Programare Logică
Cursul I

Claudia MUREȘAN
cmuresan@fmi.unibuc.ro, c.muresan@yahoo.com

Universitatea din București
Facultatea de Matematică și Informatică
București

2019–2020, Semestrul II

Prescurtări uzuale care vor fi folosite în lecții

- **i. e.** (*id est*) = adică
- **a. î.** = astfel încât
- **ddacă** = dacă și numai dacă
- **ș. a. m. d.** = și așa mai departe
- $\frac{}{//}$: să se demonstreze că
-  : contradicție

Paradigme ale programării calculatoarelor

Programarea imperativă:

- programatorul trebuie să–i spună calculatorului, pas cu pas, ce să facă: "parcurge cu un indice această mulțime, la fiecare iterație verifică această condiție..." etc..

Limbaje de programare imperativă: *Pascal, C, Java* etc..

Programarea logică/declarativă:

- programatorul descrie un cadru de lucru, și cere calculatorului să rezolve o cerință în acel cadru;
- pe baza unui backtracking și a altor tehnici de programare încorporate în interpretorul/compilatorul limbajului de programare logică folosit, calculatorul determină proprietățile pe care le poate folosi pentru a rezolva cerința respectivă și ordinea în care trebuie să le aplice.

Limbaje de programare logică/declarativă

- *Prolog* (PROGRAMMING IN LOGIC): bazat pe **predicate**, pe relații între obiecte; utilizat pentru jocuri/deducții logice, în procesarea limbajului natural etc.;

- *CafeObj, Maude:* bazate pe **specificații**, pe descrierea unor tipuri de structuri algebrice; destinate demonstrării automate de proprietăți matematice; utilizate și în verificarea sistemelor software;

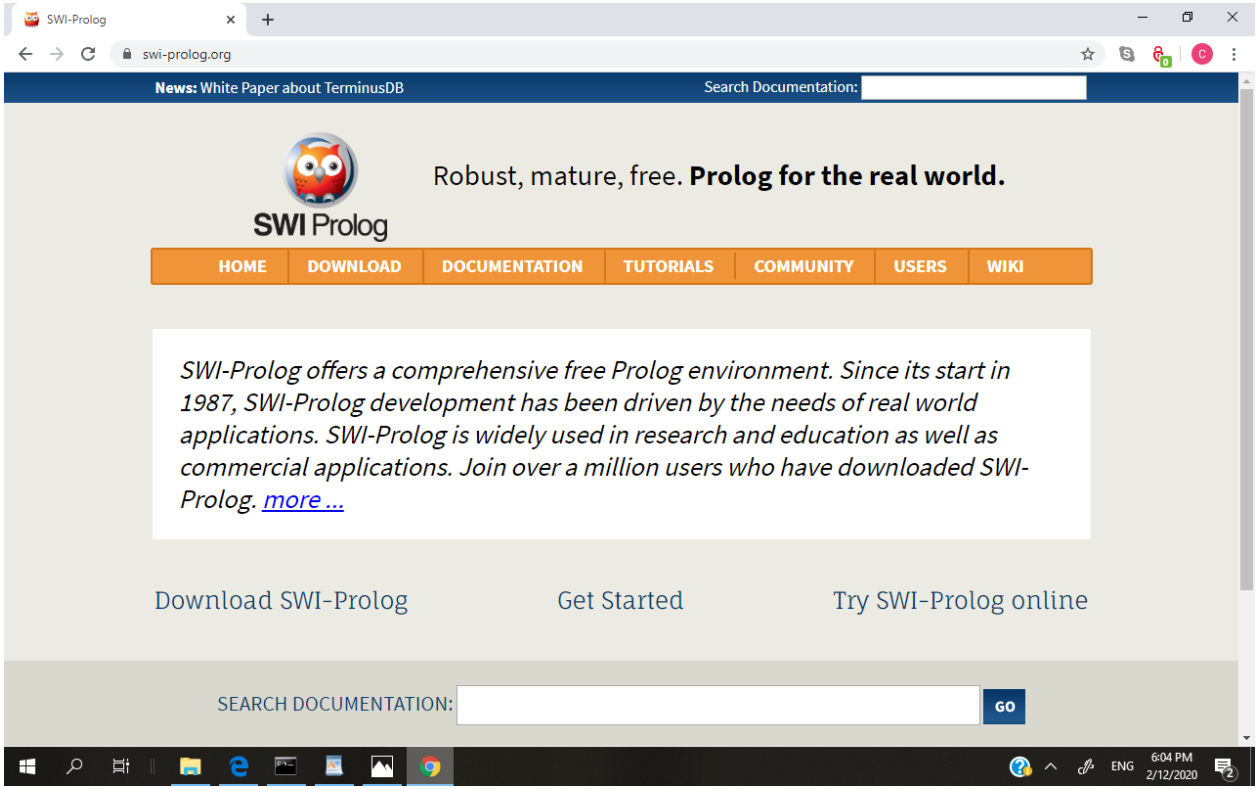
- *Haskell:* bazat pe **funcții** și **recursii**;

recursia este foarte importantă în toate limbajele de programare logică: este principalul mijloc de calcul, înlocuind, de exemplu, instrucțiunile repetitive

- etc..

Siteul SWI–Prolog

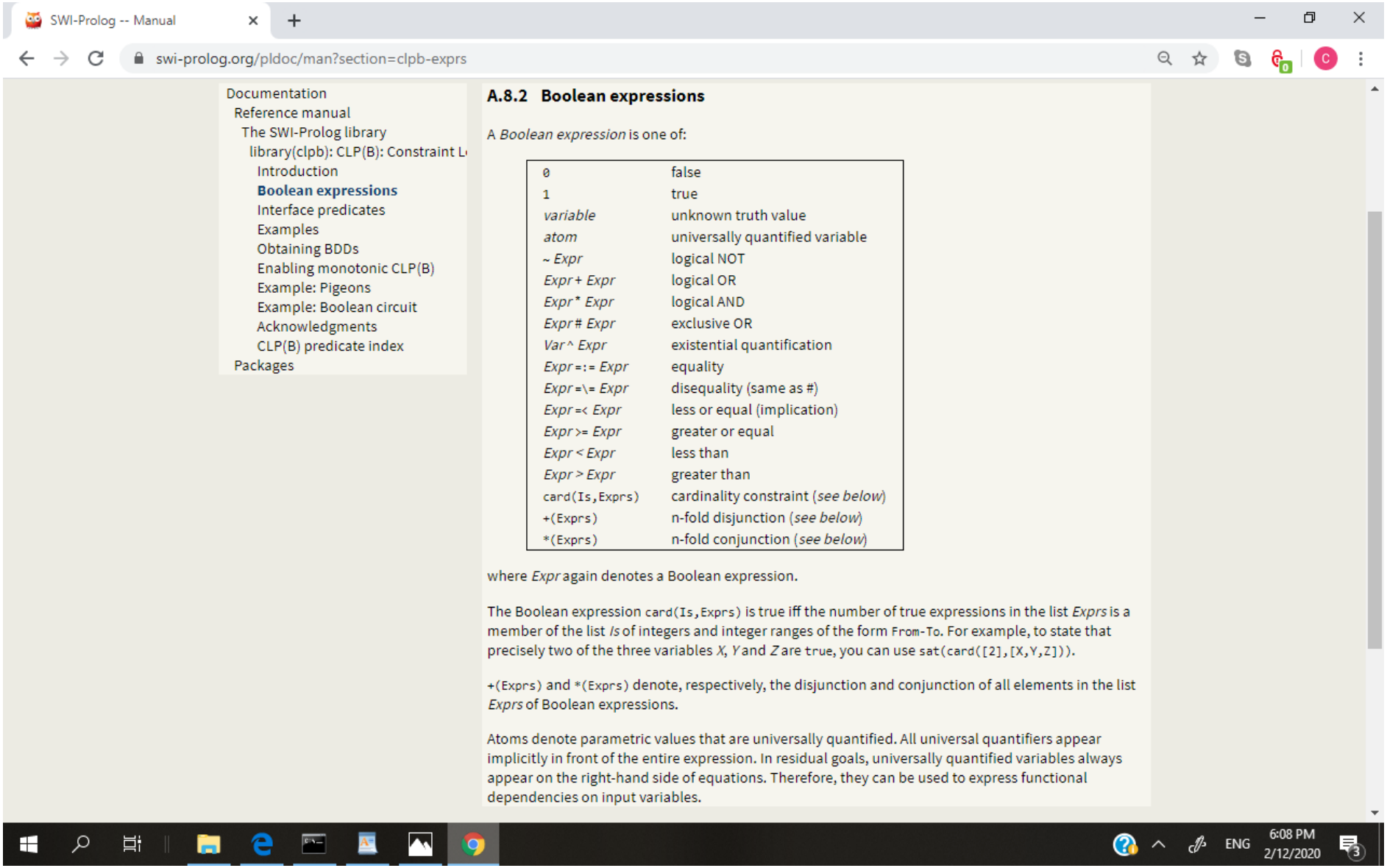
https://www.swi-prolog.org



Din josul paginii de la această adresă:

- se poate descărca SWI–Prolog;
- se poate căuta în documentația SWI–Prolog;
- se poate accesa versiunea online a SWI–Prolog.

De exemplu, dând o căutare după "boolean expressions", găsim:



Structura unui program în Prolog

Un program în Prolog este format din **clauze**; acestea sunt de trei tipuri:

- **fapte**:

propoziție.
predicat(*listă de variabile si constante*).

acestea semnifică proprietăți intotdeauna adevărate;
predicatele exprimă relații între obiecte;
propozițiile sunt predicatele fără variabile;

- **reguli**:

fapt :- succesiune de fapte.

faptele din partea dreaptă a unei reguli pot fi separate prin virgulă (care reprezintă *conjuncția logică*) sau alți conectori logici (a se vedea slideul anterior);

(semnificația unei reguli)

are loc acest fapt :- dacă au loc aceste fapte.

- **întrebări**: formate din **scopuri**.

Numele de **variabile** în Prolog încep cu *literă mare* sau *underscore* (_).
Variabilă nedenumită: *underscore* (_):

- simbol generic pentru variabile care apar o singură dată într–un fapt sau o regulă;
- sunt folosite doar pentru a indica locații de variabile, nu și pentru a lucra cu acele variabile.

Orice alt nume, inclusiv șiruri de caractere cuprinse între apostrofuri, denumește o constantă sau un predicat.

Sintaxa pentru liste în Prolog:

$[\,]$	lista vidă
$[elem_1, elem_2, \ldots, elem_n]$	lista formată din elementele $elem_1, elem_2, \ldots, elem_n$
$[elem1, elem2, \ldots, elemn \mid T]$	lista cu primele n elemente $elem_1, elem_2, \ldots, elem_n$ și coada T

Exemplu

$$[1,2,3,4,5] = [1,2,3,4,5|[\,]] = [1|[2,3,4,5]] = [1,2,3|[4,5]] = [1,2|[3|[4,5]]]$$

Exemplu

Cum putem scrie predicate pentru:

- calculul lungimii unei liste;
- concatenarea a două liste?

Să scriem un predicat *lung*(L, N), care este satisfăcut ddacă:

- L este o listă, N este un număr natural și
- N este lungimea listei L, adică numărul elementelor lui L:

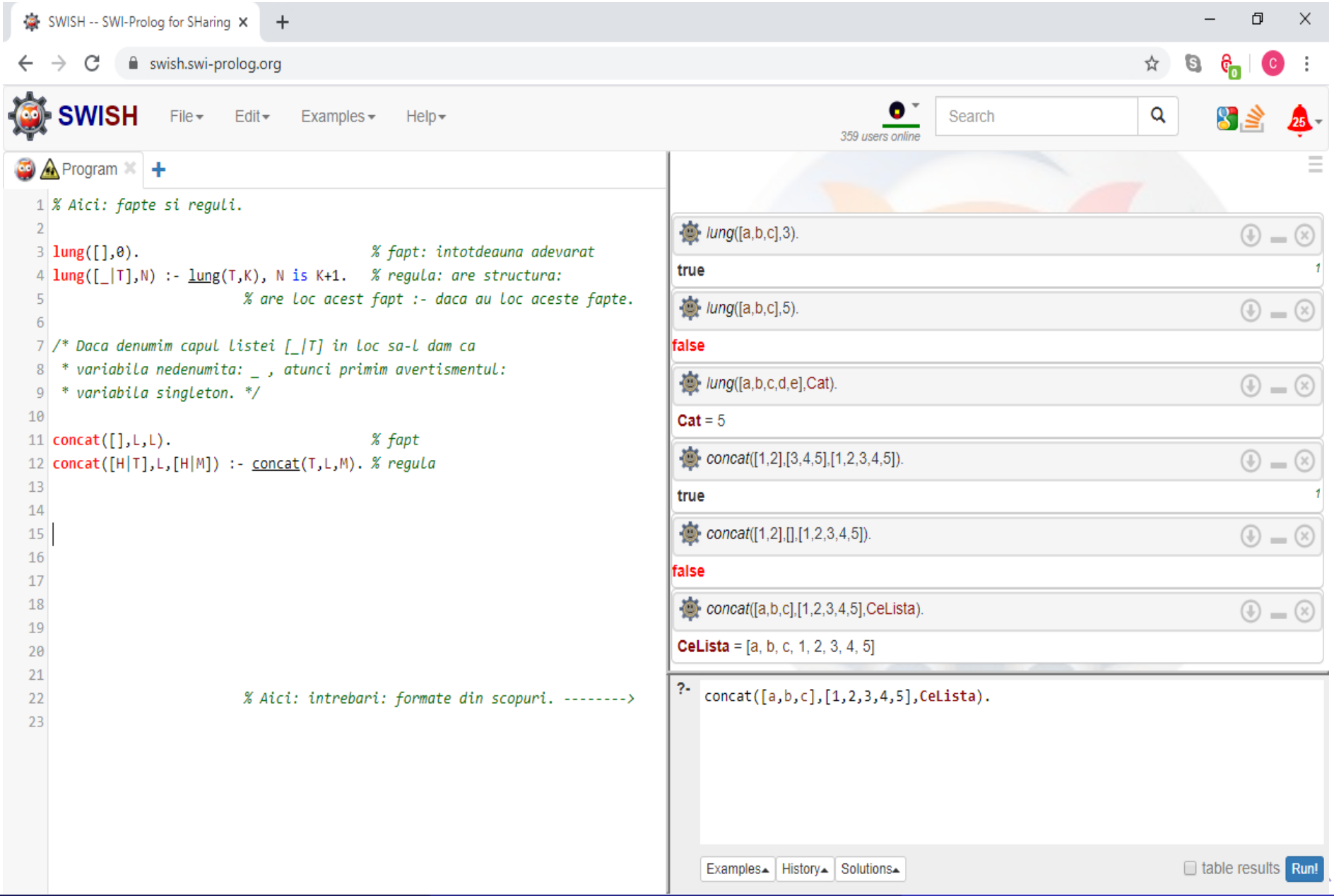
```
lung([ ], 0).
lung([_|T], N) :- lung(T, K), N is K + 1.
```

Operatorul **is** produce executarea calculului în acea expresie aritmetică.
Înlocuiți **is** cu = și vedeți ce obțineți!
Și un predicat *concat*$(L1, L2, L)$, care este satisfăcut ddacă:

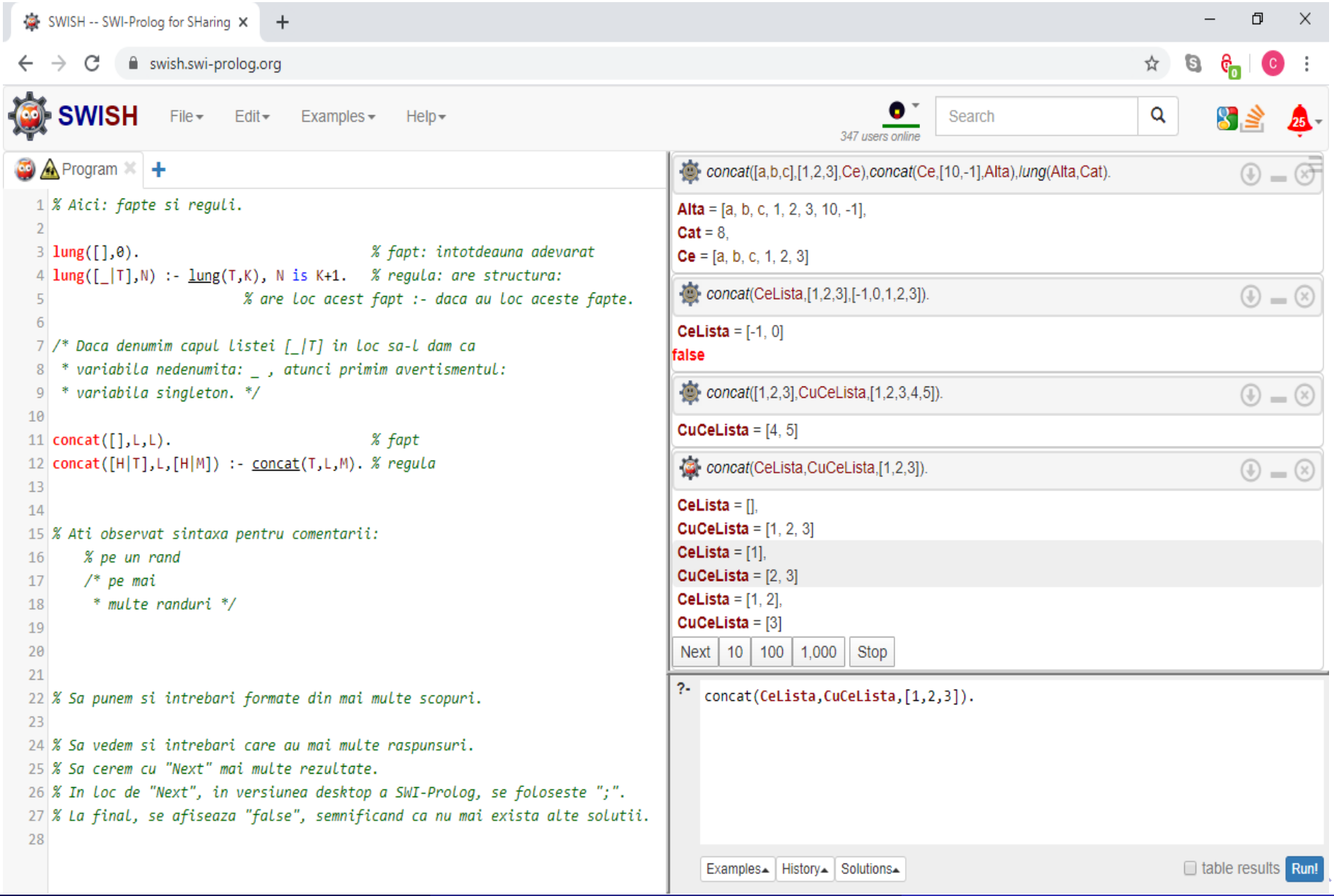
- $L1$, $L2$ și L sunt liste și
- concatenarea listei $L1$ cu $L2$ este L:

```
concat([ ], L, L).
concat([H|T], L, [H|M]) :- concat(T, L, M).
```

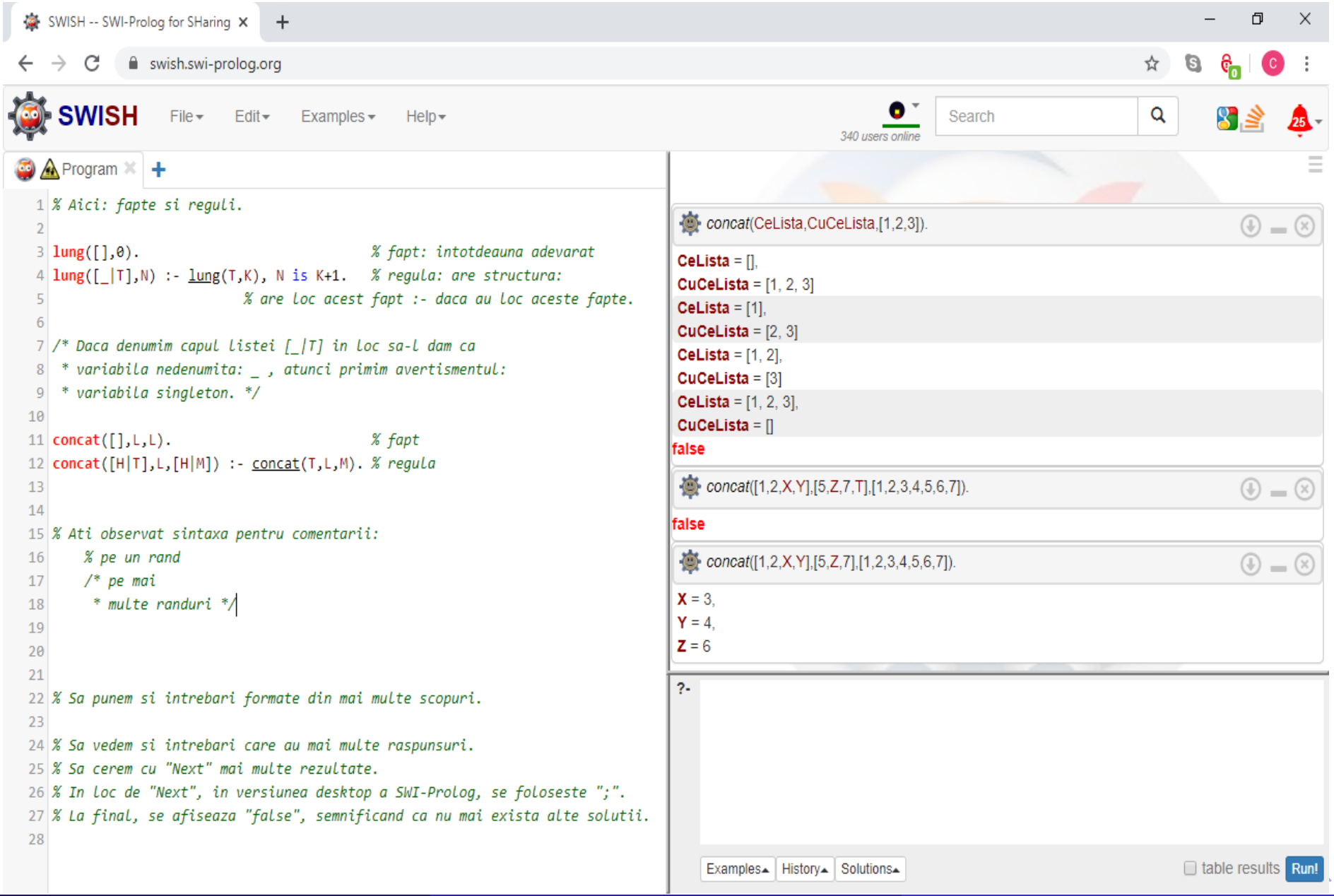
Încărcăm acest program în versiunea online a SWI–Prolog



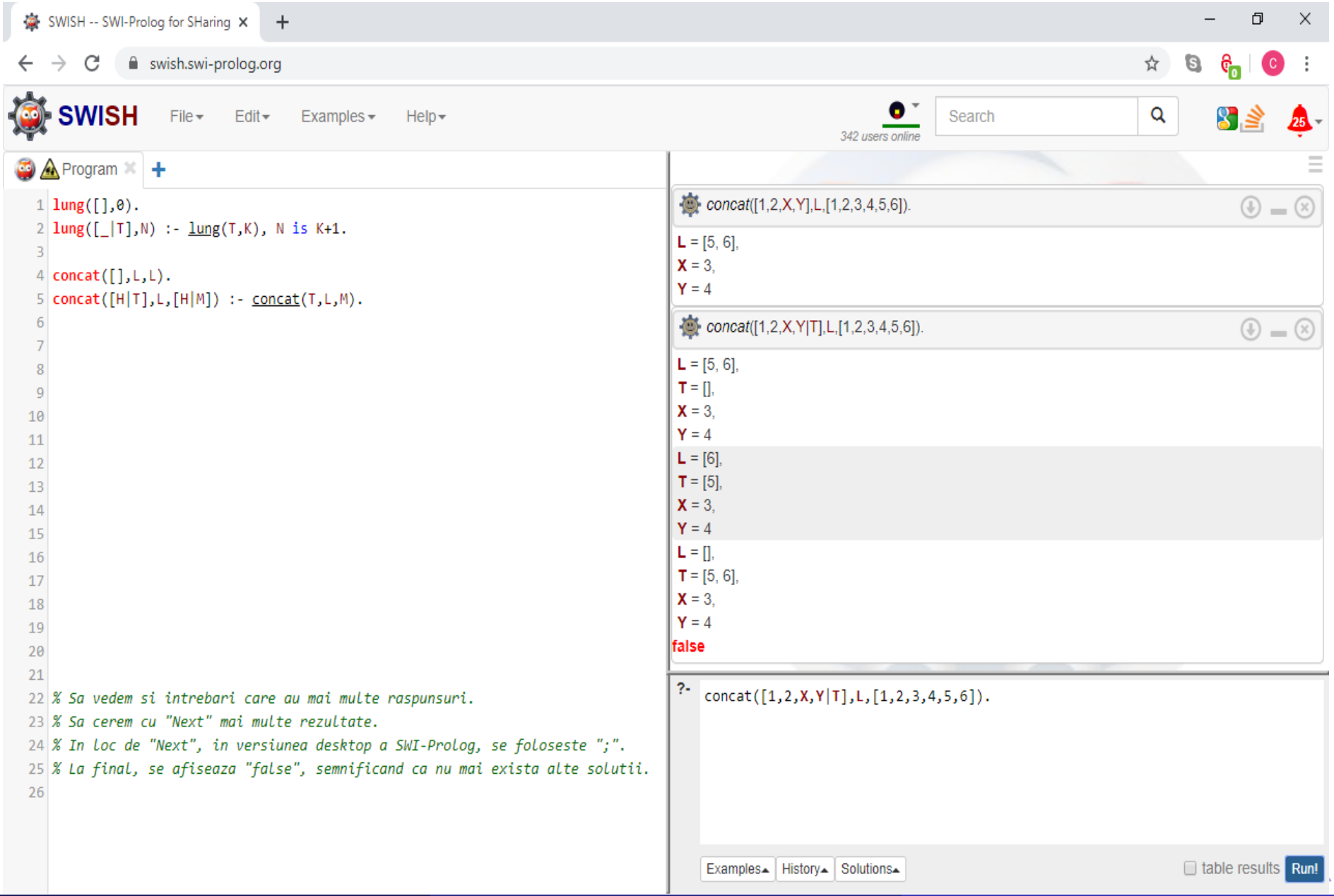
Interogări cu scopuri sau variabile multiple



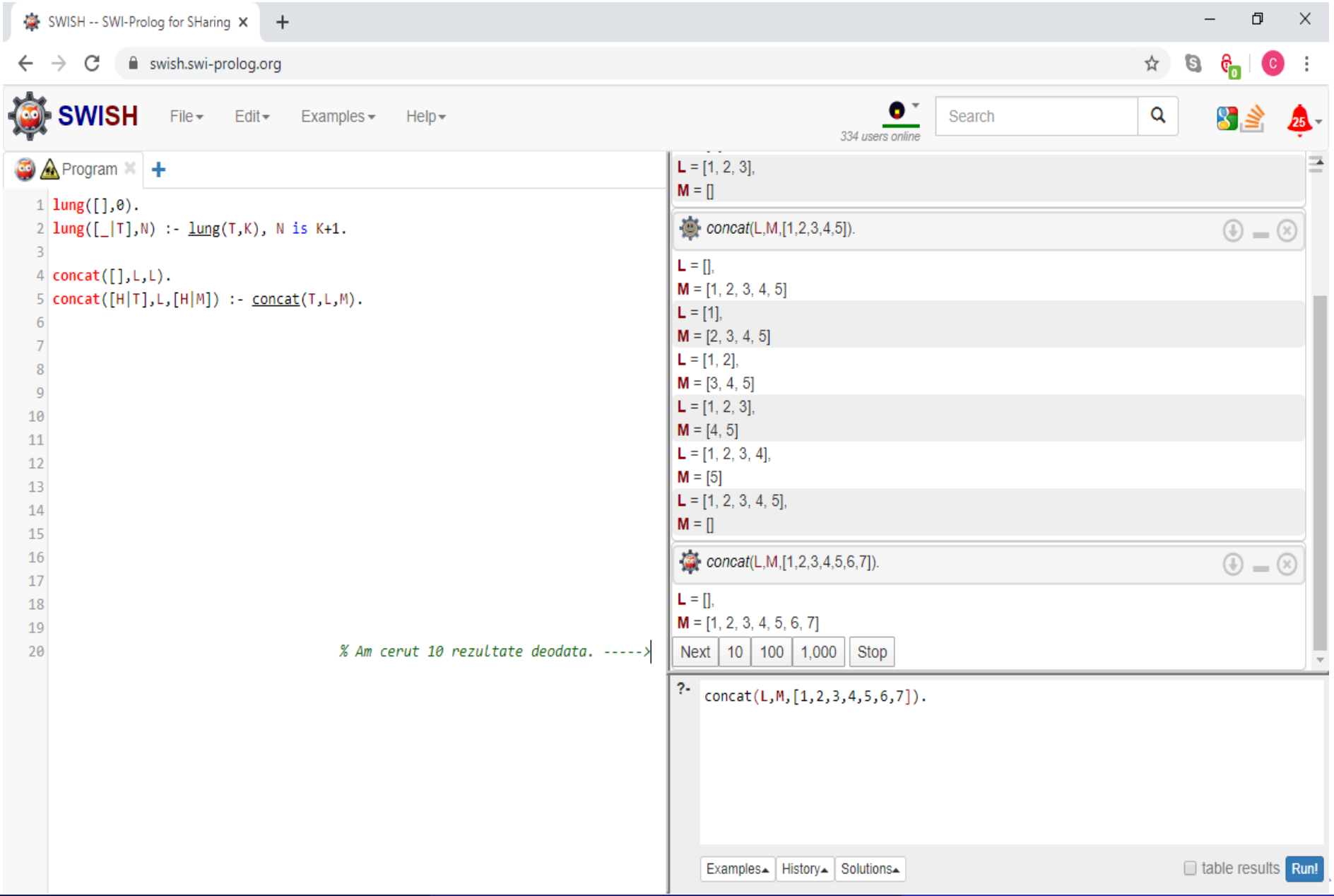
Întrebări cu mai multe răspunsuri posibile



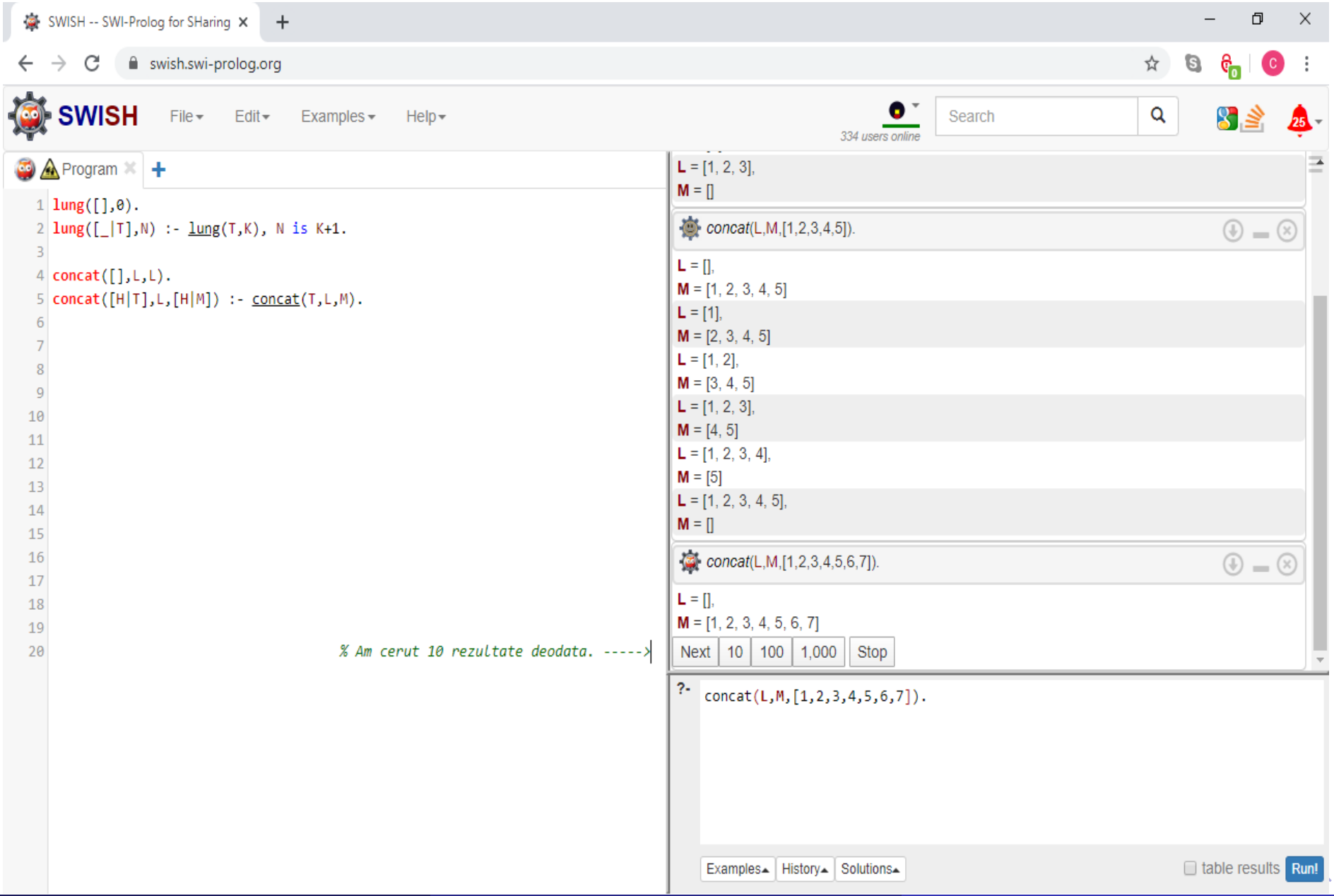
Vom vedea cum găsește Prologul aceste răspunsuri



Vedeți și celelalte opțiuni ale SWI–Prolog online

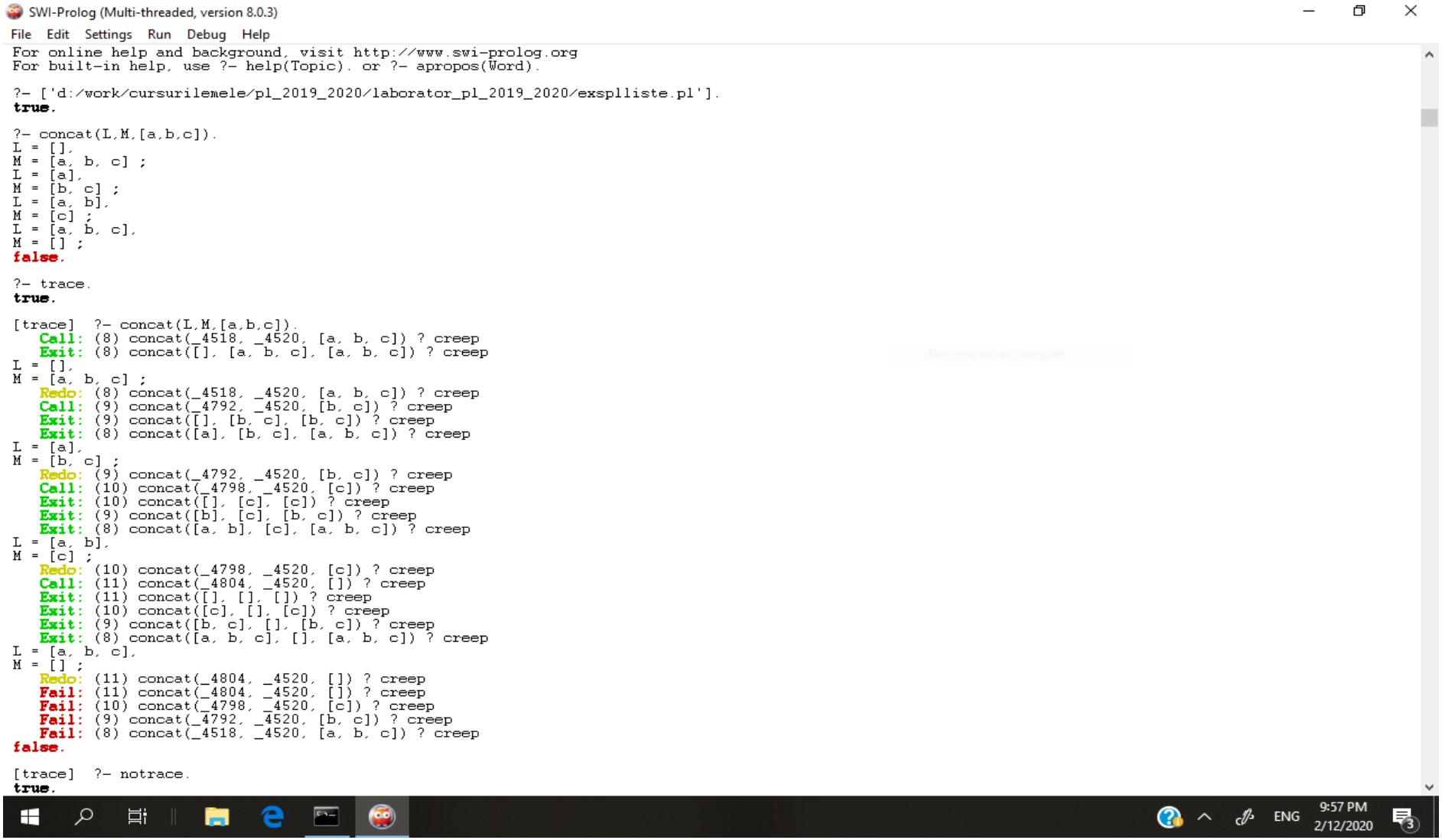


Inclusiv opțiuni pentru editare rapidă



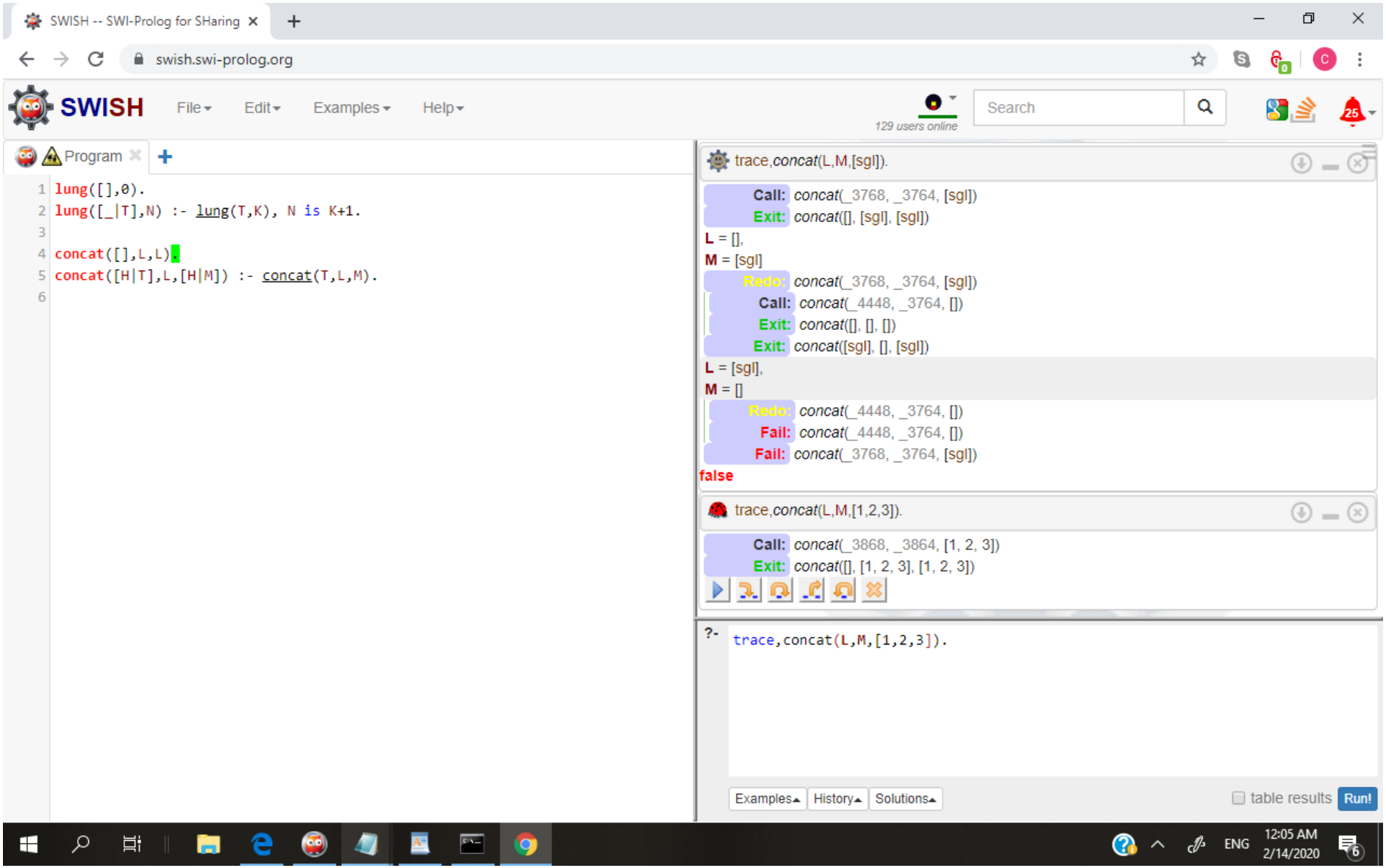
Și acum în versiunea desktop a SWI–Prolog

Am scris cele doua predicate de mai sus într–un fișier text cu extensia *.pl*:



Comanda *trace* produce detalierea pașilor urmați de Prolog pentru a rezolva interogările. Renunțare la această afișare detaliată: *notrace*.

trace și săgeată "Step into" în SWI–Prolog online



Unificare (orientativ) – detalii într–un curs următor

Două funcții **unifică** ddacă:

- acele funcții coincid
  și
- au aceleași argumente:

$$f(arg_1, \ldots, arg_n) = g(a_1, \ldots, a_k) \Longleftrightarrow \begin{cases} f = g \ (\Longrightarrow n = k) \text{ și} \\ arg_1 = a_1, \\ \vdots \\ arg_n = a_n. \end{cases}$$

Prolog–ul încearcă să unifice scopurile din interogare cu predicatele din:

- fapte;
- membrii stângi ai regulilor – dacă o astfel de unificare reușește, atunci următorul scop este membrul drept al aceleiași reguli, cu argumentele rezultate în urma unificării.

Considerăm interogarea:

```
?- concat(L1,L2,[a,b,c]).
```

Observați din pașii afișati cu *trace* că primul lucru executat de Prolog este redenumirea variabilelor din interogare în nume de forma *_număr*.

$\mathrm{concat} \neq \mathrm{lung}$, așadar $\mathrm{concat}(\mathrm{L1}, \mathrm{L2}, [\mathrm{a}, \mathrm{b}, \mathrm{c}])$ nu unifică:

- nici cu $\mathrm{lung}([\,], 0)$,
- nici cu $\mathrm{lung}([\mathrm{H}|\mathrm{T}], \mathrm{N})$.

În schimb, concat(L1,L2,[a,b,c]) unifică:

cu $\mathrm{concat}([\,], \mathrm{L}, \mathrm{L})$, pentru **L1** $= [\,]$ și **L2** $=$ **L** $= [\mathbf{a}, \mathbf{b}, \mathbf{c}]$ – aceasta este *prima soluție* a interogării,

dar și:

cu $\mathrm{concat}([\mathrm{H}|\mathrm{T}], \mathrm{L}, [\mathrm{H}|\mathrm{M}])$, pentru **H** $=$ **a**, **L1** $= [\mathbf{a}|\mathbf{T}]$, **L2** $=$ **L** și **M** $= [\mathbf{b}, \mathbf{c}]$,

așadar **următorul scop** este:

```
?- concat(T,L2,[b,c]).
```

Acesta unifică:

cu $\mathrm{concat}([\,], \mathrm{L}, \mathrm{L})$, pentru $\mathbf{T} = [\,]$ și $\mathbf{L2} = \mathbf{L} = [\mathbf{b}, \mathbf{c}]$ – așadar *a doua soluție* a interogării este: $\mathbf{L1} = [\mathbf{a}|\mathbf{T}] = [\mathbf{a}|[\,]] = [\mathbf{a}]$ și $\mathbf{L2} = [\mathbf{b}, \mathbf{c}]$,

dar și:

cu $\mathrm{concat}([\mathrm{H1}|\mathrm{T1}], \mathrm{L}, [\mathrm{H1}|\mathrm{M1}])$, pentru $\mathbf{H1} = \mathbf{b}$, $\mathbf{T} = [\mathbf{b}|\mathbf{T1}]$, $\mathbf{L2} = \mathbf{L}$ și $\mathbf{M1} = [\mathbf{c}]$,

așadar **următorul scop** este:

```
?- concat(T1,L2,[c]).
```

Acesta unifică:

cu $\mathrm{concat}([\,], \mathrm{L}, \mathrm{L})$, pentru $\mathbf{T1} = [\,]$ și $\mathbf{L2} = \mathbf{L} = [\mathbf{c}]$ – așadar *a treia soluție* a interogării este: $\mathbf{L1} = [\mathbf{a}|\mathbf{T}] = [\mathbf{a}|[\mathbf{b}|\mathbf{T1}]] = [\mathbf{a}|[\mathbf{b}|[\,]]] = [\mathbf{a}, \mathbf{b}]$ și $\mathbf{L2} = [\mathbf{c}]$,

dar și:

cu $\mathrm{concat}([\mathrm{H2}|\mathrm{T2}], \mathrm{L}, [\mathrm{H2}|\mathrm{M2}])$, pentru $\mathbf{H2} = \mathbf{c}$, $\mathbf{T1} = [\mathbf{c}|\mathbf{T2}]$, $\mathbf{L2} = \mathbf{L}$ și $\mathbf{M2} = [\,]$,

așadar **următorul scop** este:

?- concat(T2,L2,[ ]).

Întrucât:

[ ] nu unifică cu $[\mathrm{H3}|\mathrm{M3}]$,

avem:

$\mathrm{concat}(\mathrm{T2}, \mathrm{L2}, [\,])$ nu unifică cu $\mathrm{concat}([\mathrm{H3}|\mathrm{T3}], \mathrm{L}, [\mathrm{H3}|\mathrm{M3}])$.

Prin urmare:

$\mathrm{concat}(\mathrm{T2}, \mathrm{L2}, [\,])$ unifică numai cu $\mathrm{concat}([\,], \mathrm{L}, \mathrm{L})$, pentru **T2** $= [\,]$ și **L2** $=$ **L** $= [\,]$ – așadar *a patra* și *ultima soluție* a interogării este:
L1 $= [\mathbf{a}|\mathbf{T}] = [\mathbf{a}|[\mathbf{b}|\mathbf{T1}]] = [\mathbf{a}|[\mathbf{b}|[\mathbf{c}|\mathbf{T2}]]] = [\mathbf{a}|[\mathbf{b}|[\mathbf{c}|[\,]]]] = [\mathbf{a}, \mathbf{b}, \mathbf{c}]$ și **L2** $= [\,]$.

În cazul unui scop compus, de exemplu:

?- concat(A,[2,3],[1,2,3]),concat(A,[4,5],C).

toate scopurile trebuie satisfăcute, cu **aceleași valori** pentru *variabilele comune*.
Pentru exemplul de mai sus, *unica soluție*: **A** $= [\mathbf{1}]$, **C** $= [\mathbf{1}, \mathbf{4}, \mathbf{5}]$.