AUTOMATING GIS PROCESSES

FINAL ASSIGNMENT

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

FINAL ASSIGNMENT

60 % of course grade

Two deadlines to choose from:

- 1st deadline: Thursday the **31st December 2020**
- 2nd deadline Friday the **15th of January 2021**

(if submitting earlier, send a Slack msg to Vuokko)

The main goal is to practice and demonstrate skills learned during Geo-Python and AutoGIS ☺

COURSE TOPICS

1	Shapely and geometric objects (points, lines and polygons)
2	Managing spatial data with Geopandas (reading and writing data, projections, table joins)
3	Geocoding and spatial queries
4	Reclassifying data, overlay analysis
5	Visualization: static and interactive maps
6	OpenStreetMap data (osmnx) and Network analysis (networkx)
7	Course recap and preparing for the final assignment
Extra	Raster processing (rasterio), Python in QGIS

REMINDER: AUTOGIS LEARNING GOALS 1/2

- After completing this course, the students are able to
  - test and produce **modular code** in the Python programming language
  - **manage spatial data** programmatically (for example, reading different data formats, re-projecting, re-classifying and storing data),
  - **apply spatial analysis methods** in Python (such as buffering, network analysis and spatial joins)
  - create **visualizations** (graphs and maps) from geographic data using Python
  - design and implement a geographical **data analysis workflow**

REMINDER: AUTOGIS LEARNING GOALS 2/2

- After completing this course, the students are able to
  - Independently **search for information** regarding programming methods
  - **Apply new methods** based on online documentation
  - **Critically evaluate** the available methods and information sources
  - Understand the importance of **version control** for practical tasks and scientific purposes
  - **Communicate** their analysis workflow in written format
  - Complete assignments **on time** ☺

FINAL ASSIGNMENT

The task in the final assignment is to develop a GIS analysis workflow using Python that includes:

- **Data acquisition** (Reading data from files or online sources)
- **Data analysis** (Enriching and analyzing the data, eg. spatial join, overlay, buffering, other calculations..)
- **Visualization** (Visualizing the results as maps and graphs)

→ ideally, the same code could be applied on multiple inputs with minimum amount of manual edits.

STRUCTURE

- The final work will be a set of Jupyter Notebook files (.ipynb) and/or Python script files (.py)
- The README (.md) should contain all relevant information about the work, including links to all files
- IF you are using large input files, DON'T upload them to GitHub! You can provide sample data for demonstrating your workflow, and/or download instructions for the whole data set.
- Remember to use informative variable names, inline comments, docstrings etc.

→ anyone who gets a copy of your repository should be able to run your code AND read your code.

FINAL ASSIGNMENT ASSESMENT

60 points points for main components of the workflow (20 per each):

- reading and manipulating data
- analyzing data
- visualizing data

10 points for overall documentation of the work

Extra points available for other merits in the work

→ **Grade 1-5**

CHECKLIST..

- Is the overall aim and structure of the submission is clearly documented in the README.md file
- Is the documentation of the analysis process and related functions clear (docstrings, comments, markdown texts readme)?
- Are there visualizations? At least a single map should be in place, after all, this is a GIS course!
- Does the code work? Does the code work with different inputs?
- Does the code avoid unnecessary repetition (e.g. by using functions and for-loops)?
- Is the extent of the work sufficient? (consider that the final work replaces an exam!)

Final Assignment

Status

Once you are finished with the final assignment, edit this readme and add "x" to the correct box:

- [ ] Submitted
- [x] I'm still working on my final assignment.

Instructions

Read the final assignment instructions from the course webpages https://autogis.github.io. Remember to write readable code, and to provide adequate documentation using inline comments and markdown. Organize all your code(s) / notebook(s) into this repository and **add links to all relevant files to this `README.md` file.** In sum, anyone who downloads this repository should be able to **read your code and documentation** and understand what is going on, and **run your code** in order to reproduce the same results! :)

Modify this readme so that anyone reading it gets a quick overview of your final work topic, and finds all the necessary input data, code and results.

YOU CAN REMOVE THESE INSTRUCTIONS FROM THE FINAL SUBMISSION

Note: If your code requires some python packages not found in the cscmpgeobooks environment, please mention them also in this readme and provide installation instrutions.

Note: Don't upload large files into GitHub! If you are using large input files, provide downloading instructions and perhaps a small sample of the data in this repository for demonstrating your workflow.

Fill in details of your final project below. You can remove this instructions-section from the README-file if you want.

Topic:

Structure of this repository:

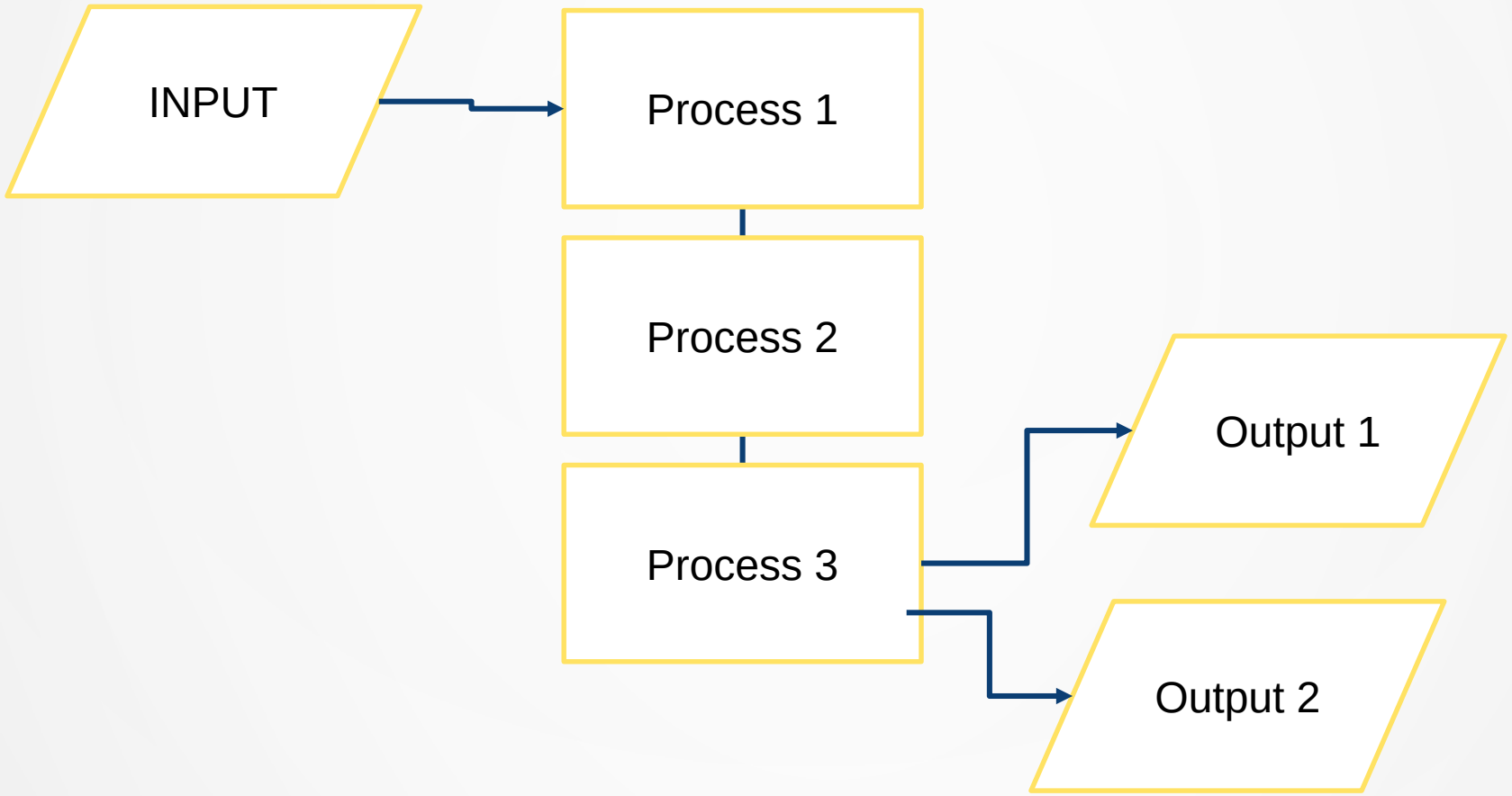
Input data:

Analysis steps:

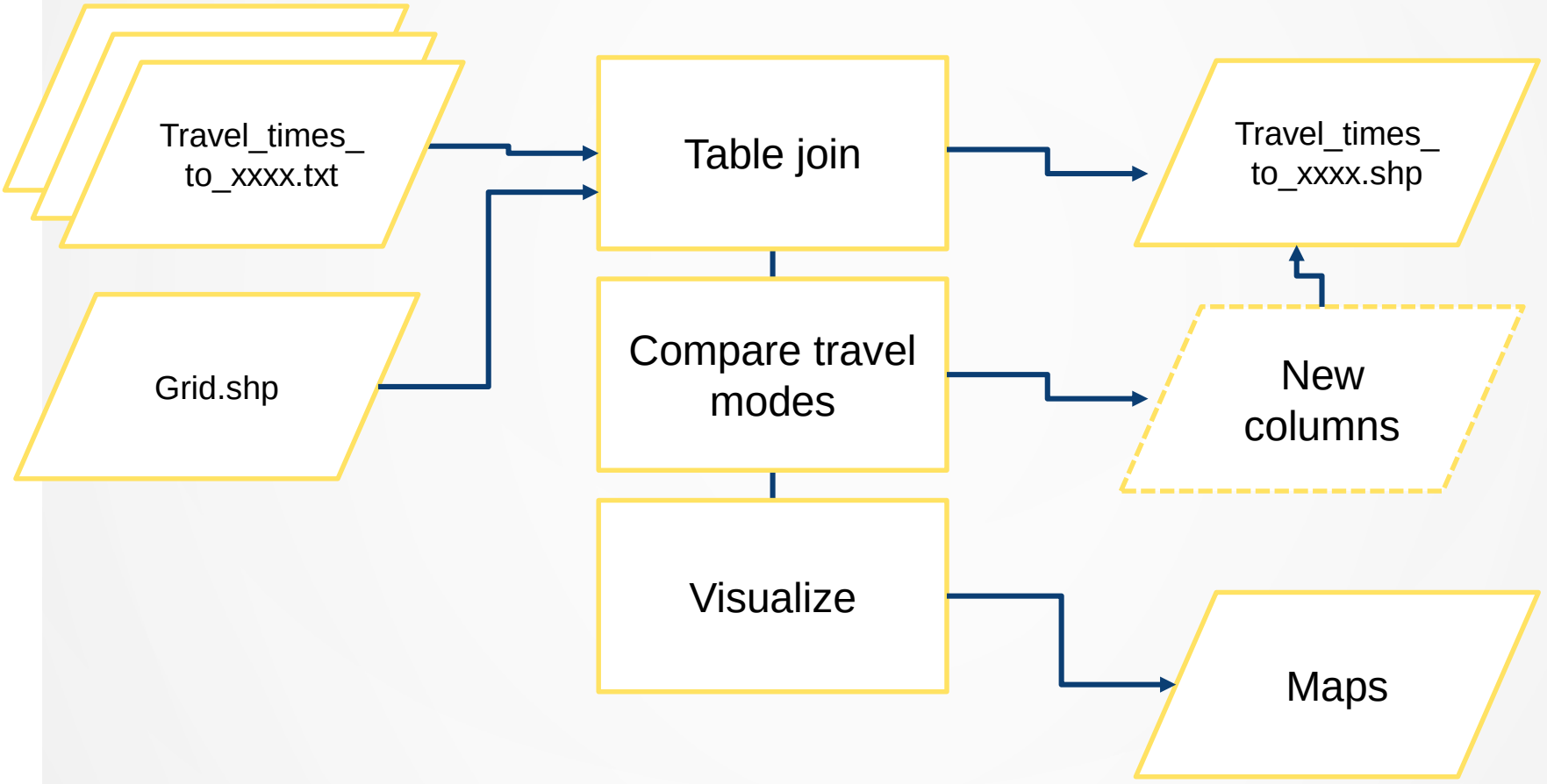
Results:

References:

WORKFLOW



WORKFLOW


Travel_times_
to_xxxx.txt
Grid.shp
Table join
Compare travel
modes
Visualize
Travel_times_
to_xxxx.shp
New
columns
Maps

ACCESS VIZ

AccessViz is a Python tool (i.e. a set of Notebooks and/or Python script files) for managing, analyzing and visualizing the Travel Time Matrix data set. AccessViz consist of Python functions, and examples on how to use these functions.

AccessViz has four main components for accessing the files, joining the attribute information to spatial data, visualizing the data and comparing different travel modes:

1. **FileFinder**

2. **TableJoiner**

3. **Visualizer**

4. **Comparison tool**

+ four other optional components

URBAN INDICATORS

Develop an urban analytics tool and apply it to at least two cities or neighborhoods(e.g. Helsinki and Tampere, or neighborhood areas in Helsinki).

The main idea is to calculate a set of metrics / indicators based on the urban form and/or population, and to compare the cities/regions based on these measures.

You should use 2-4 different indicators, for example some of these:

Population distribution and demographics

Urban population growth

Accessibility:

Green area index

Street network metrics

Building density

YOUR OWN TOPIC?

Develop your own topic! In general, your own topic should also contain these sections:

1. **Data acquisition** (Fetching data, subsetting data, storing intermediate outputs etc.)
2. **Data analysis** (Enriching and analyzing the data, eg. spatial join, overlay, buffering..)
3. **Visualization** (maps and graphs)

But feel free to be creative! Your own project might be, for example, related to your thesis or work project. Remember to describe clearly what you are doing in the final assignment repository README.md -file.

Preferably, present your idea to the course instructors before the winter holidays.

FINALLY..

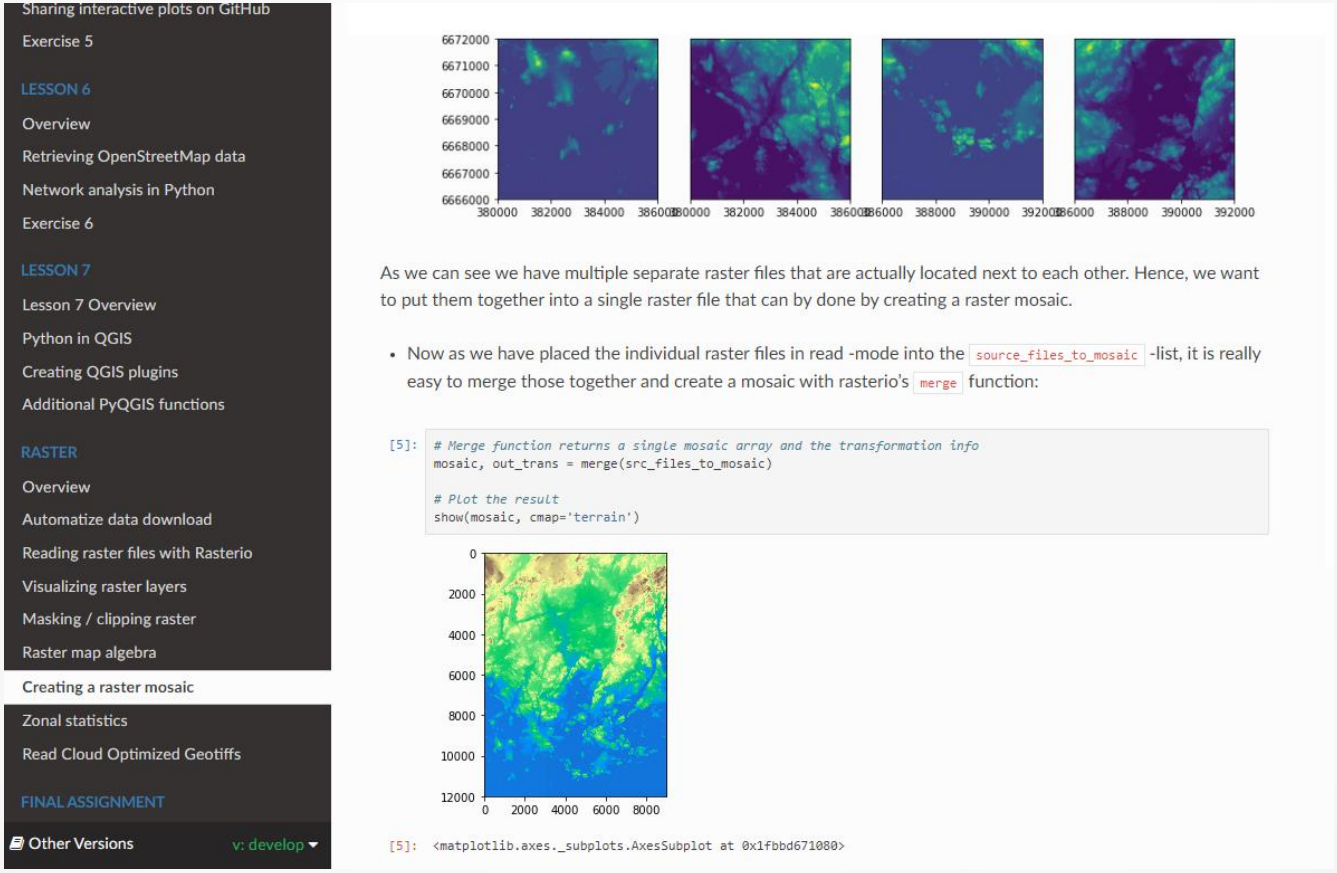
There are several guidelines how to do programming "in a proper way". These best practices when doing programming are well described in Wilson et al. (2014)
http://journals.plos.org/plosbiology/article?id=10.1371/journal.pbio.1001745

1. *"Write programs for people, not computers."*
2. *"Let the computer do the work."*
3. *"Make incremental changes."*
4. *"Don't repeat yourself (or others)."*
5. *"Plan for mistakes."*
6. *"Optimize software only after it works correctly."*
7. *"Document design and purpose, not mechanics."*
8. *"Collaborate."*

ADDITIONAL RESOURCES

The cool stuff we didn't (yet) cover during the lessons..

RASTER DATA PROCESSING



https://automating-gis-processes.github.io/site/lessons/Raster/overview.html

ONLINE RESOURCES..

- **MovingPandas** for analysing trajectories:
  - https://github.com/anitagraser/movingpandas
  - https://anitagraser.com/2019/09/11/movement-data-in-gis-24-movingpandas-hands-on-tutorials/
- **Plotly express** for interactive maps:
  - https://plot.ly/python/plotly-express/
  - https://plot.ly/python/maps/
- Raster data visualization using **datashader**
  - https://datashader.org/user_guide/Geography.html

…. And many more!

LET'S GET STARTED ☺!

https://autogis.github.io