

САА – Упражнение 13

Графи

1. Основни понятия

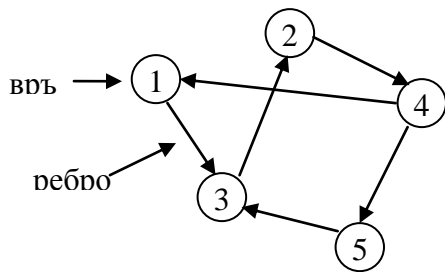
Графът $G = (V, E)$ е структура от данни, която се дефинира чрез две множества:

- множество на **върховете** (възлите) в графа $V = \{v_1, v_2, \dots, v_n\}$ и
- множество на неговите **ребра** (клони) $E = \{e_1, e_2, \dots, e_m\}$, където всеки елемент $e_k \in E$ ($k = 1, \dots, m$) е двойка (v_i, v_j) , $v_i, v_j \in V$.

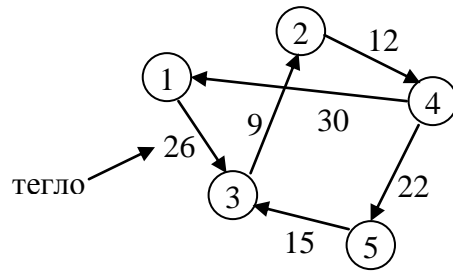
Графът може да бъде ориентиран или неориентиран. Ако множеството E за един граф се състои от насочени ребра, то този граф е **ориентиран**. В противен случай, графът се нарича **неориентиран**.

Ако е дадена числова функция f , която на всяко ребро e_k от графа съпоставя тегло $f(e_k)$, то графът се нарича **тегловен (претеглен)**.

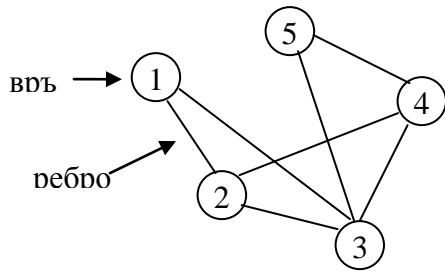
Ако два върха в графа са свързани с ребро, то тези върхове се наричат **съседни**.



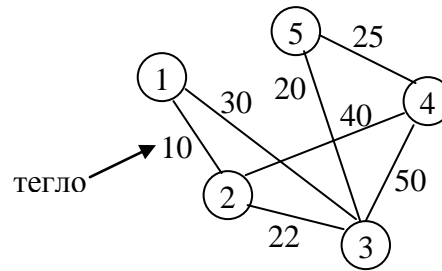
А) Ориентиран граф



Б) Ориентиран тегловен граф



В) Неориентиран граф



Г) Неориентиран тегловен граф

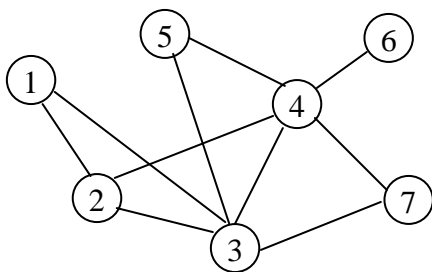
Всяка съвкупност от обекти с дефинирани връзки между тях може да бъде представена като граф. Ето няколко примера за подобни абстракции:

- 1) Няколко града могат да бъдат представени като върхове на граф, а преките пътища между тях — като ребра. Теглата на ребрата ще бъдат дължините на преките пътища.
- 2) Компютърна мрежа може да бъде представена чрез граф, като компютрите са върховете на графа, а всяко ребро между два върха показва, че съответните компютри са пряко свързани в мрежата:

- 3) Няколко химични съединения могат да бъдат представени като върхове на граф. Всяко ребро от графа ще показва дали съответните химични съединения могат да си взаимодействат. Аналогичен пример е върховете да представят видове декоративни риби, а ребрата да показват дали два вида риби могат да съжителстват заедно, или — не.
- 4) Процесите в изработването на едно изделие могат да се представят с върховете на граф, а с ребрата — кой процес след кой трябва да следва в изработката.

2. Обхождане на граф в дълбочина

Идеята на алгоритъма за обхождане в дълбочина е следната: като започнем от стартовия връх, на всяка стъпка преминаваме към първия необходим съсед на текущия връх, като се спускаме се все 'по-дълбоко' в графа. Спускането продължава, докато не се стигне до връх, който няма необходими съседи. В този случай се прави стъпка назад към най-скоро обходения връх, който все още има необходими съседи. Този процес на спускане и връщане продължава, докато се обходят всички върхове в графа.



Стартов връх 1:

1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7

стъпка назад към връх 4

3. Задачи:

1. Съставете програма, която реализира алгоритъма за обхождане на граф в дълбочина. Графът да бъде зададен чрез матрица на съседство.
2. Съставете програма, която проверява дали даден граф съдържа цикли. За целта използвайте алгоритъма за обхождане в дълбочина, реализиран в зад. 1.

Алгоритъмът за проверка на цикличност се състои в следното: Извършва се обхождане в дълбочина. На всяка стъпка, ако върха i , който се разглежда в момента има съсед, който вече е обходен, различен от предшественика на i , значи графът съдържа цикъл.

Този алгоритъм може да бъде реализиран чрез модификация на функцията за обхождане в дълбочина като се добави още един параметър, който ще показва предшественика на върха, който е текущ). Ако текущият връх i има за съсед връх, който вече е бил обходен, различен от предшественика, значи се е затворил цикъл.