Wstęp do Sztucznej Inteligencji - rok akademicki 2023/2024

Przed rozpoczęciem pracy z notatnikiem zmień jego nazwę zgodnie z wzorem: `NrAlbumu_Nazwisko_Imie_PoprzedniaNazwa`.

Przed wysłaniem notatnika upewnij się, że rozwiązałeś wszystkie zadania/ćwiczenia.

Temat: Prolog - Programowanie w logice cz. I

Zapoznaj się z treścią niniejszego notatnika czytając i wykonując go komórka po komórce. Wykonaj napotkane zadania/ćwiczenia.

Informacje wstępne

Na zajęciech będziemy korzystać z `SWI Prolog` w wersji online poprzez https://swish.swi-prolog.org/. Możliwe jest również pobranie i zainstalowanie prologa na komputerze: http://www.swi-prolog.org/download/stable.

Prolog jest językiem programowania w logice. W przeciwieństwie do popularnych proceduralnych języków programowania (C, PASCAL,...) Prolog jest językiem deklaratywnym. Oznacza to, że pisząc program w Prologu, skupiamy się na definicjach obiektów i związkach między nimi, a nie na sposobie wykonania danego zadania. Nacisk zatem jest położony na to **CO** obliczyć zamiast **JAK** obliczyć. Zadaniem programisty jest zapisać logiczną strukturę problemu, sterowanie wykonaniem natomiast zostawiamy tzw. interpreterowi Prologa.

Prolog opisany został w 1972 przez Alain Colmerauer (standardy ISO w 1995 i 2000). Używany jest w takich dziedzinach jak:

- relacyjne bazy danych,
- logika matematyczna,
- problemy abstrakcyjne,
- języki naturalne,
- automatyka,
- algebra symboliczna,
- biochemia,
- sztuczna inteligencja.

Logika użyta w Prologu jest standardową dwuwartościową logiką matematyczną, opisywaną algebrą Boole'a. Podstawowe operacje to:

- I (AND)
- LUB (OR)
- NEGACJA (NOT)

- JEŚLI (IF)

Zaś wartości logiczne to `true` oraz `false`.

Prolog stara się uzgodnić wszystkie zdania w programie do logicznej wartości `true`.

Podstawowymi pojęciami w prologu są **termy**, wiązane **uporządkowanymi relacjami**.

Przykład:

- termem mogą być `jas`, `malgosia`
- relacjami mogą być `brat`, `siostra`
- relacje są uporządkowane, co oznacza, że `jas` jest w relacji `brat` z termem `malgosia` , ale niekoniecznie na odwrót

Schemat programu w prologu

1. deklaracja faktów (baza danych termów i relacji)
2. deklaracja reguł dozwolonych do manipulowania faktami
3. deklaracja problemu (zadanie pytania)

Hello World w Prologu

Ucząc się nowego języka programowania, często pierwszym programem pisanym przez początkującego programistę jest program wyświetlający pozdrowienia na ekranie. W Prologu podobna funkcje pełni program, który odpowiada na pytanie co kto lubi.

Ćwiczenie 1:

- Jeśli używasz Prologa zainstalowanego na komputerze: Stwórz nowy plik tekstowy i zapisz go pod nazwa `lubi.pl`. Edytuj jego treść. Zapisz w nim:

```
lubi(jarek, jablko).
lubi(jarek, gruszka).
lubi(kasia, piwo).
lubi(kasia, hamburger).
```

Pamiętaj o kropce na końcu!

Plik `lubi.pl` stanowi tzw. bazę wiedzy, aby ją załadować wykonaj polecenie `consult('sciezka_do_pliku').` lub krócej `['sciezka_do_pliku'].` (po wcześniejszym uruchomieniu `swipl`). `prolog ?- ['lubi.pl'].`

- Jeśli używasz Prologa online to treść pliku `lubi.pl` umieść w komórce `Program`.

Następnie wykonaj następujące zapytania.

```
?- lubi(jarek, piwo).

?- lubi(kasia, hamburger).
```

Baza znanych faktów zostaje przeszukana w takiej kolejności w jakiej fakty zostały wprowadzone i jeśli napotkany zostanie fakt, o który pytamy, odpowiedź będzie pozytywna, a zatem prawdziwość faktu zostanie potwierdzona. Jeśli taki fakt nie zostanie znaleziony, Prolog odpowiada „nie". „Nie" w Prologu nie oznacza, ze coś jest nieprawdziwe, a raczej, że Prolog na podstawie znanych mu faktów (i reguł) nie jest w stanie go potwierdzić.

Fakty

W powyższym przykładzie zostały zapisane fakty typu: "jarek lubi jabłka". Fakty mogą również przyjmować postać:

- `czlowiek(jarek).` -> jarek jest człowiekiem
- `rodzice(maria, stefan, henio).` -> maria oraz stefan są rodzicami henia
- `ladna_pogoda.` -> fakt; jeśli zapytamy `ladna_pogoda.` to otrzymamy odpowiedź twierdzącą.

W powyższych przykładach `czlowiek`, `rodzice`, `ladna_pogoda` są **predykatami**. Należy pamiętać o kropkach kończących deklaracje każdego faktu.

Fakty mogą być również zapisane za pomocą bardziej skomplikowanych struktur:

`ksiazka(tytul('Ogniem i mieczem'), autor(henryk, sienkiewicz), 42353).`

Stałe i zmienne

Stałe w prologu pisane są z małej litery (np. `jarek`, `ksiazka`). Zmienne pisane są z dużej litery lub zaczynają się od znaku podkreślenia (np. `X`, `_zmienna`, `Zmienna`). Zmienne można wykorzystać do zadawania bardziej interesujących pytań Prologowi.

Ćwiczenie 2:

Wykonaj zapytanie:

```
?- lubi(kasia, X).
```

Naciśnięcie `ENTER` powoduje zakończenie poszukiwań kolejnych rozwiązań. Naciśniecie `;` powoduje poszukiwanie alternatywnych rozwiązań. Średnik jest w Prologu oznaczeniem logicznej operacji `OR`.

Wykonaj kolejne zapytania:

```
?- lubi(X, jablko).

?- lubi(X,Y).
```

Do zmiennych przypisywane są kolejne znalezione wartości (**uzgadnianie**), które pasują do zapytania. Odbywa się to w procesie zwanym **nawracaniem**.

Reguły

Reguła składa się głowy reguły oraz ciała. Aby spełniona była przesłanka reguły, spełnione muszą być wszystkie jej podcele, w tym przypadku oddzielone przecinkiem, co w Prologu oznacza logiczne `AND`. Przykład:

```
lubi(stefan, X) :- slodkie(X), zdrowe(X).
     GŁOWA      IF        WARUNKI
```

Powyższa reguła mówi nam, że `stefan` lubi `coś` co jest `slodkie` i `zdrowe`.

Ćwiczenie 3:

Wprowadź do bazy (`lubi.pl`) poniższe fakty i regułę:

```
slodkie(jablko).
slodkie(czekolada).
zdrowe(jablko).
lubi(stefan, X) :- slodkie(X), zdrowe(X).
```

Zapytaj co lubi stefan.

Jak widać, stefan lubi jabłka. Prolog znalazł tą odpowiedź, mimo że w zbiorze faktów nie było wprost powiedziane, ze stefan lubi jablka. Stefan nie lubi natomiast czekolady, gdyż mimo iż słodka, nie jest zdrowa (przynajmniej nic o jej zdrowotnym wpływie nie jest Prologowi wiadomo...).

Zadanie 1:

Stwórz nowy plik z następującą zawartością i załaduj go do pamięci:

```
mezczyzna(adam).
mezczyzna(stefan).
mezczyzna(staszek).
mezczyzna(marek).
kobieta(ala).
kobieta(alina).
kobieta(maria).
kobieta(ania).
rodzice(stefan, staszek, maria).
rodzice(ala, staszek, maria).
rodzice(ania, marek, alina).
```

Predykat `rodzice(X,Y,Z)` określa, ze ojcem X jest Y, a matką Z.

Zadaj pytania:

1. kim są rodzice stefana?
2. kogo ojcem jest staszek?

1 - Rodzice to Ala i Staszek

2 - Jest ojcem Stefana

Uwaga: Jeśli chcemy zapytać na przykład tylko o ojca stefana, możemy w miejsce zmiennej oznaczającej matkę wstawić zmienną anonimową `_`.

```
rodzice(stefan, Ojciec, _).
```

Zapytania oraz uzyskane odpowiedzi na nie umieść w komórce poniżej.

YOUR ANSWER HERE

Zadanie 2:

Do bazy wiedzy z zadania 1 dodaj regułę określającą, kiedy X jest siostrą Y.

Uwaga! Pamiętaj, że `,` w Prologu pełni rolę AND, natomiast `;` rolę OR.

```
siostra(X, Y) :- ...  X/==Y, rodzice(X, A, B), rodzice(Y, A, B).
```

YOUR ANSWER HERE

Arytmetyka

Prolog umożliwia operacje na liczbach, pamiętać jednak trzeba o różnicach w zapisie:

```
X = 3*4+2.
```

a zapisem

```
X is 3*4+2.
```

Pierwsza komenda wypisze string, a druga dokona obliczeń

Wypróbuj również zapytanie:

```
X=3*4+2, display(X).
```

Na stronie http://www.swi-prolog.org/pldoc/man?section=functions sprawdź, jakie możliwości daje SWI-Prolog jeśli chodzi o działania arytmetyczne.

Ćwiczenie 4:

Co zrobić, aby jeśli X jest mniejsze od 4 Wynik miał taką samą wartość jak X?

Przyjrzyj się przykładowi poniżej:

```
pom(X,W):-X>4, W is X*2.
pom(X,W):-W is X.
```

Widać, że jeśli nie jest spełniony warunek w pierwszej klauzuli predykatu `pom`, to wykorzystana jest druga reguła. Prawidłowo zatem Prolog odpowie na pytanie:

```
?- pom(2, Wynik).
```

Problem jest natomiast gdy zadamy pytanie:

```
?- pom(5, Wynik).
```

Pierwsza odpowiedz jest prawidłowa, jeśli jednak zapytamy o alternatywę (naciśniemy średnik), dostaniemy drugą odpowiedź, wywnioskowaną z drugiej reguły. Aby uniknąć takich sytuacji często stosuje się mechanizm odcięcia zapisywany jako wykrzyknik `!`.

```
pom(X,W):-X>4, W is X*2, !.
pom(X,W):-W is X.
```

Odcięcie powoduje, że po dojściu do danego miejsca, Prolog nie będzie próbował uzgadniać ponownie celów stających po lewej stronie znaku „!", w naszym przypadku chodzi o cel główny jakim jest pytanie `pom(5,Wynik).`.

Sprawdź działanie poprawionego predykatu `pom`.

Zadanie 3:

Jak obliczyć dwukrotną wartość podanej liczby, jeśli ta liczba jest większa od 4?

```
pomnoz_2(X, Wynik) :- X>4, Wynik is 2*X.
```

Czy

```
pomnoz_2(X, Wynik) :- X>4, Wynik = 2*X.
```

należy wybrać pierwszą opcję

YOUR ANSWER HERE

Zadanie 4:

Zbuduj bazę wiedzy, w której zawarta będzie informacjach, w jakich latach rządzili królowie np.

```
krol(jan_I, 1323, 1355)
```

oznacza, że jan_I rzadzil od 1323 roku do 1355 roku.

Napisz regułę umożliwiającą zadawanie pytań, kto rządził w danym roku.

```
rzadzil(Krol, Rok) :- ...
```

Stworzoną bazę wiedzy, reguły oraz kilka przykładowych zapytań z odpowiedziami umieść w komórce poniżej.

rzadzil(Krol, Rok) :- krol(Krol, A, B), Rok=<A, Rok>=B.

YOUR ANSWER HERE

Dodatkowe polecenia

Sprawdź w dokumentacji

http://www.swi-prolog.org/pldoc/man?section=quickstart

za co odpowiadają następujące polecenia/predykaty:

- `make.`
- `listing.`
- `[user].`