Exame de Programação em Lógica
Duração: 120min

NOME: _____

1. Considere o seguinte predicado:

```
n([],L,L).
n([H|L],NL,L) :-
   n(L,[H|NL], L).
```

   esse predicado implementa:

   - a cópia de uma lista
   - o inverso de uma lista.
   - o programa não funciona.

2. Considere as seguinte chamadas **(a)** `3+Y == Z+2` **(b)** `3+Y =:= Z+2` **(c)** `3+Y is 2+Z`.

   Se `Y=2` e `Z=3`:

   - todas são verdade.
   - **(a)** e **(b)** são verdade, mas **(c)** é falsa.
   - **(a)** e **(c)** são verdade, mas **(b)** é falsa.

3. Qual dos seguintes programas é equivalente ao programa `c/3` na questão 1:

```
(a). n([]) --> [].          (b).  n([]) --> [].
     n([H|L]) -->                    n([H|L]) -->
           [H], n(L).                      n(L), [H].
```

   - (a)
   - (b)
   - nenhum deles.

4. Dado o predicado `x/3` definido pelo fato `x(X,Y,Y-X)` :

   - `maplist(x,[2,3],[4,5],L)` tem como resposta [4-2,5-3];
   - `maplist(x,[2,3],[4,5],L)` tem como resposta [4,5,2,3];
   - `maplist(x,[2,3],[4,5],L)` falha.

5. considere agora `maplist(x,[2,3],[4,5],L),maplist(x,A,B,L)` . Tem como resposta: **(1)** A=[2,3], B=[4,5]; **(1)** A=[4,5], B=[2,3]; **(1)** falha.

6. Uma vantagem de tabulação é:

   - Garante que qualquer programa termine;
   - programas que não criam estruturas de dados recursivas terminam.
   - A terminação é um tema de investigação em aberto.

7. Para gerar uma lista de números consecutivos entre A e B escolheria: **(1)** between(A,B,I) **(1)** maplist(between,A,B,L) **(1)** forall(I,between(A,B,I),L).

8. Em Datalog é comum escrever um programa assim:

```
sum(X, max(Z)) :-
    a(X, Y),
    b(Y, Z).
```

   Isto funciona pq:

   - os predicados só têm escalares;
   - a cabeça só pode ter variaveis;
   - Datalog executa bottom-up.

9. Qual dos seguintes programas tem uma e apenas uma soluçao para `b(X,Y)`:

```
(a). b(X,Y) :- c(Y,X), !. (b) b(X,Y) :- !, c(X,X). | c(3,4). c(7,7).
     b(_).                                         |
```

   (1) a. **(2)** b **(3)** os dois

10. Qual dos seguintes programas tem uma e apenas uma soluçao para `b(X,7)`:

```
(a). b(X,Y) :- c(Y,X), !. (b) b(X,Y) :- !, c(X,X). | c(3,4). c(7,7).
     b(_).                                         |
```

   (1) a. **(2)** b. **(3)** os dois

11. No seguinte programa inc(X) :- ?(i(X)), X1 is X+1, ??(i(X1))

    (a) ? = assert e ?? = retract

    (b) ?? = assert e ? = retract

    (c) Não é correcto.

12. Qual destas chamadas emtra em ciclo sem gerar soluções:

    (1) `append(X,[2],X)` **(2)** `append(X,X,X)` **(3)** `append(X,Y,X)`

1 Pratica

1. Escreva um programa para calcular o número de Fibonacci $f(1) < -f(0) < -1 \wedge f(N) \leftarrow f(N-1) + f(N-2)$

   nnnxu

   (a) Usando a definição;

   (b) Garantindo tempo polinomial.

2. Dada uma lista de sublista, em que cada sublista tem dois elementos e representa um intervalo $[X, Y], X \leq Y$:

   ```
   [[1,2],[5,7],[6,10],[12,15]]
   ```

   Escreva um programa para calcular o maior intervalo contíguo. Pode assumir que as entradas estão ordenadas pelo primeiro elemento.

   No ex, 5:10

Syntax:

- Atoms: `hello`, `;`, `'quoted atoms'`
- Numbers: `22`, `55.8`, `3888888888888888888`
- Strings: ``` ``double quoted'' ``` in SWI, ``` `back-quoted` ``` in YAP;
- Lists Of Codes: ``` `back quoted` ``` in SWI, ``` ``double-quoted in YAP'' ```
- Lists of Chars: `[r,e,a,d,s,' ',e,a,s,i,e,r]`.
- Compound Terms, eg $\texttt{f}(\texttt{t}_1, \texttt{t}_2, \ldots, \texttt{t}_\texttt{n})$
- Lists are a compound term `[h|T]`
- Variables
- Facts: `team(mu).`
- Rules: `best(Team) :- champion(Team).`
- Directives`:- champion(Team).`

Builtins:

- control: `!`, `once(G)`, `forall(G,P)`, `(Guard->FirstYes;No)`, `(Guard->AllYes,No)`,
- Meta-Programming: `call(G)`, `call(G,Extra)`;
- Collecting Solutions: `findall(ho , G , L)`, `setof(Who,G, L).`
- Generating Alternatives: `repeat`, `between(I,J,N)`
- Input/Output of streams: `open(F,M,S)`, `close(S)`, `stream_property(S,P)`;
- Input/Output of characters: `get_code(C)`, `put_char(C)`,`get_code(S,C)`, `put_char(S,C)`,
- Input/Output of terms: `read(T)`, `write(T)`,`nl`,`read_term(S,T,OARIALGIA)`,
- Type conversion: `atom_to_codes(A,S)`, `atom_to_number(A,N)`,
- Lists: `append(LA,LB,LAB)`, `length(L,Sz)`, `member(E,Els)`
- App: `maplist(P,L)`, `select(P,L,P1)`, `foldl(P,N0,NF)`