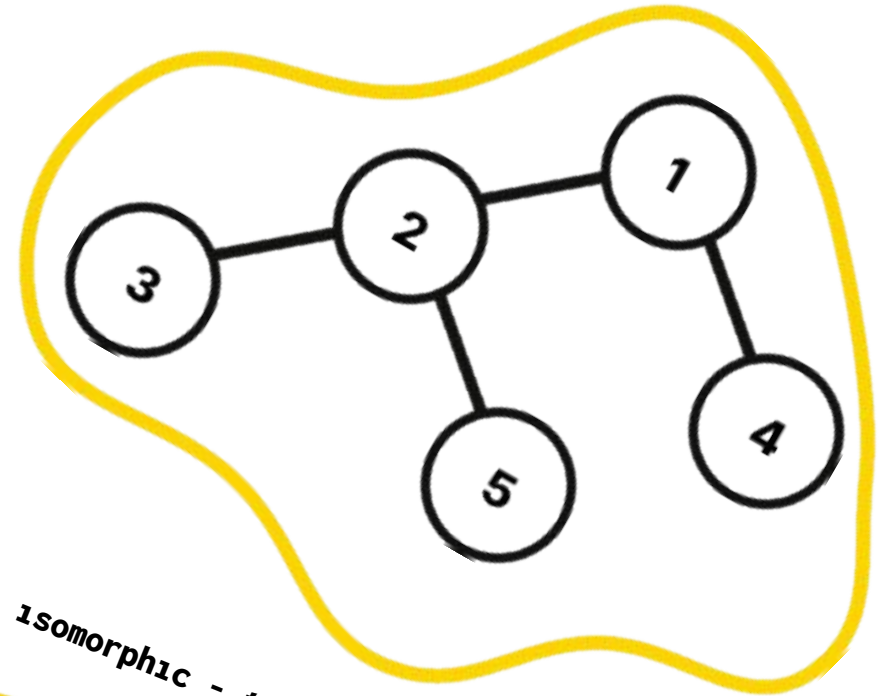
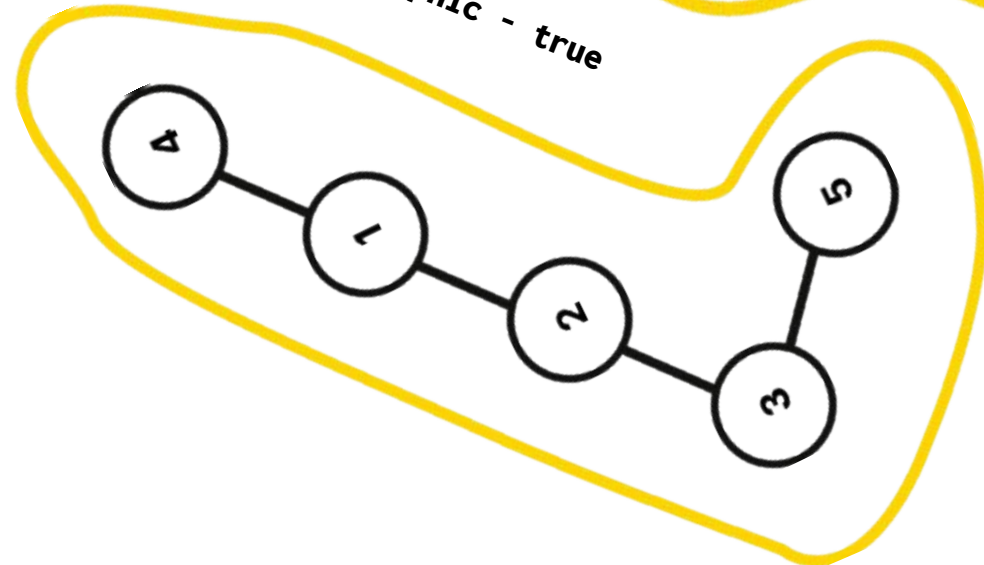



ИЗОМОРФИЗМ ДЕРЕВЬЕВ



Trees are isomorphic - true



ФОРМАЛЬНАЯ ПОСТАНОВКА ЗАДАЧИ

1. изучить алгоритм **определения изоморфности деревьев**
2. реализовать этот алгоритм на выбранном языке программирования
3. выполнить тестирование с помощью unit – тестов
4. результаты работы выложить на платформу  **GitHub**

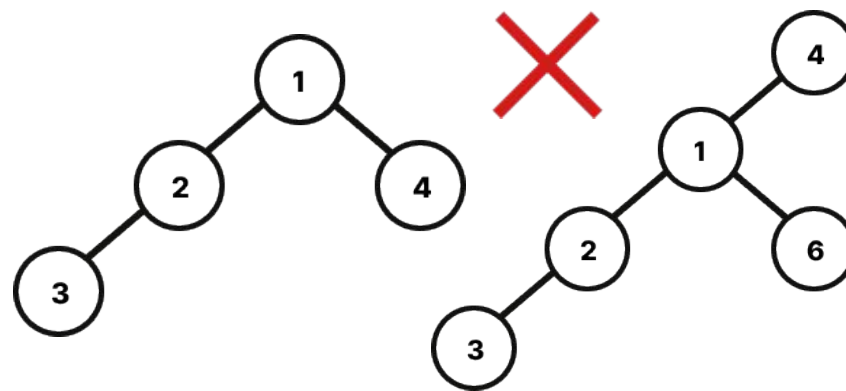
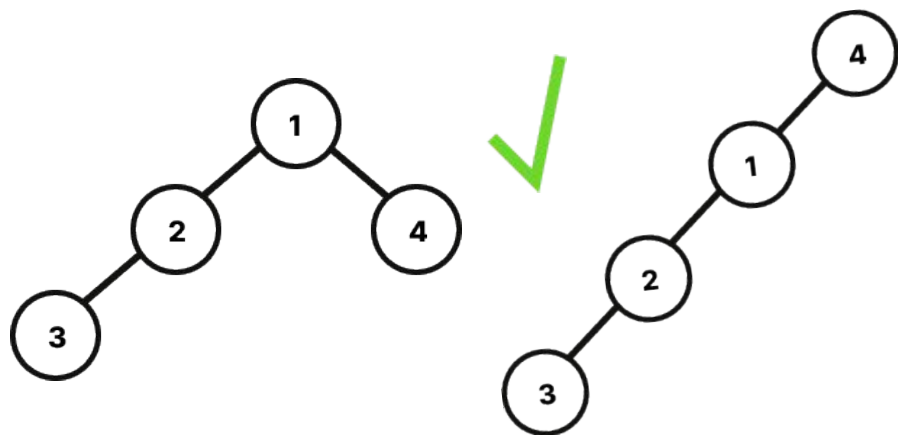


ИЗОМОРФИЗМ И ДЕРЕВЬЯ

ИЗОМОРФИЗМ – логико-математическое понятие, выражающее одинаковость структуры

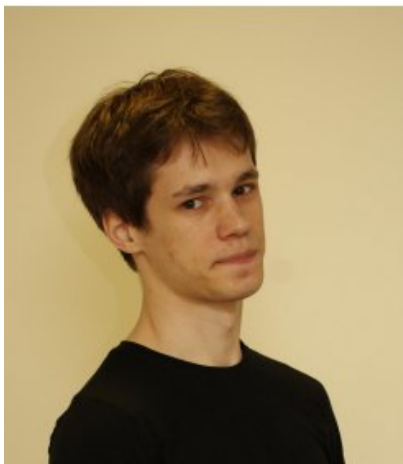
ДЕРЕВО – структура данных, связный граф без циклов, с неориентированными и невзвешенными ребрами

ИЗОМОРФНЫЕ ДЕРЕВЬЯ – деревья с одинаковой структурой



ИСТОРИЯ

1. **Первое** «открытие» теории графов
2. **Второе** «открытие» теории графов
3. **Третье** «открытие» теории графов
4. **Четвертое** «открытие» теории графов
5. **Пятое** «открытие» теории графов



**Копелиович Сергей
Владимирович**

описал алгоритм
определения
изоморфности
деревьев

1. **1736** – Леонард Эйлер применил идеи теории графов.
2. **1847** – Густав Кирхгоф разработал теорию деревьев
3. **1857** – Артур Кэли открыл важный класс графов – деревья.
4. **1859** – Уильям Гамильтон придумал игру «Вокруг света».
5. **1869** – Камиль Жордан исследовал деревья в рамках математики. XX – Денеш Кенинг ввел термины.



ИЗОМОРФИЗМ И ДЕРЕВЬЯ

- Пусть дерево задается матрицей смежности. Два дерева с матрицами смежности и называются изоморфными, если существует такая перестановка, $\exists p$, что $\forall i, j : c_1[pi, pj] = c_2[i, j]$.
- Хеш функция - $31^n + \sum_{i=1}^n 31^{n-i} * a_i$
- Алгоритм определения изоморфности деревьев требует корректности входных данных.
- Количество узлов дерева не может превышать **2^{32}**

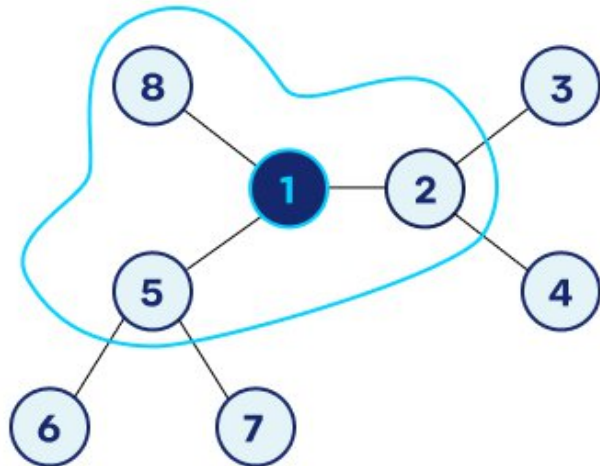
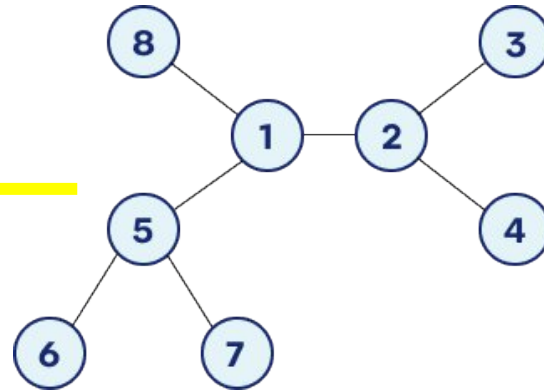


СТРУКТУРА

данные вида **число:** – индексы узлов.

данные вида **число: число*1* число*2* ... число*n*** – индексы узлов, на которые он ссылается.

1:	2 5 8
2:	1 3 4
3:	2
4:	2
5:	1 6 7
6:	5
7:	5
8:	1



nodes – список (вектор) всех узлов в дереве.

numbers – ассоциативный массив (Map) хешей сортированных списков (List) и их номеров.

nodes

1	2	3	4	5	6	7	8
0	1	2	3	4	5	6	7

numbers

{ []=1, [1, 1]=2, [1, 2]=3, [1, 1, 3]=4 }

4 – номер хеша узла 2

subNodes – список узлов, с которыми есть связь (ребра).

subNodes

8	5	2
0	1	2



ОПИСАНИЕ АЛГОРИТМА

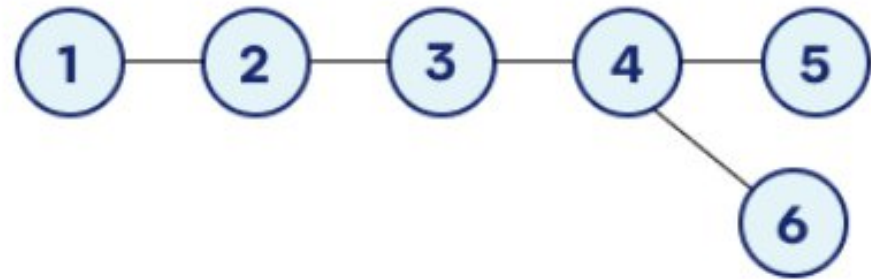
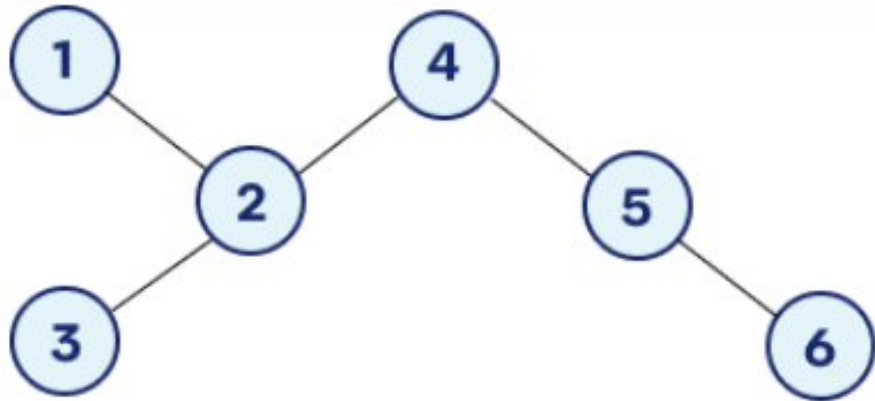
Даны два некорневых дерева. Проверяем, изоморфны ли они.

- Возьмем случайно выбранный узел из первого дерева за корень и определим его хеш рекурсивно, как хеш отсортированного списка (вектора) хешей детей.
- Деревья изоморфны, если хеши корней совпадают, поэтому проходимся по узлам второго дерева, определяя их хеши, и если хоть один совпадает с хешем корня первого дерева, то деревья изоморфны.
- При этом хеш функция должна обладать следующим свойством: хеши различных списков (векторов) различны.



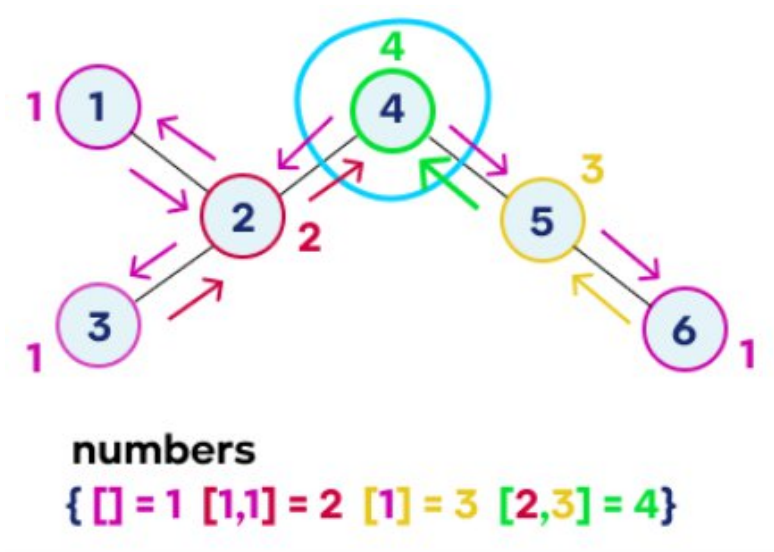
ОПИСАНИЕ АЛГОРИТМА

1. рассмотрим алгоритм на примере таких деревьев:



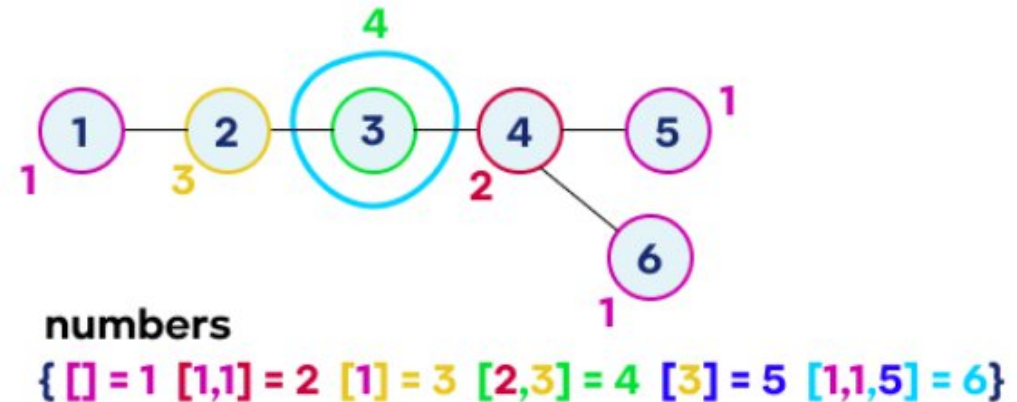
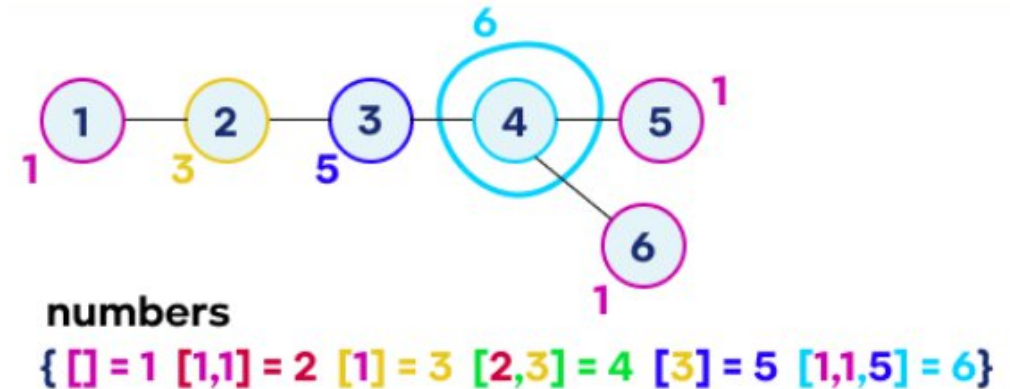
ОПИСАНИЕ АЛГОРИТМА

- выбираем корнем первого дерева узел 4
- рекурсивно переберем потомков и узнаем номера их хешей



если хотя бы один номер хеша потомков совпадает с первым деревом – деревья изоморфны

- аналогично выполняем действия для второго дерева, только за корень определяем каждый узел



ТЕСТИРОВАНИЕ

Sample1_6nodes_structure1

SampleX – номер примера.
Xnodes – кол-во узлов.
structureX – структура дерева.

Тестируется класс Tree:

- успешных пройденных тестов 9
- проваленных тестов 0

- testIsomorphicStruct1() – проверяет деревья с одинаковой структурой
- testIsomorphicStruct2() – проверяет деревья с одинаковой структурой:
- testNonIsomorphicStruct1Struct3() – проверяет деревья с разной структурой
- testNonIsomorphicDifferentQuantityNodes() – проверяет деревья с разным количеством узлов
- testIsomorphicItself() – проверяет один и тот же объект дерева:
- testIsomorphicEmpty() – проверяет пустые деревья
- testIsomorphicEmptyStruct5() – проверяет пустое и непустое деревья
- testNonExistentFile() – пытается считать данные из несуществующего файла
- testInvalidData() – пытается считать некорректные данные

