

Githublink

<https://github.com/Vegarderv/ITGK-Eksamensforelesning-H20>
23

Vi starter 12:15

Sett opp Menti mens du venter:

www.menti.com, bruk kode 6558 1314

Og hent slides lokalt her:

<http://bit.ly/abakusitgk2023>



Kræsjkurs



TDT4109 Informasjonsteknologi Grunnkurs

Ha Menti og slides lokalt på PCen underveis:

www.menti.com, bruk kode **5544 366**

Hent slides lokalt her:

<http://bit.ly/abakusitgk2023>



Agenda for dagen

- **12:15 - 12:30**
 - Intro og **Variabler, typer, conditionals, løkker, lister, funksjoner**
- **13:15 - 14:00:**
 - **Strengåndtering, slicing, dictionaries, sets, filbehandling**
- **14:15 - 15:00:**
 - **Rekursjon, tips og triks, numpy, nyttige funksjoner**
- **15:15 - 17:00:**
 - **Spørsmål og gjennomgang av oppgaver. Avslutning**

Hvem er jeg?

- Vegard, 26 år fra Bamble
- Kunnskapsingeniør i Computas
- Gikk ut av datateknologi-studiet i 2024
- Fordypning i kunstig intelligens, masteroppgave om optimalisert allokering av ambulanseressurser
- Dette blir min 6. eksamensforelesning :o
- Veldig glad i programmering og generelt interessert i oppbyggingen av maskiner og programvare



Forventninger til dagens sesjon

- Dere vil få en oversikt over noe av det viktigste i pensum, forklart med “nye ord”.

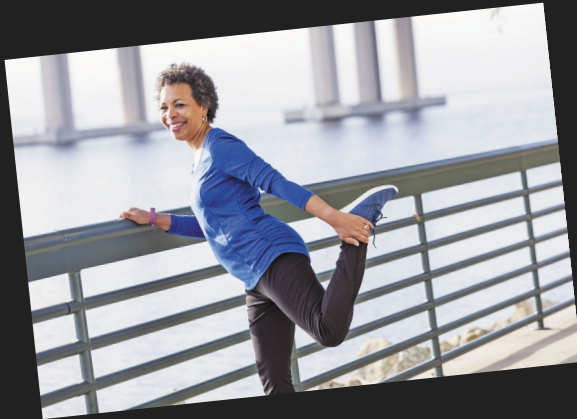
Går på handels, og dette var dritbra. Aldri slutt, han foreleseren var helt villt funny. Snev av autisme, men det digger vi. Stå på!

- Husk at jeg vet ingenting om hva som kommer på årets eksamen.
- Det er mye tekst og mange slides men vi rekker ikke gå gjennom alt idag, tanken er at man kan bruke dette som et “kompendium” i etterkant

KODETID



Kaffe, strekk og kos
15 min



Oppgaveløsning

Svar noe på alle oppgavene

- Om du ikke har tid til å skrive kode på en oppgave, forklar **hva du ville ha gjort**
- Husk: **Forståelsen er det viktigste**, og du kan få god uttelling selv om det ikke er 100% korrekt syntaks.
 - **Semantikk > Syntaks**
- Husker du ikke syntaks? Skriv **pseudokode**
- **Det er alltid bedre å svare noe enn ingenting**

```
Organise everything together;  
Plug in kettle;  
Put teabag in cup;  
Put water into kettle;  
Wait for kettle to boil;  
Add water to cup;  
Remove teabag with spoon/fork;  
Add milk and/or sugar;  
Serve;
```

Antagelser

- Skriv **antagelser** hvis nødvendig
 - Oppgaver kan være tvetydige/vage, det er helt innafor å gjøre en antagelse, skrive den i starten av oppgaven og løse ut i fra denne
- Pass på at du gjør en **rimelig** antagelse.

3.5 **rec re r**

Lag funksjonen ``letter_count`` som tar i mot:

1. En streng `s`
2. En dictionary `d`

Oppførsel og krav:

- Funksjonen **SKAL** være rekursiv, og resultatet av kallet til funksjonen skal være:
- En dictionary med tegnene i strengen `s` som nøkler, og antallet ganger de opptrer som verdier.
- Du før ikke lov til å bruke streng-funksjonen `count`.
- Ja, jeg vet at denne er enkel å løse uten rekursjon. Poenget med oppgaven er å se om du klarer å løse oppgaven med de verktøyene du er gitt.

3.6 Trippel 1 - jsinn

data er en éndimensjonal liste som inneholder navn og poengene de har fått i et spill, og er resultatet av forrige oppgave. Listen inneholder først navn, så dennes poeng, så neste navn og dennes poeng.

Lag funksjonen *load_game()*

- Funksjonen skal lese inn fra filen *game.json* som befinner seg i samme mappe som funksjonen.
- Funksjonen skal returnere verdien som er knyttet til nøkkelen *results*.
- Merk: json-strukturen inneholder litt mer data enn det som skal brukes.

Json-filen har følgende innhold:

```
{
"game": "kurrong",
"date": "20250717",
"results": ["Carl", "4", "8", "13", "Berit", "11", "15", "44", "Ada", "24", "1"]
}
```

Eksempel på bruk:

```
>>>data = load_game()
>>> print(data)
['Carl', '4', '8', '13', 'Berit', '11', '15', '44', 'Ada', '24', '1']
>>> print(type(data))
<class 'list'>
```

Skriv svaret ditt her. Endringer blir lagret automatisk.

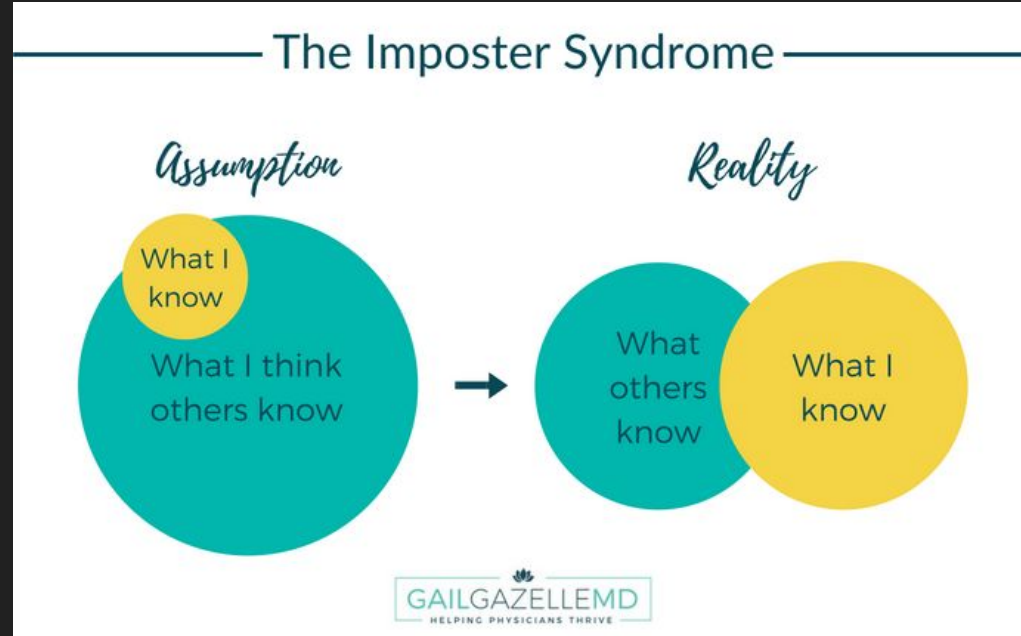
Helt til slutt

“The most disastrous thing
that you can ever learn is
your first programming
language.”

- Alan Kay

Impostor syndrome

- Dette er noe man må lære seg å leve med, man blir aldri helt kvitt det (etter min erfaring)
- Alle har alle allerede drevet med “programmering” tidligere. Dere må bare lære dere en ny “syntaks”.



“Jeg forstår jo ikke LF engang!”

- Husk at LF = Løsnings**forslag**
- Mange LF-er er **lite pedagogiske**, dvs. de bruker avanserte metoder for å gjøre koden kortere.
- Som regel kan man løse problemet med mye enklere “ordforråd” (men da gjerne med litt lengre kode).
 - Eksempelvis: Bruke lister i stedet for dictionary, sets, etc.



Det viktigste er **tydelighet**, ikke hvor kort koden er...

```
variabel = None  
if 6 > 5:  
    variabel = "Sant"  
else:  
    variabel = "Usant"
```



Hvilken av disse er egentlig mest “lesbar” av mennesker? Øyet som ser, men jeg vil påstå at nok mange “forstår” den øverste raskere

```
variabel = "Sant" if 6 > 5 else "Usant"
```

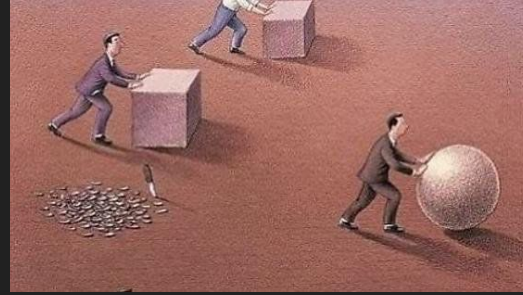
Slapp av!

- Mange av problemene du møter på eksamen kan kokes ned til **IF/ELIF/ELSE**-statements samt **FOR** og **WHILE**-løkker. Er du rutta på disse samt funksjoner og lister, samt kan enkelt filbehandling så kommer du langt bare med dette.
- Det er ingen svar i ITGK som er for lange eller kompliserte! Men forsøk å begrense deg så du ikke bruker all tiden på én oppgave.

Sist men ikke minst

- ITGK og programmering generelt er et modningsfag.
- Det krever mye mengdetrening for å bli god.
- Ikke fortvil og gi opp selv om du ikke får det til akkurat nå
 - En dårlig karakter i ITGK betyr ikke at du ikke bør satse videre på en karriere innen IT.
 - Det finnes vanvittig mange roller som må fylles innen IT, både fra det mest tekniske til helt ikke-tekniske roller, hvis man foretrekker det.
- Mange starter med å foretrekke å jobbe ikke-teknisk og oppdager en lidelse for programmering senere i livet. Andre gjør motsatt.

Noen ganger går det bare ikke!



- Noen ganger mangler man motivasjon, man kan ha en dårlig dag, eller rett og slett bommer på hva man bør øve på.
- Ingen skam å måtte ta opp emner, men ha også i bakhodet at det kan være usunt å fokusere for mye på å skulle ha perfekte karakterer. Man vinner (erfaringsmessig) lite på å ta opp stå-karakterer.
- Jobb smartere, ikke hardere. Dette gjelder spesielt ITGK
- Ikke pugg deg forderva på LF, i programmering så handler det om å finne de riktige verktøyene, ikke å kunne pugge og memorere kodesnutter (disse finner du uansett på google!)

Tilbakemelding ønskes



LYKKE TIL PÅ EKSAMEN – Dette klarer du!

