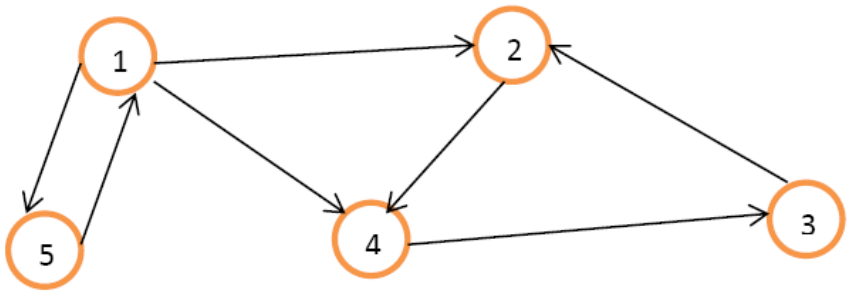
Algorytmy i złożoność obliczeniowa

Laboratorium 5: Grafy. Macierzowa i listowa reprezentacja grafu.

1. **Grafy – wprowadzenie.**

Graf jest strukturą składającą się ze zbioru wierzchołków (węzłów) i zbioru krawędzi łączących wierzchołki ze sobą. Zbiór wierzchołków będziemy oznaczać przez V (od ang. vertex), a zbiór krawędzi przez E (ang. edge). Najbardziej intuicyjną reprezentacją grafu jest reprezentacja graficzna. Węzły reprezentowane są najczęściej jako koła, natomiast krawędzie jako linie (w grafach nieskierowanych) lub strzałki (w grafach skierowanych) łączące ze sobą węzły. Poniższy rysunek przedstawia graf skierowany składający się z 5 wierzchołków i 7 krawędzi:



Ten graf jest wyznaczony przez zbiór wierzchołków **V={1, 2, 3, 4, 5}** oraz następujący zbiór krawędzi: **E={ (1, 5), (5, 1), (1, 4), (1, 2), (2, 4), (4,3), (3, 2) }**. Zwróćmy uwagę, że krawędź jest wyznaczona przez parę wierzchołków, a zbiór krawędzi to zbiór par wierzchołków. Formalnie E jest podzbiorem produktu kartezjańskiego V x V.

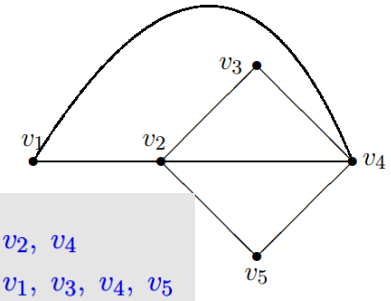
2. **Macierzowa reprezentacja grafu.**

Reprezentacja graficzna jest intuicyjna i czytelna dla człowieka (zwłaszcza dla grafów o niewielkich rozmiarach), ale nie nadaje się do automatycznego przetwarzania przez programy komputerowe. Jedną z reprezentacji grafów ułatwiającą przetwarzanie grafów przez algorytmy jest macierz sąsiedztwa.

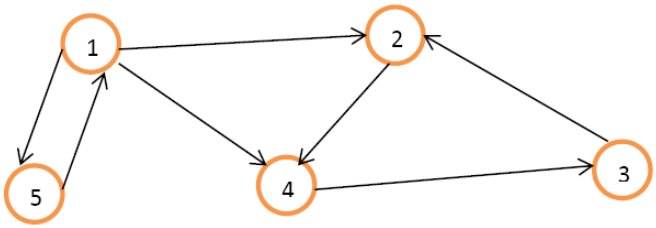
Macierz sąsiedztwa to macierz kwadratowa o rozmiarze równym liczbie wierzchołków grafu. Wiersze i kolumny macierzy są etykietowane kolejnymi wierzchołkami grafu. Element $\mathbf{a_{ij}}$ macierzy zawiera informację o krawędzi (lub jej braku) pomiędzy wierzchołkiem **i** a wierzchołkiem **j**. W najprostszym przypadku (dla grafów nieetykietowanych) możemy przyjąć, że jeśli wierzchołki **i** oraz **j** są połączone krawędzią, to $\mathbf{a_{ij}} = 1$. Jeśli taka krawędź nie istnieje, to $\mathbf{a_{ij}} = 0$. Poniżej macierz sąsiedztwa dla grafu przedstawionego na rysunku:

	1	2	3	4	5
1	0	1	0	1	1
2	0	0	0	1	0
3	0	1	0	0	0
4	0	0	1	0	0
5	1	0	0	0	0

3. **Listowa reprezentacja grafów.**
   Dla każdego wierzchołka v budujemy listę, której elementami są wierzchołki połączone krawędzią z v.
   Listy dla grafów z poniższych rysunków wyglądają następująco:



v_1 : v_2, v_4
v_2 : v_1, v_3, v_4, v_5
v_3 : v_2, v_4
v_4 : v_1, v_2, v_3, v_5
v_5 : v_2, v_4



1 : 2, 4, 5
2: 4
3: 2
4: 3
5: 1

4. **Klasy potrzebne do rozwiązania zadań.**

Dana jest klasa abstrakcyjna AGraph, którą należy pobrać bezpośrednio ze strony WWW.

Zadania.

1. Utworzyć klasę **TGraph** dziedziczącą z klasy AGraph wykorzystującą **macierz** sąsiedztwa do reprezentacji grafu. Zaimplementować metody abstrakcyjne. **Uwaga:** konstruktor klasy TGraph jako pierwszą instrukcję musi wołać konstruktor klasy bazowej za pomocą **super(i)**. **[3p]**

2. Utworzyć klasę **LGraph** dziedziczącą z klasy AGraph wykorzystującą **listę list sąsiedztwa** do reprezentacji grafu. Zaimplementować metody abstrakcyjne.
   Wskazówka: Jako listy użyj klasy java.util.**ArrayList** lub Java.util.**LinkedList**.
   Np. zadeklaruj w klasie LGraph pole ***List<List<Integer>> lista*** **[3p]**

3. W obydwu klasach (**TGraph** oraz **LGraph**) zaimplementować metodę, która
   a. wypisze wierzchołki grafu nie mające żadnych sąsiadów. **[1p]**
   b. sprawdzi, czy graf jest nieskierowany **[1p]**
   c. sprawdzi czy graf jest kliką. **[1p]**
   d. przeprowadzi transpozycję grafu. **[1p]**

Uwaga: Implementacje mogą być jednakowe dla obu klas jeśli wykorzystują wyłącznie metody zdefiniowane w klasie AGraph. W takim wypadku sensownie byłoby zaimplementować je w klasie AGraph. Wyjątkiem jest tu transpozycja – należy utworzyć nowy graf, więc musi to być TGraph lub LGraph, bo nie można utworzyć instancji AGraph.
Jednak poszczególne operacje mogą być bardziej efektywne, gdy wykonamy je oddzielnie dla każdej klasy wykorzystując cechy charakterystyczne dla danej reprezentacji grafu.

Praca domowa:

1. Zaimplementować wypisywanie tablicy incydencji grafu.
2. Zaimplementować metodę *equals* sprawdzającą, czy dwa grafy są identyczne.

Na następnych zajęciach:
Grafy. Algorytm BFS.