OHJE MURREPIIRTEIDEN HAKUUN PAIKKAKUNTAKOHTAISISTA KORPUKSISTA

paikkakunnan annotoitu korpus ajetaan annorivit.py läpi, jonka outputtina saadaan tällä skriptillä käsiteltävä tiedosto

```
zcat Kanta-Häme_kaikki.gz | python3 annorivit.py | python3 d_vaihtelu.py
```

etsittavat_sanat = lemmat, joita etsitään korpuksesta

```
etsittavat_sanat = ["kahtoa", "katsoa", "kattoa", "itte", "itse", "ihte", "ruotti", "ruotsi", "ruohti", ]
```

alustetaan muuttujat

```
tt_vaihtelu = []
ts_vaihtelu = []
t_vaihtelu = []
ht_vaihtelu = []
muu = []

laskuri = Counter()
```

siistitään conllu-annotoinnit luettavaan muotoon *(tämä löytyy myös annorivit.py, voisi siistiä paremmin)*

```
def annojen_erittely(line):
    pattern = r"'.*?'"
    annot = re.findall(pattern, line)
    annot = [i.strip("'") for i in annot]
    return annot
```

luokitellaan etsittava_sana muuttujassa määritellyt lemmat, if lauseissa määritellään ehdot kullekin “luokalle”

```
for line in sys.stdin:
    annorivi = annojen_erittely(line)
    if len(annorivi) >= 2:
        etsittava_sana = annorivi[2].lower()
        if etsittava_sana in etsittavat_sanat:
            if "ts" in annorivi[1].lower():
                ts_vaihtelu.append(annorivi)
            elif "tt" in annorivi[1].lower():
                tt_vaihtelu.append(annorivi)
            elif "ht" in annorivi[1].lower():
                ht_vaihtelu.append(annorivi)
            elif "t" in annorivi[1].lower() and "h" not in annorivi[1].lower() and "ts" not in annorivi[1].lower
                t_vaihtelu.append(annorivi)
            else:
                muu.append(annorivi)
```

vertailun vuoksi lasketaan esiintymät

```
# lasketaan kaikki esiintymät ja osuudet
total_matches = len(tt_vaihtelu) + len(ts_vaihtelu) + len(t_vaihtelu) + len(ht_vaihtelu) + len(muu)

if total_matches > 0:
    percentages = {
        "tt_vaihtelu": len(tt_vaihtelu) / total_matches * 100,
        "ts_vaihtelu": len(ts_vaihtelu) / total_matches * 100,
        "t_vaihtelu": len(t_vaihtelu) / total_matches * 100,
        "ht_vaihtelu": len(ht_vaihtelu) / total_matches * 100,
        "muu": len(muu) / total_matches * 100,
    }

    for key, value in percentages.items():
        print(f"{key} osuus esiintymistä: {value:.2f}%")
    print()
```

tarvittaessa tulostetaan esiintymät, tämän voi kuitenkin jättää suorittamatta

```
print("Kaikki tt esiintymät:", len(tt_vaihtelu))
for item in tt_vaihtelu:
    print(item)
print("Kaikki ts esiintymät:", len(ts_vaihtelu))
for item in ts_vaihtelu:
    print(item)
print("Kaikki t+kato esiintymät:", len(t_vaihtelu))
for item in t_vaihtelu:
    print(item)
print("Kaikki ht esiintymät:", len(ht_vaihtelu))
for item in ht_vaihtelu:
    print(item)
print("Muut esiintymät:", len(muu))
for item in muu:
    print(item)
```