Introduction to Programming

For Archaeologists

Part 1: Introduction

2021-2022

Universiteit Leiden
The Netherlands

Discover the world at Leiden University

About Me

Alex Brandsen

- Ba / MSc in Archaeology
- Background in Web Development
- PhD in Text Mining in archaeology
- Currently: Postdoc
    - Building search engine for excavation reports

Introductions

- Your name
- Why you want to learn programming
- If you have coded before

Topics of this lecture series

1. Introduction: Python, variables, comments (R?)
2. Lists & Loops
3. Loading and manipulating data
4. Graphs & Plots
5. SQL & Databases
6. Advanced methods: Machine Learning, QGIS integration

- Short lecture + exercises every week
- Assignment every 2 weeks
- Exam at the end

Timetable

- 8 April: 13.15 - 15.00 - Introduction
- 15 April: **No lecture** due to Good Friday, university is closed
- 22 April: 13.15 - 15.00 - Lists and Loops
- 29 April: 13.15 - 15.00 - Loading and Manipulating Data
- 6 May: 13.15 - 15.00 - Graphs
- 13 May: 13.15 - 15.00 - SQL
- 20 May: 13.15 - 15.00 - Advanced Methods

Timetable

Assignment deadlines

- Assignment 1: 22 April
- Assignment 2: 6 May
- Assignment 3: 20 May

Exam

24 May, 13.00

Materials

Brightspace

- Literature
- Slides

Github

- Modules
- Assignments (download, complete, then upload on Brightspace)
- Slides

Topics of this lecture

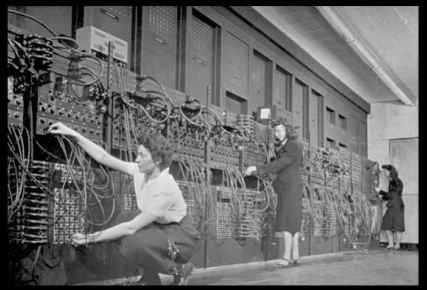
- What is programming?
- Why should you learn to code?
- What is it used for in archaeology?
- What is Python, and why do we use it?
- Basics: variables, syntax, comments, print, if/else
- Debugging
- Getting started with the exercises:
  - Anaconda
  - Jupyter Notebook
  - Github

After this lecture:

- You can conceptually describe programming
- You know why programming can be useful
- You can list a couple of applications of code in archaeology
- You know what the following concepts are:
  - Variables
  - Comments
  - The `print()` command
  - If / else statements
- You know how to open Jupyter Notebooks in Anaconda (after the exercise)

What is Programming?

What my friends think I do

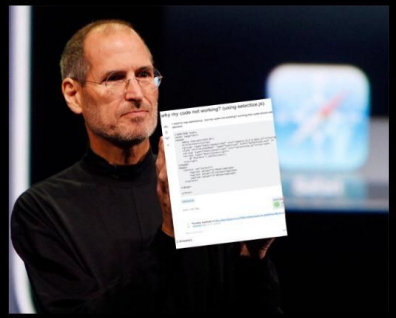


What my mom thinks I do

What the media thinks I do

What my co-workers think I do

What I think I do



What I really do

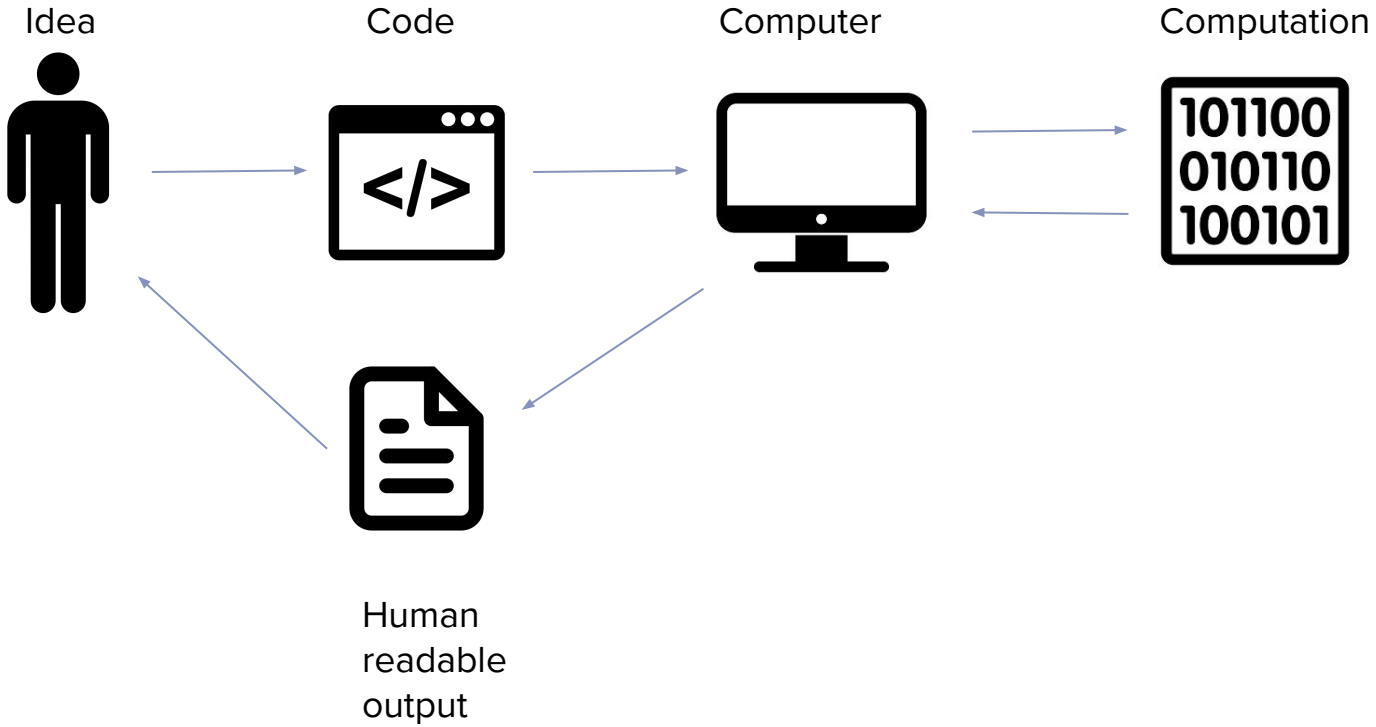
What is Programming?

- “the process of performing a particular computation (or more generally, accomplishing a specific computing result), usually by designing/building an executable computer program.”

- “Translating instructions for a computer from human language to a language a machine can understand. This code tells the computer how to behave and what actions to perform.”

- Basically, writing code to create software that can do things for you

What is Programming?



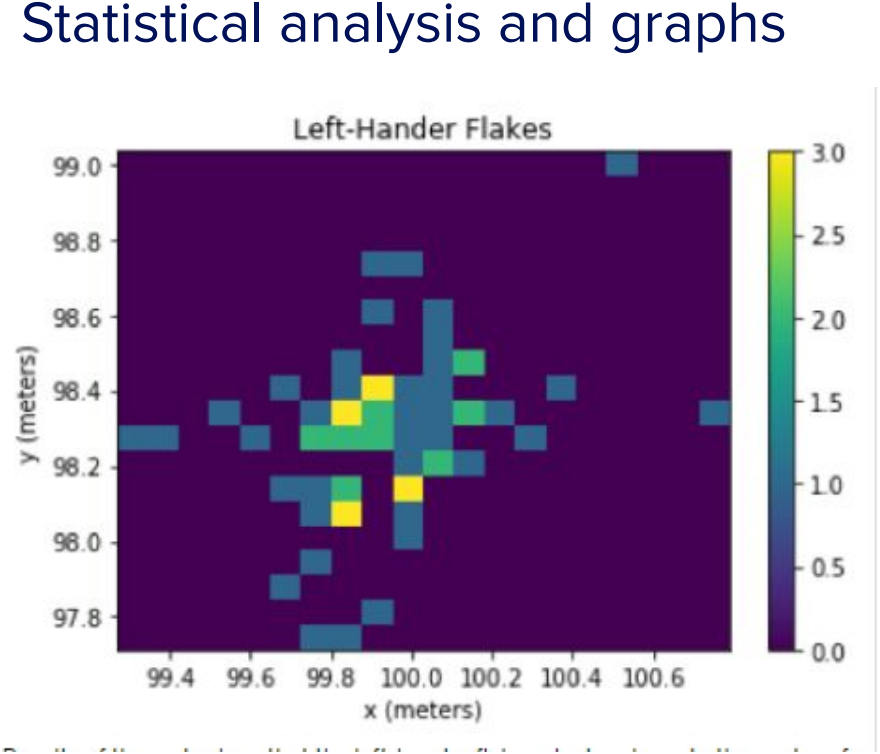
Why should you learn to code?

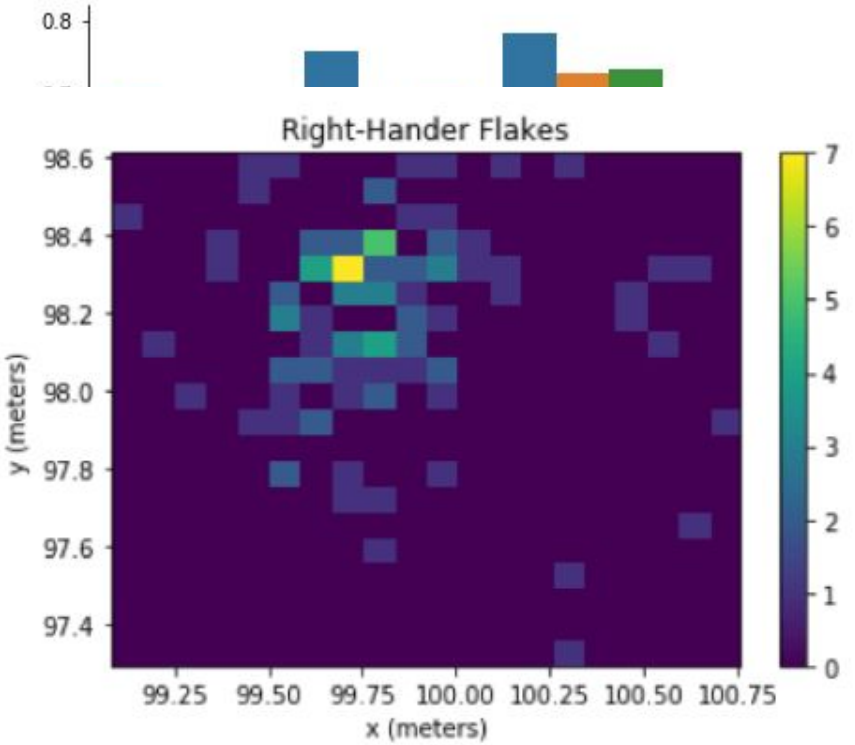
Why should you learn to code?

- Saving time
- Doing boring things for you
- Soft skills:
  - Problem solving
  - Modular / structured thinking
  - Creativity
- Jobs
- Open Science / reproducibility
- Look into ‘black box’
- Doing cool stuff…

What is coding used for in archaeology?

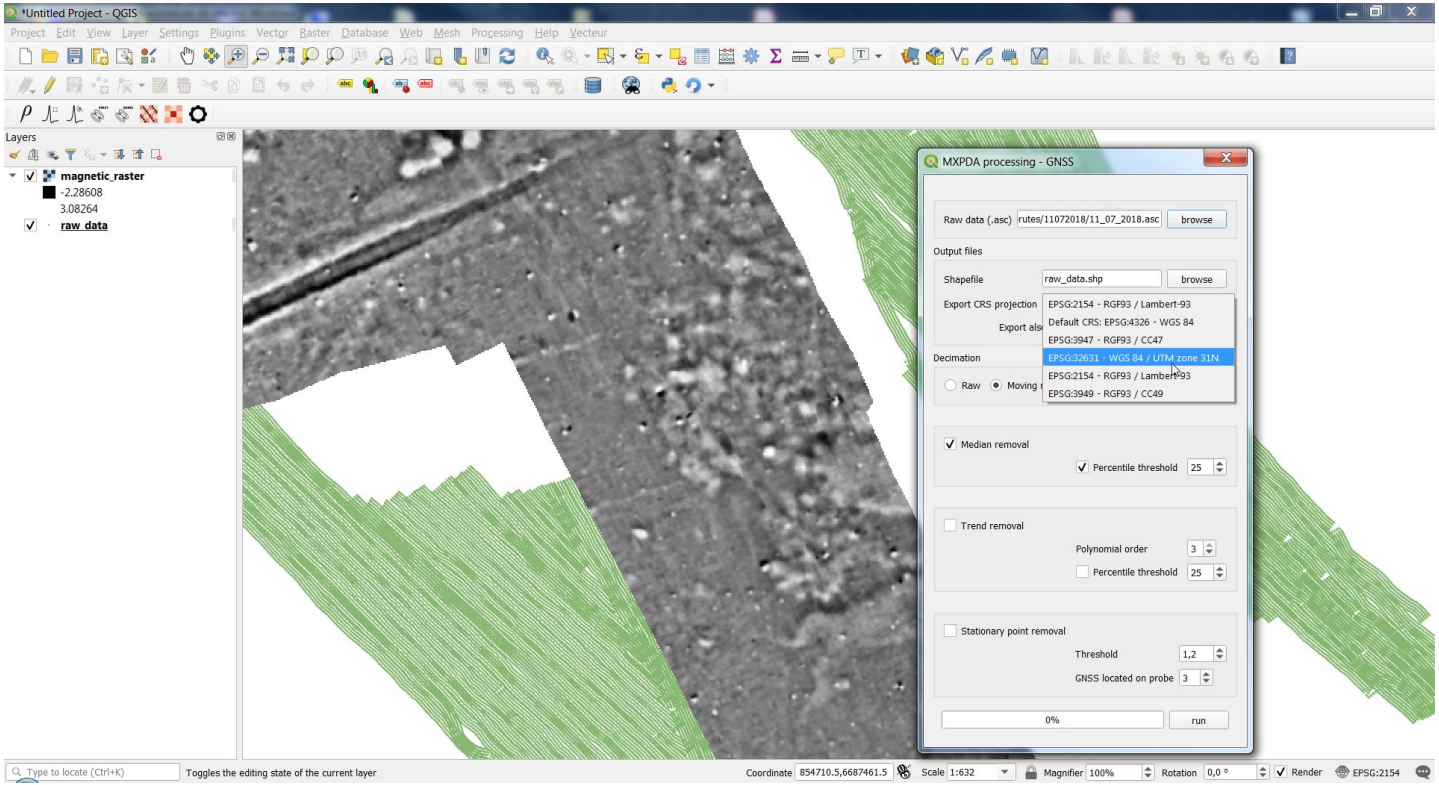
Statistical analysis and graphs





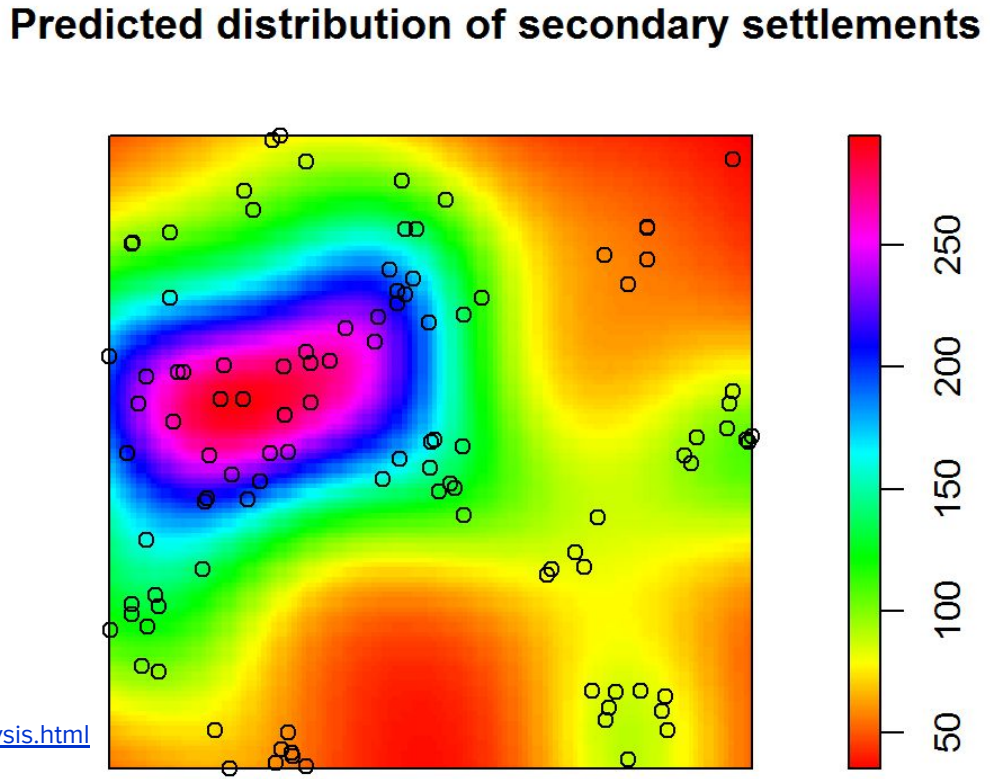
What is coding used for in archaeology?

QGIS integration



What is coding used for in archaeology?

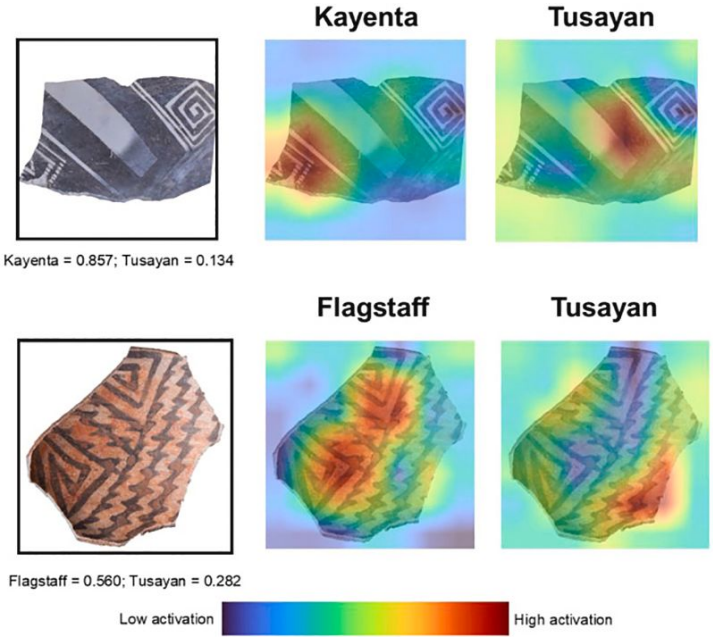
Spatial analysis /
Predictive modeling

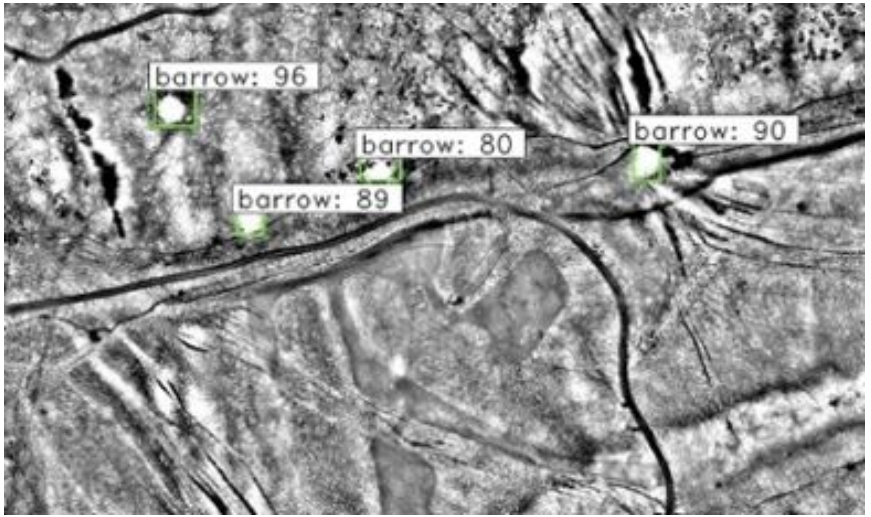


https://benmarwick.github.io/How-To-Do-Archaeological-Science-Using-R/basic-spatial-analysis-in-r-point-pattern-analysis.html

What is coding used for in archaeology?

Machine Learning



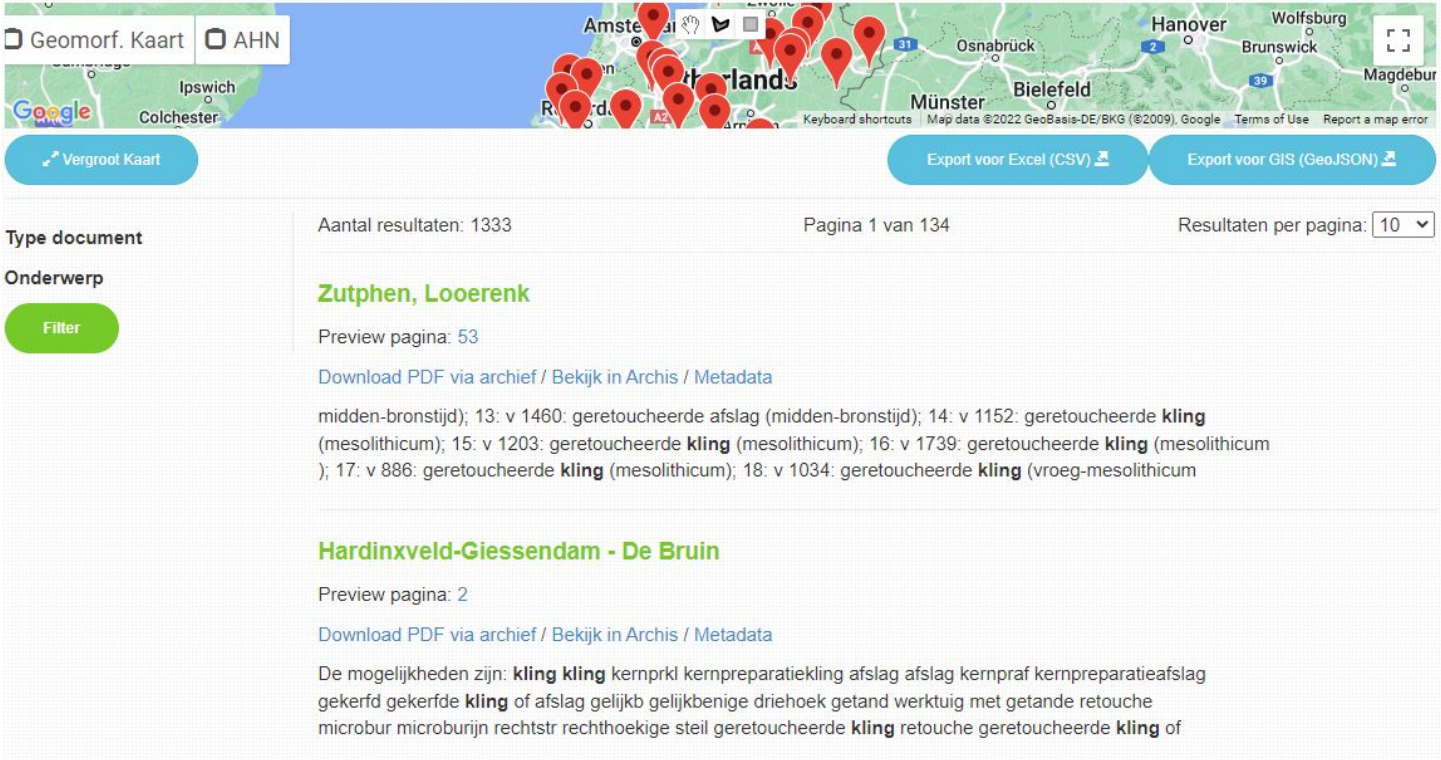


What is coding used for in archaeology?

Citizen Science

What is coding used for in archaeology?

Building websites



What is Python?

- Programming language
- Used to build websites, software, automation, do data analysis
- General purpose: many uses!
- Beginner-friendly: relatively easy to learn
- Popular: 2nd most used (JavaScript is 1st)
- Due to this: many resources

Syntax

- The ‘grammar’ of a programming language
- Rules which your code needs to follow (otherwise: error!)
- Code editors highlight different elements

```
# assign a variable: save the number 42
number_of_sherds = 42
```

```
print(number_of_sherds)
```

```
42
```

Variables

- Have a name
- Contain something (called the value)

Looks like:

```
pots = 100
flint = 50
site = 'Oss'
periods = ['Bronze age','Iron age']
```

Printing!

- Showing the value of a variable

```
pots = 100
print(pots)
```

→ 100

Variables

- Can be combined

```
pots = 100
flint = 50
total_artefacts = pots + flint

print(total_artefacts)
```

Comments

- To explain what's going on!

```
# assign a variable: save number of pots
pots = 100

# print number of pots
print(pots)

flint = 50 # number of flint artefacts
```

Comments - tips

- Comment on everything that's not immediately clear
- Comment like you're explaining the code to someone else
  - (sometimes that 'someone else' is you, 4 months later!)
- Extra work, but saves time when you revisit your code

Variable types

```
# integer (number, no decimals)
pots = 100

# float (number with decimals)
axe_weight = 24.56

# string (a piece of text)
site = 'Oss'

# boolean (True or False, yes or no, 1 or 0, on or off)
gold_found = True
```

Variable types - more types!

```
# list
trench_numbers = [1, 2, 3]

# dictionary
archaeologist = {'name': 'Indiana Jones', 'age': 42}

# data frame
artefacts = pd.read_csv('artefacts.csv')
```

Combining Variables

Can generally only combine same type, otherwise: error

```
pots = 100 # integer
flint = 50 # integer
total_artefacts = pots + flint # this works!

site = 'Oss' # string
site_pots = site + pots # this doesn't work!
```

Naming Variables

- You can use (almost) any name for a variable
- Choose one that is indicative of the value(s) it will hold
- Make it as easy as possible to understand!

```
# good variable name
spearhead_max_width = 56.98

# bad variable names
smw  = 56.98
a = “flint”
```

Naming Variables

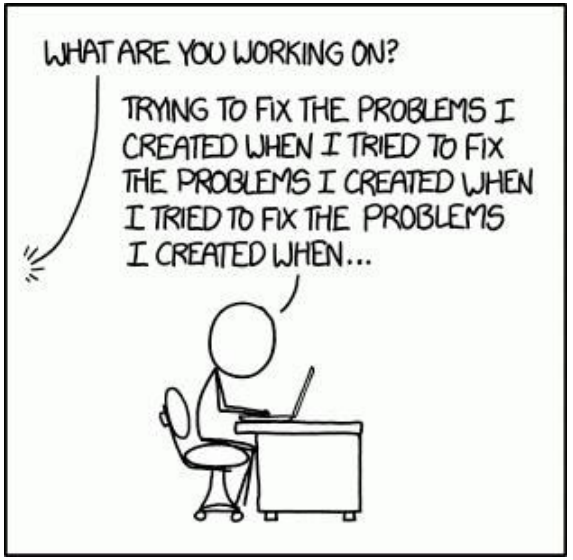
Different ways to use multiple words in name:

- `snake_case = “slithery”`
- `kebab-case = “tasty”`
- `camelCase = “humpy”`
- `PascalCase = “pascal-y?”`

Choose one you like, but be consistent where possible

Debugging

- Figuring out error messages
- Fixing the problem(s)

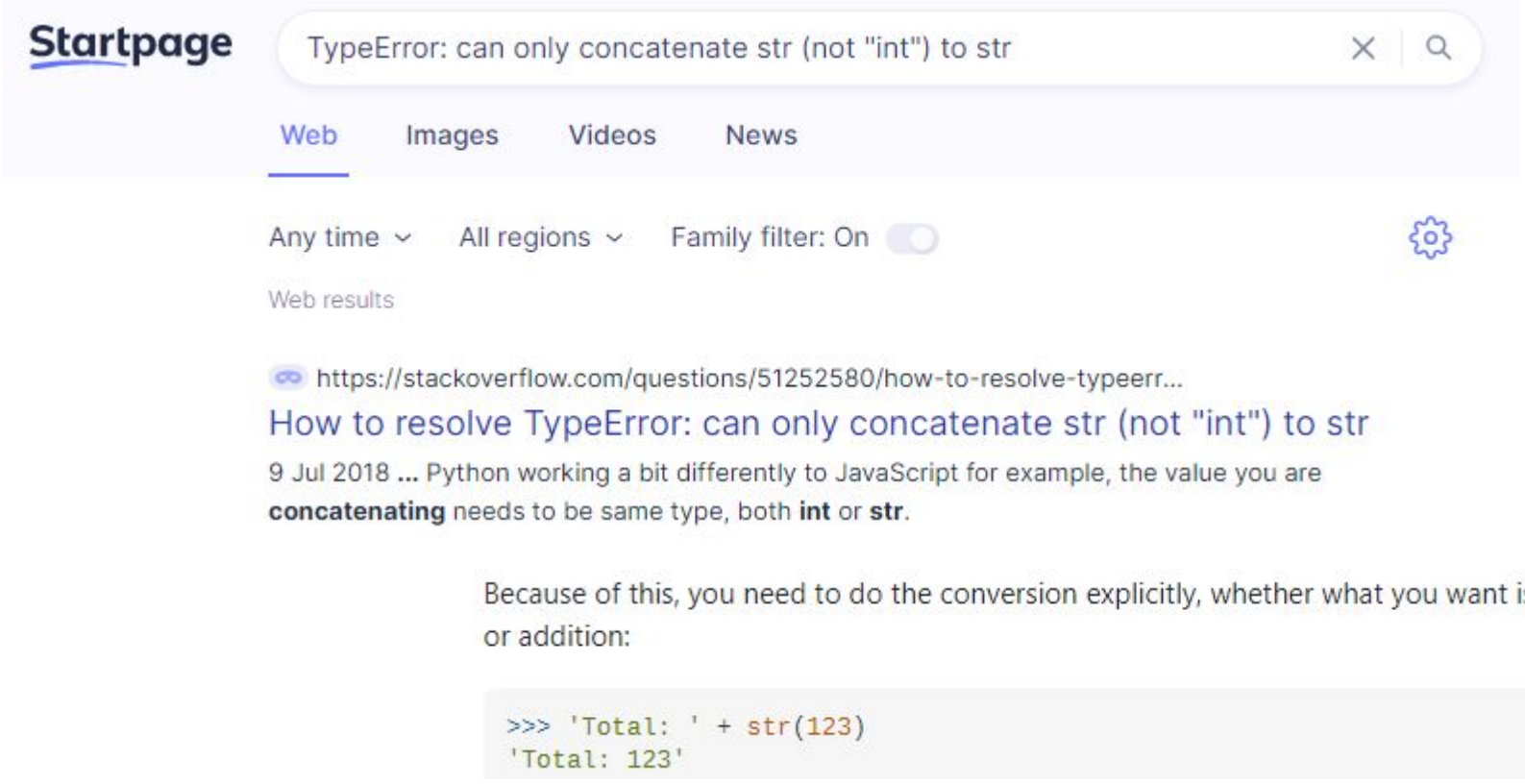



```
pots = 100
site = “Oss”

site_pots = site + pots

TypeError: can only concatenate str (not "int")
to str
```

Debugging

- Use Google! (or another search engine)
- Copy and paste error message
- And/or describe what you are trying to do
- Look on stackoverflow.com

Debugging


Startpage
TypeError: can only concatenate str (not "int") to str
Web
Images
Videos
News
Any time
All regions
Family filter: On
Web results
https://stackoverflow.com/questions/51252580/how-to-resolve-typeerr...
How to resolve TypeError: can only concatenate str (not "int") to str
9 Jul 2018 ... Python working a bit differently to JavaScript for example, the value you are concatenating needs to be same type, both int or str.
Because of this, you need to do the conversion explicitly, whether what you want is concatenation or addition:
>>> 'Total: ' + str(123)
'Total: 123'

Debugging

```
site_pots = site + pots

TypeError: can only concatenate str (not "int")
to str

site_pots = site + str(pots) # outputs 'Oss100'
```

If Statements

If statements check if a condition is true, and if so, executes a bit of code

Looks like:

```
gold_found = True
if gold_found:
    print('We are rich!')
```

Always end 'if' with colon

indent!

Or / And

Used to evaluate multiple conditions:

```
bones_found = True
preservation = 'good'

if bones_found and preservation == 'good':
   print('Make osteology report)
```

('=': assignment, '==': is it equal to x)

Else Statements

Looks like:

```
gold_found = True

if gold_found:
   print('We are rich!')
else:
   print('Back to work tomorrow…')
```

Github

- A place to store your code online
- Kind of like iCloud / Google Drive for code
- Share code with others
- Collaborate on code with others
- For now: just a place to get your exercises / assignments

After this lecture:

- You can conceptually describe programming
- You know why programming can be useful
- You can list a couple of applications of code in archaeology
- You know what the following concepts are:
  - Variables
  - Comments
  - The `print()` command
  - If / else statements
- You know how to open Jupyter Notebooks in Anaconda (after the exercise)

Questions?

- **Any questions about any of the subjects?**

- Contact me at
  - a.brandsen@arch.leidenuniv.nl

Slides are available on Brightspace

Minor “AI in Society”

- Part of SAILS
- Starting in September
- Contact Francine Dechesne
  - f.dechesne@law.leidenuniv.nl

Exercises

github.com/alexbrandsen/Introduction-to-Programming-for-Archaeologists

- Download Anaconda (https://www.anaconda.com/products/distribution)
- Install Anaconda
- Create folder to hold all your scripts
- Download code from Github, save in your Scripts folder
- Start Jupyter Notebook from Anaconda
- Navigate to the exercise folder, select relevant Notebook
  - Read explanations
  - Run cells by pressing ‘play’ button, or press SHIFT+ENTER
  - Edit code in cells where needed
  - Do exercises (marked with `## EXERCISE ##`)