



GIS
Python



Python for GIS

USP Digital Urban 2018

Lecturer: Henrikki Tenkanen
henrikki.tenkanen@helsinki.fi

12.2.2018

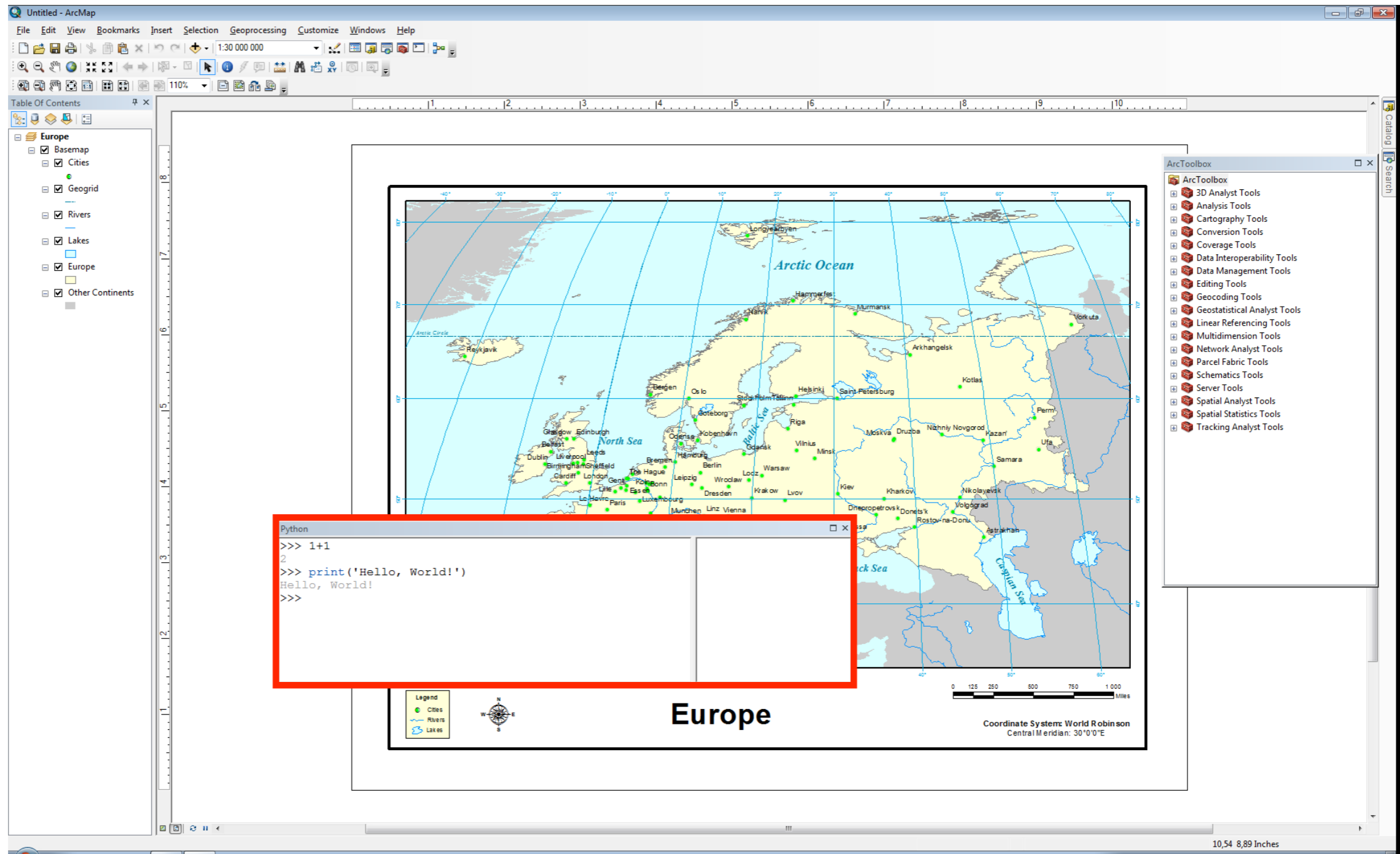


Why learn to program?

```
dhcp-eduroam-hy-55-157 whip ~ > python
Python 2.7.5 (default, Aug 25 2013, 00:04:04)
[GCC 4.2.1 Compatible Apple LLVM 5.0 (clang-500.0.68)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> Average_Geologist = 100
>>> Programming_Factor = 1000
>>> Quantitative_Geologist = Average_Geologist * Programming_Factor
>>> Quantitative_Geologist > Average_Geologist
True
>>> []
```

- Rather than being restricted to using existing software, you will have the ability to develop your own solutions when solutions do not exist or are inefficient
- Many software packages offer the ability to extend their capabilities by adding your own short programs (e.g., ArcGIS, ParaView, Google Earth, etc.)

Python can be called directly from QGIS/ArcGIS



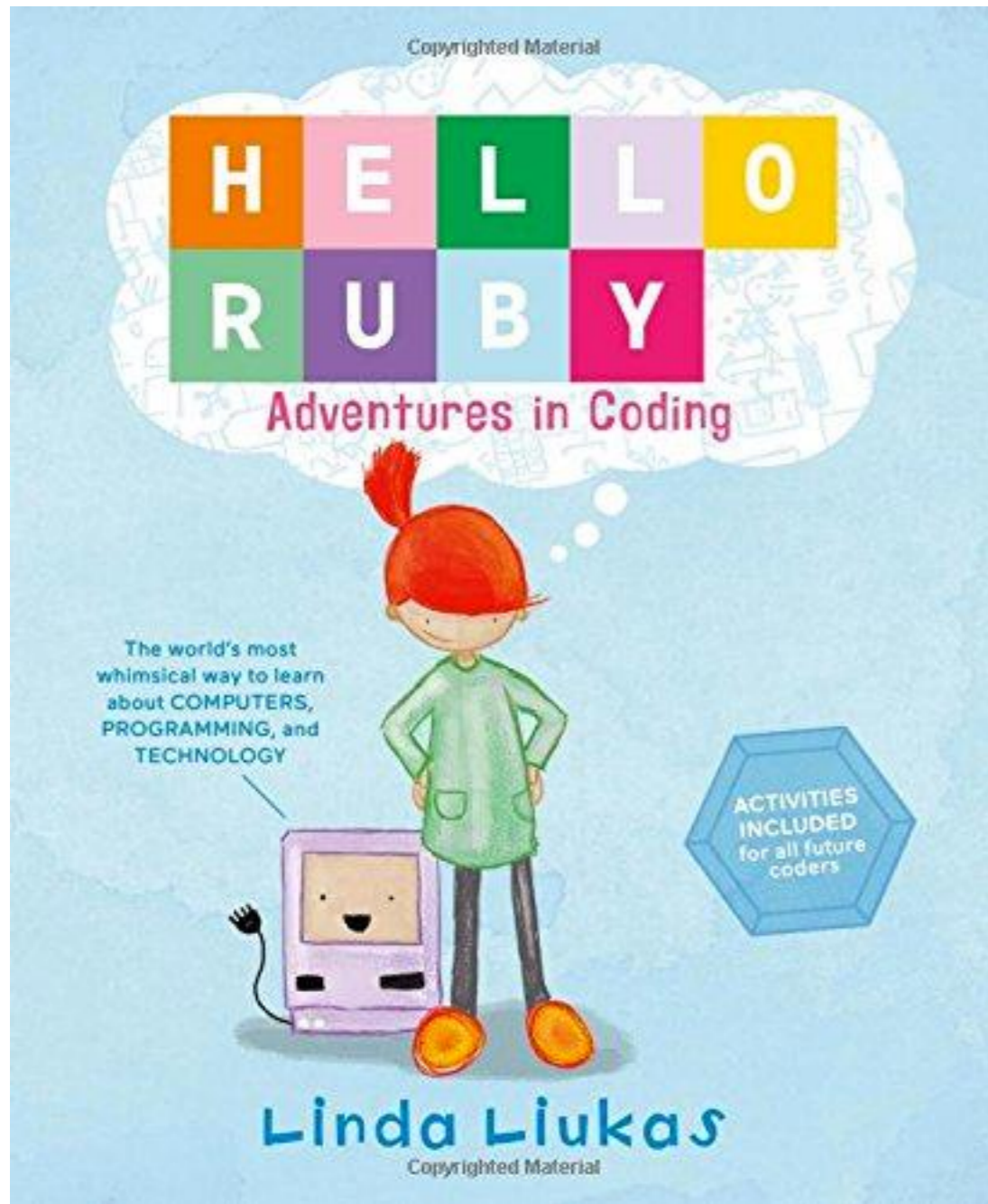
The screenshot displays the ArcMap interface with a map of Europe. A Python console window is open in the foreground, showing the following code and output:

```
Python
>>> 1+1
2
>>> print('Hello, World!')
Hello, World!
>>>
```

The map shows various cities and geographical features. The ArcToolbox is visible on the right side of the interface, listing various tool categories such as 3D Analyst Tools, Analysis Tools, and Geostatistical Analyst Tools.



Why learn to program?



- Believe it or not, **programming is fun!** It involves
 - Breaking complex problems down into simpler pieces
 - Developing a strategy for solving the problem
 - Testing your solution
- All of this can be exciting and rewarding (when the code works...)



Programming



“Computer programming (often shortened to programming) is a **process** that leads from an original formulation of a computing problem to executable computer programs. Programming involves activities such as analysis, developing understanding, generating algorithms, verification of requirements of algorithms including their correctness and resources consumption, and implementation (commonly referred to as coding) of algorithms in a target programming language.”

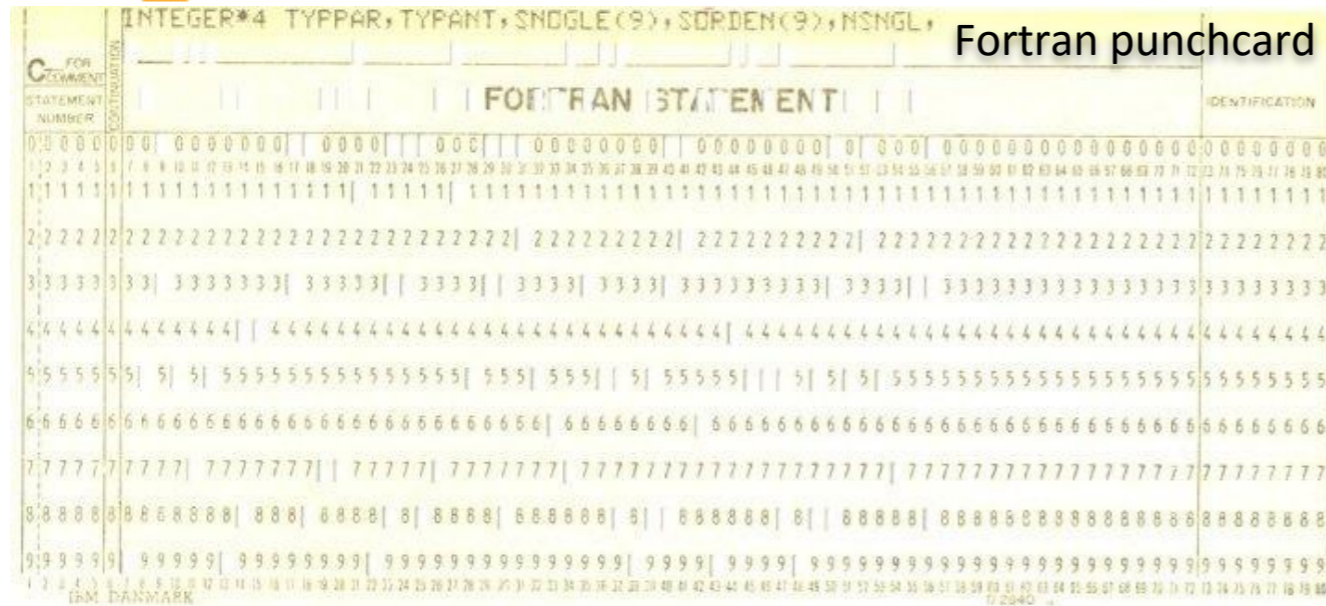
-Wikipedia (2015)

“A program is like a recipe. It contains a list of ingredients (called variables) and a list of directions (called statements) that tell the computer what to do with the variables.”

- Webopedia (2015)



What is a program?



```
# Define plot variables
misfit = NA_data[:,0]
var1 = NA_data[:,1]
var2 = NA_data[:,2]
var3 = NA_data[:,3]
clrmin = round(min(misfit),3)
clrmax = round(min(misfit),2)
trans = 0.75
ptsize = 40
```

Python source code

- A **program** is a detailed list of step-by-step instructions telling the computer exactly what to do
- The program can be changed to alter what the computer will do when the code is executed
- **Software** is another name for a program



What is a programming language?

- A **computer language** is what we use to ‘talk’ to a computer
- Unfortunately, computers don’t *yet* understand our native languages
- A **programming language** is like a code of instructions for the computer to follow
 - It is exact and unambiguous
 - Every structure has a precise form (**syntax**) and a precise meaning (**semantics**)
- Python is just one of many programming languages



Programming



Various different programming languages exist

- Python
- Perl
- Ruby
- JavaScript
- Java
- C++
- C
- Fortran

+ Many others

High level
programming



Low level
programming

Less
code



More
code

Faster
to write



Slower
to write

Slower to
execute



Faster to
execute



Programming



Programming languages remind our spoken languages!

- There are different ways to express the same meaning
- Some languages are more similar to each other than others
- After learning one language it is easier to learn other ones!

Moikka, Hello, Hej, Hallo, Hola

Spoken languages



```
echo "Moikka", print("Hello"), console.log("Hej"),  
puts("Hallo"), System.out.println("Hola")
```

Programming languages



Programming



Why to learn / use Python?

- Easy to learn (a good programming language for the beginners)
- Easy to read and understand – elegant code
- Powerful – widely used e.g. by NASA for scientific programming
- Countless ready-made modules / libraries to use
- Supportive, large and helpful user community
- Widely supported in different GIS-softwares
 - ArcGIS, QGIS, Erdas Imagine, IDRISI, uDig, ILWIS, PostGIS ...
- **Extremely useful for automating GIS processes**



Developing a program

- Coming up with a specific list of instructions for the computer to follow in order to accomplish a desired task is not easy
- The following list will serve us as a **general software development strategy**
 1. Analyze the problem
 2. Determine specifications
 3. Create a design
 4. Implement the design
 5. Test/debug the program
 6. Maintain the program (if necessary)



Let's consider an example

- As an American, David was raised in a country that uses Fahrenheit for temperatures
 - 70°F is lovely
 - 90°F is hot
 - Water freezes at 32°F
- The problem here in Finland is that he doesn't always know what he should wear to work when he finds weather reports with temperatures in degrees Celsius
 - A simple program could help



Developing a program

1. Analyze the problem

- Before you can solve a problem, you must figure out exactly what should be solved



Developing a program

1. Analyze the problem

- Before you can solve a problem, you must figure out exactly what should be solved

2. Determine specifications

- Describe exactly what the program will do
- Don't worry about how it will work. Determine the input and output values and how they should interact in the program



Developing a program

3. Create a design

- What is the overall structure of the program? How will it work?
- It is often helpful to write out the code operation in **pseudocode**, precise English (or Finnish) describing the program. Be specific!



Developing a program

3. Create a design

- What is the overall structure of the program? How will it work?
- It is often helpful to write out the code operation in **pseudocode**, precise English (or Finnish) describing the program. Be specific!

4. Implement the design

- If you've done a good job with the previous steps, this should be fairly straightforward. Take your pseudocode and 'translate' it into Python



Developing a program

5. Test/debug the program

- Now you can put your new Python code to the test (literally) by running it to see whether it reproduces the expected values
- For any test, you should know the correct values in advance of running your code. How else can you confirm it works???



Developing a program

5. Test/debug the program

- Now you can put your new Python code to the test (literally) by running it to see whether it reproduces the expected values
- For any test, you should know the correct values in advance of running your code. How else can you confirm it works???

6. Maintain the program

- If you've written something that will be shared by other users, a helpful programmer will continue to add features that are requested by the users



References

Zelle, J. M. (2010). *Python programming: an introduction to computer science* (2nd ed.). Franklin, Beedle & Associates, Inc.