Kryptografi *Oppgave 1, 2019*

2. januar 2019

Den første oppgaven er å konstruere et krypteringssprogram, en «Enigma-maskin».

Denne oppgave er ment som en repetisjon fra høstens Python-introduksjon. Oppgaven trenger derfor ikke noe annet enn det som diskutertes i introduksjonskursen.

– *Programmet* –

Programmet skal

(a) spørre etter input, d.v.s. «tekst», fra tastatur (eller fil).

(b) Denne input skal *krypteres* til «tekst» som presenteres på skjerm (eller skrives til fil).

Programmet skal også kunne

(c) *dekryptere*, d.v.s., ta en kryptert «tekst» (fra tastatur eller fil) og omvandle denne til leselig tekst.

– *Tenk etter først!* –

Før dere setter dere ved datoren er det veldig lurt å tenke gjennom hvordan programmet skal operere: hva skal være input? hva skal være output? skal det finnes valg ved start av programmet? hvilke funksjoner (eller deler) skal programmet inneholde? hvilken algoritm skal jeg bruke å hvordan implementerer jeg den i koden? skal algoritmen være en egen funksjon? e.t.c..

Gjør gjerne et diagram!

Observer at disse tanker ikke trenger å være spesielt detaljerte i første omgang. Men det er veldig lurt å ha en mental bilde (helst på papir!) av programmets struktur og funksjon før man starter. Deretter kommer dere å oppdage at det kommer til å bli (noen ganger store) endringer underveis. Derfor trenger det ikke være mye detaljer i den første strukturskissen.

– *Krypteringsalgoritm (-skjema)* –

Som dere sikkert vet fins det mange måter å kryptere en tekst på, men dere trenger i første omgang bare å velge en veldig enkel måte. Deretter, hvis dere ønsker, kan dere konstruere mer kompliserte krypteringsalgoritmer.

En første enkel variant kunne være et *Ceasar-chiffer*: hver bokstav erstattes med bokstaven «rett etter» (eller med noen annen shift), for

eksempel et a i teksten blir et b, et b blir et c, . . . , og til sist, et å blir et a.

En noe mer komplisert variant kunne være å omvandle hvert ord til et heltall. Krypteringen kunne da være å bruke en (matematisk) funksjon på dette heltall for å få et nytt tall. Tenk da på at det skal være mulig å «gå tilbake», d.v.s., dekryptere. Funksjonen må være *injektiv*.

Dere kan bare bruke deres egen fantasi for å konstruere mer kompliserte krypton. Men som sagt, tenk på at det skal være mulig å dekryptere også.

En ting som kan være lurt å fundere på når dere implimenterer en spesifikk krypteringsidé, er "Hvor vanskelig er det å knekke krypteringsskjemaet?" Slike spørsmål er generelt veldig vanskelig å svare på.

– *Konsept* –

Dere kommer sikkert ha bruk før *bland annet* følgende konsepter.

- `list`
- `array`
- `dictionary`
- `range`, `arange`, `linspace`
- `for __ in __ :`
- `while __ :`
- `if __ :`
- `def __ :`
- `print` og bruket av dennes «metoder»
- typene `int`, `float` og `str`, og i tillegg overgang (eng. conversion) mellom disse
- diverse funksjoner for manipulasjon av strenger (eng. strings).

– *Presentasjon* –

Denne oppgave kommer ikke til å inngå i vurderingsmappen, men det er likevel viktig at dere nøye tenker gjennom hvordan man best presenterer oppgaven. Derfor ønsker jeg at dere sender meg .py-filen og en kort rapport (maksimalt to a4-sider) i .pdf-format. Denne rapport skal inneholde en forklaring av den algoritm dere bruker og en kort beskrivelse av programmet og dets struktur.

Tenk på å bruke mye kommentarer i programfilen som beskriver delene på en god måte for en som ikke selv har skrevet programmet!