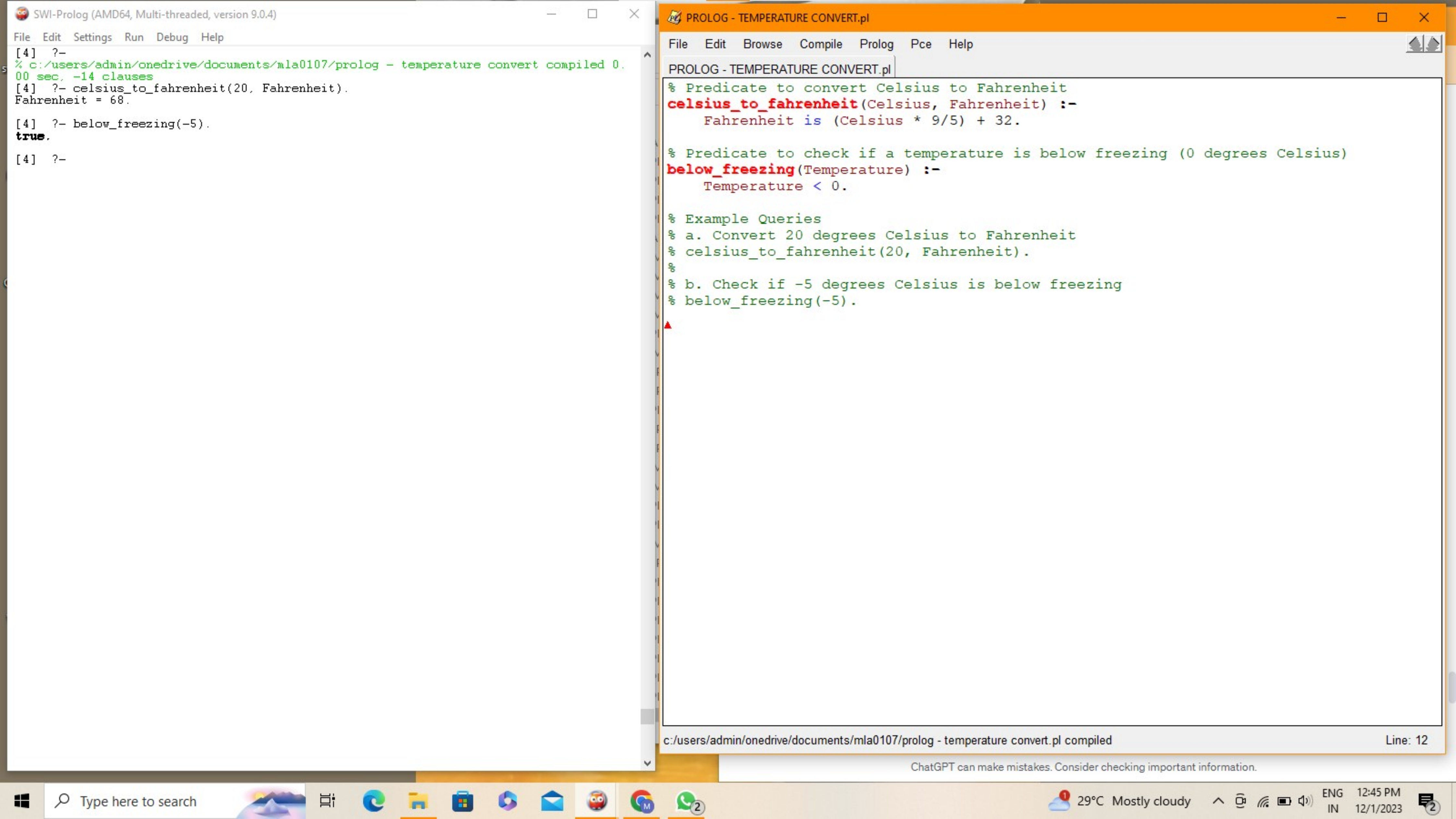
PROLOG - SUM OF INTEGERS.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - SUM OF INTEGERS.pl

```
sum_up_to_n(1, 1).
sum_up_to_n(N, Sum) :-
    N > 1,
    Prev is N - 1,
    sum_up_to_n(Prev, PrevSum),
    Sum is PrevSum + N.
```

Colourising buffer ... done, 0.00 seconds, 17 fragments

Line: 7

Type here to search
26°C Mostly cloudy
ENG
IN
9:30 PM
11/30/2023



SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File
Edit
Settings
Run
Debug
Help

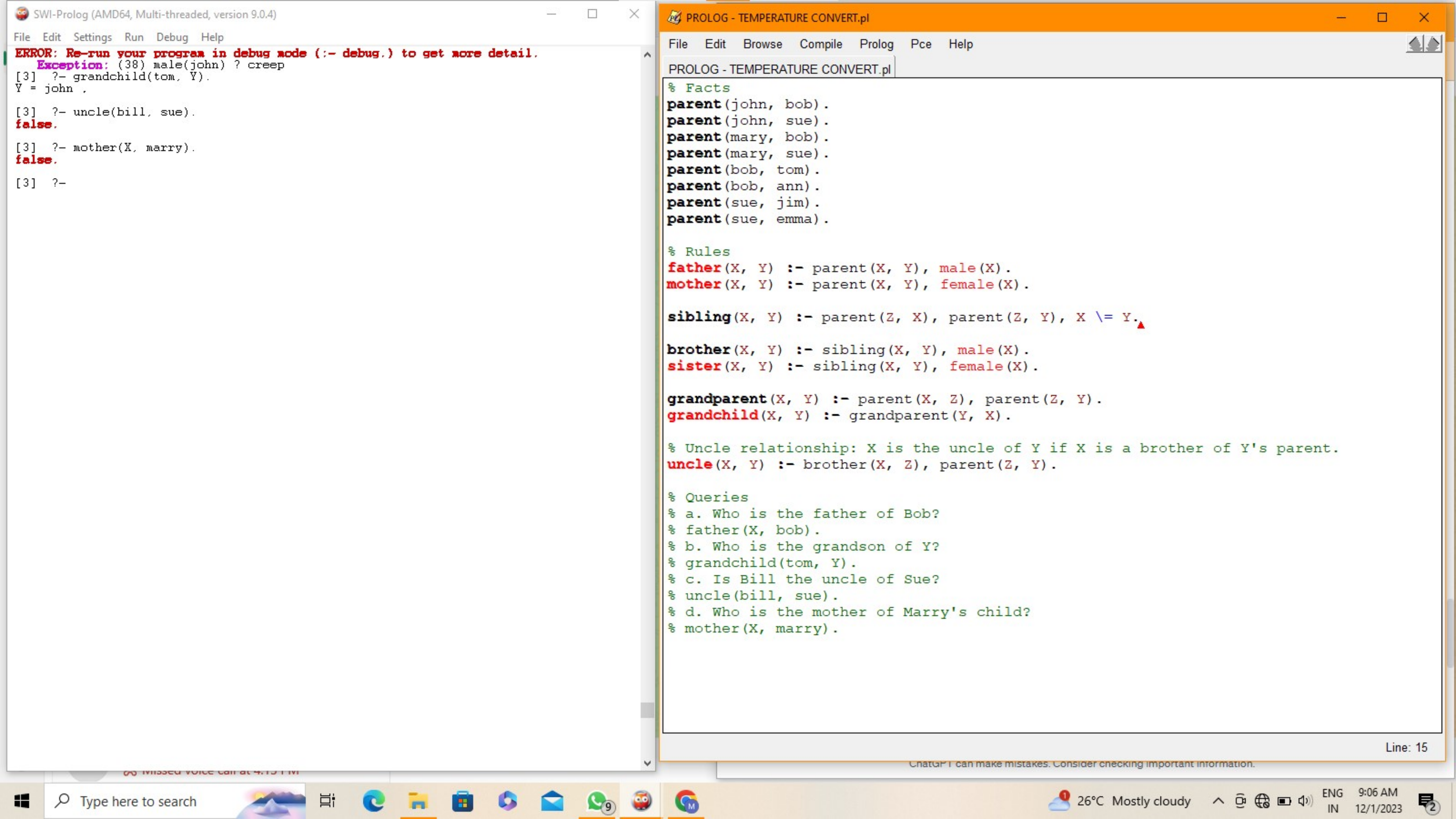
```
[4]  ?-
% c:/users/admin/onedrive/documents/mla0107/prolog - temperature convert compiled 0.
00 sec, -14 clauses
[4]  ?- celsius_to_fahrenheit(20, Fahrenheit).
Fahrenheit = 68.

[4]  ?- below_freezing(-5).
true.

[4]  ?-
```

PROLOG - TEMPERATURE CONVERT.pl
File
Edit
Browse
Compile
Prolog
Pce
Help
PROLOG - TEMPERATURE CONVERT.pl

```
% Predicate to convert Celsius to Fahrenheit
celsius_to_fahrenheit(Celsius, Fahrenheit) :-
    Fahrenheit is (Celsius * 9/5) + 32.

% Predicate to check if a temperature is below freezing (0 degrees Celsius)
below_freezing(Temperature) :-
    Temperature < 0.

% Example Queries
% a. Convert 20 degrees Celsius to Fahrenheit
% celsius_to_fahrenheit(20, Fahrenheit).
%
% b. Check if -5 degrees Celsius is below freezing
% below_freezing(-5).
```

c:/users/admin/onedrive/documents/mla0107/prolog - temperature convert.pl compiled   Line: 12

ChatGPT can make mistakes. Consider checking important information.


SWI-Prolog (AMD64, Multi-threaded, version 9.0.4)
File Edit Settings Run Debug Help
PROLOG - TEMPERATURE CONVERT.pl
File Edit Browse Compile Prolog Pce Help
PROLOG - TEMPERATURE CONVERT.pl
Line: 15
Missed voice call at 4:15 PM
ChatGPT can make mistakes. Consider checking important information.
Type here to search
26°C Mostly cloudy
ENG
IN
9:06 AM
12/1/2023

```
ERROR: Re-run your program in debug mode (:- debug.) to get more detail.
   Exception: (38) male(john) ? creep
[3]  ?- grandchild(tom, Y).
Y = john ,

[3]  ?- uncle(bill, sue).
false.

[3]  ?- mother(X, marry).
false.

[3]  ?-
```

```
% Facts
parent(john, bob).
parent(john, sue).
parent(mary, bob).
parent(mary, sue).
parent(bob, tom).
parent(bob, ann).
parent(sue, jim).
parent(sue, emma).

% Rules
father(X, Y) :- parent(X, Y), male(X).
mother(X, Y) :- parent(X, Y), female(X).

sibling(X, Y) :- parent(Z, X), parent(Z, Y), X \= Y.

brother(X, Y) :- sibling(X, Y), male(X).
sister(X, Y) :- sibling(X, Y), female(X).

grandparent(X, Y) :- parent(X, Z), parent(Z, Y).
grandchild(X, Y) :- grandparent(Y, X).

% Uncle relationship: X is the uncle of Y if X is a brother of Y's parent.
uncle(X, Y) :- brother(X, Z), parent(Z, Y).

% Queries
% a. Who is the father of Bob?
% father(X, bob).
% b. Who is the grandson of Y?
% grandchild(tom, Y).
% c. Is Bill the uncle of Sue?
% uncle(bill, sue).
% d. Who is the mother of Marry's child?
% mother(X, marry).
```