Informação

Destacar pergunta

Texto informativo

Considere uma base de conhecimento que indica o nome, sexo e peso à nascença de bebés - *child(Name,Gender,WeightAtBirth)* - a evolução do peso - ***weight(Baby,DaysSinceBirth,Weight)*** - marcas e modelos de fraldas, os pesos mínimo e máximo indicados e o preço por fralda - *diapers(Brand, Model,MinWeight,MaxWeight,PricePerDiaper)* - e as compras de fraldas para os bebés, e em que dia tiveram lugar - *boughtFor(Baby,Brand,Model,DaysSinceBirth):*

```
child(john, m, 3.5).
child(mary, f, 4.1).

weight(john, 5, 3.3).
weight(john, 10, 3.5).
weight(john, 15, 3.8).
weight(mary, 5, 4.1).
weight(mary, 10, 4.5).
weight(mary, 15, 4.9).

diapers(dodot, small, 3, 5, 0.5).
diapers(dodot, medium, 4.5, 6, 0.4).
diapers(libero, small, 2, 4, 0.7).
diapers(libero, medium, 3.5, 5, 0.7).

boughtFor(john, dodot, small, 1).
boughtFor(mary, libero, small, 0).
boughtFor(john, dodot, small, 5).
boughtFor(mary, dodot, small, 8).
```

Responda às perguntas seguintes **SEM** utilizar os predicados *findall, setof* e *bagof*.

Pergunta 1

Respondida
Pontuou 1,00 de 1,00

Destacar pergunta

Enunciado da pergunta

Implemente o predicado ***priceDiapers(+Baby,+DaysSinceBirth,-Brand, -Model, -Price)*** que devolva em *Brand*, *Model* e *Price* a marca e modelo de fraldas, bem como o preço por fralda, das fraldas compradas para o bebé *Baby*, *DaysSinceBirth* dias depois de ter nascido.

Comentários

Comentário:

Pergunta 2

Respondida
Pontuou 0,20 de 1,00

Destacar pergunta

Enunciado da pergunta

Os bebés cujo peso nunca esteja abaixo do peso registado à nascença são conhecidos como bebés saudáveis (*healthy babies).*

Implemente o predicado ***healthyBaby(+Gender, -Baby)*** que devolva em *Baby* um bebé saudável do sexo *Gender.*

Comentários

Comentário:

Pergunta 3

Respondida
Pontuou 0,90 de 1,00

Destacar pergunta

Enunciado da pergunta

Implemente o predicado ***weightsAtBirth(+Babies, -WeightsAtBirth)*** que devolva em *WeightsAtBirth* a lista com os pesos à nascença de cada bebé na lista *Babies*. Os Pesos devem aparecer pela mesma ordem dos nomes respetivos.

Comentários

Comentário:

Pergunta 4

Respondida
Pontuou 0,45 de 1,00

Destacar pergunta

Enunciado da pergunta

Implemente o predicado ***diapersBoughtForHealthyBabies(+Brands,+Gender, -Babies)*** que devolva em *Babies* uma lista de pares *Brand-Baby* que indique para cada marca na lista *Brands* o nome de um bebé saudável do sexo *Gender* para quem tenham sido compradas fraldas dessa marca.

Se para uma dada marca, não existir um bebé com essas características, então a marca não deve aparecer no resultado. Por outro lado, o mesmo bebé pode aparecer várias vezes.

Conforme explicador anteriormente, os bebés cujo peso nunca esteja abaixo do peso registado à nascença são conhecidos como bebés saudáveis (*healthy babies).*

Comentários

Comentário:

Pergunta 5

Respondida
Pontuou 0,10 de 1,00

Destacar pergunta

Enunciado da pergunta

Implemente o predicado ***suitableDiapers(+Baby,+DaysSinceBirth)*** que escreva no ecrã todas as marcas e modelos (*Brand e Model*) que são adequadas para o bebé *Baby* quando tem *DaysSinceBirth* dias.

Comentários

Programação em Lógica

Informação

Destacar pergunta

Texto informativo

A Universidade Fechada para Férias (UFF) decidiu substituir o seu antigo sistema, SICADA, por uma implementação em Prolog, de forma a guardar informação sobre disciplinas e alunos. Alguns dos predicados implementados são os seguintes:

course(Code, Name, Acronym, SchoolYear, Semester, Credits).
student(Code, Name, FirstYear).
score(StudentCode, CourseCode, Year, Result).

O predicado *course* apresenta informação sobre disciplinas; o predicado *student* apresenta informação sobre estudantes (código, nome e ano de primeira inscrição na UFF); o predicado *score* apresenta informação sobre inscrições de estudantes em cadeiras e resultado obtido. O resultado pode ser um valor numérico (de 0 a 20, sendo que para obter aprovação é necessária uma nota igual ou superior a 10) ou o átomo *missed* (indicando que o estudante não foi avaliado).

Apresenta-se de seguida um extrato da base de conhecimento do sistema.

```
course(eic0026, 'Planilhas Orientadas a Gamers', 'PLOG', 3, 1, 5).
course(eic0084, 'Luau de Animação e Interação Gestual', 'LAIG', 3, 1, 7).
course(eic0024, 'Estimativas de Sofrimento', 'ESOF', 3, 1, 6).
course(eic0032, 'Realidade Comatosa', 'RCOM', 3, 1, 6).
course(eic0112, 'Lágrimas e Tremores para a Web', 'LTW ', 3, 1, 6).

student(2012012270, 'Artemisa Antonieta', 2012).
student(2012490160, 'Bernardete Bernardes', 2012).
student(2012687310, 'Cristalina Coronado', 2012).
student(2012380501, 'Demétrio Dourolindo', 2012).
student(2012380401, 'Eleutério Elisandro', 2012).
student(2012746621, 'Felismina Felizardo', 2012).

score(2012012270, eic0026, 2014, 20).      score(2012490160, eic0026, 2014, missed).
score(2012012270, eic0084, 2014, 16).      score(2012490160, eic0084, 2014, 7).
score(2012012270, eic0024, 2014, 17).      score(2012490160, eic0024, 2014, 10).
score(2012012270, eic0032, 2014, 12).      score(2012490160, eic0032, 2014, 4).
score(2012012270, eic0112, 2014, 18).      score(2012490160, eic0112, 2014, missed).
score(2012687310, eic0032, 2014, missed).  score(2012380501, eic0032, 2014, 11).
```

Pergunta 6

Respondida
Pontuou 1,00 de 1,00
Destacar pergunta

Enunciado da pergunta

Implemente o predicado *nCourses(?Count)*, que obtém em *Count* o número total de disciplinas existentes no sistema. Exemplo:

```
| ?- nCourses(C).
C = 5 ? ;
no
```

Comentários

Comentário:

Pergunta 7

Respondida
Pontuou 0,20 de 1,00
Destacar pergunta

Enunciado da pergunta

Implemente o predicado *studentAverage(+Student, ?Average)*, que obtém em *Average* a média do estudante *Student*. Note que a média é obtida tendo apenas em conta os resultados positivos (>=10) do estudante. Exemplo:

```
| ?- studentAverage(2012490160, Avg).
Avg = 10.0 ? ;
no

| ?- studentAverage(2012490490, Avg).
Avg = nd ? ;
no
```

Comentários

Comentário:

Pergunta 8

Não respondida
Pontuação 1,00
Destacar pergunta

Enunciado da pergunta

Implemente o predicado *cadeirao(?Course)*, que obtém em *Course* uma lista com o(s) nome(s) da(s) disciplina(s) da UFF com mais inscrições. Exemplo:

```
| ?- cadeirao(C).
C = ['Realidade Comatosa'] ? ;
no
```

Pergunta 9

Respondida
Pontuou 0,60 de 1,00
Destacar pergunta

Enunciado da pergunta

Considere o código seguinte:

```
whatDoIDo(Out):- student(C, Out, FY), score(C, _, EY, _), !, A is EY-FY, A > 6.
```

O que faz o predicado *whatDoIDo/1*? Que nome lhe daria?

O cut usado no predicado é verde ou vermelho? Justifique.

Comentários

Comentário:

Programação em Lógica

Informação

Destacar pergunta

Texto informativo

A rede social ***CHAT'inha*** permite que os utilizadores sigam as publicações uns dos outros. Essas relações podem ser vistas como um grafo dirigido, o qual pode ser representado pelo predicado *follows(Seguidor, Seguido)*, indicando que o primeiro utilizador segue o seguindo utilizador. Exemplo:

```
follows(asdrubal, capitulina).
follows(capitulina, asdrubal).
follows(capitulina, marciliano).
follows(irineu, epifanio).
follows(marciliano, epifanio).
```

Pergunta 10

Não respondida
Pontuação 1,50
Destacar pergunta

Enunciado da pergunta

Uma das medidas para análise da rede social é a sua densidade, sendo esta especificada como o rácio entre as ligações existentes na rede e o total de ligações possíveis. Implemente o predicado *density(Density)*, que obtém em *Density* a densidade da rede CHAT'inha (valor entre 0 e 1). Exemplo:

```
| ?- density(D).
D = 0.25 ? ;
no
```

Pergunta 11

Não respondida
Pontuação 2,50
Destacar pergunta

Enunciado da pergunta

Uma outra medida de análise da rede é a sua heterogeneidade, a qual pode ser visualizada através de um histograma. Implemente o predicado *heterogeneity(Histogram)*, que obtém em *Histogram* uma lista pares de valores que permitem construir o histograma da rede CHAT'inha. Cada par de valores representa uma quantidade de seguidores e quantos utilizadores têm esse número de seguidores. Exemplo:

```
| ?- heterogeneity(H).
H = [0-1, 1-3, 2-1] ? ;
no
```

Este resultado advém de existir 1 utilizador (irineu) com 0 seguidores; 3 utilizadores com 1 seguidor; e 1 utilizador com 2 seguidores.

Programação em Lógica

Informação

Destacar pergunta

Texto informativo

A partir daqui pode usar a biblioteca CLPFD

Pretende-se distribuir uma série de objetos num conjunto de N sacos. Cada objeto pode ser colocado num dos sacos de um subconjunto S ⊆ N.

Pergunta 12

Respondida
Pontuou 0,00 de 2,00
Destacar pergunta

Enunciado da pergunta

Pretende-se que objetos consecutivos sejam colocados sempre em sacos diferentes. Implemente, em PLR, o predicado *distribute(+Domains,-Vars)*, que recebe no primeiro argumento os subconjuntos de sacos onde os objetos podem ser colocados e devolve no segundo argumento a alocação de objetos a sacos. Exemplo:

```
| ?- distribute([[1,2],[1,3],[2,3],[3],[2,3]],Vars).
Vars = [1,3,2,3,2] ? ;
Vars = [2,1,2,3,2] ? ;
Vars = [2,3,2,3,2] ? ;
no
```

(Nota: o predicado *list_to_fdset/2*, da biblioteca clpfd, permite converter uma lista num conjunto FD, e o operador *in_set* permite definir o domínio de uma variável como sendo um conjunto FD.)

Comentários

Comentário:

Pergunta 13

Respondida
Pontuou 0,20 de 2,00
Destacar pergunta

Enunciado da pergunta

Ignorando a restrição da alínea anterior, considere agora que se pretende que cada saco fique ou vazio ou com um número de objetos entre um intervalo [MinObj,MaxObj] definido. O predicado *distribute(NBags,Domains,MinObj,MaxObj,Vars)* recebe o número de sacos a utilizar, os subconjuntos de sacos para cada objeto e os limites inferior e superior para o número de objetos em cada saco, e devolve no quinto argumento a alocação de objetos a sacos. Exemplo:

```
| ?- distribute(3,[[1,2],[1,3],[2,3],[3],[2,3]],2,3,Vars).
V = [1,1,3,3,3] ? ;
V = [2,3,2,3,2] ? ;
V = [2,3,2,3,3] ? ;
V = [2,3,3,3,2] ? ;
no
```

Implemente um programa usando PLR que permita fazer a distribuição referida.

Comentários

Comentário:

Programação em Lógica

Informação

Destacar pergunta

Texto informativo

Organizou-se uma conferência sobre Programação em Lógica, subdividindo-a em diferentes sessões, de acordo com o número expectável de artigos aceites.

Após a seleção efetiva dos artigos a serem apresentados na conferência, solicitou-se a cada orador que fornecesse as suas preferências relativamente às sessões em que poderia fazer a sua apresentação. Assim, cada orador forneceu uma lista, por ordem decrescente de preferência, das sessões possíveis. Esta informação foi depois compilada numa única lista: por exemplo, [[4,1,2],[6,2,1],[3,2],...] indica que o primeiro orador prefere a sessão 4 (depois a 1, depois a 2), o segundo prefere a 6, etc.

Cada sessão tem um número de apresentações associado. A lista [4,4,3,5,...] indica que as duas primeiras sessões podem ter até 4 apresentações, a terceira só pode ter 3, etc. Ao fazer a atribuição de apresentações às sessões, pretende-se que estas fiquem equilibradas (por exemplo, é preferível ter duas sessões com 50% das apresentações do que uma sessão sem apresentações alocadas).

Pergunta 14

Não respondida
Pontuação 1,50
Destacar pergunta

Enunciado da pergunta

Defina em PLR um predicado que calcule o custo de uma solução (atribuição de apresentações a sessões), tendo em conta as preferências dos oradores e a intenção de obter sessões equilibradas: *cost(+Prefs,+Sessions,+Schedule,-Cost)*.

A lista *Prefs* contém, como sublistas, as preferências de cada orador (o índice de um elemento na lista é o orador, e as sublistas contêm números de sessões). A lista *Sessions* contém os tamanhos de cada uma das sessões previstas (o índice de um elemento na lista é o número da sessão). A lista *Schedule*, de variáveis de domínio, contém a atribuição de oradores a sessões (o índice de um elemento na lista é o orador, e o valor é o número da sessão).

Sessões fora da lista de preferência de um orador podem ser-lhe atribuídas, mas obtendo todas elas igual baixa satisfação.

Pergunta 15

Não respondida
Pontuação 1,50
Destacar pergunta

Enunciado da pergunta

Implemente um programa em PLR que permita criar o programa da conferência, obtendo a solução ótima de acordo com a função de custo definida na alínea anterior. Exemplo:

```
| ?- conf_schedule([[4,1,2],[6,2,4,3,1],[3,2,1],...], [4,4,3,4,3,5,4,...], Schedule).
Schedule = ...
```