**Fr. Conceicao Rodrigues College of Engineering**

**Department of Computer Engineering**

EXPERIMENT 2

|  |  |
| --- | --- |
| **Practical No:** | 2 |
| **Title:** | Program in prolog to implement simple facts and Queries |
| **Date of Performance:** | 5-02-2025 |
| **Date of Submission:** | 11-02-2025 |
| **Roll No:** | 9913 |
| **Name of the Student:** | Mark Lopes |

**Rubrics for Evaluation:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr. No** | **Performance Indicator** | **Excellent** | **Good** | **Below Average** | **Total Score** |
| 1 | On time Completion & Submission (01) | 01 (On  Time ) | NA | 00 (Not on Time) |  |
| 2 | Logic/Theory understanding(02) | 02(Correct) | NA | 01 (Tried) |  |
| 3 | Coding Standards (03): Comments/indention/Naming conventions  Output/Test Cases | 03(All used) | 02 (Partial) | 01 (rarely followed) |  |
| 4 | Post Lab Assignment (04) | 04(done well) | 3 (Partially Correct) | 2(submitted) |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Academic Year** | **2024-25** | **Estimated Time** | **Experiment No. 2 – 02 Hours** |
| **Course & Semester** | **T.E. (CE) – Sem. VI** | **Subject Name** | **CSC604: Artificial Intelligence** |
| **Chapter No.** | **04** | **Chapter Title** | **Knowledge and Reasoning** |
| **Experiment Type** | **Knowledge and Reasoning** | **Software** | **Prolog** |

#### **AIM: Write a program in prolog to implement simple facts and Queries**

#### **Ram likes mango.**

#### **Seema is a girl.**

#### **Bill likes Cindy.**

#### **Rose is red.**

#### **John owns gold.**

#### **Clauses**

#### **likes(ram ,mango).**

#### **girl(seema). red(rose).**

#### **likes(bill ,cindy).**

#### **owns(john ,gold).**

#### **Goal**

#### **?- likes (ram,What).**

#### **What = mango.**

#### **1 solution.**

**Assignment:**

Aim: Write facts for following:

* 1. Ram likes apple.
  2. Ram is taller than Mohan.
  3. My name is Subodh.
  4. Apple is fruit.
  5. Orange is fruit.
  6. Ram is male.

**AIM: Write simple queries for following facts.**

**Simple Queries**

Now that we have some facts in our Prolog program, we can consult the program in the listener and query, or call, the facts. This chapter, and the next, will assume the Prolog program contains only facts. Queries against programs with rules will be covered in a later chapter.

Prolog queries work by pattern matching. The query pattern is called a **goal**. If there is a fact that matches the goal, then the query succeeds and the listener responds with 'yes.' If there is no matching fact, then the query fails and the listener responds with 'no.'

Prolog's pattern matching is called **unification**. In the case where the logic base contains only facts, unification succeeds if the following three conditions hold.

* The predicate named in the goal and logic base are the same.
* Both predicates have the same arity.
* All of the arguments are the same.

Before proceeding, review figure 3.1, which has a listing of the program so far.

The first query we will look at asks if the office is a room in the game. To pose this, we would enter that goal followed by a period at the listener prompt.

?- room(office). yes

Prolog will respond with a 'yes' if a match was found. If we wanted to know if the attic was a room, we would enter that goal.

?- room(attic). no

**Solution:-**

**clauses**

likes(ram ,mango).

girl(seema).

red(rose).

likes(bill ,cindy).

owns(john ,gold).

**queries**

?-likes(ram,What).

What= mango

?-likes(Who,cindy).

Who= cindy

?-red(What).

What= rose

?-owns(Who,What).

Who= john

What= gold

**Prolog:**

female(pam).

female(liz).

female(pat).

female(ann).

male(jim).

male(bob).

male(tom).

male(peter).

parent(pam,bob).

parent(tom,bob).

parent(peter,jim).

parent(bob,pat).

parent(bob,peter).

parent(liz,pat).

parent(liz,peter).

mother(X,Y):- parent(X,Y),female(X).

father(X,Y):- parent(X,Y),male(X).

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

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

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

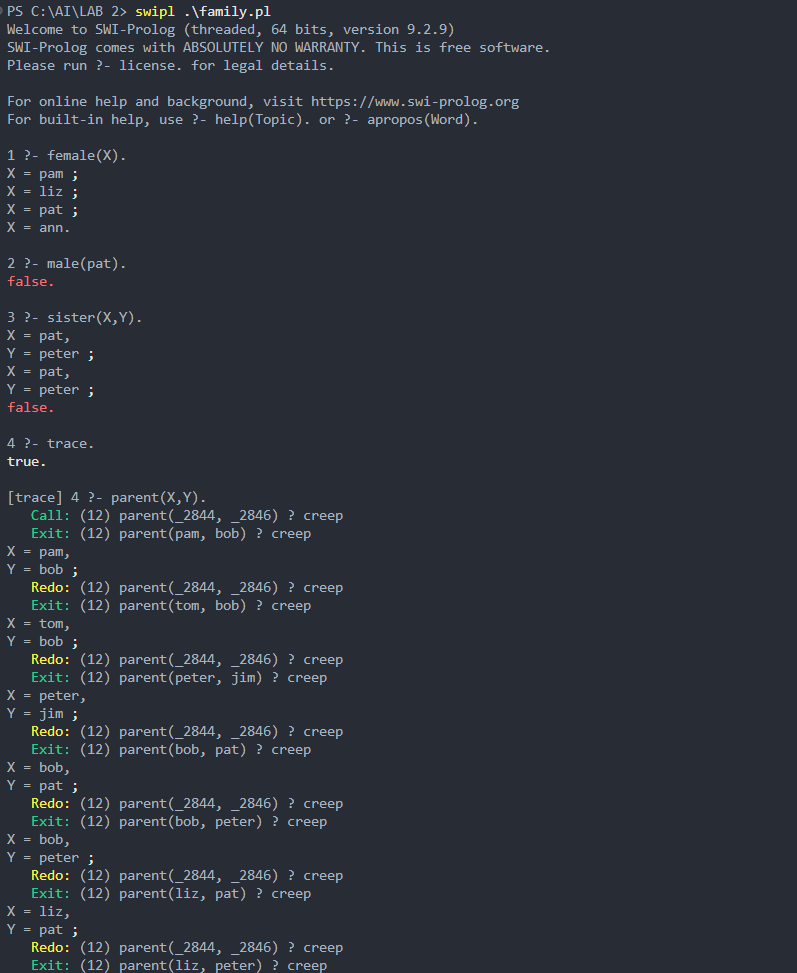
grandmother(X,Z):- mother(X,Y),parent(Y,Z).

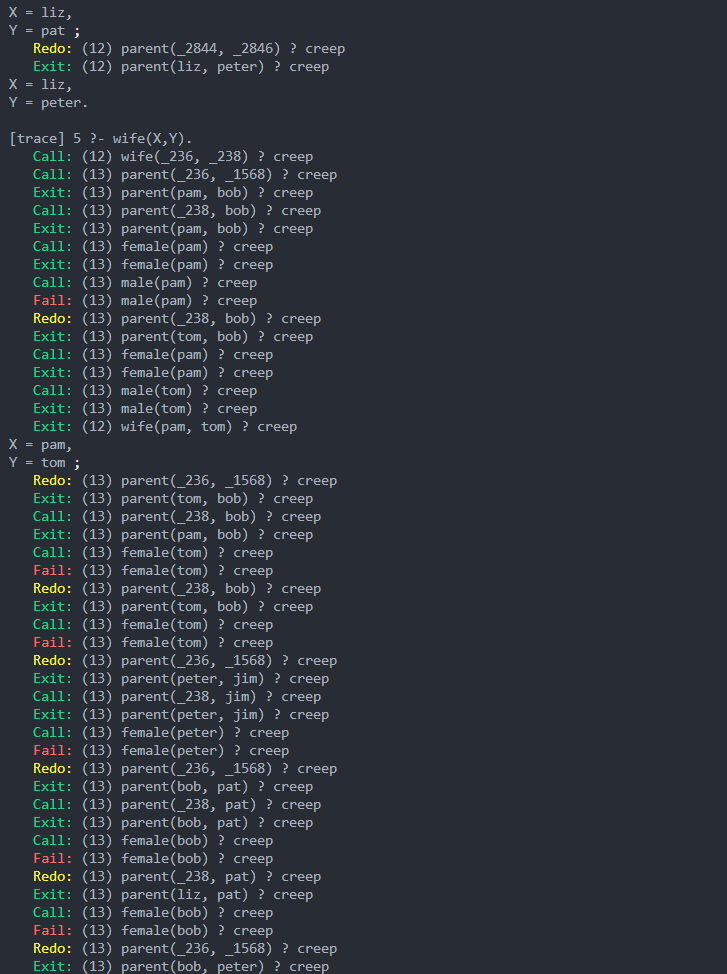
grandfather(X,Z):- father(X,Y),parent(Y,Z).

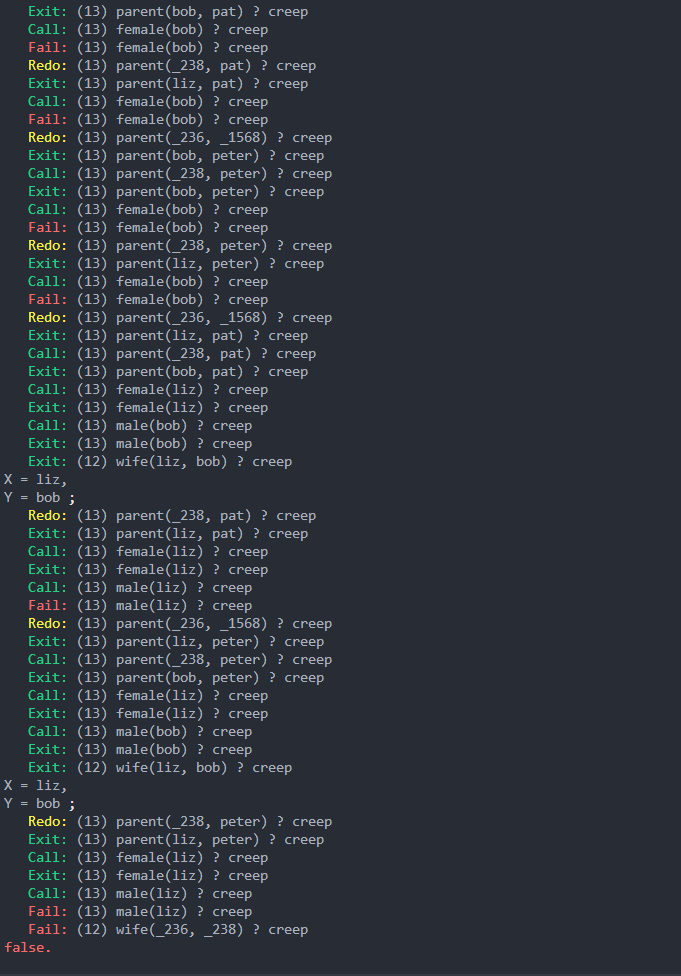
wife(X,Y):- parent(X,Z),parent(Y,Z),female(X),male(Y).

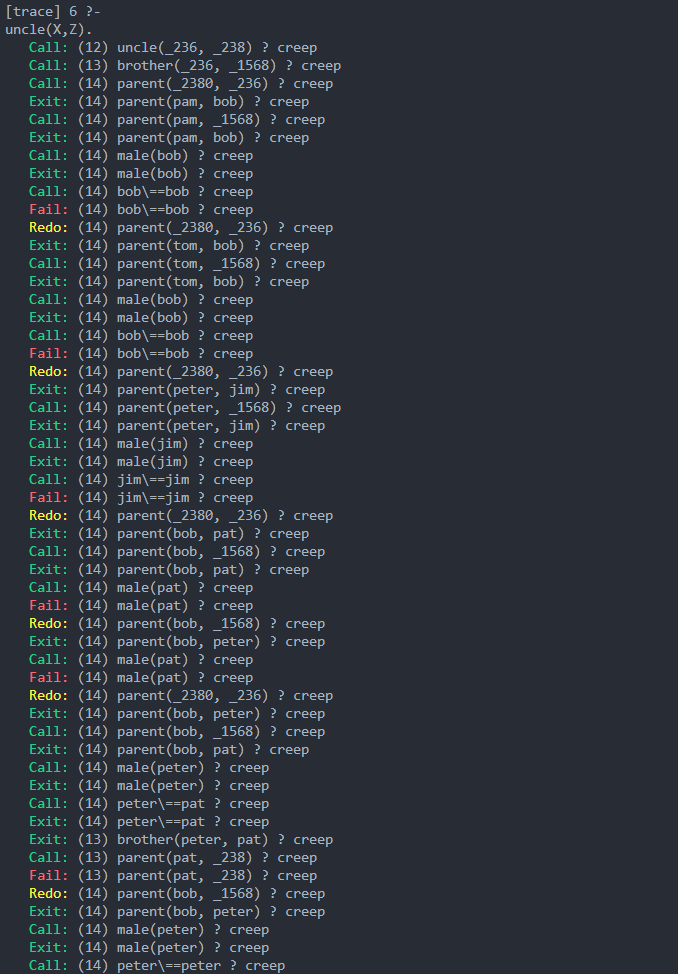
uncle(X,Z):- brother(X,Y),parent(Y,Z).

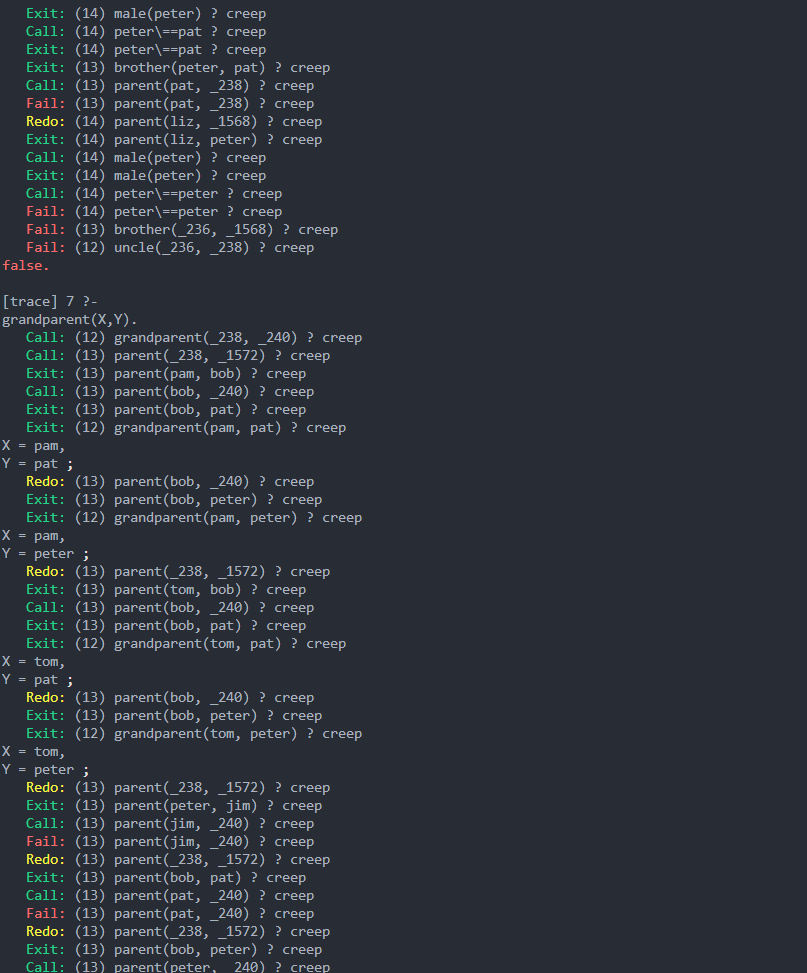
**Output:**

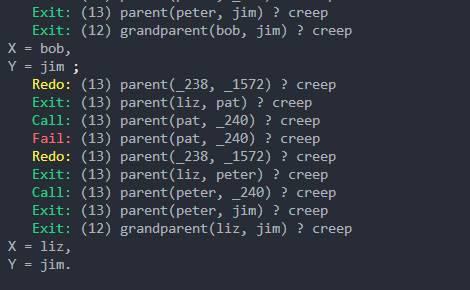












**PostLab Assignment:**

**Aim:** Using the following facts answer the question

* 1. Find car make that cost is exactly 2,00,000/-
  2. Find a car make that costs less than 5 lacs.
  3. List all the cars available.
  4. Is there any car which costs more than 10 lacs.

**Prolog:**

**car(mazda, 200000).**

**car(toyota, 350000).**

**car(honda, 150000).**

**car(bmw, 800000).**

**car(mercedes, 1200000).**

**car(ford, 400000).**

**car(nissan, 450000).**

**car(audi, 950000).**

**car(chevrolet, 600000).**

**car(kia, 300000).**

**car(volvo, 700000).**

**car(jaguar, 1300000).**

**car(subaru, 500000).**

**car(lexus, 1100000).**

**car(suzuki, 250000).**

**car(tesla, 3000000).**

**car(land\_rover, 1500000).**

**car(ferrari, 25000000).**

**car(porsche, 1800000).**

**car(fiat, 180000).**

**% Rule 1: Find car make that costs exactly 2,00,000**

**find\_2(Make) :- car(Make, 200000).**

**% Rule 2: Find car make that costs less than 5,00,000**

**find\_5(Make) :- car(Make, Price), Price < 500000.**

**% Rule 3: List all available cars**

**list\_all\_cars :-**

**car(Make, Price),**

**write(Make), write(' costs '), write(Price), nl,**

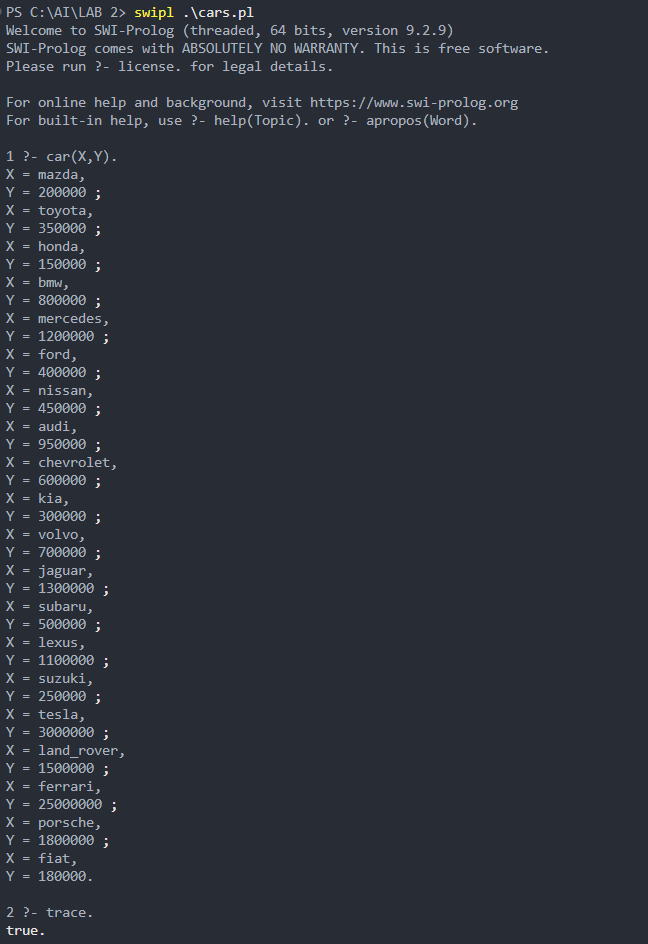
**fail.**

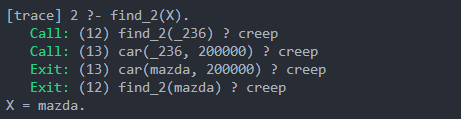
**list\_all\_cars.**

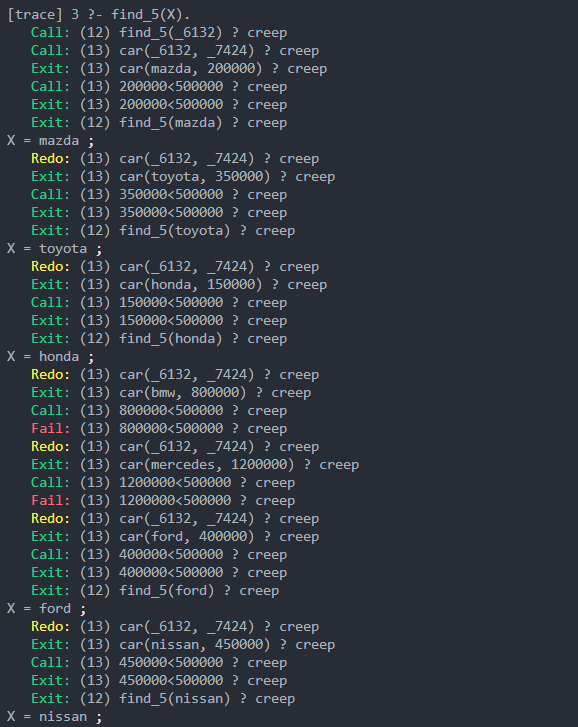
**% Rule 4: Check if any car costs more than 10 lacs**

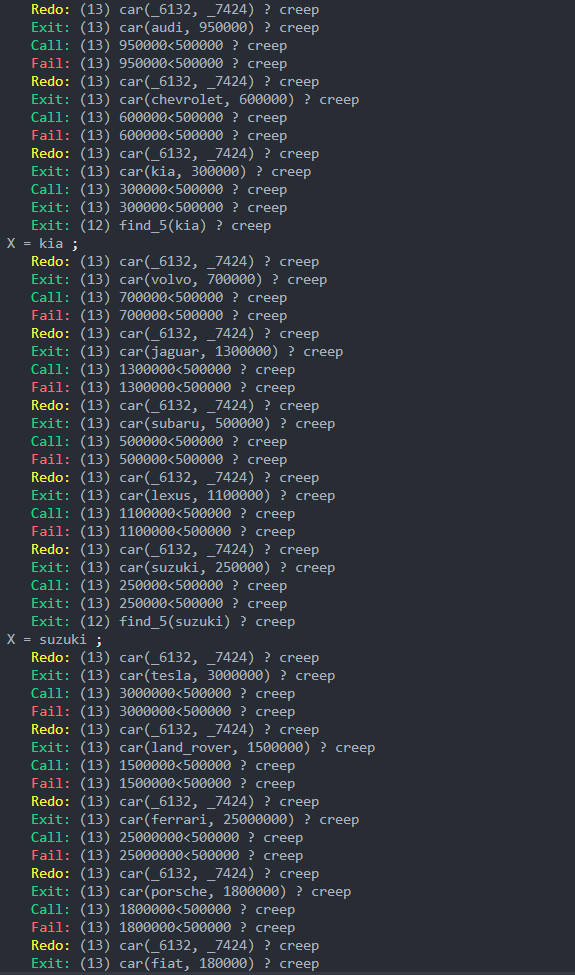
**car\_10\_lacs :- car(\_, Price), Price > 1000000.**

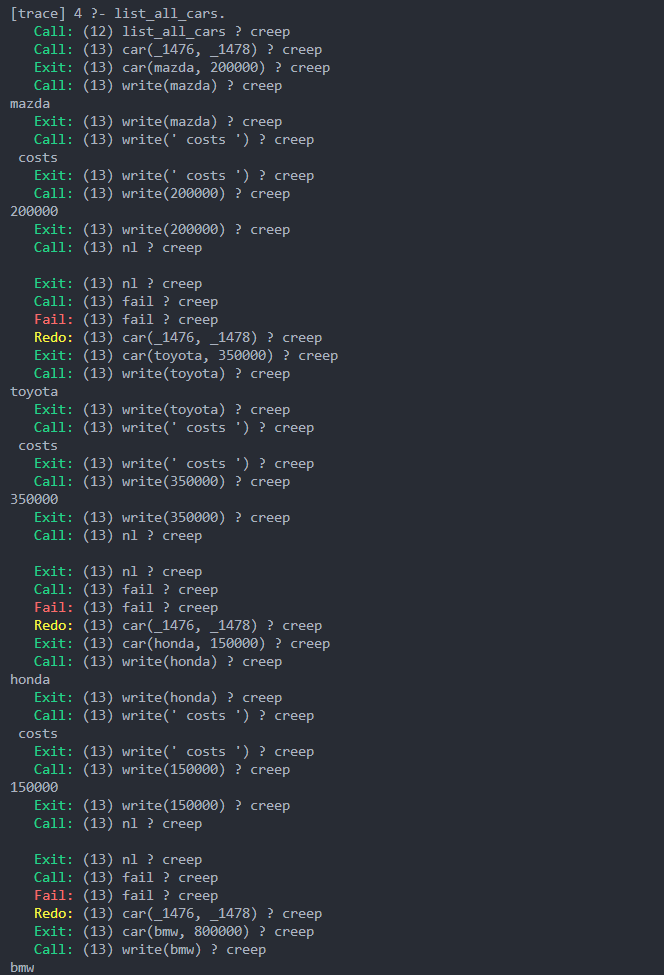
**Output:**

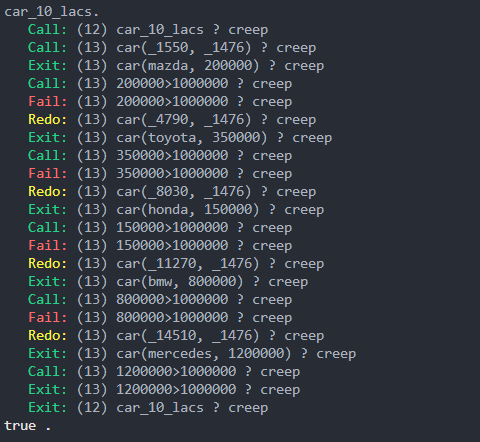
****

****

****

****

****

****