Gequoteerde zitting Prolog: Cocktails maken

NAAM: RICHTING:

Enkele praktische afspraken

- Je krijgt twee uur om deze opdracht **individueel** op te lossen.
- Je raadpleegt **enkel afgedrukte kopies** van de slides (eventueel met handgeschreven nota's) en de ingebouwde manual van SWI-Prolog (gebruik bv ?-help(write). of ?-apropos(select).)
- In de map **1415_Gequoteerde/Prolog_donderdag** op Toledo vind je de bestanden `cocktailsfacts.pl` and `run.pl`. Ook de indienmodule staat daar.
  - Het bestand `cocktailsfacts.pl` bevat de facts die gebruikt werden voor de voorbeeld queries van deze opdracht. Het bestand `run.pl` kan worden gebruikt m.b.v. `swipl -f run.pl` om de voorbeeldqueries uit te voeren. Dit bestand bevat ook de verwachte uitvoer voor deze queries.
  - Als de opdracht expliciet de naam (en ariteit) van een predicaat vermeldt, ben je verplicht om dezelfde naam (en ariteit) te gebruiken in je oplossing.
  - Je oplossing zet je in een bestand `prolog.pl` en de eerste lijnen van dit bestand moeten je naam, studentennummer en richting bevatten.

    ```
    % Jan Jansen
    % r0123456
    % master cw
    ```

  - Na twee uur, of wanneer je klaar ben, dien je het `prolog.pl` bestand in via Toledo.

Deel 1: Bestellingen opnemen

Iemand organiseert een breng-je-eigen-ingrediënten cocktail feestje. De gasten die komen naar het feestje zijn gegeven als `guest/1` predicaat.

```
guest(gastNaam).
```

Waarin `gastNaam` de naam is van een gast die naar het feestje komt.
Elke gast heeft zijn geliefde cocktails. Deze zijn gegeven in het `drinks/2` predicaat.

```
drinks(gastNaam,cocktailNaam).
```

Waarin `cocktailNaam` de naam is van een cocktails die `gastNaam` graag drinkt.
Er zijn nog twee predicaten die het drinkgedrag van de gasten beschrijven. `drinks_minimum/2` beschrijft het minimum aantal cocktails dat die gast zal drinken. `drinks_maximum/2` beschrijft het maximum aantal cocktails dat die gast kan verdragen.

```
drinks_minimum(gastNaam,minNr).
drinks_maximum(gastNaam,maxNr).
```

De gegeven `ingredient/2` facts beschrijven welke ingrediënten nodig zijn voor elke cocktail. Ingrediënten worden als onuitputtelijk en gedeeld over cocktails beschouwd. B.v.b. als zowel een *margarita* als een *mojito* het ingrediënt `limes` nodig hebben, dan is dit ingrediënt maar één keer nodig in totaal.

```
ingredient(cocktailNaam,ingredientNaam).
```

Sommige gasten vinden elkaar niet leuk. Dit wordt gegeven in `dislikes/2` facts, waarin de gast in het eerste argument de gast in het tweede argument niet leuk vindt. Het is onmogelijk voor gasten om zichzelf niet leuk te vinden.

```
dislikes(gastNaam1,gastNaam2).
```

Er is een limiet op hoeveel dingen elke gast kan dragen. Dit is gegeven in `guest_carry_capacity/2` facts.

```
guest_carry_capacity(gastNaam,nummer).
```

Opdracht 1 Schrijf een predicaat `drink_order(DrinkOrder)` dat `DrinkOrder` unificeert met een lijst die bestellingen bevat voor elke gast. Bestellingen voor een gast worden voorgesteld met een term `order(gastNaam, [cocktailNaam1, cocktailNaam2...])`. Je oplossing moet in staat zijn alle mogelijke oplossingen (dus alle geldige `DrinkOrder`s) terug te geven. Bijvoorbeeld

```
?- drink_order(DrinkOrder).
```

resulteert (voor het voorbeeld in cocktailsfacts.pl) mogelijks in een bestelling waar *bruno* twee margaritas drinks, en waar *sandra* en *stephen* beiden één bloody mary drinken.

```
DrinkOrder = [
  order(bruno, [margarita, margarita]),
  order(sandra, [bloody_mary]),
  order(stephen, [bloody_mary])
] ;
... (er zijn alternatieven mogelijk)
```

`DrinkOrder` moet ook voldoen aan de volgende voorwaarden:

- Elke gast heeft enkel bestellingen voor cocktails die hij of zij graag drinkt.
- Elke gast bestelt een aantal cocktails dat tussen zijn of haar minimum en maximum ligt.
- Als gast A gast B niet leuk vindt (`dislikes(gastA,gastB)`), dan drinkt A meer cocktails dan B uit jaloezie.
- Het totale aantal ingredienten dat nodig is om alle bestellingen te maken is kleiner dan of gelijk aan de totale draagcapaciteit van alle gasten samen (herinner je: dubbele benodigde ingrediënten tellen als één ingrediënt)

Opmerking: de volgorde van de elementen in je `DrinkOrder` moet niet exact dezelfde zijn als die in het voorbeeld; jouw antwoord moet enkel dezelfde elementen bevatten. (De volgorde waarin ze voorkomen maakt niet uit)

Deel 2: Gaan winkelen

Deze ingrediënten worden verkocht in winkels. Deze winkels worden gegeven als `store/1` facts.

```
store(winkelNaam).
```

Het `sells/2` predicaat beschrijft welke winkels welke ingrediënten verkopen.

```
sells(winkelNaam,ingredientNaam1).
sells(winkelNaam,ingredientNaam2).
```

Aangezien dit een breng-je-eigen-ingrediënten cocktail feestje is, wordt van de gasten verwacht dat ze alle ingrediënten zelf mee te nemen. Ze mogen eveneens ingrediënten voor de cocktails van andere mensen meenemen. We gaan er opnieuw van uit dat ingrediënten onuitputtelijk zijn en gebruikt kunnen worden voor meerdere cocktails. Dus als twee cocktails hetzelfde ingrediënt nodig hebben, dan moet het maar één keer gekocht worden.

Opdracht 2: Schrijf een predicaat `shopping_list(DrinkOrder,ShoppingList)` dat, voor een gegeven `DrinkOrder`, een `ShoppingList` maakt die voor elke gast een winkel lijstje bevat. Een winkel lijstje voor een gast wordt voorgesteld met een term `shopat(gastNaam, winkelNaam, [ingredientNaam1, ingredientNaam2 ...])`. Je oplossing moet in staat zijn alle mogelijke oplossingen (dus alle geldige `ShoppingLists`) terug te geven voor een gegeven `DrinkOrder`.

Bijvoorbeeld:

```
?- drink_order(DrinkOrder), shopping_list(DrinkOrder,ShoppingList).
```

kan (voor de `DrinkOrder` uit het voorbeeld gegeven in Deel 1) resulteren in een `ShoppingList` waar `bruno` gaat winkelen in de `carrefour` en er `limes`

en `tequila` koopt, waar `sandra` gaat winkelen in de `aldi` en `tomatoes` koopt, en waar `stephen` gaat winkelen in `delhaize` en `vodka` koopt.

```
ShoppingList = [
  shopat(bruno, carrefour, [limes, tequila]),
  shopat(sandra, aldi, [tomatoes]),
  shopat(stephen, delhaize, [vodka])
] ;
...
```

`ShoppingList` moet ook voldoen aan de volgende voorwaarden:

- Het aantal ingrediënten dat een gast koopt is kleiner dan of gelijk aan de draagcapaciteit van die gast
- Elke gast bezoekt enkel één winkel
- Als gast A de gast B niet leuk vindt (`dislikes(gastA,gastB)`), dan koopt A enkel ingrediënten wanneer
  - A het ingrediënt nodig heeft voor een van zijn eigen cocktails ***of***
  - B heeft het ingrediënt niet nodig voor een van zijn cocktails

  (dit betekent dat A geen ingrediënten meeneemt die bijdragen tot een cocktails voor B, maar niet tot een cocktails van A)

Opmerking: de volgorde van de elementen in je `ShoppingList` moet niet exact dezelfde zijn als die in het voorbeeld; jouw antwoord moet enkel dezelfde elementen bevatten. (De volgorde waarin ze voorkomen maakt niet uit)