|  |  |  |  |
| --- | --- | --- | --- |
| Texto, Carta  Descrição gerada automaticamente | **INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA**  **Campus Birigui**  **Bacharelado em Engenharia de Computação** | | |
| **Disciplina:** Inteligência Artificial | | | **Lista 3** |
| **Professor:** Prof. Dr. Murilo Varges da Silva | | | **Data:** 16/10/2023 |
| **Nome do Aluno:** Henrique Akira Hiraga | | **Prontuário:** BI300838X | |

LISTA 3 – PROLOG

1. O programa a seguir associa a cada pessoa seu esporte preferido.

joga(ana,volei).

joga(bia,tenis).

joga(ivo,basquete).

joga(eva,volei).

joga(leo,tenis).

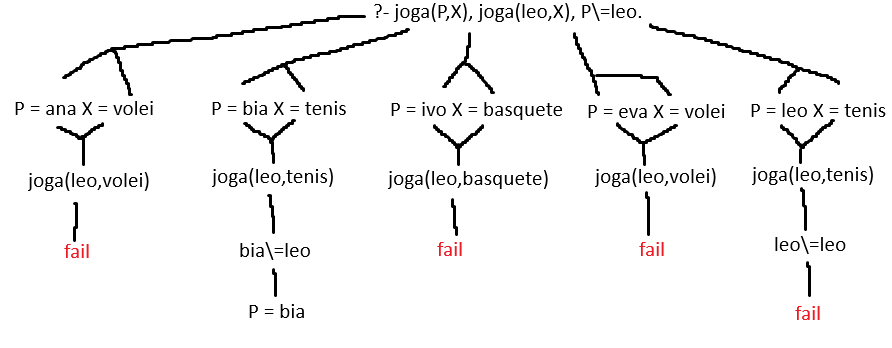
Suponha que desejamos consultar esse programa para encontrar um parceiro P para jogar com Leo. Então, podemos realizar essa consulta de duas formas:

* 1. ?- joga(P,X), joga(leo,X), P\=leo.
  2. ?- joga(leo,X), joga(P,X), P\=leo.

Desenhe as árvores de busca construídas pelo sistema ao respondercada

uma dessas consultas. Qual consulta é mais eficiente, por quê?

Resultado da árvore de busca da pesquisa (a):



Resultado da árvore de busca da pesquisa (b):

Diagrama

Descrição gerada automaticamente com confiança baixa

A consulta (b) é mais eficiente porque ela começa com uma restrição específica (Leo jogando volei) e, em seguida, encontra um parceiro que atende a essa restrição. Isso reduz a quantidade de backtracking necessário. A consulta (a) começa com a variável P e, em seguida, verifica a restrição em relação a Leo, o que pode levar a mais tentativas de unificação e, portanto, ser menos eficiente.

1. O predicado num classifica números em três categorias: positivos, nulo e

negativos. Esse predicado, da maneira como está definido, realiza retrocesso desnecessário. Explique por que isso acontece e, em seguida, utilize cortes para eliminar esse retrocesso.

num(N,positivo) :- N>0.

num(0,nulo).

num(N,negativo) :- N<0.

Considerando o predicado num, ao fazermos uma consulta como por exemplo num(5, C), ele irá achar a solução (C = positivo) e, em seguida fará o retrocesso para verificar se há outras soluções, o que é desnecessário para esse caso. O mesmo se aplica se entrarmos com um valor de N negativo, após o Prolog encontrar a solução (C = negativo) ele irá retornar e procurar outras soluções.

Para solucionarmos esses retrocessos desnecessários, basta adicionar o operador de corte ‘!’, que fará com que o Prolog não realize o retrocesso a partir desse ponto. O predicado então ficará da seguinte maneira:

num(N,positivo) :- N > 0, !.

num(0,nulo).

num(N,negativo) :- N < 0, !.

A adição do corte após as condições que satisfaçam as categorias "positivo" e "negativo" garante que o Prolog não faça retrocesso nessas regras após encontrar uma solução. Assim, o retrocesso desnecessário é eliminado, tornando o predicado mais eficiente. No entanto, a regra para o caso "nulo" não tem um corte, o que permite ao Prolog continuar a procurar por mais soluções se N for igual a zero.

1. Suponha que o predicado fail não existisse em Prolog. Qual das duas definições a seguir poderia ser corretamente usada para causar falhas?
   1. falha :- (1=1).
   2. falha :- (1=2).

A definição que ocasionará falha será a (b). Neste caso, a regra falha falhará porque a igualdade 1=2 é avaliada como falsa, o que levará à falha da regra. Isso é equivalente a usar fail para indicar uma falha, mas sem o uso direto do predicado fail.

1. Considere o programa a seguir:

animal(cão).

animal(canário).

animal(cobra).

animal(morcego).

animal(gaivota).

voa(canário).

voa(morcego).

voa(gaivota).

dif(X,X) :- !, fail.

dif(\_,\_).

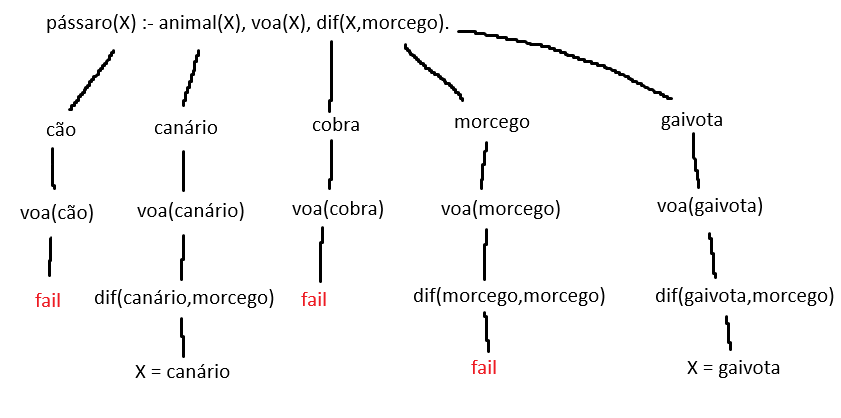
pássaro(X) :- animal(X), voa(X), dif(X,morcego).

Desenhe a árvore de busca necessária para responder a consulta

?- pássaro(X).

Em seguida, execute o programa para ver se as respostas do sistema correspondem àquelas que você encontrou.

Desenho da árvore de busca:



Execução do programa:

Texto

Descrição gerada automaticamente com confiança baixa

1. Defina um predicado recursivo para calcular o produto de dois números naturais usando apenas soma e subtração.

Predicado recursivo:

Uma imagem contendo Gráfico de dispersão

Descrição gerada automaticamente

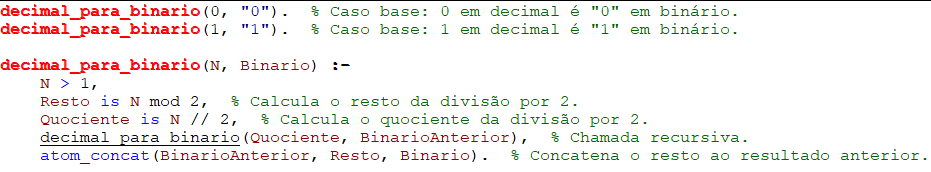
Resultado:

Gráfico

Descrição gerada automaticamente com confiança média

1. Defina um predicado recursivo exibir um número natural em binário.

Predicado recursivo:



Resultado:

Uma imagem contendo Diagrama

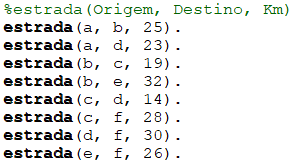
Descrição gerada automaticamente

1. O grafo a seguir representa um mapa, cujas cidades são representadas por letras e cujas estradas (de sentido único) são representados por números, que indicam sua extensão em km.

Desenho de pessoa com relógio no topo

Descrição gerada automaticamente com confiança baixa

* 1. Usando o predicado estrada(Origem,Destino,Km), crie um programa para representar esse mapa.



Tabela

Descrição gerada automaticamente com confiança baixa

* 1. Defina a relação transitiva dist(A,B,D), que determina a distância D entre duas cidades A e B.

Texto

Descrição gerada automaticamente com confiança média

Tabela

Descrição gerada automaticamente com confiança média