Oblig 2

Oppgave 1:

En IF-statement er et statement som sjekker om et spesifikt utsagn er sant, og gjør noe avhengig av resultatet. Som for eksempel:

```
if 2+2 == 4:
    print("to pluss to er lik fire")
```

vil printe «to pluss to er lik fire» i konsollen, mens:

```
if 2+2 == 5:
    print("to pluss to er lik fem")
```

vil ikke printe noe i konsollen.

En ELIF-statement (kort for Else IF) er et statement, som puttes etter et vanlig IF-statement. ELIF vil bare kjøres hvis det første IF-statementet er ikke sant. Som for eksempel:

```
if 2+2 == 5:
    print("to pluss to er lik fem")
elif 2+2 == 4:
    print("to pluss to er lik fire")
```

vil printe «to pluss to er lik fire», fordi første IF ikke er sant

```
if 2+2 == 4:
    print("to pluss to er lik fire")
elif 3+3 == 6:
    print("tre pluss tre er lik seks")
```

derav vil denne bare printe «to pluss to er lik fire», selv om begge statementsene er sanne, ignorerer koden elif-en fordi den første if-en er sann

Et ELSE- statement likner et ELIF-statement i at det bare kjøres hvis første IF ikke er sann, men er forskjellig i at et ELSE-statement har ikke noe IF-statement i seg. Så for eksempel:

```
if 2+2 == 5:
    print("to pluss to er lik fem")
else:
    print("hva er dette matte-tullballet?")
```

vil printe «hva er dette matte-tullballet?», fordi første if ikke er sann. Mens:

```
if 2+2 == 4:
    print("to pluss to er lik fire")
else:
    print("hva er dette matte-tullballet?")
```

dette eksemplet vil bare printe «to pluss to er lik fire» fordi første statement er sann, og da ignorerer den else-statementet.

Oppgave 2:

Lister i python er en måte man kan lagre flere data-«enheter» i en variabel,

Som et eksempel:

```
alfabet = ["A","B","C"]
```

her er en liste som heter «alfabet», som inneholder items «A», «B» og «C». data kan også være tall

lister kan brukes til å behandle og lagre store mengder data, eller mange variabler på en lett og kompakt måte

Oppgave 3:

Løkker er en måte å «gjenta» kode på i python. Det vil si at kode inni en løkke vil gjenta seg helt til løkken stopper.

Det fins to typer primær-løkker i python: WHILE og FOR.

En WHILE-løkke har på en måte en egen betingelse inni seg, som den sjekker hver eneste gang den looper. Hvis det er sant, vil den kjøre koden inni seg, og hvis den ikke er sann, vil den stoppe. Så for eksempel:

```
i = 1
while i < 6:
    print(i)
    i += 1
```

vil da printe 1, så 2, 3, 4 og stopper ved 5, siden på neste loop da har i blitt ikke mindre enn 6, så da kjører den ikke.

Man kan lage en evig while løkke ved å skrive «while True», fordi da sjekker den om Bool-verdien True er sann, noe som den alltid er. Evige while-løkker er nyttige for konstante oppdateringer i koden, som for eksempel i en klokke.

En for løkke funker annerledes. En FOR-løkke funker litt som å bla gjennom en bok. En FOR-løkke itererer igjennom en bestemt variabel, og kjører for så lang variabelen er. Som for eksempel:

```
for x in "eple":
    print(x)
```

vil printe «E», så «P»,«L» og «E», siden den kjører for hver bokstav i tekst-stringen.

For løkker kan iterere gjennom et mangfold av data, som for eksempel lister, Ranges, strings og mange andre som vi ikke har kommet til ennå. For-løkker er nyttige hvis man trenger å gå igjennom noe data/ kjøre noe kode et bestemt antall ganger basert på en variabel, da de gjør dette bedre enn while-løkker.