*Функция* называется **рекурсивной**, если в своем теле она содержит обращение к самой себе с измененным набором параметров. При этом количество обращений конечно, так как в итоге решение сводится к базовому случаю, когда ответ очевиден.

// Картинки показать

Написать с ними факториал. Дать самим написать арифметическую прогрессию

*Пример 1*. В арифметической прогрессии найдите an, если известны а1 = -2.5, d =0.4, не используя формулу n -го члена прогрессии.

*По* определению арифметической прогрессии, an=an-1+d, при этом

an-1=an-2+d, an-2=an-3+d,... a2=a1+d.

Таким образом, нахождение an для номера n сводится к решению аналогичной задачи, но только для номера n -1, что в свою *очередь* сводится к решению для номера n -2, и так далее, пока не будет достигнут номер 1 (*значение* а1 дано *по* условию задачи).

См. папку «Программа 1 арифм. прогрессия»

Для решения задач *рекурсивными методами* разрабатывают следующие этапы, образующие **рекурсивную триаду**:

* *параметризация* – выделяют параметры, которые используются для описания условия задачи, а затем в решении;
* база рекурсии – определяют тривиальный случай, при котором решение очевидно, то есть не требуется обращение функции к себе;
* *декомпозиция* – выражают общий случай через более простые подзадачи с измененными параметрами.

*Целесообразность применения рекурсии* в программировании обусловлена спецификой задач, в постановке которых явно или опосредовано указывается на возможность сведения задачи к подзадачам, аналогичным самой задаче. При этом эффективность рекурсивного или итерационного способов решения одной и той же задачи определяется в ходе анализа работоспособности программы на различных наборах данных. Таким образом, *рекурсия* не является универсальным способом в программировании. Ее следует рассматривать как альтернативный вариант при разработке алгоритмов решения задач.

*Пример 2*. Для целого неотрицательного числа n найдите его *факториал*.

Разработаем рекурсивную триаду.

*Параметризация*: n – неотрицательное *целое число*.

База рекурсии: для n =0 *факториал* равен 1.

*Декомпозиция*: n!=(n-1)!\*n.

См. папку «Программа 2 факториал»

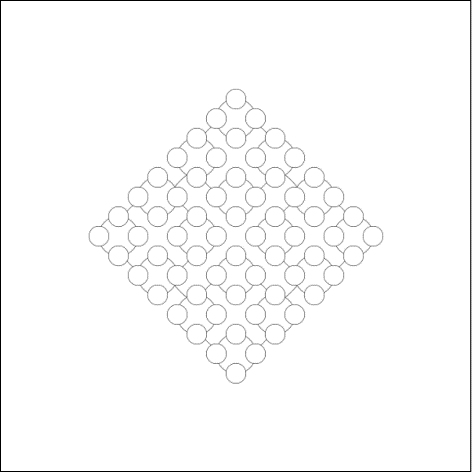
Фрактал - математическое [множество](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE), обладающее свойством [самоподобия](https://ru.wikipedia.org/wiki/%D0%A1%D0%B0%D0%BC%D0%BE%D0%BF%D0%BE%D0%B4%D0%BE%D0%B1%D0%B8%D0%B5) (объект, в точности или приближённо совпадающий с частью себя самого, то есть целое имеет ту же форму, что и одна или более частей).

Фракталы удобно рисовать с помощью… рекурсии.

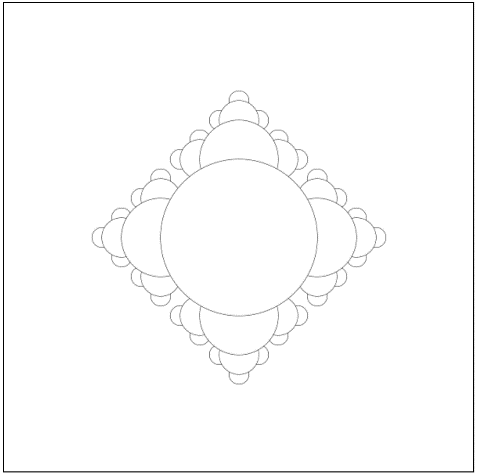
// Картинки показать

Порисовать фракталы, см. папку «Программа 3 фракталы»

В результате получится такой фрактал:



Самостоятельная работа – превратить фрактал в такой:



В программе в комментарии написано, как это сделать.

Дерево Пифагора

Если в классическом дереве Пифагора угол равен 45 градусам, то также можно построить и обобщённое дерево Пифагора при использовании других углов. Такое дерево часто называют *обдуваемое ветром дерево Пифагора*. Если изображать только отрезки, соединяющие каким-либо образом выбранные «центры» треугольников, то получается *обнаженное дерево Пифагора*.

Домашнее задание – нарисовать другие фракталы (квадратные, например), менять цвет в зависимости от глубины фрактала.

Задание со звездочкой: нарисовать дерево Пифагора.