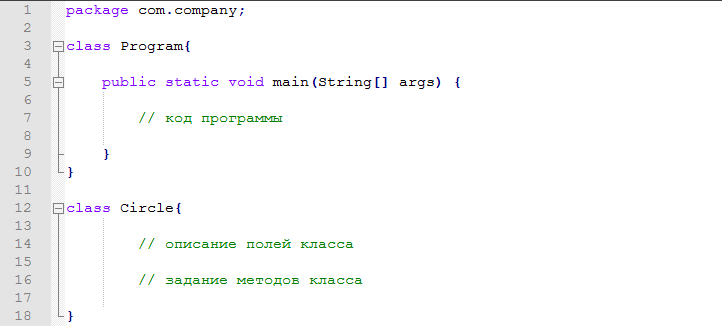
**Занятие № 21. ООП в Java.**

**Описание классов, методов. Конструкторы класса.**

Цель данной лабораторной работы - изучить основные концепции объектно-ориентированного программирования, ввести понятие класса и научиться создавать собственные классы.

**Задание 1.** Запустить на выполнение код программ, приведенных в конспекте лекционного занятия «Занятие № 6. ООП в Java (часть 2)». Выполнять только те программы, для которых приведены примеры вывода на консоль. Обращать внимание на возникающие ошибки компиляции, которые нужно исправить. Добиться успешного выполнения программ.

**Задание 2.** В качестве примера рассмотрим программу, описывающую окружность. Создадим класс Circle.



Каждая окружность имеет несколько параметров:

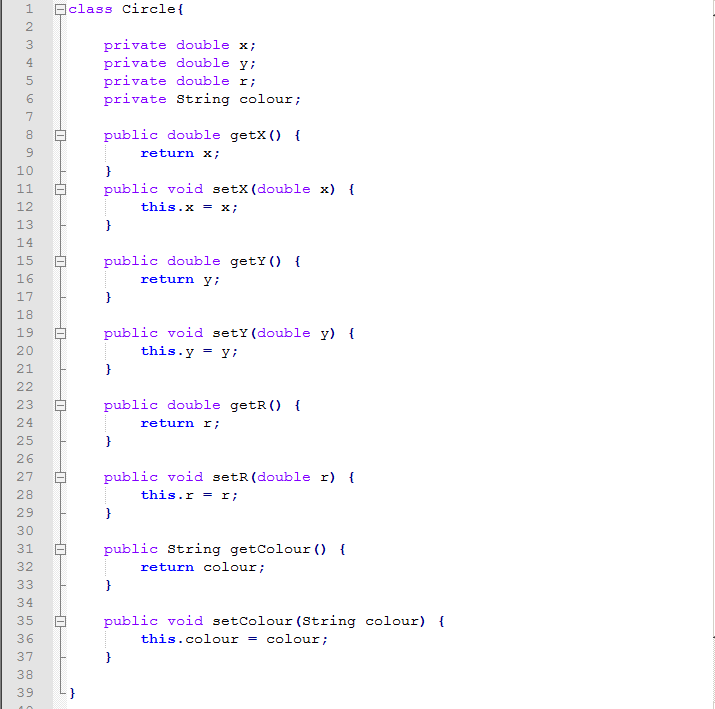
* Координаты по оси x;
* Координаты по оси y;
* Радиус r;
* Цвет colour.

Объявим наши переменные:



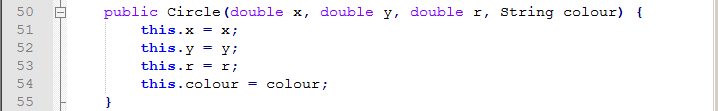
Как мы уже знаем, тип данных double применяется для числовых значений, а String для текстовых.

Одним из стандартных подходов при проектировании классов на языке Java является управление доступом к атрибутам класса через пару методов get и set. Метод get позволяет получить значение поля, set — установить новое значение. Общий принцип именования этих методов (называемых, также, геттером и сеттером).

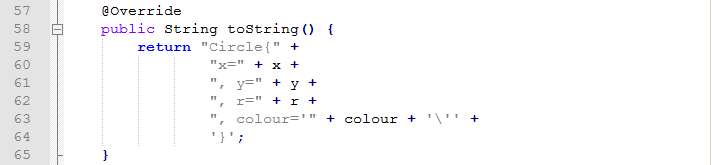


Обратите внимание, что в теле метода прописаны два вида строчек: return (для getter) и this. (для setter). Return пишется для возврата значений переменной (чтения), this. пишется в качестве ссылки на данные переменной.

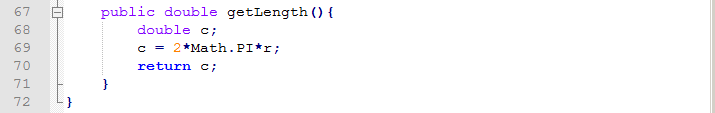
Затем также, как мы описали методы get и set, задаем метод Constructor - специальный метод, который вызывается при создании нового объекта. Не всегда удобно инициализировать все переменные класса при создании его экземпляра. Иногда проще, чтобы какие-то значения были бы созданы по умолчанию при создании объекта. По сути Constructor нужен для автоматической инициализации переменных.



Далее мы также задаем еще один метод – toString(). Метод toString в Java используется для предоставления ясной и достаточной информации об объекте (Object) в удобном для человека виде.



Последним этапом формирования класса Circle будет создание метода getLength(). В теле данного метода мы объявляем переменную *c*, которая обозначает длину нашей окружности. Длина окружности описывается формулой: с = 2\*PI\*r. В Java нельзя просто так написать PI, поэтому следует обратиться к библиотеке Math. В конце тела метода необходимо возвратить значение *c*. Получается такая запись:

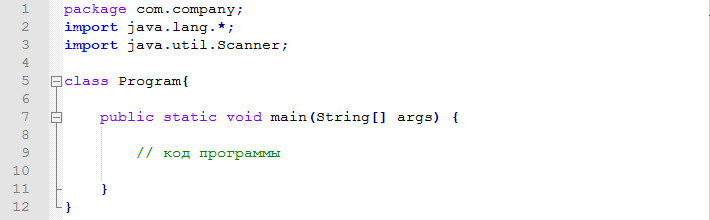


Последняя фигурная скобка не является лишней – она закрывает тело нашего класса Circle.

Следующим этапом написания нашей программы будет добавление необходимых библиотек: lang и Scanner (в дальнейшем мы будем использовать ввод с клавиатуры):

import java.lang.\*;

import java.util.Scanner;

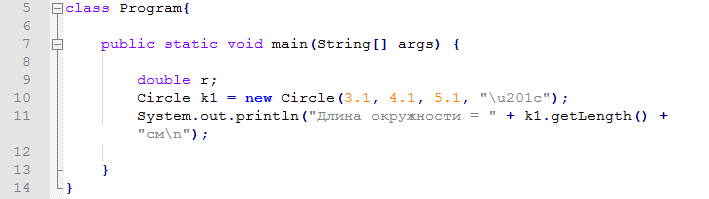


Следующим шагом будет объявление переменной *r.* Второй раз мы объявляем переменную с таким же именем, но в другом классе *Program*, чтобы ввести значение *r* с клавиатуры и посмотреть, как меняется значение *c* (строка 9).

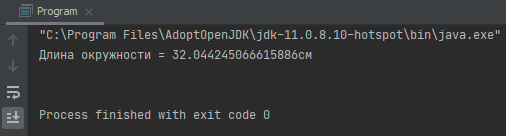
Затем мы создаем экземпляр (*k1*) класса *Circle* (строка 10).

Далее прописываем стандартную процедуру “Вывод” (строка 11).

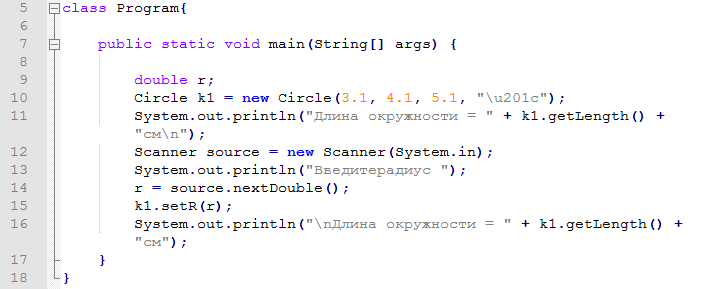
После ввода этой строчки на консоли выведется длина окружности *c*, при радиусе *r*, который мы ввели для экземпляра класса (r:5.1).



Выполняем программу:

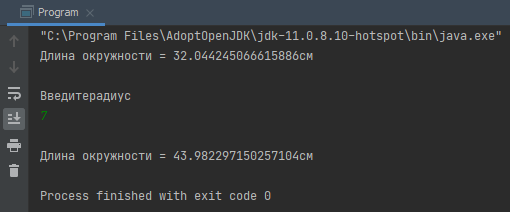


Добавляем ввод данных с клавиатуры и пишем процедуру «Вывод», в которой будет содержаться информация о том, что нам необходимо ввести, и пишем переменную, которую нам необходимо будет ввести (строки 12-14).



Пишем k1.setR(r); (строка 15). Это необходимо для того, чтобы при работе программа обращалась уже к новому значению *r*, а не к тому, которое мы ввели для экземпляра класса. Завершением класса *Program* будет написание процедуры “Вывод” (строка 16).

Выполняем программу:



Самостоятельно введите код описанной выше программы вычисления длины окружности и добейтесь ее успешного выполнения.

Отчет о выполнении практического занятия должен содержать скриншоты выполненных вами программных модулей Задания 1 и Задания 2.