# Implementacja grafu oraz algorytmów wyszukujących: DFS i BFS.

Arkadiusz Cyktor 200367

15 maja 2014

W mojej implementacji grafu użyłem **macierzy sąsiedztwa** - przechowuje ona informacje o połączeniach pomiędzy poszczególnymi **wierzchołkami**. Te natomiast tworzone są na bazie struktury "wierzcholek"i umieszczane w odpowiednim miejscu (pole odpowiadające numerowi wierzchołka) przechowującego ich wektora.
Wierzchołki łączone są **krawędziami**, za kształt których również odpowiada odpowiednia struktura - "krawedz". Dodając obiekt tego typu określamy jakie dwa wierzchołki ma ze sobą łączyć oraz jaką dane połączenie ma mieć wagę. Podobnie jak wierzchołki, krawędzie również przechowywane są w wektorze.

Algorytm wyszukujący **DFS** porusza się po grafie przy pomocy macierzy sąsiedztwa. Po odwiedzeniu przez algorytm wierzchołek jest odznaczany przez przypisanie wartości true zmiennej "czy odwiedzony", a następnie przechodzi do kolejnych nieodwiedzonych sąsiadów danego wierzchołka - i tak aż do chwili, w której każdy sąsiad zostanie odznaczony jako odwiedzony, wtedy algorytm wraca do pierwszego rozgałęzienia. Na koniec wyświetlany jest komunikat o odnalezieniu szukanego elememtu, a wszystkie wierzchołki ponownie oznaczane są jako nieodwiedzone. Algorytm ten korzysta ze stosu, z którego pobiera numery kolejnych wierzchołków do odwiedzenia.
Algorytm **BFS** również wykorzystuje macierz sąsiedztwa, ale w jego przypadku dochodzi jeszcze kolejka realizowana przez strukturę "lista BFS". W kolejce tej umieszczane są wierzchołki, które po kolei pobierane są do odczytu i zastępowane przez swoich nieodwiedzonych sąsiadów. Po odwiedzeniu przez algorytm wierzchołek jest odznaczany. Na koniec wyświetlany jest komunikat o odnalezieniu szukanego elememtu, a wszystkie wierzchołki ponownie oznaczane są jako nieodwiedzone.