

Minha página principal ► Programação em Lógica ► Provas ► Mini-Teste 2 -- 2017/01/06

Data de início Sexta, 6 Janeiro 2017, 09:07

Estado Teste enviado

Data de submissão: Sexta, 6 Janeiro 2017, 11:05

Tempo gasto 1 hora 57 minutos

Nota 12,80 de um máximo de 20,00 (64%)

Pergunta 1

Respondida

Pontuou 1,000 de 2,500

Destacar pergunta

Explique o que faz o seguinte programa em Prolog e comente quanto à sua eficiência:

```
:- use_module(library(lists)).

p1(L1,L2) :-
    gen(L1,L2),
    test(L2).

gen([], []).
gen(L1, [X|L2]) :-
    select(X, L1, L3),
    gen(L3, L2).

test([_, _]).
test([X1,X2,X3|Xs]) :-
    (X1 < X2, X2 < X3; X1 > X2, X2 > X3),
    test([X2,X3|Xs]).
```

O programa diz-nos se as listas são iguais e ordenadas, quer por ordem crescente e decrescente. Mas a sua eficiência não é máxima porque falha quando a lista só tem um elemento.

Comentário:


O que são listas iguais e ordenadas?

O que tem eficiência a ver com casos de falha?

Pergunta 2

Respondida

Pontuou 2,500 de 2,500

 Destacar pergunta

O Asdrúbal resolveu implementar uma versão equivalente usando programação em lógica com restrições (PLR):

```
:- use_module(library(clpfd)).
```

```
p2(L1,L2) :-  
    length(L1,N),  
    length(L2,N),  
    %  
    pos(L1,L2,Is),  
    all_distinct(Is),  
    %  
    labeling([],Is),  
    test(L2).
```

```
pos([],_,[]).  
pos([X|Xs],L2,[I|Is]) :-  
    nth1(I,L2,X),  
    pos(Xs,L2,Is).
```

Contudo, após estudar melhor descobriu que o seu programa não era uma implementação correta em PLR. Escolha a opção que melhor traduz essa constatação:

Selecione uma opção de resposta:


- ☒ a. As variáveis de domínio estão a ser instanciadas antes da fase de pesquisa e nem todas as restrições foram colocadas antes da fase da pesquisa.
- ☐ b. Não é possível etiquetar *Is*, pois estas variáveis não têm domínio definido, e nem todas as restrições foram colocadas antes da fase da pesquisa.
- ☐ c. As variáveis de domínio estão a ser instanciadas antes da fase de pesquisa.
- ☐ d. Não é possível etiquetar *Is*, pois estas variáveis não têm domínio definido.
- ☐ e. [Não Responder]
- ☐ f. Nem todas as restrições foram colocadas antes da fase da pesquisa.

A sua resposta está correta.

Pergunta 3

Respondida

Pontuou 2,300 de 3,000

 Destacar pergunta

Corrija o programa, de modo a obter uma implementação correta em PLR.

```
all_distinct(Is),
test2(L2),
labeling([],Is).
```

```
pos([],_,[]).
pos([X|Xs],L2,[I|Is]) :-
element(I,L2,X),
pos(Xs,L2,Is).
```

```
test2([_,_]).
test2([X1,X2,X3|Xs]) :-
(X1 #< X2, X2 #< X3; X1 #> X2, X2 #> X3),
test2([X2,X3|Xs]).
```

Comentário:

Restrições proposicionais incorretas.

Pergunta 4

Respondida

Pontuou 5,000 de 6,000

Destacar pergunta

Para os seus cozinhados natalícios, o Bonifácio dispunha de um determinado número de ovos, com prazo de validade próximo. Este era o único recurso limitado, dispondo ele de quantidades intermináveis de todos os outros ingredientes. Na sua lista de receitas de doces, cada receita inclui, entre outras coisas, o tempo de preparação e o número de ovos que leva. O Bonifácio tem um tempo limitado para cozinhar, pretende fazer 3 pratos diferentes de doce e gastar o maior número possível de ovos de que dispõe. Usando programação em lógica com restrições, construa um programa que determine que receitas é que o Bonifácio deve fazer. O predicado ***sweet_recipes(+MaxTime,+NEggs,+RecipeTimes,+RecipeEggs,-Cookings,-Eggs)*** recebe o máximo de tempo disponível (*MaxTime*), o número de ovos disponíveis (*NEggs*), os tempos (*RecipeTimes*) e ovos (*RecipeEggs*) gastos por cada receita; devolve em *Cookings* os cozinhados a realizar (índices das listas *RecipeTimes/RecipeEggs*) e em *Eggs* os ovos utilizados.

```
| ?- sweet_recipes(60,30,[20,50,10,20,15],[6,4,12,20,6],Cookings,Eggs).
Cookings = [1,3,5],
Eggs = 24
```

```
| ?- sweet_recipes(120,30,[20,50,10,20,15],[6,4,12,20,6],Cookings,Eggs).
Cookings = [1,2,4],
Eggs = 30
```

```
constraint([],_,0).
constraint([A | R],RecipeTimes,Value):-
nth1(A,RecipeTimes,Value1),
Value #= Value1+Value2,
constraint(R,RecipeTimes,Value2).
```

```
calculateEggs([],_,0).
calculateEggs([A | R],RecipeEggs,NEggs):-
nth1(A,RecipeEggs,Value),
NEggs #= NEggs1 + Value,
calculateEggs(R,RecipeEggs,NEggs1).
```

Comentário:
nth1 devia ser element.

Pergunta 5

Respondida

Pontuou 2,000 de 6,000

Destacar pergunta

O Eleutério trabalha numa loja que corta prateleiras de madeira à medida. Para efeitos deste exercício, consideramos apenas cortes numa dimensão (isto é, assumimos que todas as prateleiras têm sempre a mesma largura). A páginas tantas, o Eleutério dispõe de várias pranchas, cada qual com um determinado comprimento, e precisa de cortar um conjunto de prateleiras, cada qual com a sua dimensão. Construa um programa, usando programação em lógica com restrições, que determine em que prancha é que cada prateleira deve ser cortada. O predicado **cut(+Shelves,+Boards,-SelectedBoards)** recebe a lista de prateleiras a cortar *Shelves* com a dimensão (unidimensional) de cada uma, e a lista *Boards* com o comprimento de cada prancha; devolve em *SelectedBoards* as pranchas a utilizar para cada prateleira.

```
| ?- cut([12,50,14,8,10,90,24], [100,45,70], S).
S = [2,3,3,2,1,1,2] ? ;
S = [3,3,2,3,1,1,2] ? ;
no
```

```
cut(Shelves,Boards,SelectedBoards):-
length(Shelves,T),
length(SelectedBoards,T),
length(Boards,Max),
domain(SelectedBoards,1,Max),

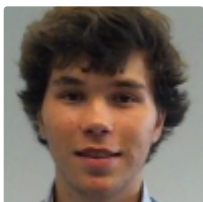
constraint2(Max,SelectedBoards,Shelves,Boards),
labeling([],SelectedBoards).

constraint2(0,_,_,[]).
constraint2(N,SelectedBoards,Shelves,[A | R]):-
constraint(Shelves,SelectedBoards,N,Value),
Value #=<= A,
```

Comentário:

Terminar revisão

NAVEGAÇÃO NO TESTE



Joao Pedro Antunes Pereira Gomes

1 2 3 4 5

Mostrar uma página de cada vez

Terminar revisão

© 2017 UPdigital - Tecnologias Educativas

Nome de utilizador: Joao Pedro Antunes Pereira Gomes (Sair)

Gestão e manutenção da plataforma Moodle U.PORTO da responsabilidade da unidade de Tecnologias Educativas da UPdigital. Mais informações:

apoio.elearning@uporto.pt | +351 22 040 81 91 | <http://elearning.up.pt>



Based on an original theme created by Shaun Daubney | moodle.org