# Домашнее задание №5
# Алгоритмы и структуры данных

# R-Дерево

**Авторы задания:** команда курса «Алгоритмы и структуры данных 2020».
Вопросы по заданию направлять:
Бузулуков Алексей Максимович ambuzulukov@edu.hse.ru
или в канал MS Teams «**Домашнее задание 5 R-дерево**»

## Отправка решения

Результатом решения должен быть один файл на языке C++ (.cpp), а также файлы тестов (входные и выходные данные). Именно файл на языке C++ (.cpp) и тесты будут оцениваться.
Решение нужно загрузить в LMS в проект «**Домашнее задание 5 R-дерево**» по дисциплине Алгоритмы и структуры данных 2020 уч. год Б 2 курс (код 131940, ОП: М090304ПИНЖ)
*При выявлении плагиата будет выставлена оценка 0 баллов*

**Дедлайн. 3 декабря 2020 г. 23:59 МСК**
Пожалуйста, загружайте работы заранее, чтобы не возникало технических препятствий в последний момент. Время на сервере LMS и вашем устройстве может отличаться. Пожалуйста, настройте доступ в LMS (даже если Вы проходите курс по ИУП): архивы с решениями не принимаются по электронной почте (даже с небольшим опозданием).

## Что нужно сделать?

1. Задание разбито на 3 уровня сложности. Выберите **только один уровень** сложности. Выполнение более сложной задачи не гарантирует автоматически более высокую оценку.
2. Решите задание (один файл .cpp).
3. Напишите 5 своих тестов (пары файлов входных и выходных данных) к задаче, пронумеруйте их соответственно. Разместите файлы с входными данными в директории input, а с выходными в директории output.
4. Создайте директорию с решением по шаблону <Фамилия и Имя>_<группа>_<уровень>, например BuzulukovAlexey_161_3. Разместите внутри нее директории input и output с тестами и один файл .cpp с решением задачи. Внимание: номер выбранного Вами уровня задания находится в имени директории (в примере BuzulukovAlexey_161_3 — это уровень 3).
5. Упакуйте корневую директорию (BuzulukovAlexey_161_3) в zip-архив и отправьте в LMS (размер файла должен быть не более 1МБ).

**Внимание**: не публикуйте свое решение в открытом доступе, например на GitHub либо в других открытых источниках. Это действие будет приравниваться к плагиату и повлечет за собой оценку 0 баллов за домашнее задание (в том числе при обнаружении такового факта уже после выставления оценки).

## Мотивация

Вы начинающий программный инженер, модерирующий сайт компании застройщиков в Москве. Для каждого нового заказного проекта перед стройкой прогнозируется занимаемое им место в виде прямоугольника, а так как местами Москва сильно застроена, бывает необходимо снести все хрущевки, пересекающие это место. Причем в качестве фигуры здания считается ее минимальный ограничивающий прямоугольник, ведь зачастую приходится сносить и пересекающую дворовую территорию. До вашего прихода несколько рабочих перед стройкой открывали карты Open Street Map и «по линейке» искали пересекающие постройки, и в качестве принятия решения о сноске зданий отдавали список идентификаторов построек Open Street Map ID. Начальник попросил вас автоматизировать этот поиск, чтобы экономить время сотрудников.

Основная ваша задача – найти идентификаторы объектов Open Street Map, чей минимальный ограничивающий прямоугольник пересекает указанную прямоугольную область. Для новой предметной области вам пришлось изучить дополнительную литературу, благодаря которой вы выяснили, что задачу можно решить тремя различными способами (см. уровни оценивания задачи).

Примечание: Open Street Map – картографический сервис со свободной лицензией, позволяющий использовать обширную географическую информацию при указании соответствующих авторских прав.

## Модель данных задания

Для реализации задачи вы выгрузили все векторные данные строений Москвы в директорию *data*. С помощью ГИС (Геоинформационной Системы) вы открыли файл *building-polygon.shp* и выяснили, что все строения представлены в виде геометрий полигонов и мульти полигонов, причем у каждого объекта есть необходимый уникальный идентификатор *OSM_ID* целочисленного типа *int*, стоящий на нулевом индексе атрибутов объекта. Точки полигонов представлены в виде двух координат – широты (x) и долготы (y) типа данных с плавающей точкой *double*. Так как область Москвы находится далеко от полюсов Земли и начальнику важно быстродействие программы, было принято решение работать с этими координатами как в декартовой системе координат: погрешность будет минимальной, а расчеты будут выполняться быстрее.

Примечание: самостоятельно ознакомиться с моделью данных можно с помощью любой Геоинформационной Системы, например QGIS (рисунок 1), открыв .shp файл с векторными данными.

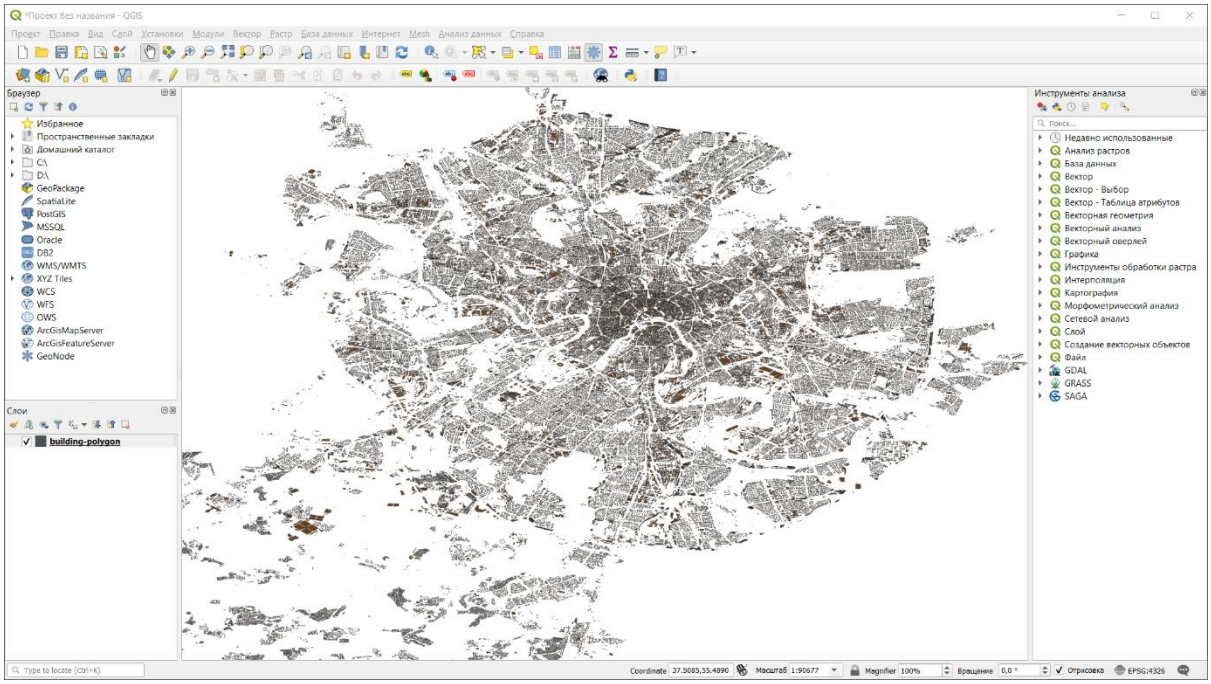


*Рисунок 1. Геометрии строений Москвы в QGIS*

## Условие задания

С помощью библиотеки GDAL первоначально необходимо загрузить векторные данные из файла *building-polygon.shp* (находится в архиве с домашним заданием). Для каждой геометрии объекта необходимо найти минимальный ограничивающий прямоугольник (MBR) и соответствующий ему атрибут OSM_ID. На вход программе передается один прямоугольник, для которого нужно найти все пересекающие его MBR загруженных геометрий соответствующим способом, в зависимости от выбранного вами уровня сложности задания (См. Уровни оценивания). В качестве выходных данных нужно вывести соответствующие атрибуты OSM_ID минимальных ограничивающих прямоугольников загруженных геометрий, пересеченных входным прямоугольником. Для уровней 2 и 3 необходимо считать, что все точки располагаются в декартовой системе координат.

## Уровни оценивания

**Уровень 1** (оценка до 6 баллов) – Необходимо загрузить данные средствами стандартной библиотеки GDAL и реализовать алгоритм нахождения пересечения прямоугольников полным перебором. Перевод координат точек в какую-либо другую систему координат не требуется.

**Уровень 2** (оценка до 8 баллов) – Для реализации алгоритма пересечения прямоугольников необходимо использовать готовую пространственную структуру данных R-tree из библиотеки boost-geometry. Загрузку данных, нахождение атрибутов и минимальных ограничивающих прямоугольников геометрий следует выполнить с помощью библиотеки GDAL. Для задания точки следует использовать тип boost::geometry::model::point<double, 2, boost::geometry::cs::cartesian>. Прямоугольник указывается типом boost::geometry::model::box<тип точки>. Значения, вставляемые в rtree, должны иметь тип std::pair<тип прямоугольника, int>, где int – соответствующий атрибут OSM_ID. Шаблонный класс rtree задается с параметрами

boost::geometry::index::rtree<тип значения, boost::geometry::index::quadratic<8, 4>>. В качестве поиска пересечений у заполненного объекта rtree следует использовать функцию query с первым параметром boost::geometry::index::intersects(искомый прямоугольник).

**Уровень 3** (оценка до 10 баллов) – Для реализации алгоритма пересечения прямоугольников необходимо использовать свою реализацию пространственной структуры данных R-Tree. Загрузку данных, нахождение атрибутов и минимальных ограничивающих прямоугольников геометрий следует выполнить с помощью библиотеки GDAL. В вашей реализации R-Tree обязательно должны быть методы:

insert – принимает на вход прямоугольник с соответствующим идентификатором OSM_ID и вставляет его в текущий инстанс объекта R-Tree.

search – принимает на вход прямоугольник поиска и возвращает идентификаторы OSM_ID всех пересекаемых им прямоугольников в текущем инстансе объекта R-Tree.

Корень R-Tree и все узлы дерева следует держать в оперативной памяти. В качестве константы минимального числа дочерних узлов у вершины следует использовать значение 4. В качестве константы максимального числа дочерних узлов у вершины следует использовать значение 8. За деталями реализации R-Tree следует обратиться к статье [1] и лекции курса [2].

Внутри реализации R-Tree разрешается использовать стандартную библиотеку шаблонов C++.

## Формат входных данных

На первой и единственной строке входного файла расположены координаты точек прямоугольника поиска в формате:

`<x min><пробел><y min><пробел><x max><пробел><y max>`

Координата точки является вещественным типом double, записанным в строке через знак точки ‘.’. Гарантируется, что все входные значения корректны и помещаются в тип double.

## Формат выходных данных

В выходной файл записываются атрибуты OSM_ID пересекаемых прямоугольником MBR геометрических объектов, каждый атрибут на новой строке (строки разделены символом конца строки `'\n'`). Порядок вывода атрибутов – по возрастанию значения. Последняя строка в файле должна быть пустой.

## Пример

| Входные данные | Выходные данные |
|---|---|
| 36.8127 55.455 36.8156 55.4561 | 345185815<br>345185820<br>345185829<br>345185833<br>345185838<br>345185849 |

## Как будет оцениваться работа?

Необходимо защитить работу на соответствующем семинаре преподавателям (устные собеседования). Перед защитой Вам нужно открыть у себя на компьютере файл с решением, который Вы загружали в LMS (преподаватели будут следить за тем, открыли ли Вы именно тот файл, который загружали в LMS).

Пожалуйста, не пропускайте соответствующий семинар, который является частью элемента контроля (элемент контроля – данное домашнее задание).

Решения будут проверяться на наборах тестов, каждый тест — это два файла, один для входных данных, другой для выходных. Таким образом, ввод и вывод данных происходит через файлы. Будут оценены: корректность кода, составленные тесты (граничные условия, сложные случаи и т.п.), декомпозиция, аккуратность кода, понимание логики решения, структуры кода, конструкций языка, структур данных.

В приложенных к условию задачи директориях расположено по 5 тестов к каждой вариации алгоритма, на которых можно проверить работоспособность решения, однако стоит учитывать, что прохождение приложенных тестов не гарантирует полное отсутствие ошибок в решении (решения могут также проверяться и на других, расширенных тестах). Файлы входных данных именуются по принципу '*test<номер теста>.txt*', файлы выходных данных (или ответов) именуются 'answer<номер теста>.txt'.

Названия файлов для входных данных и выходных данных должны быть указаны через командную строку. Кроме того, в параметрах командной строки могут указываться не только имена файлов и директорий, но и относительные/абсолютные пути к ним. Например, так будет выглядеть тестирование программы BuzulukovAlexey_161_3.cpp:

cl BuzulukovAlexey_161_3.cpp
BuzulukovAlexey_161_3 data test1.txt answer1.txt
*либо*
BuzulukovAlexey_161_3 path/to/data/ path/to/test1.txt path/to/answer1.txt

где

1-ый параметр – путь до директории с данными (файлы с векторными данными),
2-ой параметр – путь к входному файлу,
3-ий параметр – путь к выходному файлу.

Пример использования аргументов командной строки в C++:
https://docs.microsoft.com/ru-ru/cpp/cpp/main-function-command-line-args

## Список литературы


1. Antonin Guttman, R-Trees: A Dynamic Index Structure for Spatial Searching, Proc. 1984 ACM SIGMOD International Conference on Management of Data, pp. 47–57.
2. Лекция по курсу Алгоритмы и структуры данных 2020: «Пространственные структуры данных» от 17.11.2020
3. Репозиторий с примерами кода использования библиотеки GDAL: https://github.com/b1nd/gdal-example