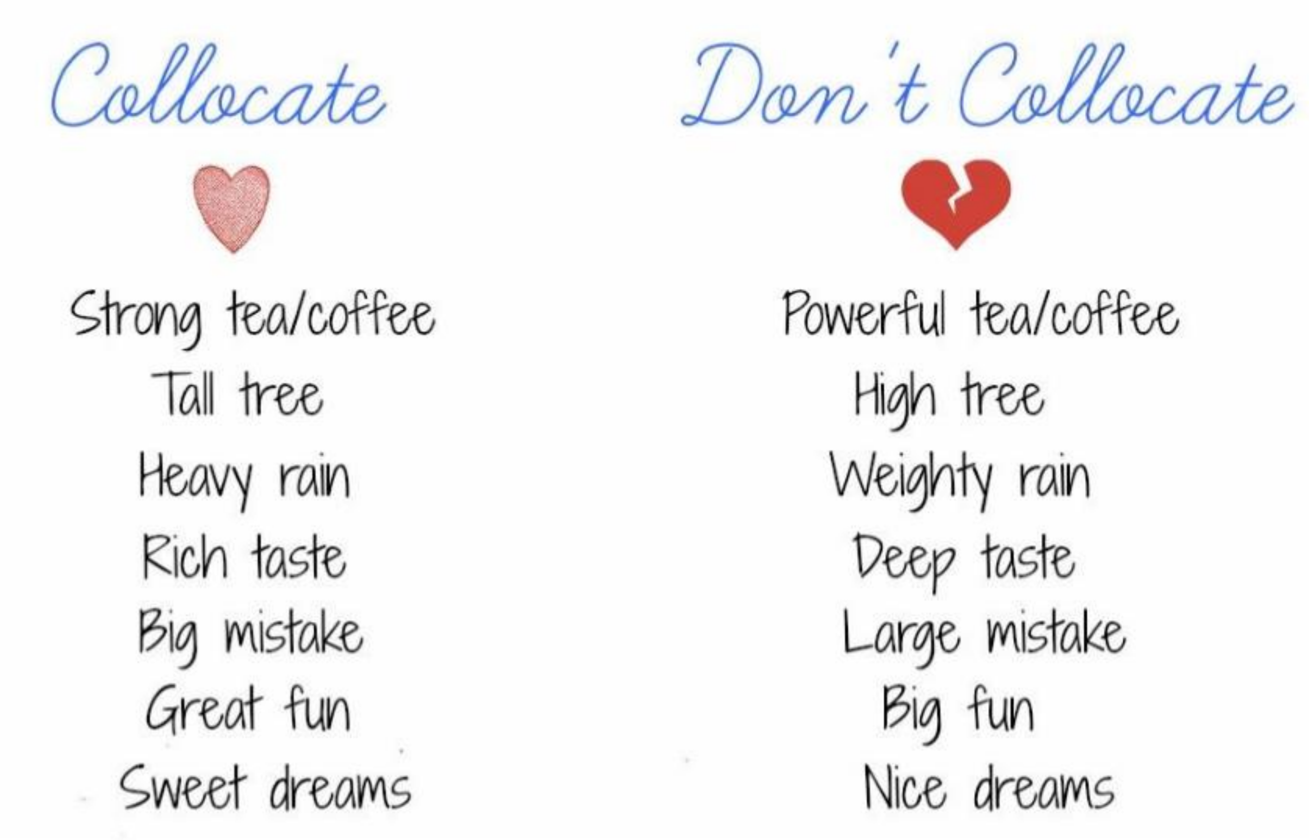
Lyrco

A small lyrics corpora engine

Corpora

- Corpora: Bodies of texts or sum of texts, digital
- Some corpora: The Brown Corpus of Contemporary American English, British National Corpus or Russian National Corpus, The Helsinki Corpus of English texts, etc

- Here however I aim to make a Corpora Engine which displays collocation and concordance. Possibly give you n-grams.
- **Collocation**: ”two or more words that often go together”
- Can be seen in statistical manner (most frequently used pairs)



Collocate ♥	Don't Collocate 💔
Strong tea/coffee	Powerful tea/coffee
Tall tree	High tree
Heavy rain	Weighty rain
Rich taste	Deep taste
Big mistake	Large mistake
Great fun	Big fun
Sweet dreams	Nice dreams

Inspiration

- Course ”Corpora and Language Technology” https://opas.peppi.utu.fi/en/course/KKLT0040/8502
- https://www.english-corpora.org/
- I believe I’m more like Rowling (JK)
- There could be a front-end sometime if the engine becomes even remotely usable

Has no plan just codes and hopes for the best. Somehow makes it work at the end of the day.

Copy pastes solid code using ton of libraries. Maybe not the fastest but gets the job done well.

Creates own language, frameworks, libraries. Top notch code for everything. Nobody knows why he does it this way, but his work is impressive.

Motivation

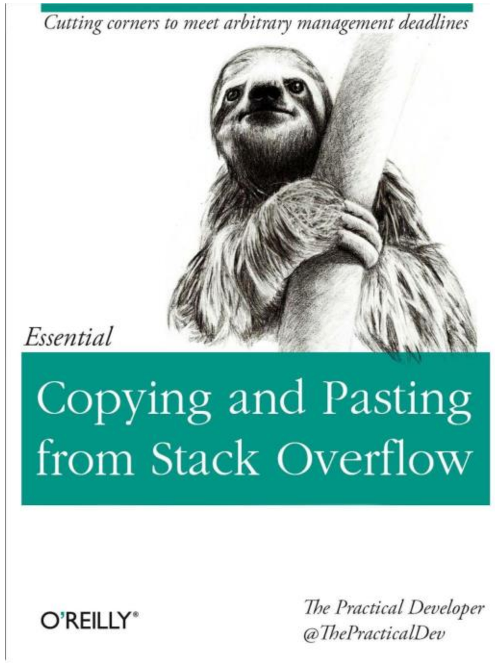
- GitHub repository is empty -> No need to even apply for jobs
- Haven’t made any python projects
- Haven’t made any language technology related projects
- Must do more projects
- Any projects
- Programming skill level is somewhere near first-bootcamp-week level

Me: why does my code take so much to run

My code:

What will be done

- Study the structure of similar projects
- Ponder what way the collocation will be fastly checked
- Create Github repository:
- [https://github.com/avhalo/lyrco]
- Create DesignDocuments
  and push them to docs folder
- Do the ever so odd Coding



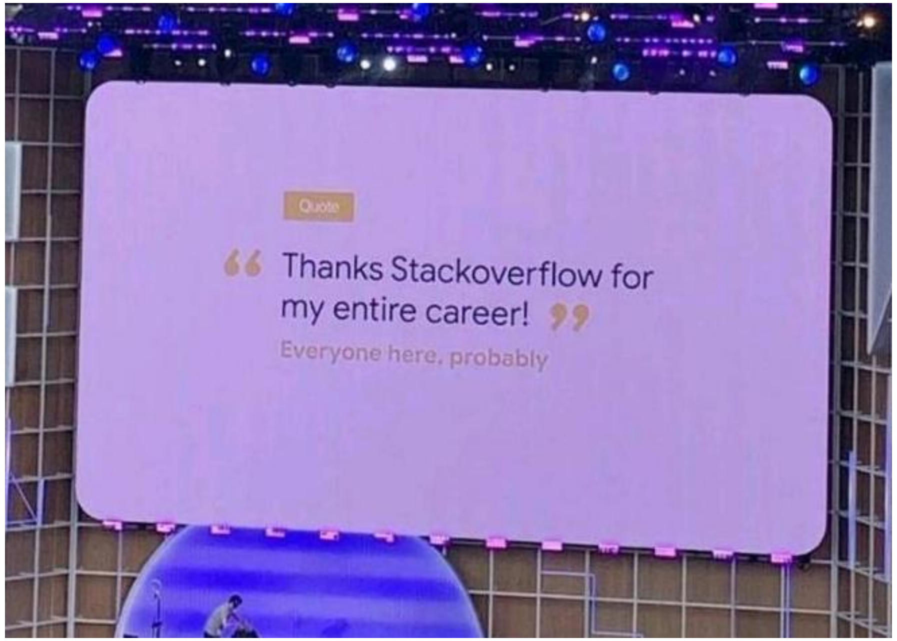
Cutting corners to meet arbitrary management deadlines

Essential

Copying and Pasting from Stack Overflow

O'REILLY®

The Practical Developer
@ThePracticalDev

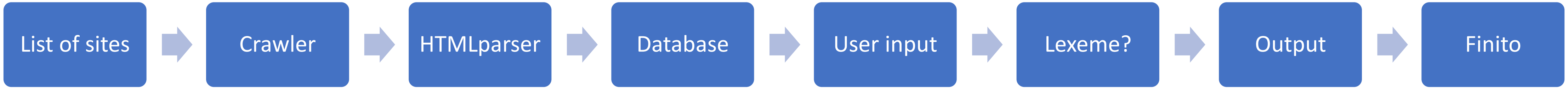


“Thanks Stackoverflow for my entire career!”

Everyone here, probably

Structure

(draft, see docs folder)



List of sites

Crawler

HTMLparser

Database

User input

Lexeme?

Output

Finito

Crawler structure

- Probably a sleeper of 1000ms
- Watch for not DoS:ing anything and be IP-banned
- Libraries will be used. Haven’t checked them out yet
- Wget?
- Will use a start of 10 sites and maybe aprox 10 000 songs for beginning the task. Most of them probably are in English since there is not that big of a selection in Finnish
- Example site: musixmatch.com genius.com
  and so on there are tons of these.
- I will use crawler in another project in the future so it better be good

Time management

Instructed 67h time (hopefully):

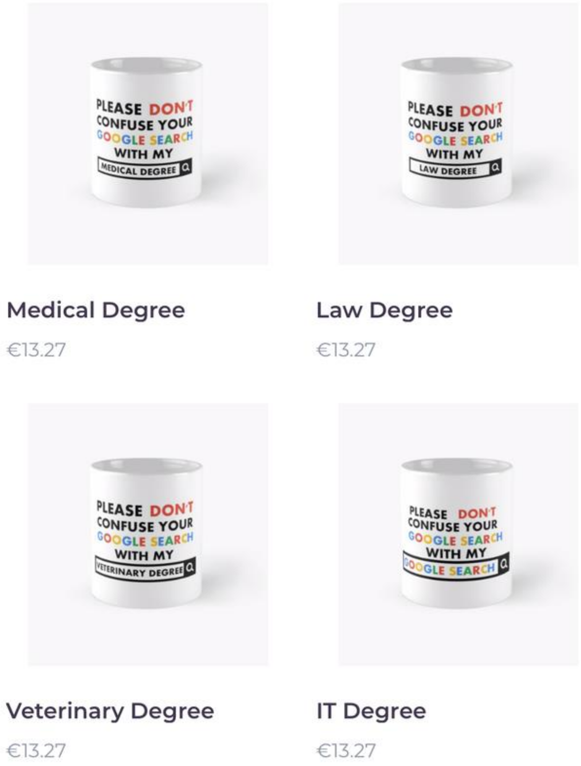
- 10h research
- 10h coding
- 37h figuring out why it doesn't work
- 10h optimizing

Start research from 15.3. Coding Starts the same day.

Milestone I at 29.3: 50% done (crawler)

Milestone II 12.4: All done as a draft -> optimize

Milestone III 26.4: 100% done


PLEASE DON'T CONFUSE YOUR GOOGLE SEARCH WITH MY MEDICAL DEGREE
PLEASE DON'T CONFUSE YOUR GOOGLE SEARCH WITH MY LAW DEGREE
PLEASE DON'T CONFUSE YOUR GOOGLE SEARCH WITH MY VETERINARY DEGREE
PLEASE DON'T CONFUSE YOUR GOOGLE SEARCH WITH MY GOOGLE SEARCH

Medical Degree
€13.27

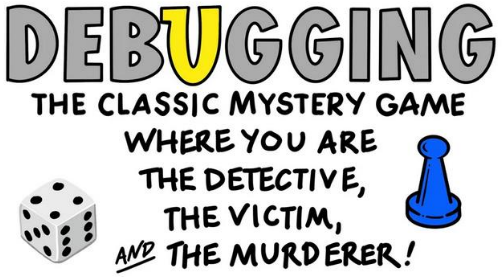
Law Degree
€13.27

Veterinary Degree
€13.27

IT Degree
€13.27

“Only half of programming is coding. The other 90% is debugging.”

Anonymous

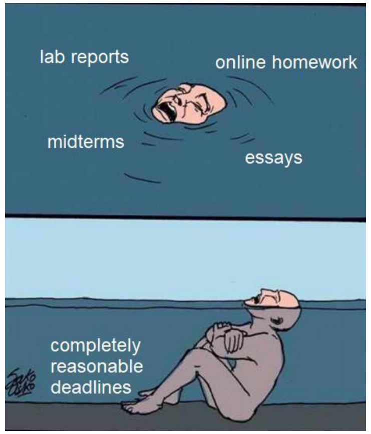


How do I A ? - bork2121

You do B. - thedudehimself

But that doesnt do A. - bork2121

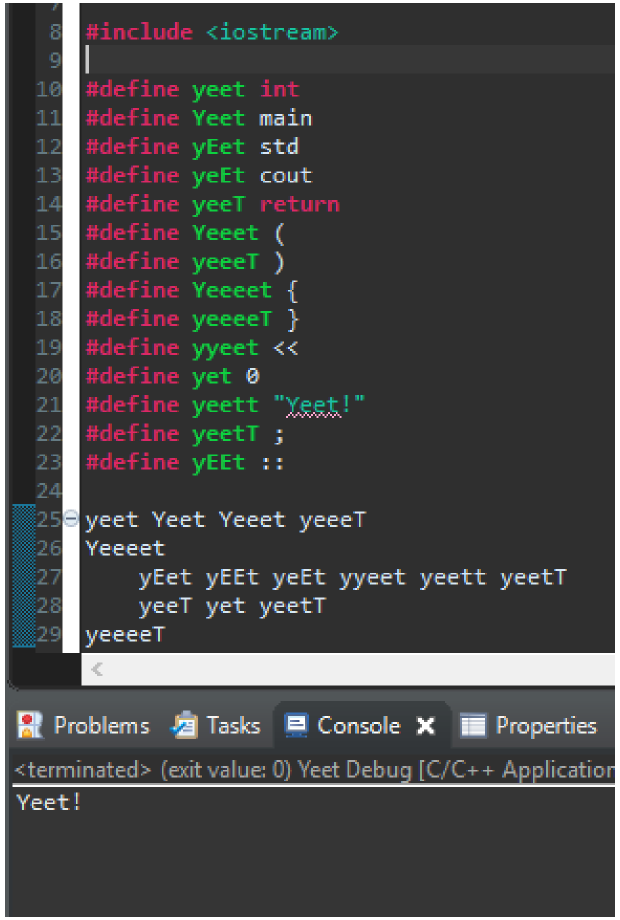
Yeah nobody does A. - thedudehimself



Clean code?

I stumbled upon this:
PEP 8
https://www.python.org/dev/peps/pep-0008/

I’ll try to follow it.
Exception: There is no way I am going to use
Four spaces instead of tabs. I will not convert tabs to
4 spaces. They will remain as ”/t”



```
#include <iostream>

#define yeet int
#define Yeet main
#define yEet std
#define yeEt cout
#define yeeT return
#define Yeeet (
#define yeeeT )
#define Yeeeet {
#define yeeeeT }
#define yyeet <<
#define yet 0
#define yeett "Yeet!"
#define yeetT ;
#define yEEt ::

yeet Yeet Yeeet yeeeT
Yeeeet
	yEet yEEt yeEt yyeet yeett yeetT
	yeeT yet yeetT
yeeeeT
```

```
<terminated> (exit value: 0) Yeet Debug [C/C++ Application
Yeet!
```

ps

- Made a list of possible projects which I then named impossible-list
- Can be checked out here

https://github.com/avhalo/the-idea-base-for-everyone

The suggestions you gave at the first lecture were good. I'll probably steal them tbh