Projekt 1 - Labirynt

Temat

Zestaw predykatów do szukania ścieżki w labiryncie oraz jej wyświetlania,

Wymagania

Labirynt zawiera się w prostokątnym obszarze i jest reprezentowany jako macierz (lista wierszy), w których atom `x` oznacza ścianę (mur), a atom `o` - wolną przestrzeń (korytarz).

```
[ [o, x, x, x, x],
  [o, o, o, o, x],
  [x, o, x, o, x],
  [x, o, x, o, x],
  [x, o, o, o, x],
  [o, o, x, x, x]]
```

1. Predykat `path(Labyrinth, Start, End, Path)` - powinien znajdować dowolną ścieżkę w labiryncie `Labyrinth` pomiędzy punktem początkowym `Start` i końcowym `End`. Parametry `Start` i `End` są listami dwuelementowymi zawierającymi współrzedne punktu poczatkowego i końcowego. Indeksy rozpoczynają się od 1.

   Znaleziona ścieżka nie musi być najkrótsza. W przypadku, gdy ściezka nie istnieje, predykat powinien zakończyć się porażką (`false`).

   Jeśli istnieje wiele ścieżek, predykat powinien je kolejno znajdować (po naciśnięciu średnika). Predykat nie powinien natomiast generować niepoprawnych rozwiązań. Nie powinien także wpadać w nieskończoną pętlę.

   Przykład.

   ```
   path([[o, x, x, x, x],
         [o, o, o, o, x],
         [x, o, x, o, x],
         [x, o, x, o, x],
         [x, o, o, o, x],
         [o, o, x, x, x]], [1,1], [5,1], Path).

   Path = [[1,1],[2,1],[2,2],[3,2],[4,2],[5,2],[5,1]]
   ```

2. Predykat `display_lab(Labyrinth)` wyświetla labirynt na konsoli zaznaczając ściany literami `x`, a przestrzenie puste - spacjami. Dla labiryntu z poprzedniego przykładu efekt powinien być nastepujący

   ```

   ```

```
  x x x x
        x
x   x   x
x   x   x
x       x
    x x x
```

3. Predykat `display_lab(Labyrinth, Path)` wyświetla labirynt na konsoli tak jak poprrzednio, ale dodatkowo zaznacza pola leżące na ścieżce kropkami `'.'`. Dla labiryntu i ściezki z przykładu 1 powinniśmy otrzymać

```
. x x x x
. .     x
x . x   x
x . x   x
x .     x
. . x x x
```

Uwagi

1. Predykaty będą testowane częściowo automatycznie i dlatego wymagane jest zachowanie **nazw predykatów** oraz kolejności i formatu argumentów zgodnie ze specyfikacją zadania.

2. Predykat `path` powinien może generować następne rozwiązania