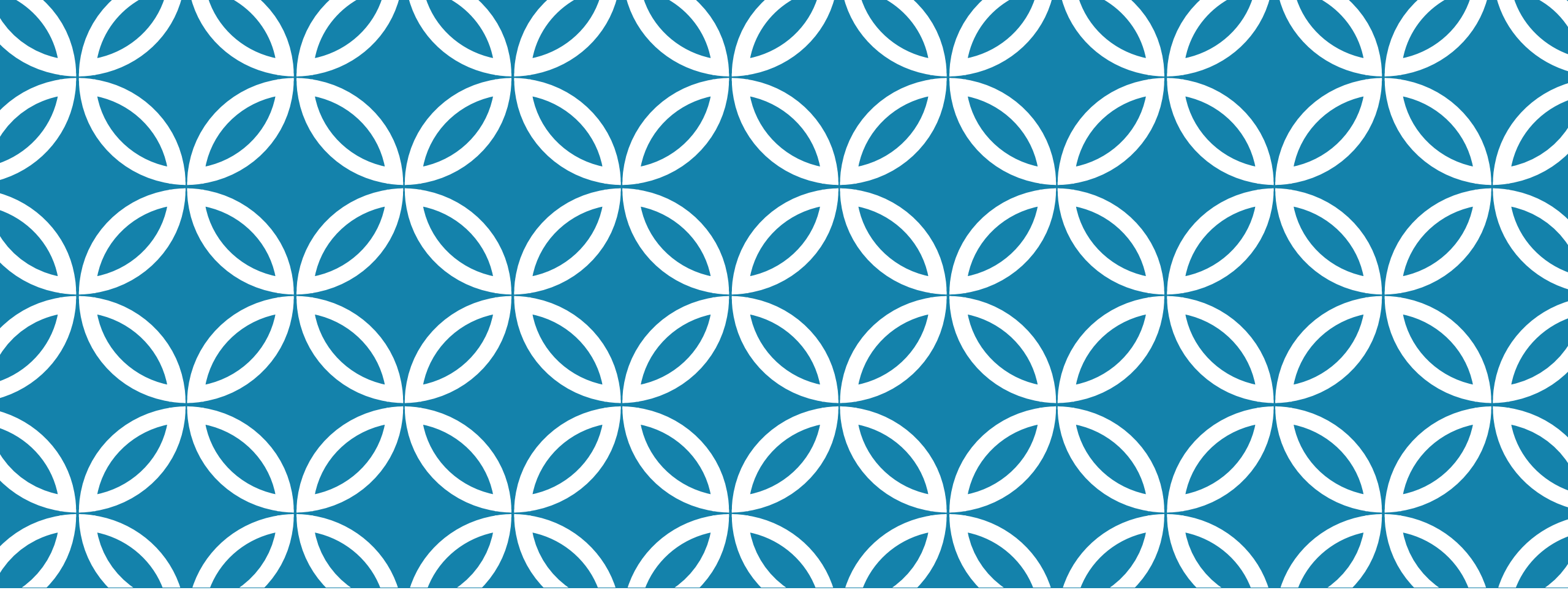

КЛАССЫ В ООП

КЛАСС

Класс — это элемент ПО, описывающий абстрактный тип данных и его частичную или полную реализацию.

Наряду с понятием «объекта» класс является ключевым понятием в ООП (хотя существуют и бесклассовые объектно-ориентированные языки, например, Self). Суть отличия классов от других абстрактных типов данных состоит в том, что при задании типа данных класс определяет одновременно как интерфейс, так и реализацию для всех своих экземпляров, а вызов метода-конструктора обязателен.

ООП

На практике объектно-ориентированное программирование сводится к созданию некоторого количества классов, включая интерфейс и реализацию, и последующему их использованию. Графическое представление некоторого количества классов и связей между ними называется диаграммой классов. Объектно-ориентированный подход за время своего развития накопил множество рекомендаций по созданию классов и иерархий классов.

ИДЕЯ КЛАССОВ

Идея классов пришла из работ по базам знаний, имеющих отношение к исследованиям по искусственному интеллекту. Используемые человеком классификации в зоологии, ботанике, химии, деталях машин, несут в себе основную идею, что любую вещь всегда можно представить частным случаем некоторого более общего понятия. Конкретное яблоко — это в целом некоторое яблоко, вообще яблоко, а любое вообще яблоко — фрукт. (Яблоки и груши - частый пример классов в учебных пособиях по объектно-ориентированному программированию.)

В объектно-ориентированной программе с применением классов каждый объект является «*экземпляром*» некоторого конкретного класса, и других объектов не предусмотрено. То есть «*экземпляр класса*» в данном случае означает не «пример некоторого класса» или «отдельно взятый класс», а «объект, типом которого является какой-то класс». При этом в разных языках программирования допускается либо не допускается существование еще каких-то типов данных, экземпляры которых не являются объектами (то есть язык определяет, являются ли объектами такие вещи, как числа, массивы и указатели, или не являются, и, соответственно, есть ли такие классы как «число», «массив» или «указатель», экземплярами которых были бы каждое конкретное число, массив или указатель).

Например, абстрактный тип данных «строка текста» может быть оформлен в виде класса, и тогда все строки текста в программе будут являться объектами — экземплярами класса «строка текста».

При использовании классов все элементы кода программы, такие как переменные, константы, методы, процедуры и функции, могут принадлежать (а во многих языках обязаны принадлежать) тому или иному классу. Сам класс в итоге определяется как список своих *членов*, а именно полей (свойств) и методов/функций/процедур. В зависимости от языка программирования к этому списку могут добавиться константы, атрибуты и внешние определения.

Как и структуры, классы могут задавать поля — то есть переменные, принадлежащие либо непосредственно самому классу (статические), либо экземплярам класса (обычные). Статические поля существуют в одном экземпляре на всю программу (или, в более сложном варианте, — в одном экземпляре на процесс или на поток/нить). Обычные поля создаются по одной копии для каждого конкретного объекта — экземпляра класса. Например, общее количество строк текста, созданных в программе за время её работы, будет являться статическим полем класса «строка текста». А конкретный массив символов строки будет являться обычным полем экземпляра класса «строка текста», так же как переменная «фамилия», имеющая тип «строка текста», будет являться обычным полем каждого конкретного экземпляра класса «человек».

В ООП при использовании классов весь исполняемый код программы (алгоритмы) будет оформляться в виде так называемых «методов», «функций» или «процедур», что соответствует обычному структурному программированию, однако теперь они могут (а во многих языках обязаны) принадлежать тому или иному классу. Например, по возможности, класс «строка текста» будет содержать все основные методы/функции/процедуры, предназначенные для работы со строкой текста, такие как поиск в строке, вырезание части строки и т. д.

Как и поля, код в виде методов/функций/процедур, принадлежащих классу, может быть отнесен либо к самому классу, либо к экземплярам класса. Метод, принадлежащий классу и соотнесенный с классом (статический метод) может быть вызван сам по себе и имеет доступ к статическим переменным класса. Метод, соотнесенный с экземпляром класса (обычный метод), может быть вызван только у самого объекта, и имеет доступ как к статическим полям класса, так и к обычным полям конкретного объекта (при вызове этот объект передастся скрытым параметром метода). Например, общее количество созданных строк можно узнать из любого места программы, но длину конкретной строки можно узнать только указав, тем или иным образом, длину какой строки будем мерить.