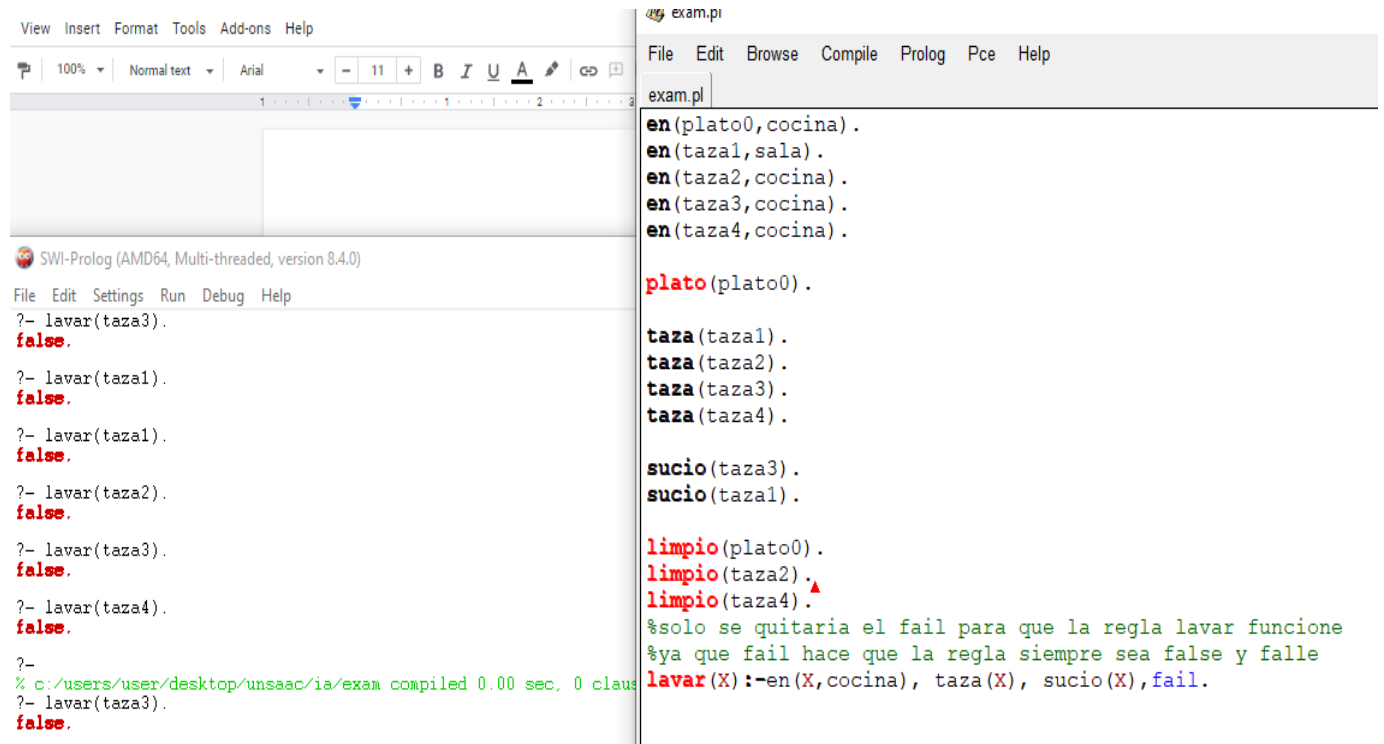


## Con fail



The screenshot shows the SWI-Prolog IDE with a file named `exam.pl`. The Prolog code defines a set of rules for a kitchen scenario. The execution results show that the `lavar` predicate always returns `false` for the given arguments.

```
en(plato0,cocina).
en(taza1,sala).
en(taza2,cocina).
en(taza3,cocina).
en(taza4,cocina).

plato(plato0).

taza(taza1).
taza(taza2).
taza(taza3).
taza(taza4).

sucio(taza3).
sucio(taza1).

limpio(plato0).
limpio(taza2).
limpio(taza4).
%solo se quitaría el fail para que la regla lavar funcione
%ya que fail hace que la regla siempre sea false y falle
lavar(X):-en(X,cocina), taza(X), sucio(X),fail.
```

```
?- lavar(taza3).
false.

?- lavar(taza1).
false.

?- lavar(taza1).
false.

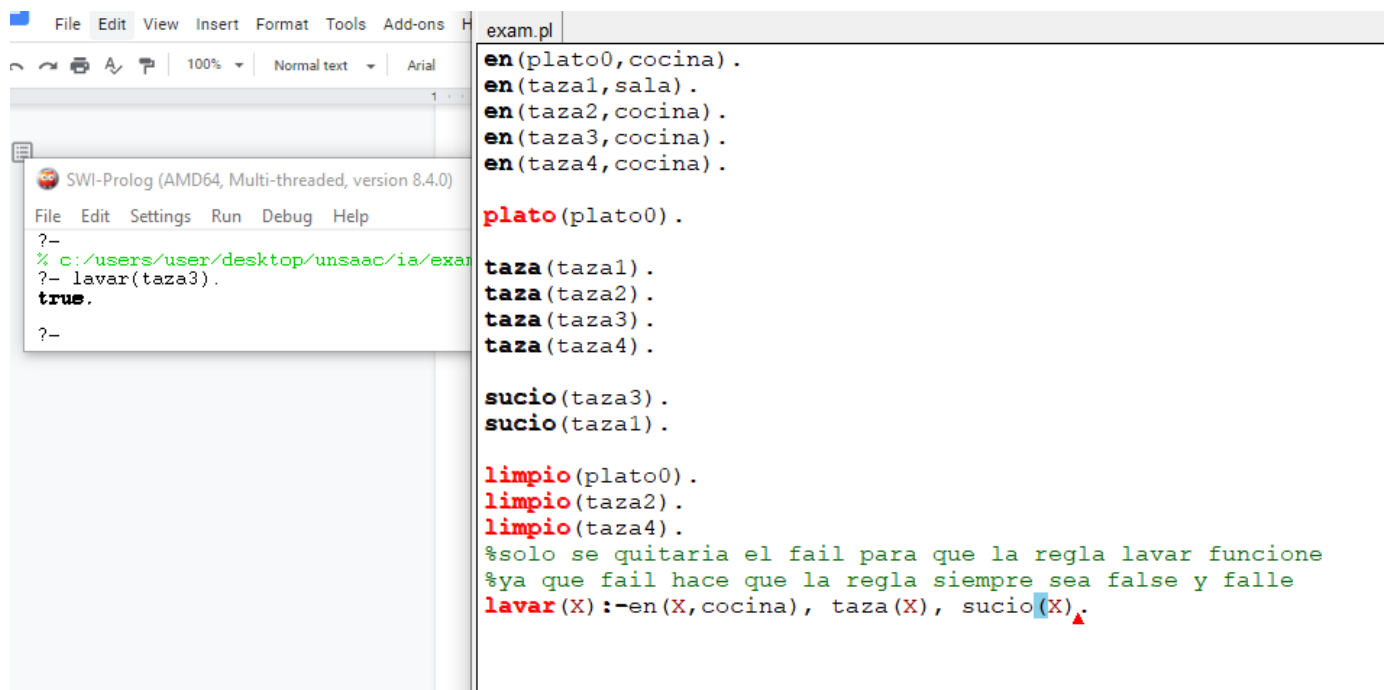
?- lavar(taza2).
false.

?- lavar(taza3).
false.

?- lavar(taza4).
false.

?- 
% c:/users/user/desktop/unsaac/ia/exam compiled 0.00 sec, 0 clauses
?- lavar(taza3).
false.
```

## Quitando el fail



The screenshot shows the same SWI-Prolog IDE with the `exam.pl` file. The `lavar` predicate has been modified to remove the `fail` predicate. The execution results show that the `lavar` predicate now returns `true` for the given arguments.

```
en(plato0,cocina).
en(taza1,sala).
en(taza2,cocina).
en(taza3,cocina).
en(taza4,cocina).

plato(plato0).

taza(taza1).
taza(taza2).
taza(taza3).
taza(taza4).

sucio(taza3).
sucio(taza1).

limpio(plato0).
limpio(taza2).
limpio(taza4).
%solo se quitaría el fail para que la regla lavar funcione
%ya que fail hace que la regla siempre sea false y falle
lavar(X):-en(X,cocina), taza(X), sucio(X).
```

```
?- 
% c:/users/user/desktop/unsaac/ia/exam compiled 0.00 sec, 0 clauses
?- lavar(taza3).
true.

?-
```