Python og NLTK i INF2820, V2015

Bakgrunnskunnskaper fra INF1820

INF1820 - Introduksjon til språk- og kommunikasjonsteknologi er anbefalt forkunnskap for INF2820, og INF2820 bygger på en del av INF1820. Vi vil bl.a. bruke noen av de samme redskapene i de to emnene: NLTK og Python. Studenter som ikke er kjent med disse redskapene, bør sette seg inn i dem så raskt som mulig.

Siden http://www.uio.no/studier/emner/matnat/ifi/INF1820/v14/pensumliste/index.html viser hvilke deler av NLTK som var pensum i INF1820 i fjor. Du finner også flere detaljer om hva som ble dekket i INF1820 på siden http://www.uio.no/studier/emner/matnat/ifi/INF1820/v14/timeplan/index.html under Øvelser. Vi vil bygge på deler av kapittel 1 og 2 i NLTK-boka (men ikke kap. 5). Du anbefales å arbeide deg gjennom følgende

- Kapittel 1 fra NLTK-boka, særlig seksjon seksjon 1.2 og 1.4.
- Gjøre oppgavene fra seksjon 1.8: 1, 2, 3, 8, 9, 10, 13, 16
- Seksjon 2.3 og seksjon 3.2
- Gjøre oppgavene fra seksjon 3.12: 2, 4, 5, 10, 11

Når du arbeider med NLTK bør du sitte ved skjermen med et Python-vindu og selv utføre det som står. Det er det du lærer av. Du vil også tjene på å arbeide mere med programmering i Python, seksjonene 4.1-4.4 I NLTK-boka. På siden med INF1820-pensum finner du også peker til mere stoff om Python,

- How to think like a computer scientist: Learning with Python
- The official Python documentation

Valg av Python-oppsett

Python/Ipython/Idle

Når du sitter på en av IFIs Linux-maskiner kan du gi kommandoen *python*. Du kommer da rett inn i et Python-shell hvor du kan arbeide interaktivt. Python er forskjellig fra Java. I Java skriver vi et program, deretter kompilerer vi det, og så kan vi kjøre det. Python kompilerer ikke programmene men interpreterer dem og legger opp til interaktivt arbeid i utviklingsfasen. Det gjør også NLTK, og vi vil bruke denne arbeidsformen i INF2820.

Alternativt til kommandoen *python* kan du gi en av kommandoene *idle* eller *ipython*. Disse gir deg også et Python-shell, men de gir også en del tilleggsfunksjonalitet som automatisk indentering (der du trenger 4 blanke), og de kan fullføre kommandonavn for deg når du har skrevet første delen. Denne funksjonen er uvurderlig når vi skal arbeide med NLTK. Du oppfordres derfor til ikke å bruke det enkle Python-shellet, men i stedet Idle eller Ipython (eller et annet Python-shell du har erfaring med).

Når det gjelder valg mellom Idle og Ipython er det delvis snakk om smak og behag, og delvis spørsmål om hvilken editor du foretrekker og evt tilgang på IDE. NLTK-boka bruker Idle som standard, og vi vil anbefale Idle med mindre du har klare preferanser for noe annet.

Editor og IDE

Idle kommer integrert med en egen editor for å skrive programmer og noen integrerte debuggingsfunksjoner, til sammen en såkalt IDE. Det gir en enkel arbeidsflyt. Ulempen er at editoren kan være annerledes enn det du er vant til. Den er mer på Windows-form ann på Linux-form.

Mange editorer har modi for redigering av Python-programmer, inkludert Emacs, Vim og Notebook++. Hvis du foretrekker, kan du skrie program i en av disse og så kalle dem fra Python/Ipython/Idle-shellet. Det finnes muligheter for å integrere Python/Ipython i for eksempel Emacs og kjøre det som en prosess der, men vi har ikke funnet noe enkelt standardoppsett for dette. De som har lyst, kan selvsagt eksperimentere med dette. Men for de som vil ha en enkel løsning, er Idle bra nok.