Mentimeter

Velkommen til første gruppetime i IN2010!

3

Hva forventer dere av gruppetimene i IN2010?

gjennomgang av pensum

Gjerne svarene på oblig

Forstå pensum

Repetisjon / Oppklaringer

Hjelp til innleveringene!

Hjelp til å forstå ting

gjennomgang av vanskelige ting fra forelesning

Oblighjelp :')

2

10

Hva forventer dere av gruppetimene i IN2010?

gjennomgang av ulike oppgaver

Få hjelp med ukesoppgaver og gjennomgang av eksempler

oppgaveløsing med hjelp, eksamensoppgaver

lære om ender

eksamensoppgaver

høres bra ut det du sa:)

Hva slags type format vil dere ha på disse gruppetimene? Eks. Et eller annet før pausen med et eller annet etter pausen.

Første halvdel til det som var vanskelig fra timen

første halvdel pensum, andre halvdel oblighjelp :))

pensum før pause, egen jobbing etter

første del pensum, andre del jobbing/hjelp med oppgaver

høres bra ut

1

5

Mentimeter

Hvem er jeg?

2

Mentimeter

Hvem er dere?

Dagens plan

- Tips
- Oppsummering
- Kjøretid
- Trær
- Live koding?
- Selvstendig arbeid (lurt å jobbe i grupper)

Tips og triks for emnet

- Implementer ALT og tegn så mye som mulig
  - Gjør pseudo-kode til faktisk kode
- Diskuter med andre medstudenter
- Bruk pensum i kattis / leetkode
- Jobb jevnt og lat som innleveringer er obliger
- Bruk ressursene til Lars Tveito på emne siden

Mentimeter

Over til en kort oppsummering

Kjøretid

- Notsjonen for kjøretid er store O eller O-notasjon
- Handler om hvor **effektivt** en algoritme er og **ikke** hvor **raskt**
- Store O beskriver et worst case
  - Se for deg at input fører til at algoritmen har så mange steg som mulig

```
def funksjon(liste):
    for elem1 in liste:
        for elem2 in liste:
            if elem1 == elem2:
                print("fant en likhet!")
```

Call-stacken og Rekursjon

- Rekursjon er funksjoner som bruker seg selv inne i seg selv
- Call-stacken er en stack
  - Den holder styr på hvilken rekkefølge funksjons-kall skal bli gjort
- En kø for hvem sitt arbeid som skal bli gjort først

```
def tell_opp_til(n):
    if n > 0:
        tell_opp_til(n - 1)
    print(n)
```

Hva er et tre?

Består av noder *v* som inneholder pekere til

- Barnenoder – v.children
- Elementet noden holder på – v.element
- Foreldrenode – v.parent

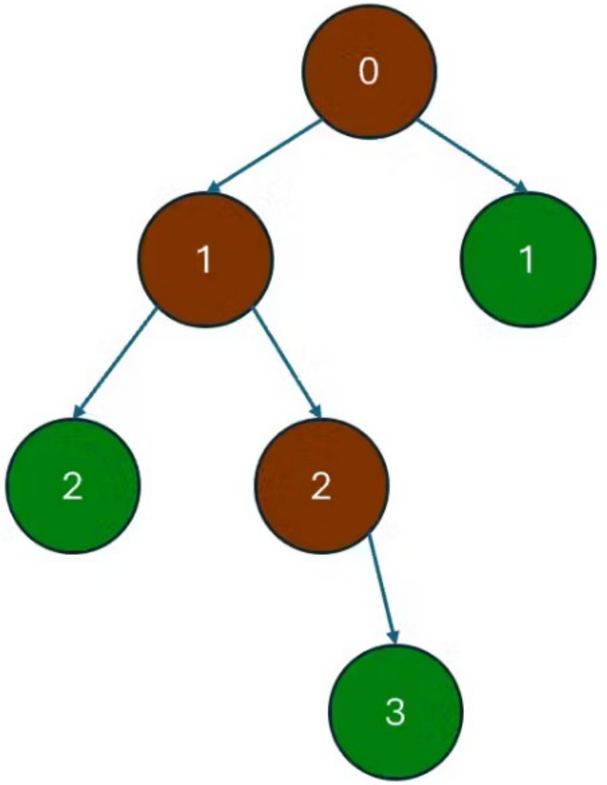
Rot* peker på den første noden *v

- I et tomt tre peker ***rot*** på null / None

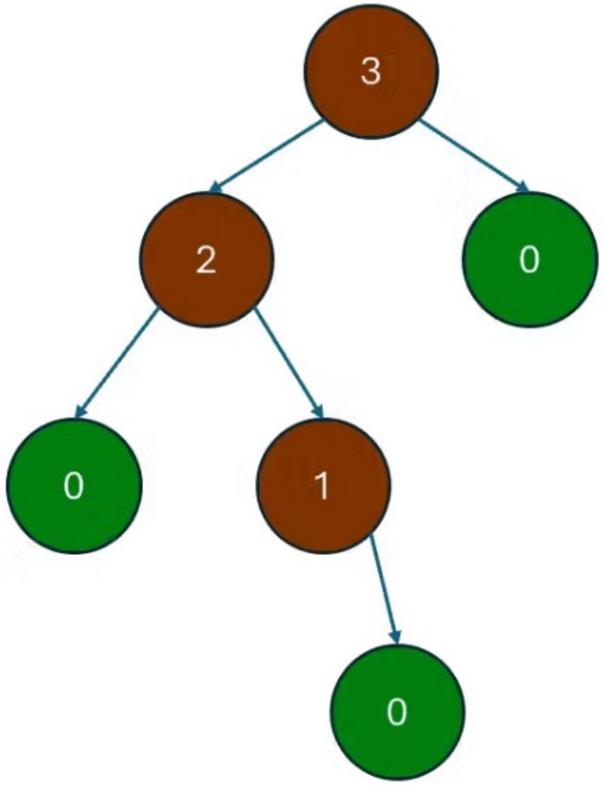
Minner om lenkeliste fra IN1010

- hode = rot
- .neste = .children (bare at children er en liste)
- .forrige = .parent

Dybde



Høyde



Binært-tre

- Samme som et vanlig tre men med en regel
- Binær -> alle noder har kun to barnenoder

Binært-søke-tre

- Samme som binært-tre med en ekstra regel
- Alle noder har maks to barnenoder der hvor:
  - venstre node er **mindre**
  - høyre node er **større**

Diskuter

Hva blir kjøretiden for følgende operasjoner på et **binært-søke-tre**?

- Insetting
- Sletting
- Søk

(Husk worst-case)

AVL-trær

- Samme som et binært-søke-tre, med en ekstra regel
- Treet skal være balansert som betyr:
  - Alle noder skal **ikke** ha en høyde forskjell på mer enn 1
- Insetting skjer vanlig, men dersom det blir ubalansert må man gjøre en rotasjon
  - Rotasjoner er forflytninger av noder slik at man beholder strukturen

Diskuter

Hva blir kjøretiden for følgende operasjoner på et **AVL-tre**?

- Insetting
- Sletting
- Søk

(Husk worst-case)

Mentimeter

Live koding eller
over til selvstendig arbeid?

2

Selvstendig arbeid
Anbefaler å gjøre:

- Ukesoppgaver
- Kattis / Lettcode
  - Kattis oppgaver:
    - Numbers On a Tree
    - Kitten on a Tree
    - Boxes
    - Kattis's Quest
- Tidligere eksamener om trær