**Problem description**

Develop an algorithm for automated spatial planning, considering set of parameters and constraints. As an input algorithm receives GeoJSON file. Algorithm outputs GeoJSON file, containing information about new objects positioned on and existing site plan and an image of land-use plan, containing input geojson and output geojson objects.

**Input parameters (algorithm must receive this inputs)**

1. site boundaries (GeoJSON format. FeatureCollection with Polygon), containing building zone(s) and constraints, like Лесная зона, Дорога, Водоем, where we cant build;
2. urban density (percent of land area, available for building);
3. building site configuration, containing information about percentage of land use for certain areas within building zone (for instance 0.5 for residential, 0.2 for commercial and 0.3 for park);
4. minimal distance between new zones (in meters);
5. minimal distance between buildings within zones (in meters);

**Example of input instance:**

{

  "type": "FeatureCollection",

  "features": [

{

   "type": "Feature",

   "properties": {"name": "Зона застройки"},

   "geometry": {

     "type": "Polygon",

     "coordinates": [[[37.6173, 55.7558], [37.6179, 55.7565], [37.6185, 55.7559], [37.6173, 55.7558]]]

   }

},

{

   "type": "Feature",

   "properties": {"name": "Лесная зона", "restriction": "no\_build"},

   "geometry": {

     "type": "Polygon",

     "coordinates": [[[37.6180, 55.7560], [37.6185, 55.7565], [37.6190, 55.7560], [37.6180, 55.7560]]]

   }

},

{

   "type": "Feature",

   "properties": {"name": "Дорога", "restriction": "no\_build"},

   "geometry": {

     "type": "LineString",

     "coordinates": [[37.6175, 55.7555], [37.6185, 55.7555]]

   }

}

  ]

}

**Algorithm goal:**

1. try to split area available for building into corresponding zones (residential, commercial, road, park. Importance rate: 1, 0.75, 0.6, 0.25 – can be set by user), considering input constraints and area importance rate;
2. position new areas within building site, considering input constraints;
3. position buildings within new areas (residential, commercial), considering input constraints;
4. visualize land use plan using matplotlib (for each type of object (residential, commercial, road, park) use different distinguishable colors; residential buildings color should be different from commercial buildings color). Input geojson objects should be present on the plot. Building site bounding box should be present on the plot. Save figure into file.
5. Return GeoJSON file with newly placed areas (not including buildings)

**Example of output instance:**

1. GeoJSON file

{

  "type": "FeatureCollection",

  "features": [

{"type": "Feature", "properties": {"type": "residential"}, "geometry": {"type": "Polygon", "coordinates": [[...]]}},

{"type": "Feature", "properties": {"type": "commercial"}, "geometry": {"type": "Polygon", "coordinates": [[...]]}},

{"type": "Feature", "properties": {"type": "road"}, "geometry": {"type": "LineString", "coordinates": [[...]]}},

{"type": "Feature", "properties": {"type": "park"}, "geometry": {"type": "Polygon", "coordinates": [[...]]}}

  ]

}

**Additional tasks:**

1. Algorithm should generate several land-use plans (default=1). For visualization each land-use plan should be placed in subplot.
2. Algorithm should accept as an input user-defined shapes for each new area (default=rectangular) (circle, triangle, square, etc.). information about shapes should be in configuration dictionary
3. Configuration dictionary should also contain information about availability of public transport. But for now don’t use this information for land use planning.