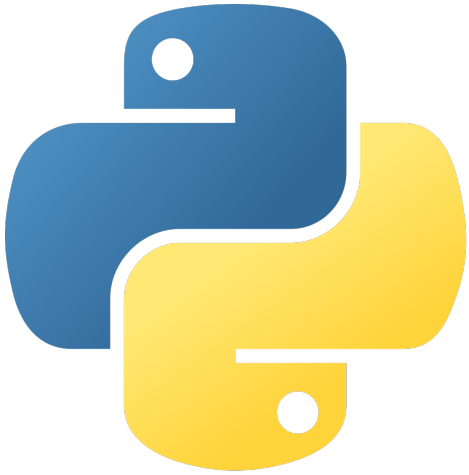

Python oppgavebank

Her er **oppgaver for hvert tema**.

Anbefaling:

- Begynne på starten
- Jobbe igjennom hvert tema

Dersom du selv opplever at du kan et tema, kan du gå videre til neste tema og sjekke om du har kontroll på det temaet.

Finner du ting her som du ikke har lært enda, er det meningen at du skal **finne ut av det selv** (Google/ChatGPT/spørre en klassekamerat, da må du skrive et godt formulert Google-søk eller en godt formulert spørring til Chatbotten slik at du finner informasjonen du leter etter).

NB! Gjerne spør om hjelp fra andre elever før du spør lærer om hjelp – det er kjempegod læring i å forklare noen andre hvordan noe fungerer. Dette gjelder både store og små ting.

PYTHON 1 - del 1: Variabler, print, input, matematikk, escape-tegn

Oppgave 1.1 – HELLO WORLD
Lag en variabel som inneholder strengen «Hello World!» og skriv den ut til konsollen.

Oppgave 1.2 – legge sammen verdier og print
Lag to variabler og legg sammen to verdier. Skriv passende tekst sammen med utskriften av svaret.

Oppgave 1.3 – aritmetikk
Gang sammen to tall. Bruk så mange variabler som du tenker kan være lurt, og skriv i kommentarer i koden hvorfor du har gjort det.

Oppgave 1.4 – input
Hent inn en string fra brukeren som du skriver ut tre ganger.

Oppgave 1.5 – casting
Hent inn tre temperaturer fra brukeren og finn gjennomsnittet av de. Skriv ut passende tekst og resultat i konsollen. Bruk escape-tegn.

Oppgave 1.6 – sette sammen flere ting
Hent inn hvor mye brukeren har i timelønn, og hvor mange timer brukeren har jobbet. Lag begge disse variablene på samme linje. Finn bruttolønnen (lønn før skatt). Hent etterpå inn hvor mye brukeren skatter i prosent. Skriv ut passende informasjon til brukeren, med forklarende tekst.

Oppgave 1.7 – kalkulator
Lag en enkel kalkulator som tar inn tall fra brukeren og som legger de sammen. Skriv kommentarer i koden din. La så brukeren bestemme selv om de vil legge til enda et tall, eller ikke. Skriv ut fornuftige ting i konsollen, og bruk den korteste skrivemåten du får til.

Oppgave 1.8 – UTFORDRING
Lag en fleksibel kalkulator som kan regne mange forskjellige regnearter, og kan skrive tall ut på vitenskapelig notasjon dersom brukeren ønsker det. Bruk funksjoner dersom du kan det fra før av (funksjoner er et tema som kommer senere, dersom du ikke kan det). Brukeren skal i starten av programmet kunne angi hvor mange siffer de ønsker etter komma, og kalkulatoren skal tilpasse seg. Bruk escape-tegn i utskriftene til brukeren.

PYTHON 1 - del 2: Sannhetsuttrykk, if/else, standardfunksjoner

Oppgave 1.9 – sannhetsuttrykk
Ta inn to tall der du sjekker om det andre er større enn det første i en variabel. Skriv ut resultatet av sjekken.

Oppgave 1.10 – standardfunksjoner
Lag seks variabler med positive og negative verdier. Finn maks med en standardfunksjon. Finn minimum med en standardfunksjon. Finn absoluttverdien av ett av tallene. Avrund ett av tallene. Skriv alt dette ut på et fint og oversiktlig format i konsoll.

Oppgave 1.11 – lengde på string
Ta inn navnet til brukeren og skriv ut hvor langt det er.

Oppgave 1.12 – if og else
Sjekk om 50/3 er større enn 20, og print ut fornuftige resultater.

Oppgave 1.13 - modulo
Prøv deg frem med %-operatoren (modulo). 10%3, 17%2, 17%3, 17%6. Skriv i koden din hvordan modulo fungerer.

Oppgave 1.14 – if
Lag en if-setning der du sjekker om en variabel er større enn en annen.

Oppgave 1.15 – if og else med forskjellige resultater
Ta inn hvor mange biler brukeren har solgt, og gi en bonus på 5000kr dersom det er solgt flere enn 70 biler. Regn så ut hvor mye brukeren får i lønn, hver bil gir 500kr i lønn. Skriv til konsoll.

Oppgave 1.16 – betingelser
Test ut de forskjellige betingelsene <, >, <=, >=, == og != på noen variabler.

Oppgave 1.17 – if og else basert på input
Godteri koster 30, og brus koster 20. Ta inn fra brukeren hvor mye penger de har. Skriv ut hva brukeren kan kjøpe.

Oppgave 1.18
Lag en enkel *nøstet* if-setning. Altså en if-setning inni en if-setning. Den skal først sjekke og en variabel er større enn et tall, og så sjekke om variabelen modulo 2 (%) blir akkurat 1. Dersom det er sant skal det printes «Oddetall!» Dersom det ikke er sant, skal det printes «Partall!»

PYTHON 2 - del 1: For-løkker, while-løkker

Oppgave 2.1 – for-løkke
Skriv en for-løkke som går fem ganger og printer ut stringen «Dette printes!».

Oppgave 2.2 – datatyper
Beskriv hva datatypene *string*, *integer*, *float* og *boolean* er. Prøv å komme med et eksempel på når det kan være lurt å bruke de forskjellige datatypene.

Oppgave 2.3 – while-løkke
Print ut tallene fra 0 til og med 7 med en while-løkke. Bruk en variabel for å holde styr på om løkken skal forsette å kjøre.

Oppgave 2.4
Print ut tallene fra 10 til og med 20 med en while-løkke. Bruk en boolsk variabel for å bestemme om løkken skal forsette å kjøre.

Oppgave 2.5
Spør om input fra bruker frem til bruker skriver 'Exit'. Skriv ut det brukeren skrev hver gang.

Oppgave 2.6
Lag en for-løkke som skriver ut tallene fra 0-5. Bruk en f-string til å formatere utskriften.

Oppgave 2.7
Lag en for-løkke som skriver ut tallene 5-9.

Oppgave 2.8
Lag en for-løkke som skriver ut partallene mellom 0 og 10.

Oppgave 2.9
Lag en for-løkke som skriver ut tallene 1-5 i motsatt rekkefølge.

Oppgave 2.10 - UTFORDRING
Lag et program som skriver ut tallene fra 1 til og med 9 i et 'rutenett':

1 2 3
4 5 6
7 8 9

PYTHON 2 - del 2: in, continue, break

Oppgave 2.11 - in
Lag en if-setning som sjekker om 'e' er i en string.

Oppgave 2.12 - UTFORDRING
tekst = 'Uten mat og drikke, duger helten ikke'
vokaler = 'aeiouyæøå'

Print ut alle vokaler i «tekst»-variabelen.
Print ut alle konsonanter i «tekst»-variablenen.

Hint: bruk «vokaler»-variabelen og sannhetsuttrykket «in», f.eks.: 'h' in vokaler gir False, 'y' in vokaler gir True.

Oppgave 2.13 - continue
Skriv en for-løkke som går over tallene 1 til 10, og printer ut alle tallene utenom 6. Bruk continue på dette tallet. Søk opp hvordan continue fungerer. Skriv til skjermen når det hoppes over.

Oppgave 2.14 - break
Lag en for-løkke som går fra 10 til 20 som legger til ett og ett av disse tallene frem til totalverdien er over 50. Bryt ut av løkken med *break* og print ut verdien.

PYTHON 3 - del 1: funksjoner

Oppgave 3.1
Lag en funksjon som skriver ut teksten “Hello World!”. Kall på funksjonen tre ganger.

Oppgave 3.2
Beskriv begrepene mtp. funksjoner i Python (gjerne kopier fra et internettsøk, så lenge du er enig i det som står der og forstår det):

- Deklarere en funksjon
- Funksjonskall
- Parametre
- Argumenter
- Returverdi
- Global variabel
- Lokal variabel

Oppgave 3.3
Deklarer en global variabel. Lag en funksjon som skal endre på den globale variabelen. Lag også en lokal variabel inni funksjonen. Kall på funksjonen, og printe de forskjellige variablene etterpå. Skriv en kommentar i koden for hva som skjer.

Oppgave 3.4
Lag en funksjon som tar inn et tall som parameter. Funksjonen skal skrive “Hello World!” så mange ganger som argumentet tilsier.

Oppgave 3.5
Lag en funksjon som regner ut lønn fra en timelønn og et antall timer. Timelønn og antall timer skal hentes inn som parametre.

Oppgave 3.6
Lag en funksjon som regner ut gjennomsnittsalderen på et antall mennesker. Antallet mennesker skal sendes inn som parameter. Brukeren skal skrive inn alderen for hvert menneske, og det legges til en totalverdi. Skriv ut gjennomsnittet.

Oppgave 3.7
Lag en funksjon som sjekker om en string inneholder en av bokstavene 'æ', 'ø' eller 'å'. Stringen skal hentes inn som argument. Print ut 'JA' dersom den gjør det.
Variabler som kan brukes:
tekst = 'Norsk setning med Æ'
ny = 'English without the letters'

Oppgave 3.8
Lag en funksjon som skriver ut en bokstav på en bestemt plass i en gitt streng. Posisjonen, eller indeksen, skal bli gitt som argument og stringen skal bli gitt som argument.

PYTHON 3 - del 2: Funksjoner - returverdi, nøstede funksjonskall

Oppgave 3.9
Lag en funksjon som spør om fornavn, og så etternavn. Returner hele navnet satt sammen. Lag en variabel som settes lik funksjonskallet.

Oppgave 3.10
Lag en funksjon som tar inn navn som parameter. Spør brukeren (med navn) om alder og bosted. Returner alder og bosted. Kall på funksjonen og lagre returverdiene i variabler.

Oppgave 3.11
Lag en funksjon som tar inn to tall som parametre. Ta inn input fra bruker i form av '/', '*', '//' eller '%'. Funksjonen skal returnere resultatet av den typen matematisk operasjon gjort på de to tallene. Skriv ut resultatet.

Oppgave 3.12
Lag en funksjon som tar inn en streng. Iterer over stringen og returner den første vokalen du finner. Dersom det ikke finnes noen vokal i strengen, returner en beskrivende tekst.

Oppgave 3.13
Lag en funksjon som tar inn to parametre, timelønn og antall_timer. Funksjonen skal returnere lønn gitt antall timer. Lag så en funksjon som tar inn timelønn og antall timer fra brukeren og returnerer det. Gjør et *nøstet* funksjonskall (funksjonskall INNI funksjonskall) der den første funksjonen blir kalt først, og den andre etterpå. Print ut resultatet.

Oppgave 3.14 - UTFORDRING, TAR EN DEL TID
Du skal nå lage et terningspill som kombinerer alt du har lært hittil. Husk: Google er din venn!

a. Importer random-modulen (du trenger kun randint).
b. Lag en info()-funksjon som skriver ut info om spillet, samt hvordan man avslutter.
c. Ta inn navnet til en bruker i en funksjon. Returner navnet. Bruk funksjonen til å lage to spillere.t
d. Lag en funksjon som heter trill_terninger. De to brukerne skal trille terninger om og om igjen så lenge brukeren vil. Hold styr på hvor mange poeng de har. Gi ett poeng til den som får høyest terningkast, ingen poeng dersom de får samme terningkast. Returner hvor mange poeng spillerene har etter funksjonen er ferdig.
e. Lag en funksjon som heter start_spill() og som er ansvarlig for å kalle på alle de andre funksjonene.
f. Lag en funksjon som tar inn to parametre, score_en og score_to. Funksjonen skal skrive ut poengscore og hvem som vinner.

PYTHON 4: Lister

Oppgave 4.1
Lag en liste med fem verdier i listen. Skriv ut listen som helhet.

Oppgave 4.2
Lag en liste med fem verdier i listen. Skriv ut listen som helhet.

Oppgave 4.3
Bruk listen fra forrige oppgave. Hent ut elementer på tre plasser som du skriver ut.

Oppgave 4.4
Lag en liste som inneholder 10 elementer. Skriv ut listen. Endre på det femte elementet (hvilken indeks blir det?), og skriv ut listen på nytt. Legg til ett tall på slutten av listen og skriv ut listen. Legg til ett tall med *insert* på indeks 4 i listen. Fjern ett tall fra listen med *del*. Fjern det siste elementet fra listen med *pop*. Skriv ut resultatet.

Oppgave 4.5
Bruk denne listen: [‘Hei’, ‘på’, ‘deg’, ‘!’]
Iterer over listen på denne måten: for *element* in min_liste og skriv ut *element* hver gang.

Oppgave 4.6
Bruk denne listen: [‘Hei’, ‘på’, ‘deg’, ‘!’]
Iterer over listen med range(4), og skriv ut hvert element i listen.

Oppgave 4.7
Lag en liste med fem heltall (int). Finn maksverdien og minimumsverdien, skriv de ut.

Oppgave 4.8
Finn det største tallet, men du skal nå gjøre det manuelt (ikke bruke maks/min). Du skal iterere over listen, og lagre den største verdi i en variabel. Du skal sammenligne to og to elementer fra listen, som du så skal finne den største av før den da lagres og du itererer videre. Skriv ut for hver iterasjon hvilke to tall som blir sammenlignet og hva som er lagret i variabelen.

Oppgave 4.9
Utvid forrige oppgave til å være *skalerbar* - du skal lage det som en funksjon som henter inn en liste som parameter og returnerer det største tallet til slutt.

Oppgave 4.10
Lag en tom liste. Så lenge brukeren vil så skal brukeren skrive inn ett og ett tall som legges til i listen. Gå så igjennom alle tall, og gjør de om til float. Skriv til slutt ut summen av alle tall i listen.

Oppgave 4.11
Lag en liste med 7 temperaturer og en annen liste med de syv dagene i uken. Skriv ut hver dag med tilhørende temperatur. Bruk f-string. Utvid med enda en liste som inneholder

vindhastighet, samt en liste med nedbørsmengde. Skriv ut på pent format som er forståelig og oversiktlig for brukeren.

PYTHON 5: Dictionaries

Oppgave 5.1
Lag en dictionary av lengde 2 med informasjon om deg selv som for eksempel bosted og alder. Legg til en nøkkel med en verdi, og print ut hele dictionaryen før og etter du legger til det nye nøkkel-verdi paret.

Oppgave 5.2
Lag en dictionary *temperaturer.* Nøklene skal være bynavn med små bokstaver, og så skal du skriv inn forskjellige temperaturer som verdier. Skriv nå ut hvert bynavn og temperatur sammen. Hint: bruk "for key in dictionary" og "dictionary[key]".

Oppgave 5.3
Utvid forrige oppgave til å spørre brukeren om de ønsker å legge til en ny temperatur for en annen by. Skriv ut fornuftige ting til brukeren.

Oppgave 5.4
Ta i bruk følgende dictionary:
person = {'name': 'Alex', 'age': 17, 'drivers license': False, 'height': 175, 'sex': 'non-binary'}
Skriv ut verdien som er tilknyttet nøkkelen "age". Skriv ut verdien som er tilknyttet nøkkelen "height". Spør brukeren hva de ønsker å vite noe om, og skriv ut verdien tilknyttet den nøkkelen.

Oppgave 5.5
Bruk samme dictionary som forrige oppgave. Fjern nøkkelen "sex" med metoden pop(). Sjekk om nøkkelen "height" finnes i dictionaryen. Skriv ut alle nøklene til dictionaryen med metoden keys. Skriv ut alle verdiene med metoden values().

Oppgave 5.6
Ta i bruk følgende dictionary:
aksjekurs = {'EQNR': 233.90, 'DNB': 210.50,
'NHY': 71.04, 'PMG': 18.52}
Hver nøkkel er her en "ticker" og hver verdi er markedsverdi. Iterer over *aksjekurs* og skriv ut hver ticker.

Oppgave 5.7
Bruk samme dictionary som forrige oppgave. Skriv ut en oversikt over hvilken nøkkel som peker på hvilken verdi, bruk for-løkke.

Oppgave 5.8 - UTFORDRING
Bruk strukturen under. Informasjonen er på følgende format: {ticker: [pris_per_aksje, aksje, volum]}
aksjekurs = {'EQNR': [233.90, 'equinor', 3700000], 'DNB': [210.50, 'dnb bank asa', 382664], 'NHY': [71.04, 'norsk hydro', 3300000], 'PMG': [18.52, 'play magnus', 26063]}

Hent ut prisen for en aksje i PMG. Hent ut det fulle navnet til PMG. Lag en funksjon som returnerer en liste med alle navnene til selskapene (ikke ticker). Skriv ut alle fulle navn, men slik at første bokstav er stor.

Oppgave 5.9
Lag en tom dictionary *konti* (flertall av konto). Lag en funksjon legg_til_konti() som ber brukeren om å skrive inn navnet på en konto de vil lage, og hvor mye penger som står på kontoen. Funksjonen skal legge til den nye kontoen med tilhørende pengeverdi i dictionaryen.

Oppgave 5.10
Oppgaven hører sammen med forrige oppgave.
Lag en funksjon oppdater_saldo(nokkel, verdi) som tar inn to parametre *nokkel* og *verdi*. Funksjonen skal aksessere kontoen via nøkkelen og oppdateresaldo på kontoen.

Oppgave 5.11
Oppgaven hører sammen med de to forrige oppgavene.
Lag en funksjon banktjeneste() der brukeren får velge hva de vil gjøre (lage ny konto, oppdatere saldo eller avslutte) og som kjører nødvendige funksjoner.

PYTHON X: UTFORDRINGSHJØRNET - algoritmisk tenking

Oppgave X.1
Lag et rutenett av størrelse 5x5. Bruk lister og for-løkker for å gjøre dette. Tallene skal være tilfeldige.

Oppgave X.2
Søk på hva en «grådig algoritme» er. Lag nå en grådig algoritme som går fra det ene hjørnet (0,0) til det andre hjørnet (4,4) og får høyest mulig «score». Bruk rutenettet fra oppgave X.1. Den har kun lov å bevege seg til høyre eller nedover. Skriv ut for hvert «trekk» både total score, posisjon og score for posisjonen.

Eksempel for hvordan rutenettet som algoritmen skal jobbe seg gjennom kan se ut:

5	4	0	10	7
3	8	4	8	1
1	1	5	7	10
10	0	4	2	0
3	6	1	9	2

Oppgave X.3
Oppdater fra oppgave X.1 slik at du har en funksjon som kan lage rutenett av den størrelsen brukeren ønsker.

Oppgave X.4
Oppdater fra oppgave X.2 slik at algoritmen din kan kjøre på rutenett av forskjellige størrelser.

Oppgave X.5
Oppdater fra oppgave X.3 slik at du kan lage rutenett som inneholder «blokkader», altså «X».

Oppgave X.6
Oppdater fra oppgave X.4 slik at algoritmen nå tar hensyn til blokkadene.

Oppgave X.7
Lag en annen algoritme som jobber seg gjennom rutenettet (kanskje en som gjør det samme som sist, men ser ett steg ekstra frem før den går dit? To steg frem?).

Oppgave X.8
Sammenlign kjøringen fra den første algoritmen og den siste. Hvilke forskjeller er det? Hvilken algoritme fungerer best på datasettet (rutenettet)?

Oppgave X.9
Lag til slik at rutenettet skrives til en tekst-fil (.txt-fil).

Oppgave X.10
Hent inn innholdet i tekstfilen slik at algoritmen kan kjøre på den.

PYTHON Y – diverse og objektorientert programmering

Oppgave Y.1
Lag en chat-bot i Python. Du må først ta inn navnet til brukeren, og så skal "samtalen" gå frem til brukeren skriver "bye". Brukeren skal kunne kommunisere og få svar.

Oppgave Y.2
https://www.techgeekbuzz.com/blog/python-object-oriented-programming-exercise/

Oppgave Y.3
Her skal du begynne å lære deg objektorientert programmering, ved å starte å lære om klasser og instanser. Følg denne tutorialen, og prøv deg frem selv:
https://www.youtube.com/watch?v=ZDa-Z5JzLYM

Oppgave Y.4
Dersom du ønsker å forsette å lære deg objektorientert programmering kan du følge tutorialen videre. Gjør slik han gjør i videoen, og ikke gå videre før du faktisk har litt kontroll på det som skjer og forstår det.

Oppgave Y.5
Lag spillet Tic-Tac-Toe i Python.

Oppgave Y.6
Lag spillet Tic-Tac-Toe objektorientert i Python.

PYTHON Z – nyttige funksjoner, moduler og triks

Oppgave Z.1
Lag et program som benytter seg av list-comprehension i Python. Du skal lage to lister, der du itererer over den ene vanlig og den andre baklengs og sammenligner elementer. Lag en strategi for hva som skjer dersom listene ikke er like lange, og begrunn i koden din hvorfor du gjør det slik. Alt skal kodes inni funksjoner som er fornuftige og godt strukturerte.

Oppgave Z.2
Lag en liste som inneholder tall.
Lag en ny liste ved hjelp av list-comprehension som kvadrerer alle tallene i den forrige listen.

Oppgave Z.3
Søk på lambda-funksjoner i Python. Lag nå en lambda-funksjon som ganger sammen to tall.

Oppgave Z.4
Lag en lambda-funksjon som tar inn et svar fra brukeren. Dersom brukeren skriver "n" skal man returnere "Så trist!", dersom de skriver "y" skal det returneres "Happy days". Gjør et funksjonskall av lambda-funksjonen din.

Oppgave Z.5
Søk på hvordan man bruker matplotlib i Python. Du skal nå lage en "pai" med matplotlib. Utvid etterpå til at grafen viser beskrivende tekst.

Oppgave Z.6
Du skal nå lage en graf med matplotlib. Utvid etterpå til at grafen viser beskrivende tekst.

Oppgave Z.7
Du skal nå lage en "pai" med matplotlib. Finn informasjon på internett om befolkningsstørrelse i Oslo, Bergen og Trondheim og plot den med et pai-diagram. Vis relevant informasjon i plottet.

Oppgave Z.8
Importer numpy-modulen som np (import numpy as np).
Lag en *array* som settes til verdien np.arange(5). Print ut arrayen. Beskriv forskjellen fra en vanlig liste.

Oppgave Z.9
Importer numpy. Lag en array med arange fem tall og steg 0.1. Print ut arrayen.

Oppgave Z.10
Lag forskjellige arrays der du bruker np.zeroes(), np.ones. Bruk etterpå reshape() på arrayene. Skriv ut både før og etter reshape.

Oppgave Z.11

Lag to arrays av samme størrelse som du legger sammen. Skriv ut resultatet.

Oppgave Z.12
Lag en liste med stemmetall for ulike partier.
Lag nå en funksjon prosent_array(stemmer) som tar inn en liste med stemmer. Funksjonen skal returnere den prosentvise fordelingen mellom de som en array. Hint: Du kan utføre en matematisk operasjon på en hel array uten å bruke en løkke. Når du returnerer listen kan du bruke np.around(returverdi, 2) for å runde til 2 desimaler.

Oppgave Z.13
Du skal nå lære deg å bruke Python med Django – Python for Webutvikling.
Gå inn på denne, og følg tutorialen:
https://docs.djangoproject.com/en/4.2/intro/tutorial01/

Oppgave Z.14
Her skal du lære deg å koble sammen et fullstack prosjekt der du skal bruke Flask i Python, og koble til en MySQL-database. Følg denne tutorialen:
https://code.tutsplus.com/creating-a-web-app-from-scratch-using-python-flask-and-mysql--cms-22972t

Oppgave Z.15
Lyst å lære Webscraping med Python? Følg denne:
https://www.edureka.co/blog/web-scraping-with-python/

Eller denne:
https://realpython.com/beautiful-soup-web-scraper-python/

PYTHON SUPER X – utvikle algoritmer

Oppgave SUPER X.1

Vi starter litt rolig. Søk på Bubble Sort. Du skal nå implementere denne algoritmen i Python, og kunne forstå og forklare din egen kode. Lag egne datasett som du kan kjøre algoritmen på.

Oppgave SUPER X.2

Søk på Quick Sort. Du skal nå implementere denne algoritmen i Python, og kunne forstå og forklare din egen kode. Lag egne datasett som du kan kjøre algoritmen på.

Oppgave SUPER X.3

Vi går over på en litt vanskeligere algoritme. Søk på Radix Sort. Du skal nå implementere denne algoritmen i Python, og kunne forstå og forklare din egen kode. Lag egne datasett som du kan kjøre algoritmen på.