

PODSTAWY PROGRAMOWANIA W JĘZYKU PYTHON

Dzień 1



Agenda

- Część pierwsza – *podstawy programowania* – spotkania 1-8
 - Podstawowe pojęcia, pamięć;
 - Instrukcje warunkowe, pętle;
 - Kolekcje.
- Część druga – *wstęp do obiektowości* – spotkania 9-15
 - Obiektowość;
 - Klasy, hierarchia klas, dziedziczenie;
 - Praktyczne wykorzystanie.



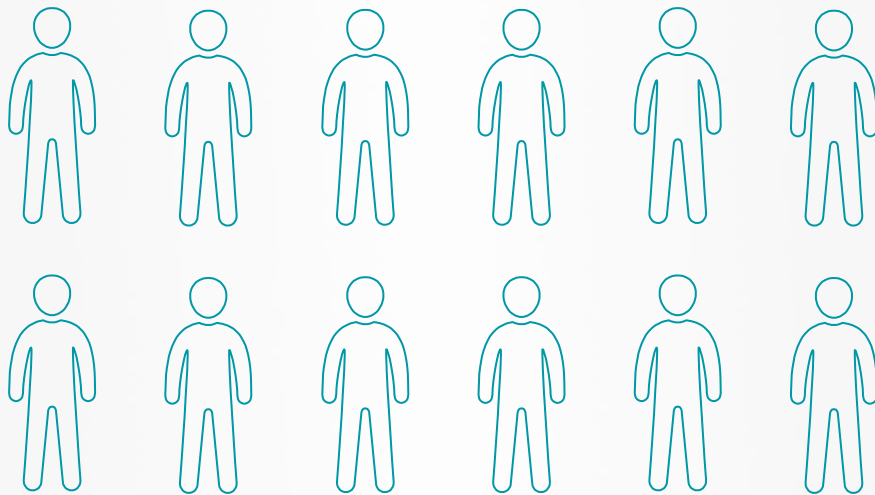
Hello!!

Maciej Kwiatkowski

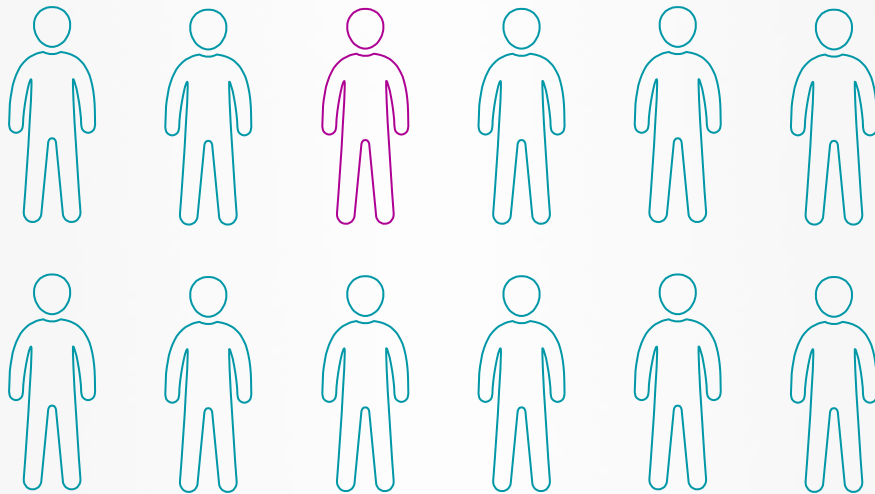
✉ maciej.kwiatkowski92@gmail.com

🐙 <https://github.com/sucharinio>

Poznajmy się



Poznajmy się



TWOJA DROGA

Nie liczy się to, jak daleko
będziesz stosunku do innych,
ale to, jak daleko znajdziesz się
za 7 tygodni, w stosunku do
siebie
z dnia dzisiejszego.

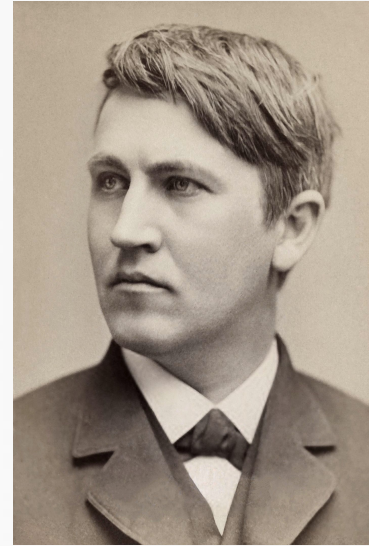
Cel kursu

- Fundamentalne pojęcia
- Analiza problemów
- Dobre praktyki
- Składnia języka



P.U.S.Z.

- **P**róbuj
- **U**ważnie czytaj
- **S**zukaj
- **Z**adawaj pytania



*I have not failed.
I've just found 10.000 ways
that won't work*

T. A. Edison

Zasoby

- **Google**

- Dokumentacja Python:

<https://docs.python.org/3/>

- StackOverflow:

<https://stackoverflow.com/>

- 4programmers:

<https://4programmers.net/Forum/Python>

- GitHub:

<https://github.com/infohareacademy/python6-warszawa>

Agenda - dzień 1

- Systemy liczbowe i kodowanie
- Algorytm
- Pseudokod
- Języki kompilowane vs interpretowane
- Python
- Terminal / wiersz poleceń
- Git / GitHub

Systemy liczbowe

System binarny

0, 1

System dziesiętny

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

System dziesiętny

1 2 3

System dziesiętny

100

1

10

2

1

3

System dziesiętny

$$\begin{array}{ccc} 100 & 10 & 1 \\ \mathbf{1} & \mathbf{2} & \mathbf{3} \\ 100 \times 1 & + 10 \times 2 & + 1 \times 3 \end{array}$$

System binary

4 2 1

0 0 0

System binary

4 2 1
0 0 1

System binary

4 2 1
0 1 0

System binary

4 2 1
0 1 1

Kod ASCII

A	B	C	D	E	F	G	H	I	J	K	L	M
65	66	67	68	69	70	71	72	73	74	75	76	77
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
78	79	80	81	82	83	84	85	86	87	88	89	90

Kod ASCII

i

105

S

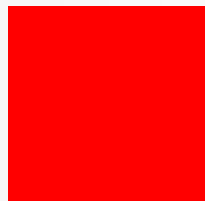
83

A

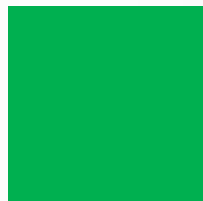
65

Kodowanie koloru

R



G



B

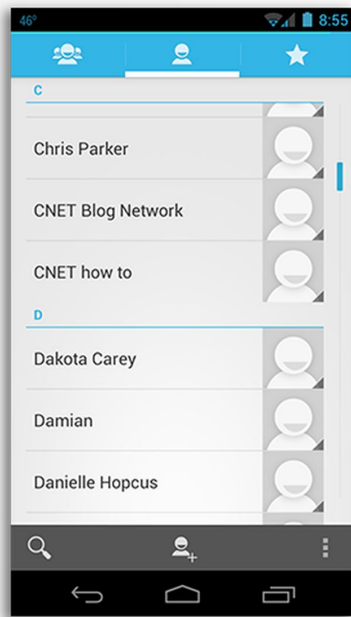


Algorytm

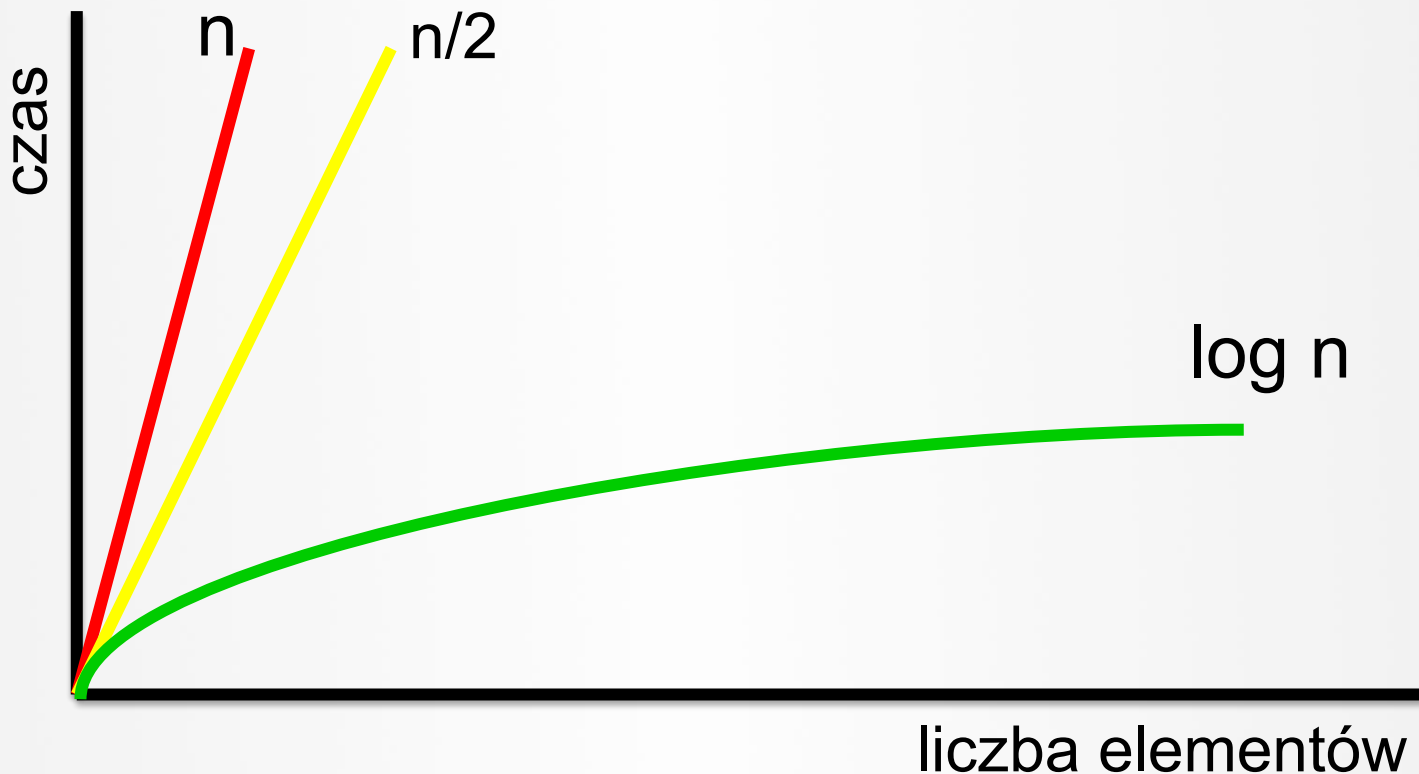
Algorytm

Skończony zestaw instrukcji potrzebnych do wykonania zadania.

Algorytm



Złożoność obliczeniowa (czasowa)



Pseudokod

Pseudokod

1. Weź książkę telefoniczną.
2. Otwórz książkę na środku.
3. Sprawdź nazwiska na otwartych stronach.
4. Jeśli "Wojtkowiak" jest wśród osób:
 - a. Zadzwoń do niego
5. W przeciwnym razie jeśli "Wojtkowiak" jest wcześniej w książce:
 - a. Otwórz lewą połowę po środku.
 - b. Idź do kroku 3.
6. W przeciwnym razie jeśli "Wojtkowiak" jest później w książce:
 - a. Otwórz prawą połowę po środku.
 - b. idź do kroku 3.
7. W przeciwnym razie:
8. poddaj się.

Polecenia

1. **Weź książkę telefoniczną.**
2. **Otwórz książkę na środku.**
3. **Sprawdź nazwiska na otwartych stronach.**
4. Jeśli "Wojtkowiak" jest wśród osób:
 - a. Zadzwoń do niego
5. W przeciwnym razie jeśli "Wojtkowiak" jest wcześniej w książce:
 - a. **Otwórz lewą połowę po środku.**
 - b. Idź do kroku 3.
6. W przeciwnym razie jeśli "Wojtkowiak" jest później w książce:
 - a. **Otwórz prawą połowę po środku.**
 - b. idź do kroku 3.
7. W przeciwnym razie:
8. poddaj się.

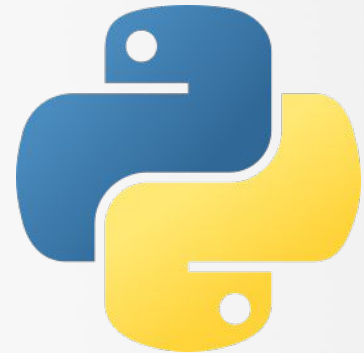
Instrukcje warunkowe

1. Weź książkę telefoniczną.
2. Otwórz książkę na środku.
3. Sprawdź nazwiska na otwartych stronach.
4. **Jeśli "Wojtkowiak" jest wśród osób:**
 - a. Zadzwoń do niego
5. **W przeciwnym razie jeśli " Wojtkowiak" jest wcześniej w książce:**
 - a. Otwórz lewą połowę po środku.
 - b. Idź do kroku 3.
6. **W przeciwnym razie jeśli " Wojtkowiak" jest później w książce:**
 - a. Otwórz prawą połowę po środku.
 - b. idź do kroku 3.
7. **W przeciwnym razie:**
8. poddaj się.

Pętle

1. Weź książkę telefoniczną.
2. Otwórz książkę na środku.
3. Sprawdź nazwiska na otwartych stronach.
4. Jeśli "Wojtkowiak" jest wśród osób:
 - a. Zadzwoń do niego
5. W przeciwnym razie jeśli "Wojtkowiak" jest wcześniej w książce:
 - a. Otwórz lewą połowę po środku.
 - b. Idź do kroku 3.**
6. W przeciwnym razie jeśli "Wojtkowiak" jest później w książce:
 - a. Otwórz prawą połowę po środku.
 - b. idź do kroku 3.**
7. W przeciwnym razie:
- 8. poddaj się.**

PYTHON



Dlaczego Python?

- Prosta składnia, zbliżona do języka naturalnego
- Kod niezależny od systemu operacyjnego (bo interpretowany)
- Wszechstronność (Big Data, AI, Web development, pentesting)
- Popularność

Język kompilowany vs. interpretowany

KOMPILOWANE

- Cały program jest kompilowany;
- Z reguły szybszy;
- Poprawka błędu wymaga ponownej kompilacji;
- Dużo klamer `{}` i średników `..... ;`.

INTERPRETOWANE

- Interpretowana jest linijka po linijce;
- Z reguły wolniejszy;
- Prostsza składnia;
- Łatwiejszy w obsłudze.

Tworzenie kodu

- Interpreter
- Edytor kodu:
 - zwykły edytor tekstu (pliki z rozszerzeniem *.py)
 - IDE (*Integrated Development Environment*):
 - podpowiedzi,
 - kolorowanie składni,
 - debugger,
 - automatyzacja uruchamiania (kodu, testów),
 - wtyczki (repozytorium kodu, klient bazy danych)

Interaktywna konsola

- Wykonuje kod linijka po linijce.
- Bardzo przydatna do testowania rozwiązań “na boku” lub sprawdzenia działania/wywołania funkcji.

Interaktywna konsola

- **uruchamianie**

w wierszu poleceń wpisujemy `python`
lub `python3.6` i naciskamy enter

- **wychodzenie**

wpisujemy `exit()` i naciskamy enter

KONSOLA PYTHON

```
C:\Users\ArkadioG>python
Python 3.6.2 (v3.6.2:5fd33b5, Jul 8 2017, 04:57:36) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

Cheatsheet

- Mac OS:

<https://bit.ly/mac-ter>

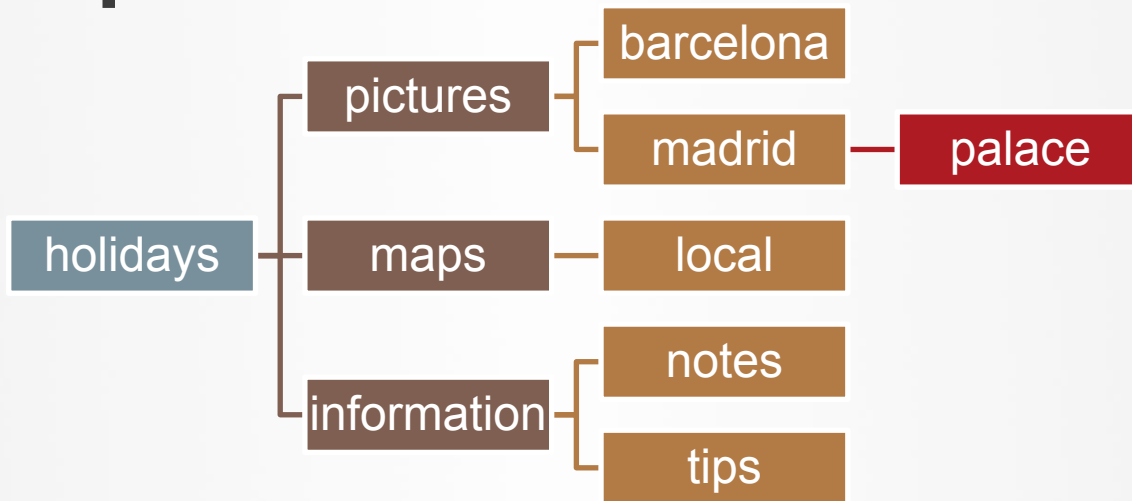
- Windows:

<http://bit.ly/wincommands>

Wiersz poleceń

- `/?` `-help` `-- help` `--h` - pomoc
- `mkdir` / `md` – tworzenie folderu
- `rmdir` / `rd` – usuwanie folderu
- `move` – przenoszenie zmiana nazwy
- `dir` / `ls` – wyświetlenie zawartości folderu
- `cd` – przejście do innej lokalizacji
- `pwd` – obecny folder (linux, osx)
- `type` / `cat` – wyświetlenie zawartości pliku
- `touch` – utworzenie pliku
- `echo Lalalala > plik.txt`

Wiersz poleceń



1. Utwórz strukturę folderów
2. Zmień nazwę folderu *barcelona* na *valencia*
3. Usuń folder *information*
4. Będąc w folderze *pictures* dodaj folder *barcelona* do folderu *maps*
5. Będąc w folderze *maps* wyświetl zawartość folderu *madrid*

Git & GitHub

Git

Autor: Linus Torvalds

Rozproszony system wersjonowania plików.

Każdy developer może pracować nad częścią kodu.

Aplikacja/usługa może mieć kilka wersji kodu.

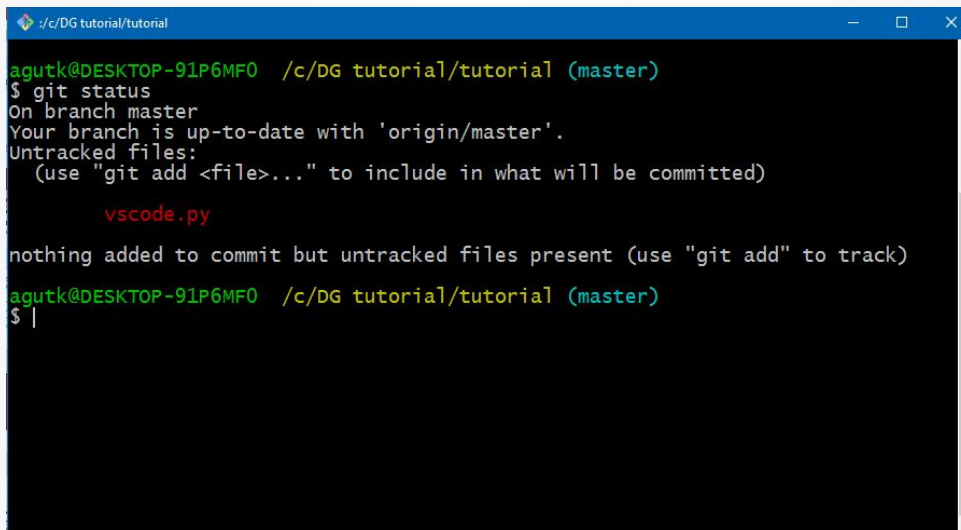
Umożliwia cofanie zmian, łączenie podzielonego kodu.

<https://git-scm.com/>



Git - konsola

wydajemy polecenia konsolą
(wymagana instalacja git na Windows)

A screenshot of a Windows command prompt window titled "C:/DG tutorial/tutorial". The prompt shows a user named "agutk@DESKTOP-91P6MF0" in the directory "C:/DG tutorial/tutorial" on the "master" branch. The user enters the command "git status". The output indicates the branch is up-to-date with 'origin/master' and lists "vscode.py" as an untracked file. The prompt then shows the user entering a second command, which is partially visible as "git |".

```
agutk@DESKTOP-91P6MF0 /c/DG tutorial/tutorial (master)
$ git status
On branch master
Your branch is up-to-date with 'origin/master'.
Untracked files:
  (use "git add <file>..." to include in what will be committed)

        vscode.py

nothing added to commit but untracked files present (use "git add" to track)
agutk@DESKTOP-91P6MF0 /c/DG tutorial/tutorial (master)
$ |
```

<https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf>

GitHub

Dokumentacja, video, książka Git Pro

<https://git-scm.com/doc>

Książka w wersji polskiej (1 edycja):

<https://git-scm.com/book/pl/v1>

Online tutorial z komend:

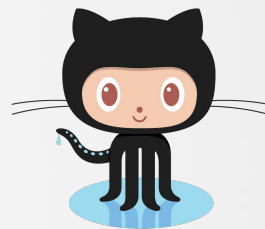
<https://learngitbranching.js.org/>



GitHub

<https://github.com>

- repozytoria kodu w chmurze
 - bezpłatne publiczne repozytoria
 - najpopularniejsze miejsce z projektami opensource
 - must-have dla programisty
-
- Konkurencja - gitlab, bitbucket - dają bezpłatne prywatne repozytoria



GitHub

- zakładamy konto
- tworzymy repozytorium
- klonujemy repozytorium na swój komputer
- zmieniamy kod
- commitujemy zmiany (zapisujemy do lokalnego repo)
- synchronizujemy z github
- Polecenia w wierszu komend

- Pliki ignorowane:

<https://help.github.com/articles/ignoring-files/>

GitHub

- Wprowadzenie:
<https://guides.github.com/introduction/git-handbook/>
- Cheatsheet:
<https://education.github.com/git-cheat-sheet-education.pdf>
- Pomoc: <https://help.github.com/>
- Video:
 - [https://www.youtube.com/githubguides,](https://www.youtube.com/githubguides)
 - <https://youtu.be/HVsySz-h9r4>

GitHub

■ Klienci desktopowi:

- <https://desktop.github.com/>
- <https://git-scm.com/downloads/guis>

The screenshot shows the GitHub desktop application interface. At the top, there's a header bar with a search icon, a link to 'muan/sort-search-results', and buttons for 'Pull request', 'No uncommitted changes', and a settings gear. Below the header, there's a dark bar with 'Update from master' and 'View branch' buttons. A 'Sync' button is on the right. The main area displays a commit history graph with a blue dot indicating the current commit. Below the graph, there's a list of recent commits:

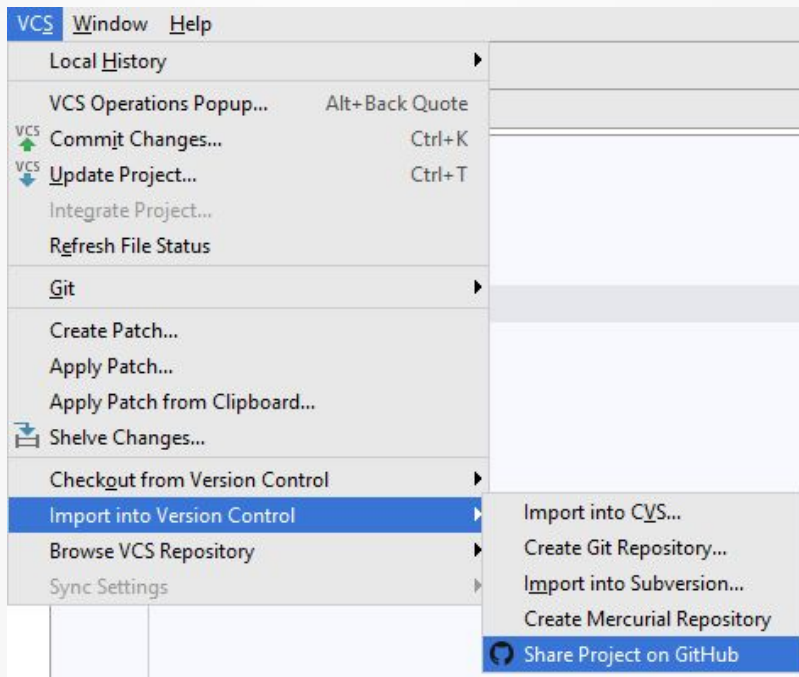
- Use a loop
9 hours ago by Ben Ogle
- Use .localeCompare instead of >...
9 hours ago by Mu-An Chiou
- Default out of range so results won't...
9 hours ago by Mu-An Chiou

On the right side, there's a commit detail for 'Prevent rendering elements indefinitely' by Mu-An Chiou (f9848ba). It includes a 'GitHub' link, 'Revert' button, and 'Collapse all' button. The commit message reads: 'By making a mark when screen is filled with results and only add more results to above the fold if the insertPoint is above the fold.'

On the left side, there's a sidebar with a 'Filter repositories' input field and a list of repositories: atom, electron, find-and-replace (selected), libgit2, libgit2sharp, mojibar, and octokit.net.

GitHub & PyCharm

- tworzymy projekt
- w ustawieniach łączymy się z github (login, hasło)
- menu VCS:



Wiersz poleceń

- `git clone https://...` - skopiowanie (klonowanie) zdalnego repo na komputer
- `git pull` - pobranie zmian ze zdalnego repo
- `git init` - zainicjowanie nowego lokalnego repozytorium
- `git add` - dodanie plików do wersjonowania
- `git commit` - utworzenie wpisu w repozytorium
- `git push` - wysłanie wpisu na zdalne repo
- `git status` - sprawdzenie stanu repozytorium

Podsumowanie

- Systemy liczbowe i kodowanie
- Algorytm
- Pseudokod
- Języki kompilowane vs interpretowane
- Python
- Terminal / wiersz poleceń
- Git / GitHub



Thanks!!