



SYSTEM RERUTACJI OPARTY O ŚRODOWISKO PROLOG

Wykonał Paweł Pauszek

Ideą programu było stworzenie systemu służącego do rekrutacji wojskowej. Program zadaje użytkownikowi pytania a ten na nie odpowiada. Można powiedzieć, że program pełnił także rolę chatbota. Wszystko opiera się o środowisko języka Prolog, czyli języka logicznego, w którym nie opisuje się co ma program robić (tak jak ma to miejsce w klasycznych językach programowania), tylko co chce się osiągnąć. Logika programu jest wyrażona w kategoriach relacji, reprezentowanych jako fakty i reguły a obliczenia są inicjowane przez wykonanie zapytań dotyczących tych relacji.

Cały program działa na zasadzie systemu ekspertowego, czyli takiego, który wykorzystuje opisaną bazę wiedzy (wcześniej zdefiniowane fakty oraz reguły) do podejmowania decyzji przez człowieka eksperta, w celu rozwiązania problemów. Zakłada on wspomaganie rozwiązywania problemów na zasadach zbliżonych do działania ludzkiego mózgu.

Jako interfejs graficzny w Prologu można wykorzystać XPCE lub moduł JPL, korzystający z elementów Javy, w tym Swinga. Wykorzystałem drugą opcję, czyli moduł JPL. Jest on mocno przestarzały co czyni jego wizualność na dzisiejsze standardy po prostu brzydkim.

Do działania programu niezbędne jest posiadanie SWI-Prolog oraz Javy JRE, które można pobrać za darmo.

Poniżej zamieszczę i opiszę działanie kodu programu:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

PS P:\Firefox\ChatBot-Prolog-main> swipl -s .\Military-Recruitment-Chatbot.pl
% Extended DLL search path with
% 'd:/Java/jre1.8.0_331/bin/server'
% 'd:/Java/jre1.8.0_331/bin'
Warning: p:/firefox/chatbot-prolog-main/military-recruitment-chatbot.pl:107:
Warning: Singleton variables: [JOP]
Warning: p:/firefox/chatbot-prolog-main/military-recruitment-chatbot.pl:128:
Warning: Singleton variables: [Kandydat,JOP]
Welcome to SWI-Prolog (threaded, 64 bits, version 8.4.2)
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software.
Please run ?- license. for legal details.

For online help and background, visit https://www.swi-prolog.org
For built-in help, use ?- help(Topic). or ?- apropos(Word).

1 ?- startuj.
□
```

W celu załadowania programu posłużyłem się poleceniem `swipl -s nazwa_pliku.pl`. Program startuje dopiero po wprowadzeniu w terminalu polecenia `startuj`.

```
Military-Recruitment-Chatbot.pl X
Military-Recruitment-Chatbot.pl
1  :- use_module(library(jpl)).
2  startuj :-
3      powitanie.
4
5  cechy(Kandydat, narkotyki) :- zapytaj(Kandydat, "Czy brales kiedykolwiek narkotyki?").
6  cechy(Kandydat, uzaleznioty) :- zapytaj(Kandydat, "Czy byles lub nadal jestes od czegos uzaleznioty?").
7  cechy(Kandydat, kategorie) :- zapytaj(Kandydat, "Czy masz kategorie wojskowa A?").
8  cechy(Kandydat, wiek) :- zapytaj(Kandydat, "Czy masz wiecej niz 18 lat?").
9  cechy(Kandydat, testp) :- zapytaj(Kandydat, "Czy wyrazasz zgode na przeprowadzenie testu psychologicznego?").
10 cechy(Kandydat, testf) :- zapytaj(Kandydat, "Czy zgadzasz sie na przeprowadzenie testu sprawnosciowego?").
11 cechy(Kandydat, patriota) :- zapytaj(Kandydat, "Czy jestes patriota?").
12 cechy(Kandydat, zainteresowanie) :- zapytaj(Kandydat, "Czy interesujesz sie szeroko pojeta militaryzacja?").
13 cechy(Kandydat, powinnosc) :- zapytaj(Kandydat, "Czy czujesz wewnetrzny obowiazek obrony swojego panstwa?").
14 cechy(Kandydat, smierc) :- zapytaj(Kandydat, "Czy jestes gotowy zginac za swoj kraj?").
15 cechy(Kandydat, psychika) :- zapytaj(Kandydat, "Czy kiedykolwiek leczyles sie psychiatrycznie?").
16 cechy(Kandydat, psychika2) :- zapytaj(Kandydat, "Czy masz lub miales stwierdzona chorobe psychiczna?").
17 cechy(Kandydat, jedzenie) :- zapytaj(Kandydat, "Czy masz dobra tolerancje gastronomiczna?").
18 cechy(Kandydat, odpornosc) :- zapytaj(Kandydat, "Czy posiadasz dobra odpornosc (chorowanie maksymalnie 2 razy do roku?).").
19
```

Najpierw deklaruje chęć używania modułu z biblioteki JPL, która odpowiada w programie za graficzny interfejs.

Po wpisaniu polecenia `startuj` uruchamiania się „interfejs” powitanie, mający na celu powitanie użytkownika oraz pobranie od niego imienia.

Powyżej są także zadeklarowane reguły dla Kandydata, który posiada określone cechy. Dla każdej z cech, zostaje zadane indywidualne pytanie. Jeżeli użytkownik odpowie twierdząco to zostają do Kandydata przypisane odpowiednie cechy.

```
20  wynik(Kandydat, idealny_zolnierz) :-
21      cechy(Kandydat, kategorie),
22      cechy(Kandydat, wiek),
23      cechy(Kandydat, testp),
24      cechy(Kandydat, testf),
25      cechy(Kandydat, patriota),
26      cechy(Kandydat, zainteresowanie),
27      cechy(Kandydat, powinnosc),
28      cechy(Kandydat, smierc).
29
30  wynik(Kandydat, dobry_zolnierz) :-
31      cechy(Kandydat, kategorie),
32      cechy(Kandydat, wiek),
33      cechy(Kandydat, testp),
34      cechy(Kandydat, testf),
35      cechy(Kandydat, patriota).
36
```

```
37  wynik(Kandydat, moze_zolnierz) :-
38      cechy(Kandydat, narkotyki),
39      cechy(Kandydat, kategorie),
40      cechy(Kandydat, wiek),
41      cechy(Kandydat, testp),
42      cechy(Kandydat, testf).
43
44  wynik(Kandydat, nie_zolnierz) :-
45      cechy(Kandydat, narkotyki),
46      cechy(Kandydat, uzaleznioty),
47      cechy(Kandydat, psychika),
48      cechy(Kandydat, psychika2).
49
50  wynik(Kandydat, ewentualnie) :-
51      cechy(Kandydat, jedzenie),
52      cechy(Kandydat, odpornosc).
53
54  wynik(_, inne).
55
```

Do podjęcia decyzji stworzyłem konkretne schematy (wyniki), które zawierają określone cechy Kandydata. Tak jak opisałem wcześniej, Kandydat określa swoje cechy poprzez interfejs graficzny Tak/Nie, a program na podstawie cech przypisuje mu konkretny schemat. W przypadku kiedy Kandydat nie pasuje pod żaden schemat przypisuje mu się schemat inne.

```
57 kd(Kandydat):-  
58     wynik(Kandydat, Decyzja),  
59     koniec(Kandydat, Decyzja),  
60     zmien.  
61  
62 pytaj(Kandydat,Pytanie):-  
63     wyswietl(Kandydat,Pytanie).  
64  
65 :- dynamic tak/1, nie/1.  
66  
67 zapytaj(K, P) :-  
68     (tak(P) ->  
69         true  
70     ;  
71     (nie(P) ->  
72         fail  
73     ;  
74         pytaj(K, P))  
75     ).  
76  
77 zmien :- retract(tak(_)),fail.  
78 zmien :- retract(nie(_)),fail.  
79 zmien.
```

„kd” uruchamia się po wpisaniu imienia przez użytkownika. Następuje tutaj dopasowanie cech pod dany schemat i wyświetlenie ostatecznej decyzji.

„pytaj” służy za wyświetlenie pytań z interfejsem graficznym.

Dynamic, assert oraz retract służą do deklaracji, że zmienne tak/nie będą się zmieniać/będą dynamiczne.

Zapytaj służy do wywoływania „pytaj” oraz pomaga przy określaniu cech (przy odpowiedzi ustala czy cecha jest true czy false).

```
81 powitanie :-  
82     jpl_new('javax.swing.JFrame', ['Chatbot'], F),  
83     jpl_new('javax.swing.JLabel', ['Rozmowa na stanowisko wojskowe'], LBL),  
84     jpl_new('javax.swing.JPanel', [], Pan),  
85     jpl_call(Pan, add, [LBL], _),  
86     jpl_call(F, add, [Pan], _),  
87     jpl_call(F, setLocation, [200,50], _),  
88     jpl_call(F, setSize, [800,700], _),  
89     jpl_call(F, setVisible, [@(true)], _),  
90     jpl_call(F, toFront, [], _),  
91     jpl_new('javax.swing.JOptionPane', [], JOP),  
92     jpl_call('javax.swing.JOptionPane', showInputDialog, [F, 'Proszę podaj swoje imie'], N),  
93     atomic_list_concat(['Witaj ', N, ' w elektronicznym systemie rekrutacji wojskowej. Zadamy Ci pare pytan, prosze odpowiadaj szczerze. '],  
94     Witaj),  
95     jpl_call(JOP, showMessageDialog, [F, Witaj], _),  
96     jpl_call(F, dispose, [], _),  
97  
98     (N == @(null) ->  
99         write('Przerwano'),  
100         fail  
101     ;  
102         kd(N)  
103     ).
```

Powitanie jest „interfejsem” powitalnym. Odpowiada on za interfejs graficzny dzięki korzystaniu z komponentów Java Swing. Odwoływanie do Swinga tworząca pomocą `jpl_call` lub `jpl_new`. Powitanie pyta także użytkownika o imię oraz wywołuje `kd`. W celu połączenia stringów stosuje `atomic_list_concat`.

```
105  wyswietl(Kandydat, Pytanie) :-
106      atomic_list_concat([Kandydat, ' ', Pytanie], Atom),
107      jpl_new('javax.swing.JFrame', ['Chatbot'], F),
108      jpl_new('javax.swing.JLabel', ['Rozmowa na stanowisko wojskowe'], LBL),
109      jpl_new('javax.swing.JPanel', [], Pan),
110      jpl_call(Pan, add, [LBL], _),
111      jpl_call(F, add, [Pan], _),
112      jpl_call(F, setLocation, [200, 50], _),
113      jpl_call(F, setSize, [800, 700], _),
114      jpl_call(F, setVisible, [@(true)], _),
115      jpl_call(F, toFront, [], _),
116      jpl_new('javax.swing.JOptionPane', [], JOP),
117      jpl_call('javax.swing.JOptionPane', showConfirmDialog, [F, Atom], N),
118      jpl_call(F, dispose, [], _),
119      write(N), nl,
120      ((N == 0) ->
121          assert(tak(Pytanie))
122      ;
123          assert(nie(Pytanie)), fail).
124
```

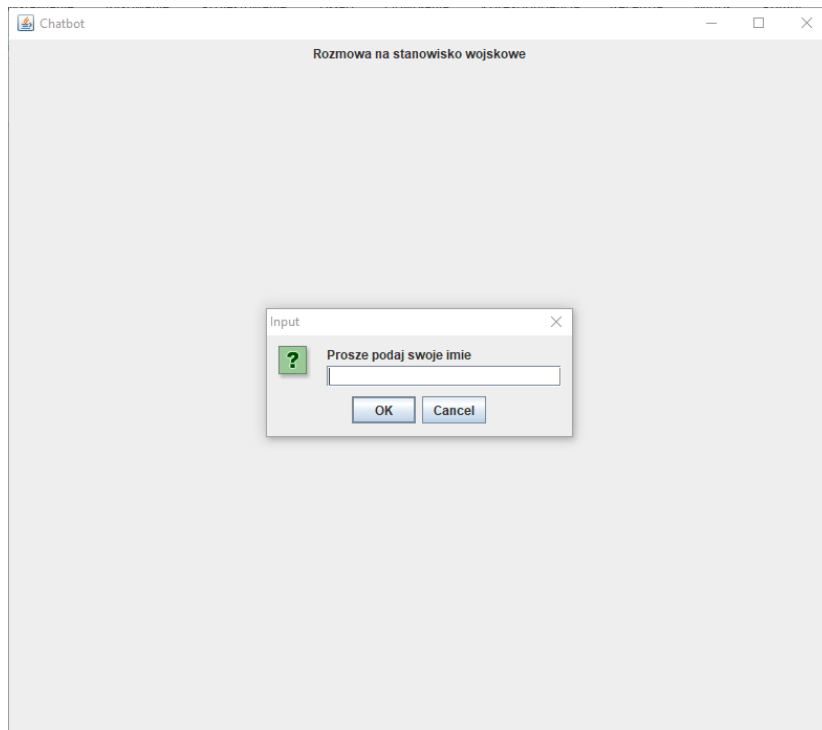
Wyswietl odpowiada za stworzenie dla każdego pytania interfejsu graficznego i zadanie samego pytania. Kliknięcie odpowiedzi jako Yes zwraca 0, a No lub Cancel zwraca 1 i 2. Przypisuje tutaj także odpowiedzi 0 czyli Yes jako „tak” a pozostałym „nie”.

```
126  koniec(Kandydat, Decyzja) :-
127      jpl_new('javax.swing.JFrame', ['Chatbot'], F),
128      jpl_new('javax.swing.JLabel', ['Rozmowa na stanowisko wojskowe'], LBL),
129      jpl_new('javax.swing.JPanel', [], Pan),
130      jpl_call(Pan, add, [LBL], _),
131      jpl_call(F, add, [Pan], _),
132      jpl_call(F, setLocation, [200, 50], _),
133      jpl_call(F, setSize, [800, 700], _),
134      jpl_call(F, setVisible, [@(true)], _),
135      jpl_call(F, toFront, [], _),
136      jpl_new('javax.swing.JOptionPane', [], JOP),
137
138      ((Decyzja == nie_zolnierz) ->
139          Decyzja2 = 'Przykro nam ale nie spelniasz podstawowych kryteriow. Nie mozesz przejsc dalej. Zostajesz odrzucony.'
140      ; (Decyzja == moze_zolnierz) ->
141          Decyzja2 = 'Niestety ale nie zakwalifikowales sie w pierwszej turze ale wciaz nasz szanse w drugiej.'
142      ; (Decyzja == dobry_zolnierz; Decyzja == idealny_zolnierz) ->
143          Decyzja2 = 'Gratulacje! Idealnie nadajesz sie na zolnierza. Na pewno sie do Ciebie odezwiemy.'
144      ; (Decyzja == inne) ->
145          Decyzja2 = 'Niestety zostales odrzucony. Nie pasujesz do schematu zolnierza. Pamietaj takze, ze wyrazenie zgody na testy jest obowiazkowe by przejsc dalej.'
146      ; (Decyzja == ewentualnie) ->
147          Decyzja2 = 'Nie mogliśmy dopasować twoich odpowiedzi do naszych wymagań dlatego zostały Ci zadane dodatkowe pytania. Z twoich odpowiedzi wynika, że zostaniesz powołany tylko w przypadku braku personelu.'
148      ),
149
150      jpl_call('javax.swing.JOptionPane', showMessageDialog, [F, Decyzja2], _),
151      jpl_call(F, dispose, [], _).
```

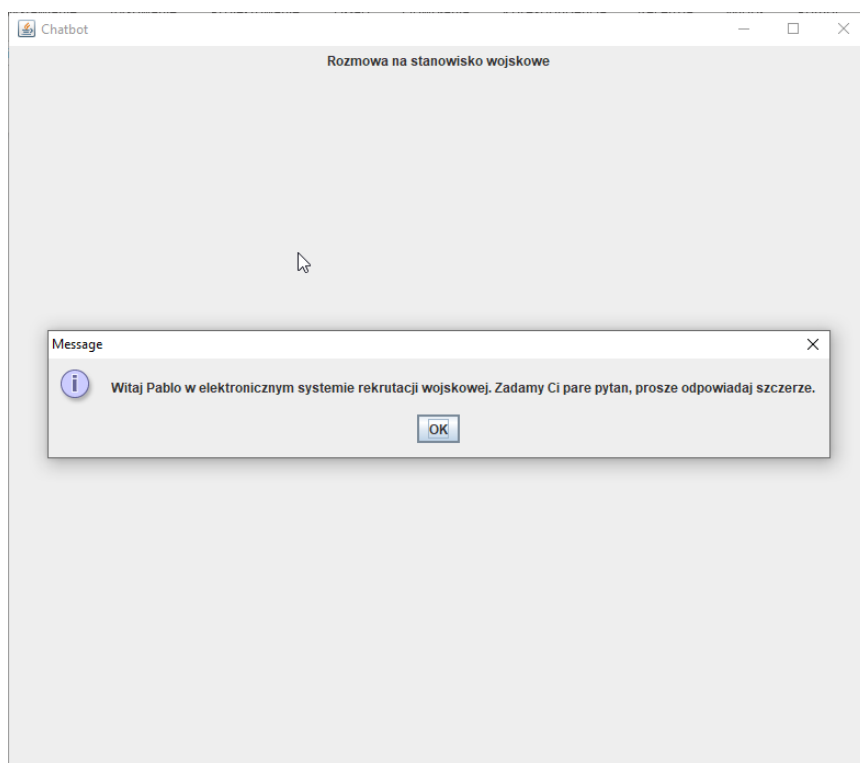
Koniec jest ostatecznym interfejsem programu. Poza ponownym stworzeniem interfejsu graficznego, odpowiada on za wypisanie decyzji w sposób przyjemny i czytelny dla oka (sprawdza decyzje i przypisuje drugiej decyzji opisowy tekst).

Demonstracja działania programu:

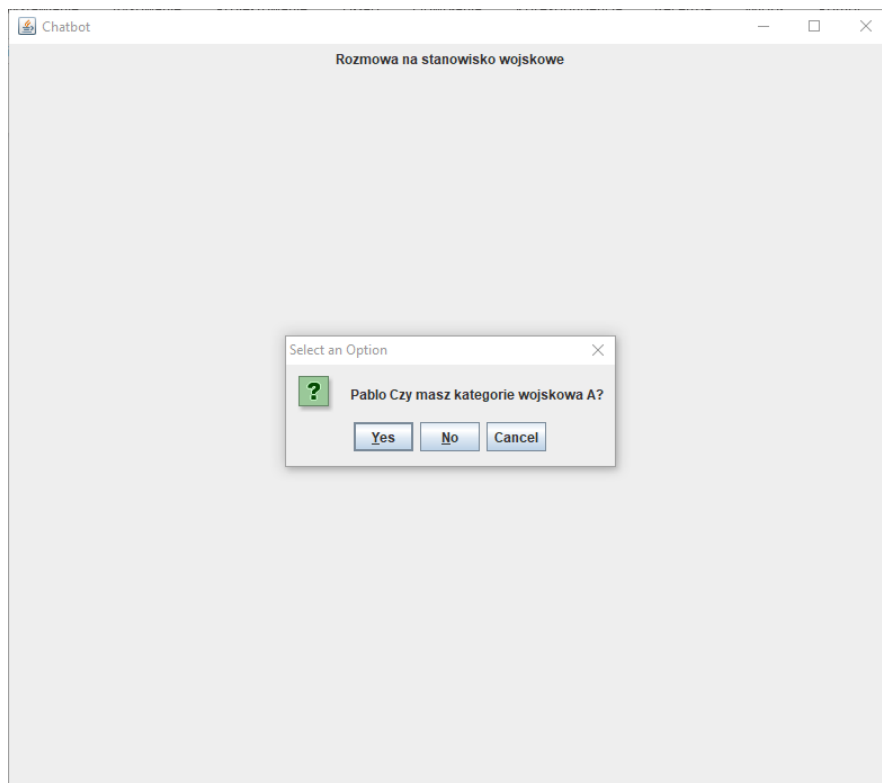
Pierwsze uruchomienie:



Wpisanie imienia:



Zadawanie pytań:



Określenie decyzji:

