



## *Деревья*

Лекции 13-14

Н.В. Белоус

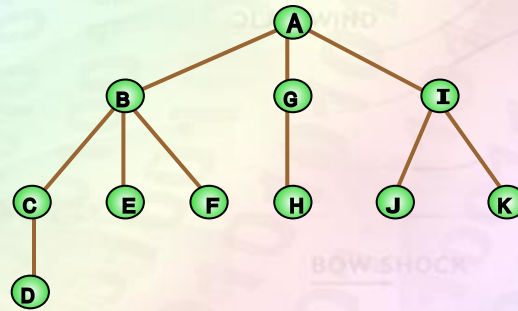
Факультет компьютерных наук

Кафедра ПО ЭВМ, ХНУРЭ

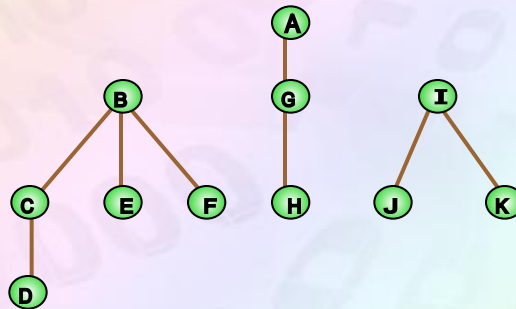


# Дерево, лес

**Деревом** называется связный неориентированный граф без циклов (ациклический), который содержит более двух вершин.



Ациклический граф, который содержит несколько компонент связности (состоит из нескольких деревьев), называют **лесом**.



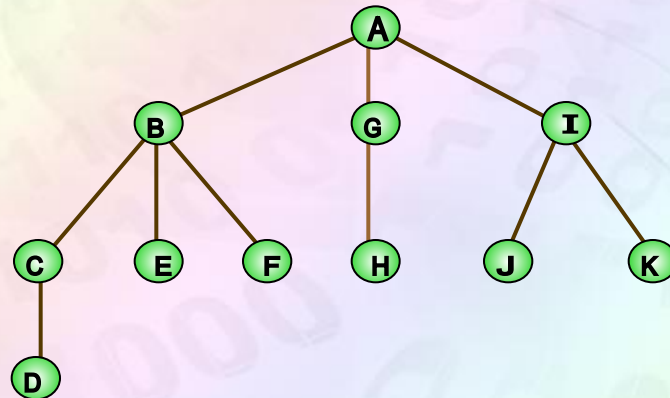


# Неориентированное дерево

! **Корнем** неориентированного дерева называют вершину с минимальным порядковым номером.

! Если в дереве  $T$  корень задан (то есть оно изображено сверху вниз, и в нем есть самая верхняя вершина), то дерево  $T$  называют **корневым**.

! В неориентированном дереве, как и в произвольном графе, пара вершин соединяется **ребром**.

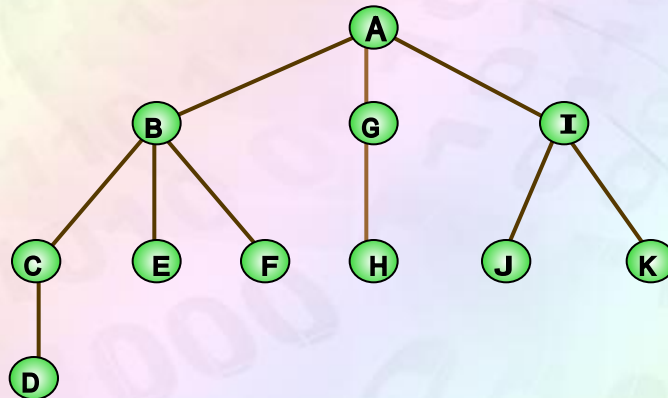




# Неориентированное дерево

**Листьями** неориентированного дерева (висячими вершинами) называются вершины, которым инцидентно лишь одно ребро.

**Узлом (внутренней вершиной)** произвольного дерева называют вершину, которая не является корнем, и не является листом.





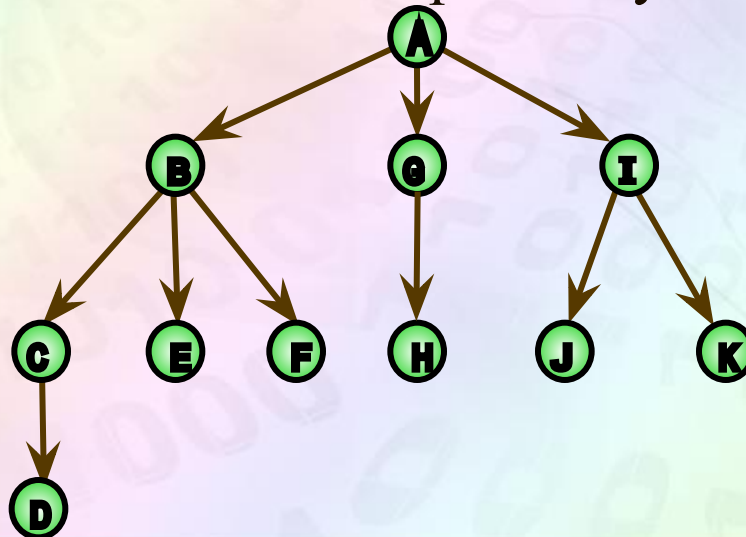
# Ориентированное дерево

**Маршрут** в любом дереве называют ветвью

В ориентированном дереве, как и в графе, ребра называют **дугами**.

**Корнем ориентированного дерева** называют вершину, полустепень захода которой равна нулю.

**Листьями ориентированного дерева** (висячими вершинами) называются вершины, полустепень захода которых равна единице, полустепень исхода равна нулю.





Для графа  $G(v,e)$  эквивалентны следующие утверждения:

- 1)  $G$  – дерево;
- 2)  $G$  – связный граф,  $|e|=|v|-1$ , где  $e$  – количество ребер,  $v$  – количество вершин графа  $G$ .
- 3)  $G$  – ациклический граф,  $|e|=|v|-1$ ;
- 4)  $G$  – граф, в котором две вершины соединяются единственной цепью;
- 5)  $G$  – ациклический граф, и добавление нового ребра приводит к появлению точно одного цикла.



При описании соотношений между узлами дерева используется терминология, принятая в генеалогических деревьях.

В дереве (или поддереве) все узлы являются *потомками* его корня, и наоборот, корень есть *предок* всех своих потомков.

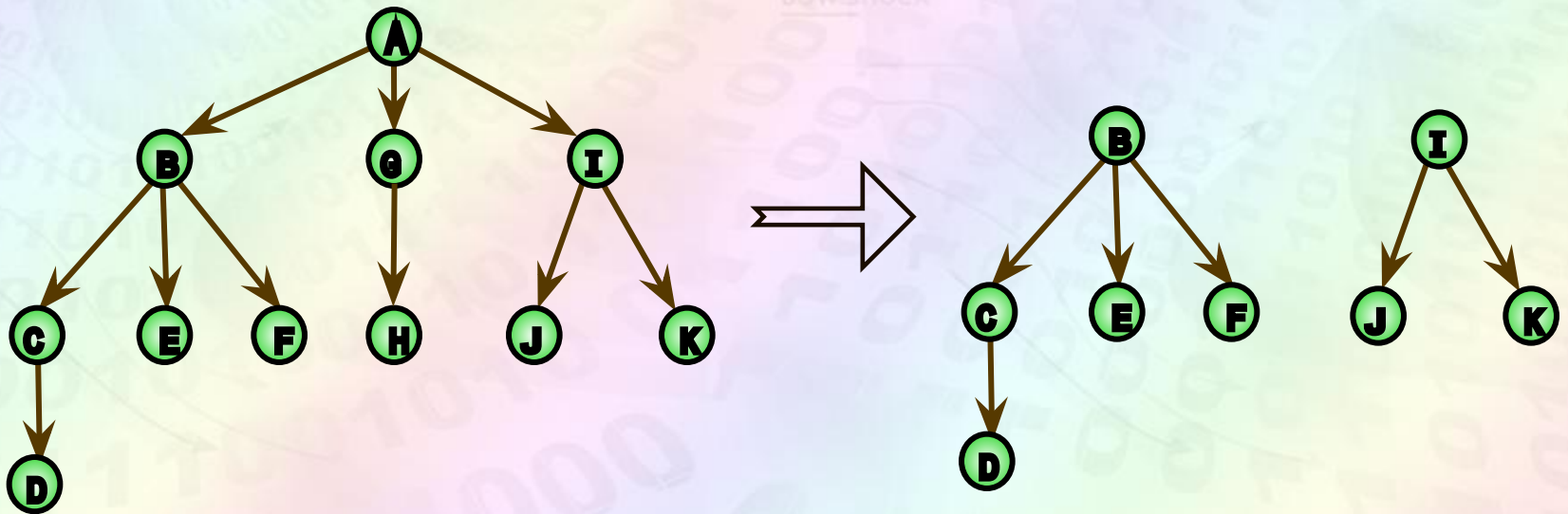
Если существует дуга, входящая из вершины А в вершину В, то А называется *отцом* вершины В, а В – *сын* А.

*Корень* является отцом корней его поддеревьев, которые в свою очередь являются сыновьями корня. Вершины, являющиеся сыновьями одного отца называются *братьями*.



# Поддереве

*Поддеревом* дерева  $T$  называют дерево  $T_1$ , которое содержит часть вершин дерева  $T$ , причем корнем дерева  $T_1$  может быть любая другая вершина дерева  $T$ .







## *n*-арное дерево

Корневое дерево называется *n*-арным, если каждая внутренняя вершина имеет не более  $n$  детей.

**Порядком** дерева называется максимальное количество потомков вершин данного дерева.

Корневое дерево называется *полным n-арным деревом*, если каждая внутренняя вершина имеет ровно  $n$  детей.

Корневое дерево называется *бинарным деревом*, если каждая внутренняя вершина имеет не более двух детей ( $n = 2$ ).

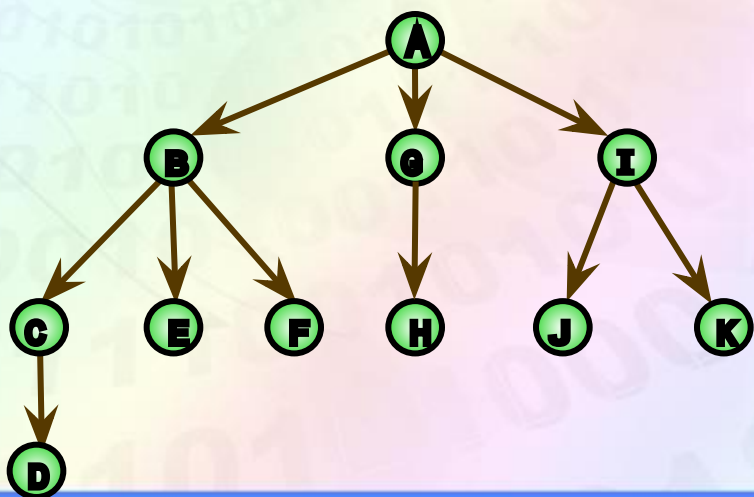


# Высота и количество уровней дерева

**Уровнем** вершины  $v_i$  в корневом дереве называется длина пути от корня дерева до данной вершины. Корень имеет уровень, равный нулю.

**Высотой**  $h$  корневого дерева  $T$  называется величина, равная максимальному из уровней вершин.

**Глубина (количество уровней)**  $d$  дерева  $T$  равняется количеству вершин, которые лежат на максимальной ветви от корня к листьям.



$$h_B = 1$$

$$h_C = 2$$

$$h_D = 3$$

$$h_E = 2$$

$$h_F = 2$$

$$h_G = 1$$

$$h_H = 2$$

$$h_I = 1$$

$$h_J = 2$$

$$h_K = 2$$

$$h = 3$$

$$d = 4$$



# Задача Кели. Пример

Пусть для некоторого множества  $M$  городов известна стоимость  $c(a, b)$  постройки дороги между любыми двумя городами  $a, b \in M$ . Какова должна быть сеть дорог между городами, входящими в  $M$ , чтобы по ней можно было проехать из любого города  $a \in M$  в любой город  $b \in M$  и чтобы стоимость этой сети была минимальной?

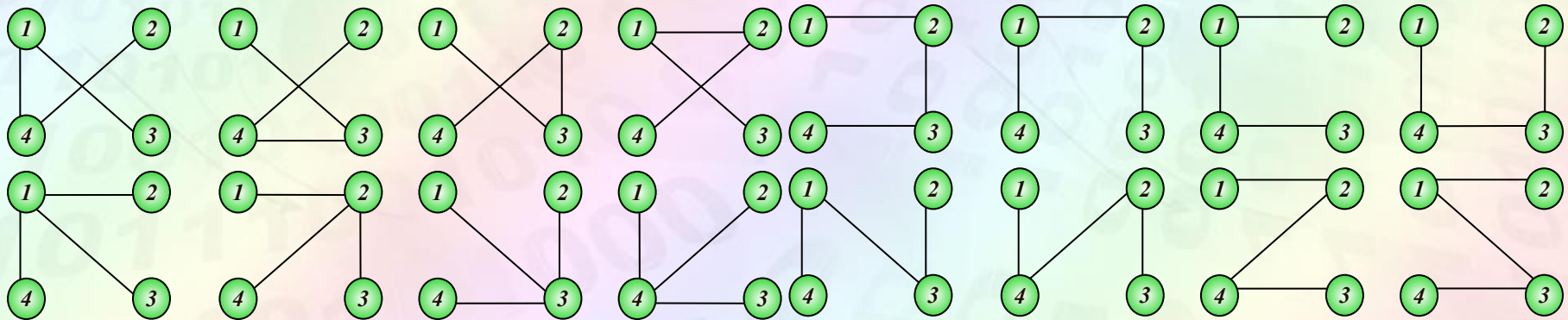
## Теорема

На  $n$  вершинах можно построить  $n^{n-2}$  деревьев.

## Пример

$$n=4$$

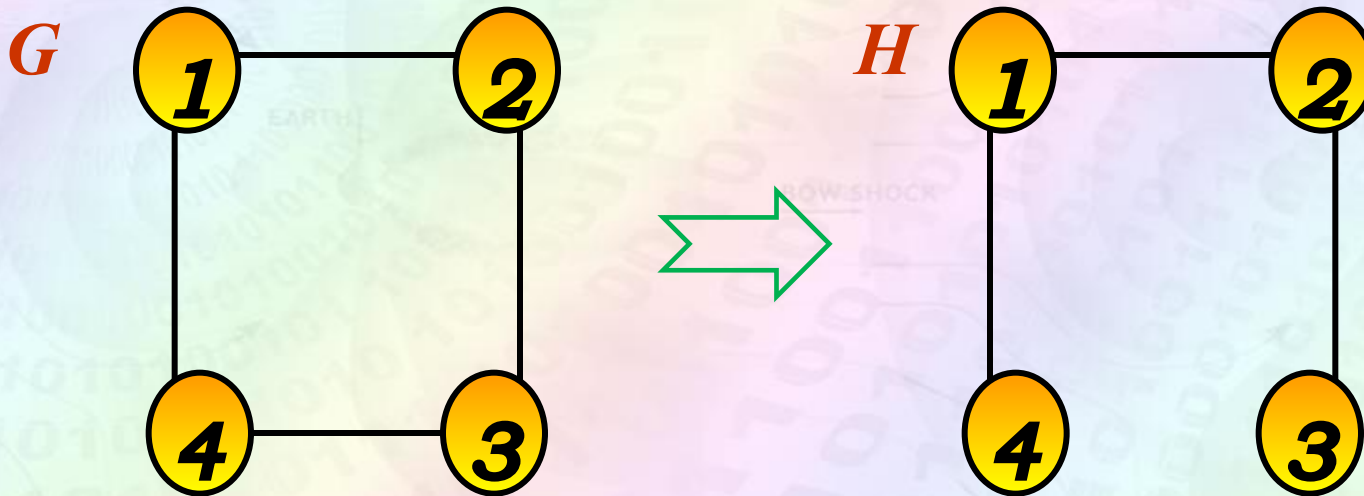
$$n^{n-2} = 4^2 = 16.$$





# Остовные деревья

! **Остовное дерево** – дерево, которое содержит все вершины исходного графа.



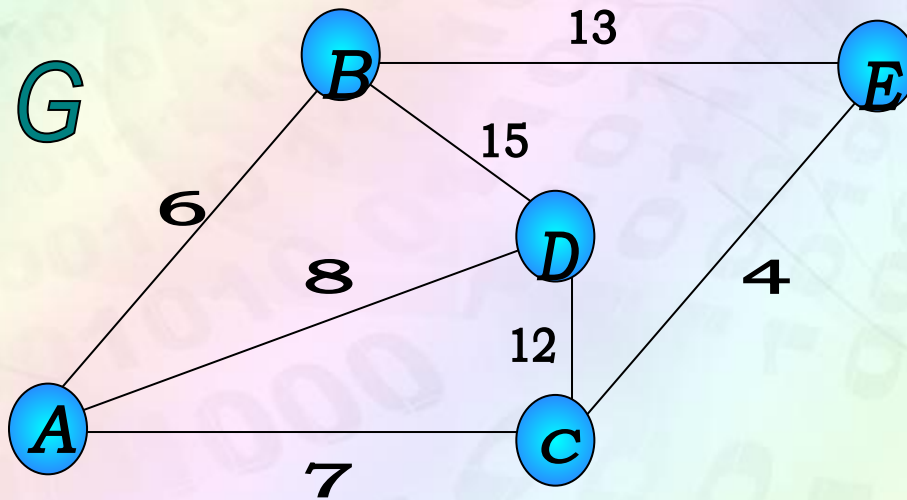
! **Хорда остовного дерева  $H$**  графа  $G$  – это ребро графа  $G$ , не принадлежащее остовному дереву  $H$ .



## Дерево минимальной стоимости (веса)

Граф  $G$  с весами на дугах называется **взвешенным графом**.

**Весом подграфа** из  $G$  называется сумма весов ребер подграфа.



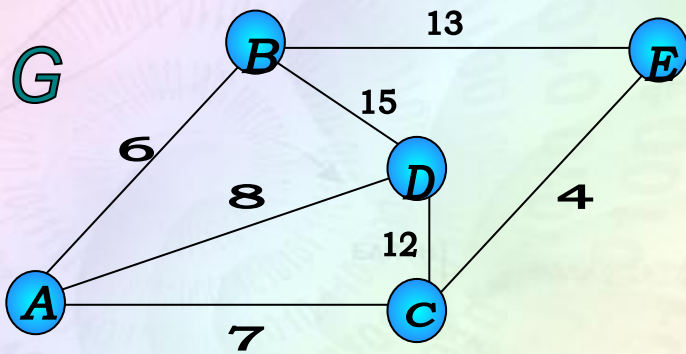


1. Упорядочиваем ребра в порядке возрастания их весов.
2. Включаем в остовное дерево, ребра в порядке их возрастания.
3. Если вновь включенное ребро образует цикл с ребрами, включенными до него, то пропускаем его.
4. Дерево построено тогда, когда в него включено  $n-1$  ребро.



# Алгоритм Борувки. Пример

Построить дерево минимальной стоимости для заданного графа  $G$ .



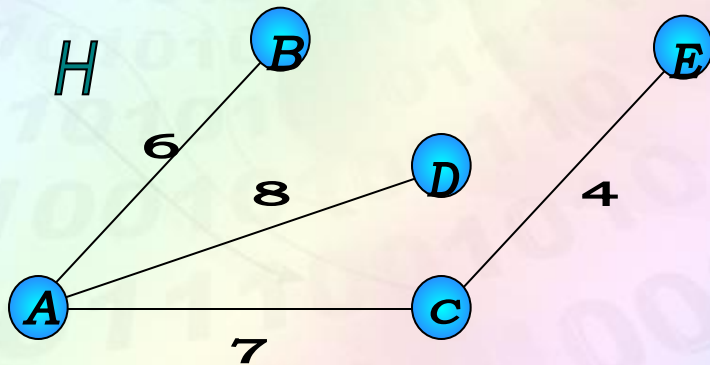
Находим ребра с наименьшей мерой. Поочередно включаем их в остовное дерево.

$$CE = 4$$

$$AB = 6$$

$$AC = 7$$

$$AD = 8$$



$$S_H = 6 + 8 + 7 + 4 = 25.$$

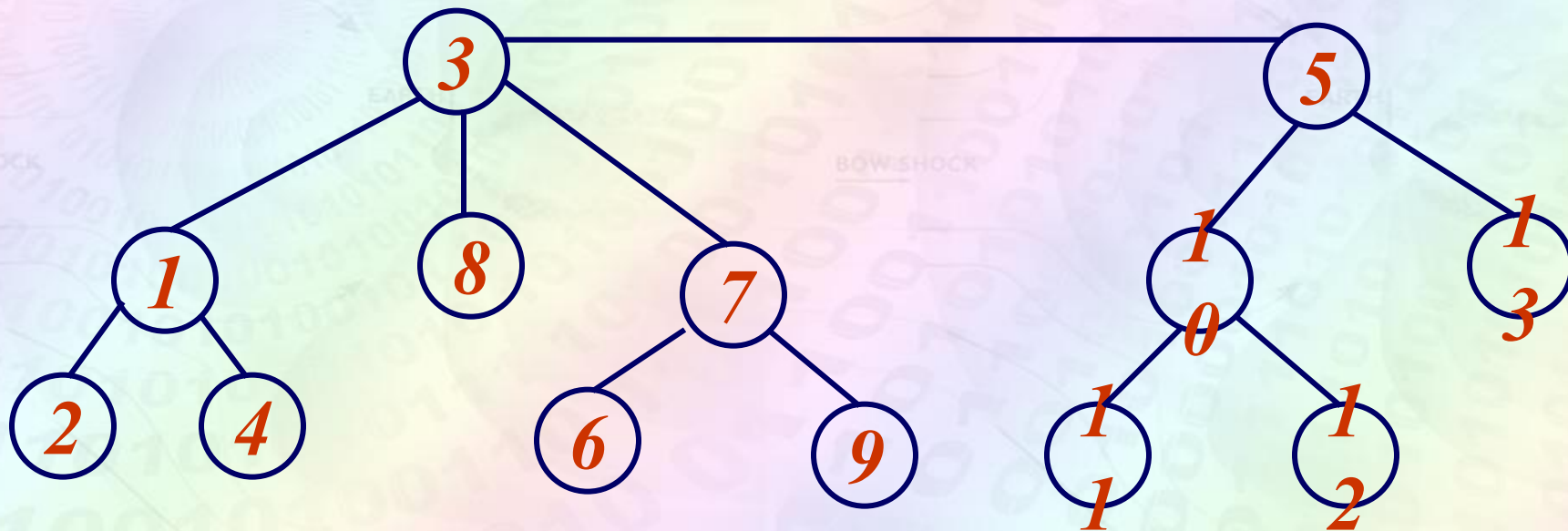
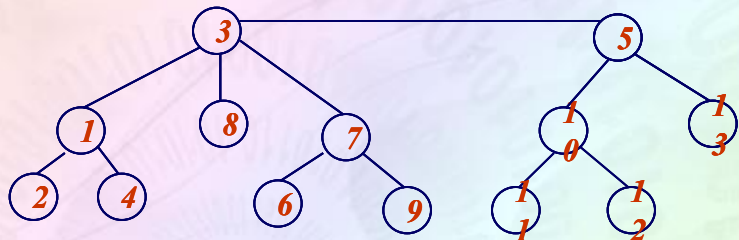


- 1) Вводится последовательность  $N_p = (1, 2, \dots, p)$ , где  $p$  – количество вершин графа.
- 2) Выбирается висячая вершина с наименьшим номером, эта вершина удаляется из последовательности  $N_p = (1, 2, \dots, p)$ , а номер связанной с ней вершины записывается.
- 3) Затем этот процесс повторяется до тех пор, пока не получим последовательность  $a(T) = (a_1, a_2, \dots, a_{p-2})$ .





# Пример кодирования деревьев



1 1 3 7 3 7 3 5 10 10 5



- 1) Находим общее количество вершин (количество вершин в коде дерева + 2);
- 2) Находим висячие вершины (т.е. которые не входят в код);
- 3) Находим висячую вершину с минимальным номером и соединяем ее с первой неиспользованной вершиной в коде. Если выбранная вершина не встречается далее в последовательности кода, записываем ее вершину в последовательности листьев;
- 4) Повторяем пункт 3), до использования всех листьев;
- 5) Последнюю вершину кода соединяем с листом с



## Пример декодирования деревьев

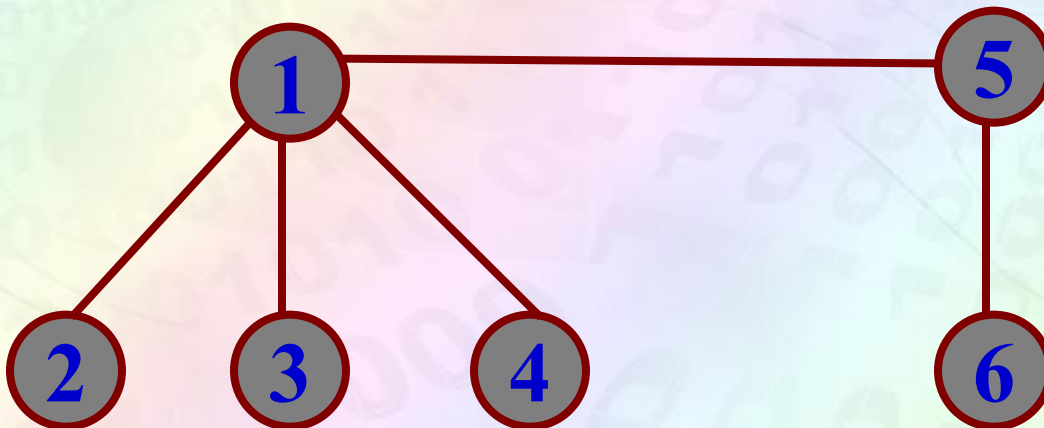
Дан код: 1, 1, 1, 5. Необходимо построить дерево.

*Решение.*

1. Общее количество вершин:  $4+2=6$ .
2. Находим висячие вершины: **2, 3, 4, 6**

Код дерева:                    1 1 1 5

Висячие вершины: 2 3 ~~5~~ 6



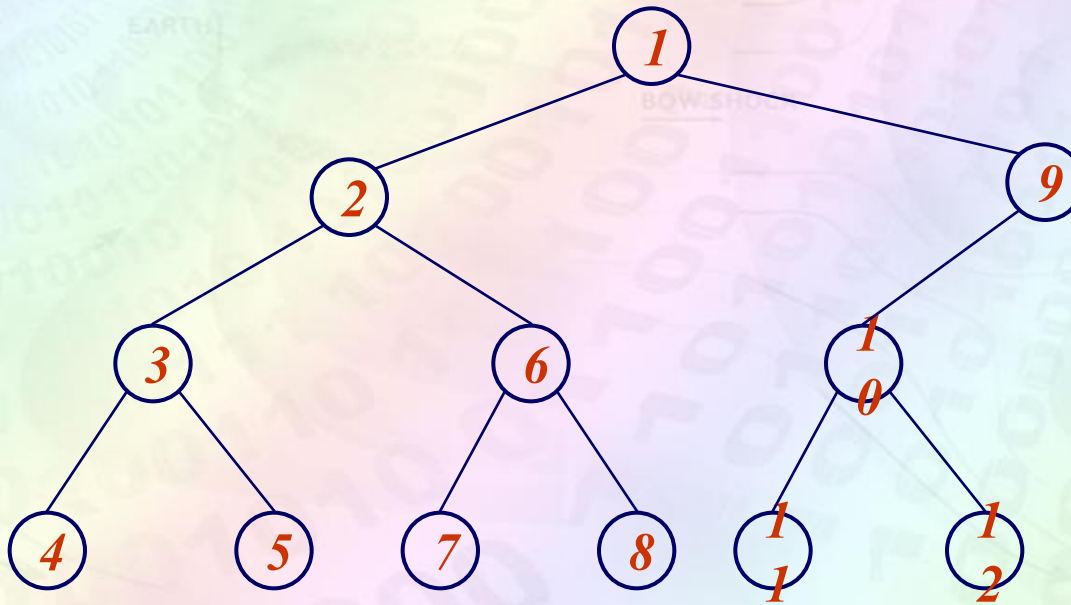


# ***Бинарные деревья***



# Бинарные деревья

Бинарным (двоичным) деревом  $T$  называется упорядоченное дерево, из каждой вершины которого может исходить не более двух дуг.





# Бинарные деревья

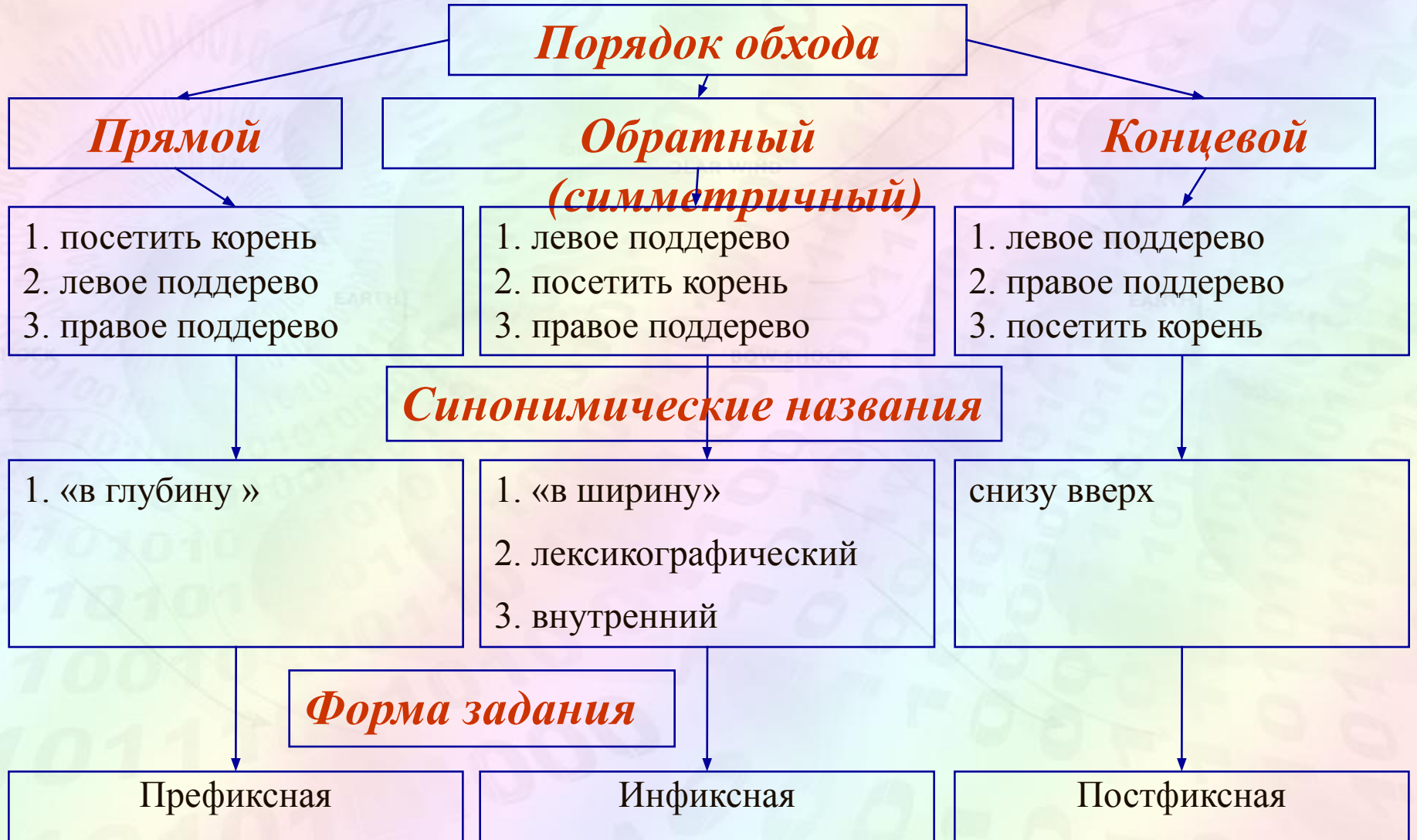
Каждая вершина бинарного дерева может иметь либо двух сыновей – левого и правого, либо иметь только левого сына, либо только правого сына, либо не иметь ни одного сына.

Поддерево, корень которого является левым сыном вершины  $v$ , называется левым поддеревом вершины  $v$ .

Поддерево, корень которого является правым сыном вершины  $v$ , называется правым поддеревом вершины  $v$ .

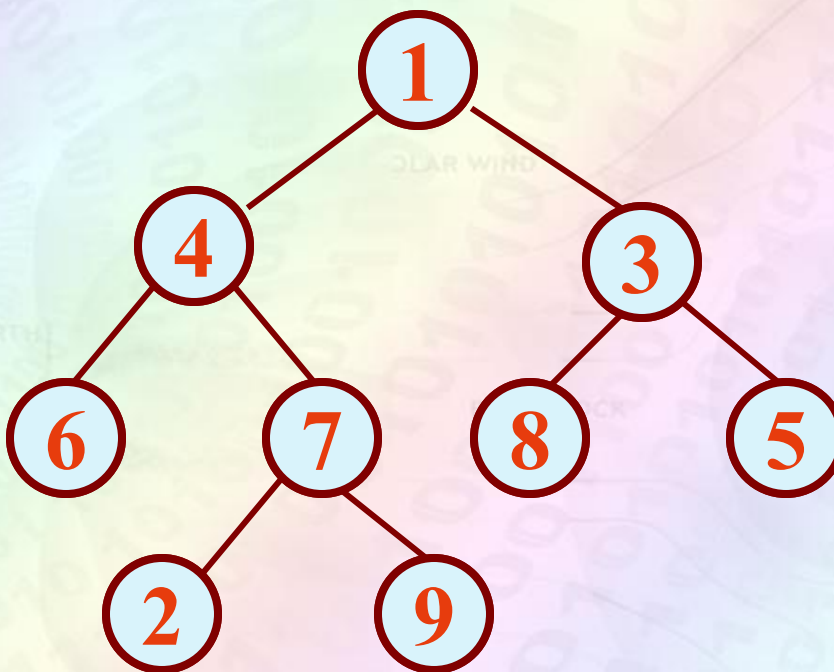


# Правила обхода бинарных деревьев





# Список выходов сервы в порядке порядке

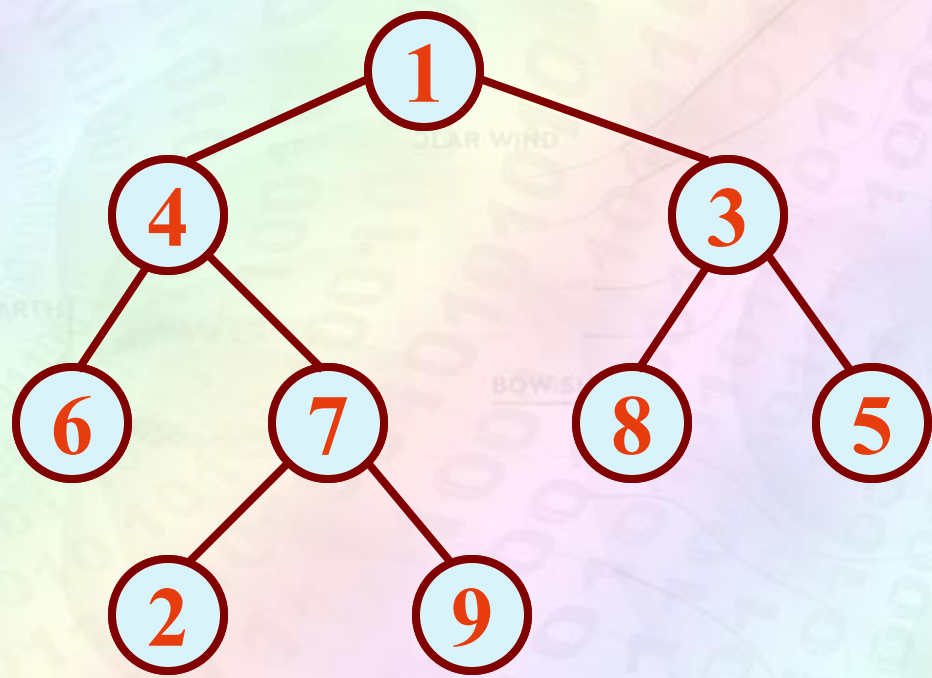


1 4 6 7 2 9 3 8 5





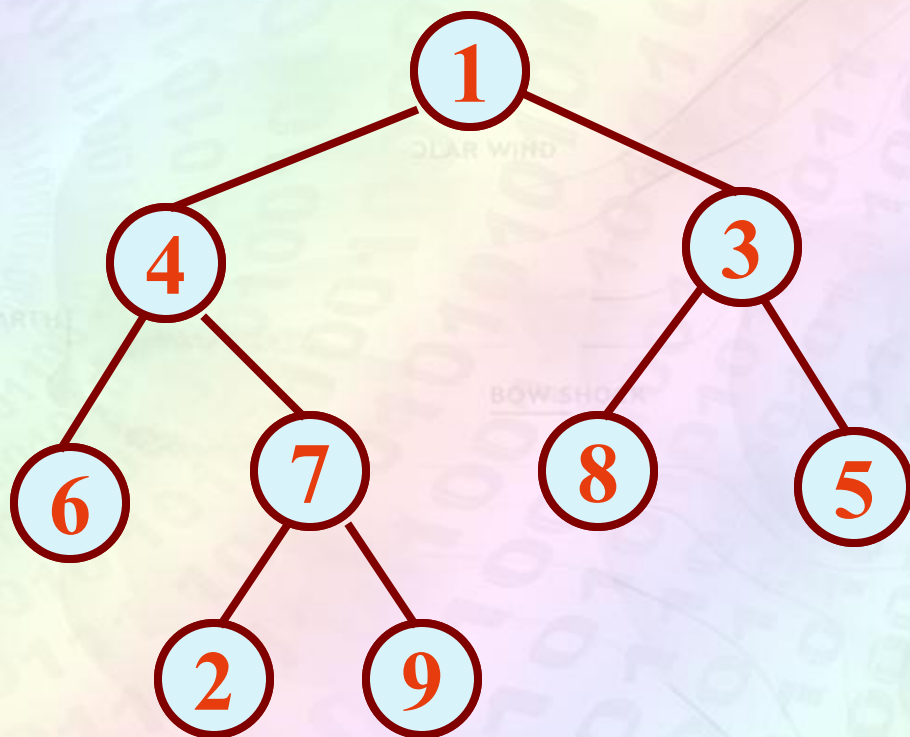
# порядке



6 4 2 7 9 1 8 3 5



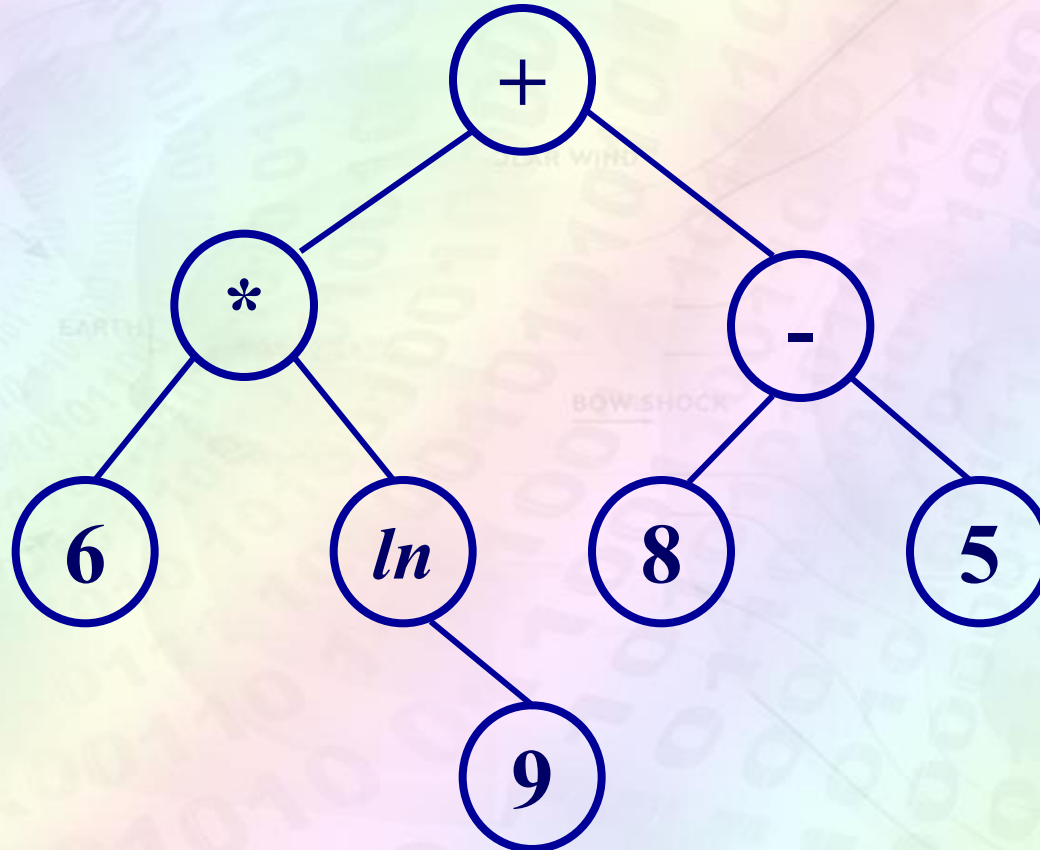
# Обход бинарного дерева в концевом порядке



6 2 9 7 4 8 5 3 1



# Запись математических выражений



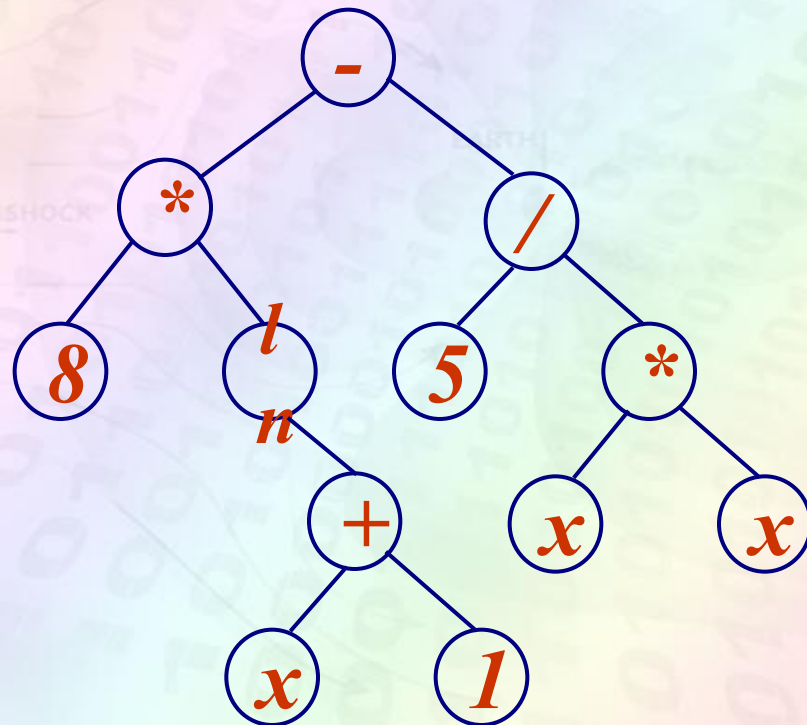
$$(6 * \ln(9)) + (8 - 5)$$



# Запись математических выражений

$$y = 8 * \ln (X + 1) - 5 / X^2$$

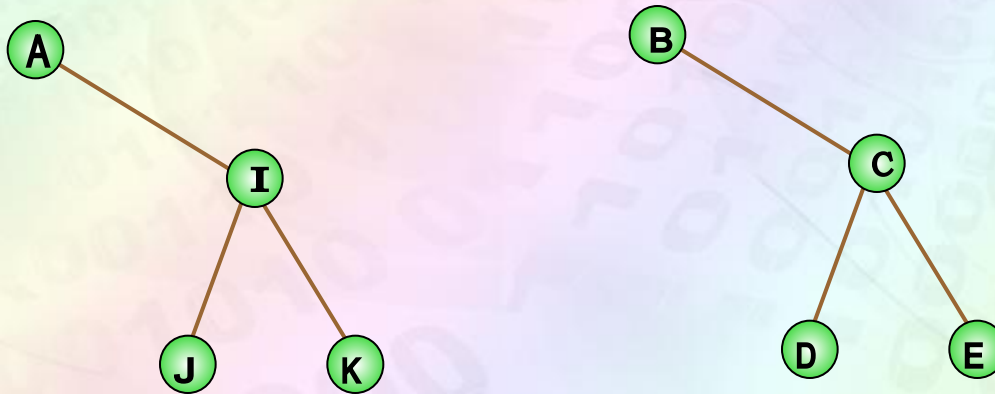
1. Построим концевые вершины, соответствующие переменным и константам заданного выражения.
2. Определим приоритет операций в выражении.
3. В соответствии с приоритетом операций добавляем в дерево вершины, соответствующие операциям и соединяем их с концевыми вершинами.





# Подобные бинарные деревья

Два бинарных дерева называются *подобными*, если они имеют одинаковую структуру, то есть либо они пусты, либо содержат одинаковое число поддеревьев и их левое и правое поддерева подобны.





# Эквивалентные бинарные деревья

Бинарные деревья *эквивалентны*, если они подобны и соответствующие узлы содержат одну и ту же информацию.

