GRUPPETIME IN1000

Datastruktur og gjennomgang av studentsystem.py

Noen magiske metoder

Oblig

•Hun heter Demi og går master i PROSA. Hun skal skrive en masteroppgave om hvordan studenter som er ny eller erfaren innen programmering prosesserer sin "sensemaking"-ferdighet, altså hvordan gjøre noe uforståelig forståelig, når de programmerer.
•Hun ønsker studenter til et parintervju, observert bare av hun og datainnsamlingsverktøy, med oppgaver hun har laget, og det blir pizza.
•All data er anonymisert, og parintervjuene kan gjerne være med to gode venner (men begge må melde seg på nettskjema).
•Telefonnummer vil være eneste måte å kontakte studentene på, og der vil det bli bestemt en passende tid for parintervjet.
•Det er også viktig at nettskjema ikke er bindene, men en mulighet for Demi å kontakte studenter som kan være interessert å være med.
•Tilslutt, det er frivillig, og selv om man har meldt seg så er det alltid mulig å backe ut når som helst.



studentsystem.py

```
> class Universitet:···


> class Student:···

> class Ansatt:···


  uio = Universitet("Universitetet i Oslo")

  tast = ""
> while tast != "STOPP":···
```

```
class Universitet:

    def __init__(self, uninavn):
        self._uninavn = uninavn
        self._ansatte = []
        self._studenter = []
```

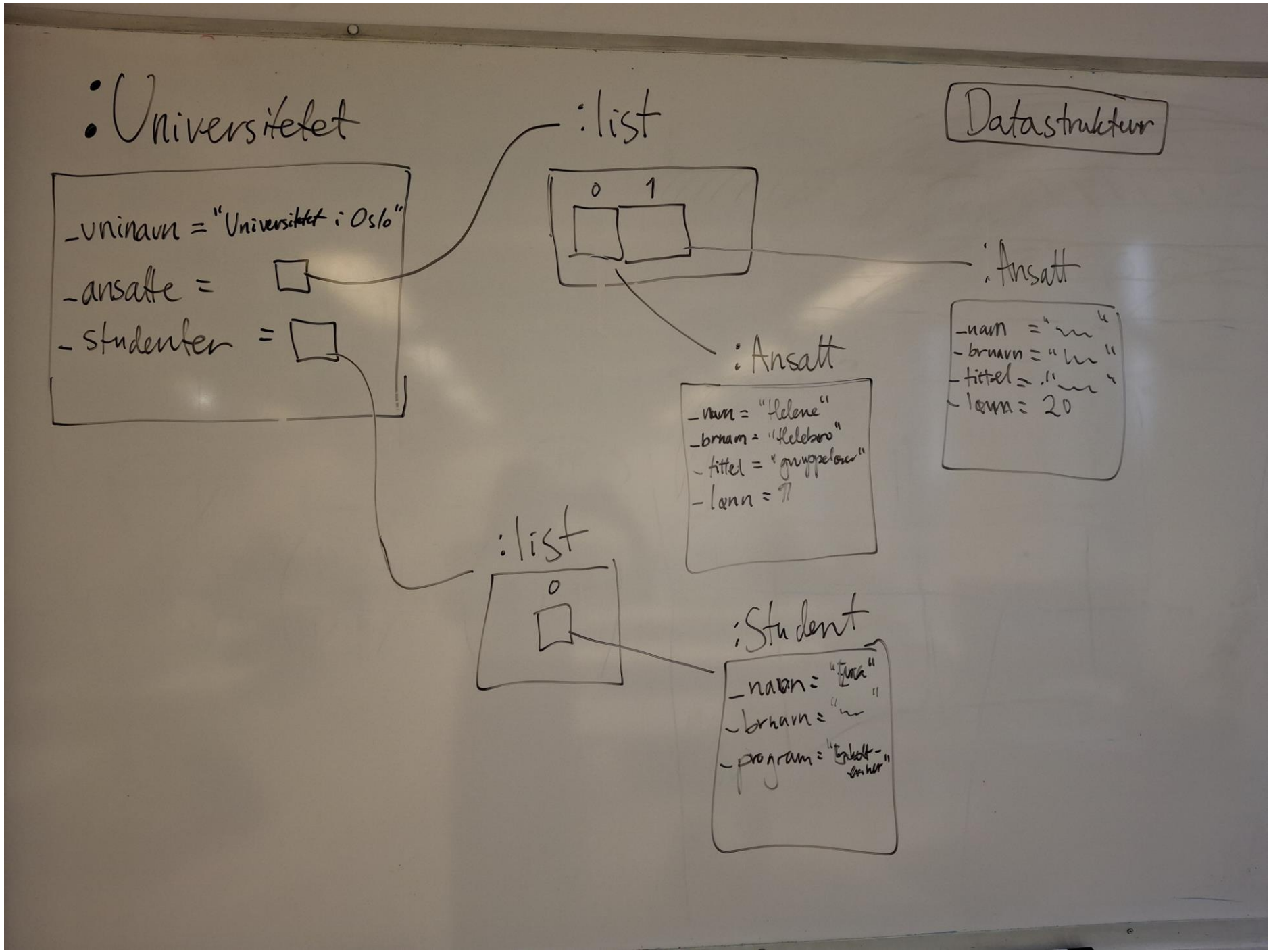
I vår datastruktur: Har trykket 1, 2, 2, STOPP

```
class Ansatt:

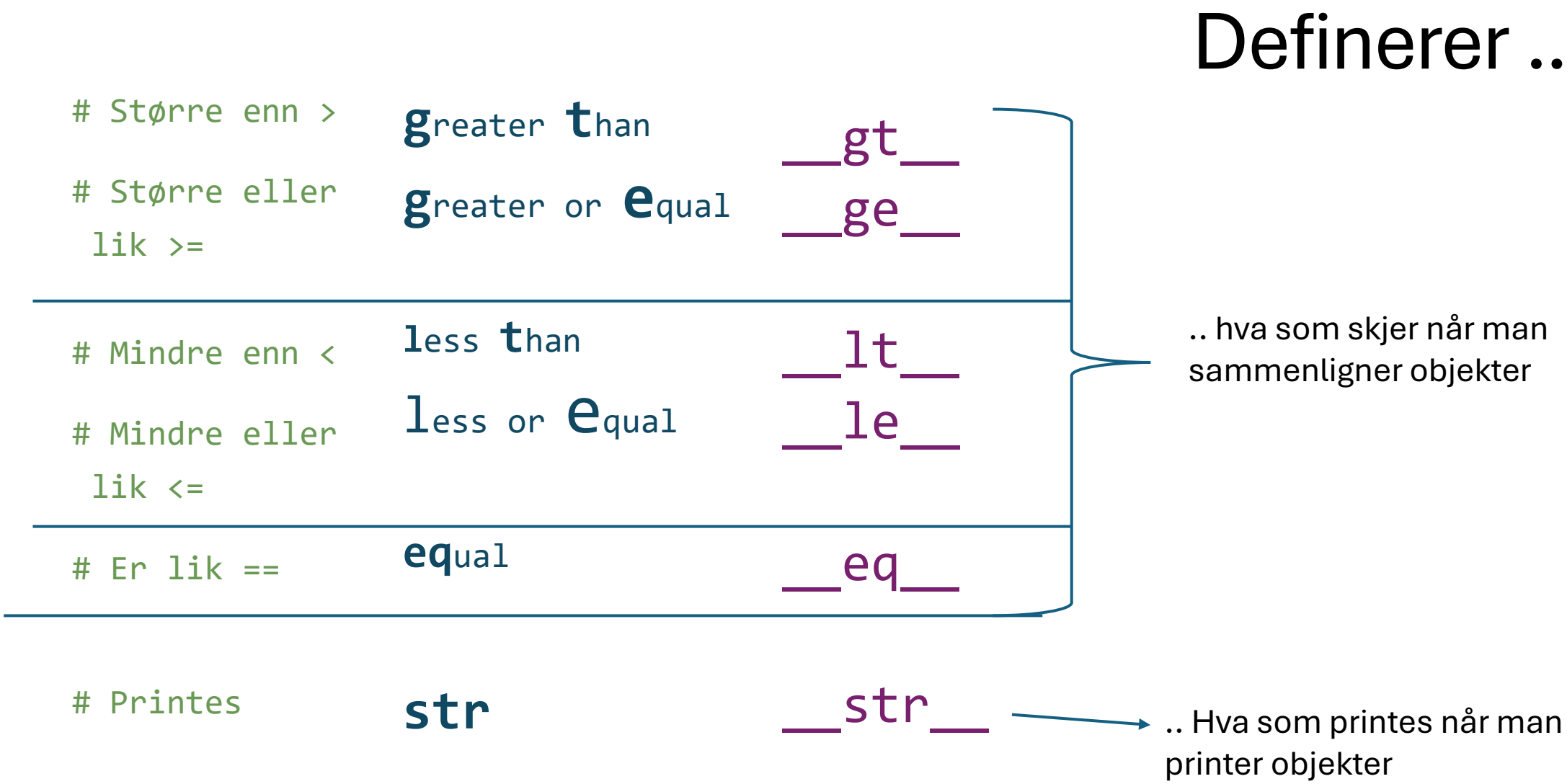
    def __init__(self, navn, brnavn, tittel, lønn):
        self._navn = navn
        self._brnavn = brnavn
        self._tittel = tittel
        self._lønn = lønn
```

```
class Student:

    def __init__(self, navn, brnavn, program):
        self._navn = navn
        self._brnavn = brnavn
        self._program = program
```


:Universitetet
_uninavn = "Universitetet i Oslo"
_ansatte =
_studenter =
:list
0 1
:Ansatt
_navn = "Helene"
_brnavn = "Helebro"
_tittel = "gruppeleder"
_lønn = 77
:Ansatt
_navn = "..."
_brnavn = "..."
_tittel = "..."
_lønn = 20
:list
0
:Student
_navn = "Tina"
_brnavn = "..."
_program = "Enkelt-fag"
Datastruktur

Noen magiske metoder



Menti

https://www.mentimeter.com/app/presentation/alwavh3tu8w8xrjf77zrz9oebfzg54jw/edit?source=share-modal

```
class Hund:
    def __init__(self, navn):
        self._navn = navn

    def __str__(self):
        return self._navn

p1 = Hund("Pluto")
p2 = Hund("Pluto")

print(p1 == p2)
```

False

```
class Hund:
    def __init__(self, navn):
        self._navn = navn

    def __eq__(self, annen_hund):
        return self._navn == annen_hund._navn

    def __str__(self):
        return self._navn


p1 = Hund("Pluto")
p2 = Hund("Pluto")

print(p1 == p2)
```

True

studentsystem.py

```
> class Universitet:···


> class Student:···

> class Ansatt:···


  uio = Universitet("Universitetet i Oslo")

  tast = ""
> while tast != "STOPP":···
```

```
class Universitet:

    def __init__(self, uninavn):
        self._uninavn = uninavn
        self._ansatte = []
        self._studenter = []
```

```
# Oppretter Ansatt-objekt og legger det til lista
    def legg_til_ansatt(self, navn, brnavn, stilling, lønn):
        ny_ansatt = Ansatt(navn, brnavn, stilling, lønn)
        self._ansatte.append(ny_ansatt)

    # Oppretter Student-objekt og legger det til lista
    def legg_til_student(self, navn, brnavn, program):
        ny_student = Student(navn, brnavn, program)
        self._studenter.append(ny_student)
```

```
# Henter informasjon om person med brukernavn, hvis finnes
  def hent_info(self, brnavn):
    info = "Ikke funnet"
    # Søker gjennom alle ansatte
    for ansatt in self._ansatte:
      if ansatt._brnavn == brnavn:
        info = "ANSATT FUNNET:\n"
        info += str(ansatt)
    # Søker gjennom alle studenter
    for student in self._studenter:
      if student._brnavn == brnavn:
        info = "STUDENT FUNNET:\n"
        info += str(student)
    # Info inneholder "ikke funnet", eller evt. informasjon om
    # en person vi har funnet.
    return info
```

```
class Student:

    def __init__(self, navn, brnavn, program):
        self._navn = navn
        self._brnavn = brnavn
        self._program = program

    def __str__(self):
        info = "Navn: " + self._navn
        info += "\nBrukernavn: " + self._brnavn
        info += "\nStudieprogram: " + self._program
        return info
```

```
class Ansatt:

    def __init__(self, navn, brnavn, tittel, lønn):
        self._navn = navn
        self._brnavn = brnavn
        self._tittel = tittel
        self._lønn = lønn
```

```
# Større enn >
  def __gt__(self, annen_ansatt):
    return self._lønn > annen_ansatt._lønn

  # Større eller lik >=
  def __ge__(self, annen_ansatt):
    if self > annen_ansatt or self == annen_ansatt:
      return True
    return False
```

```
# Mindre enn <
  def __lt__(self, annen_ansatt):
    return self._lønn < annen_ansatt._lønn

  # Mindre eller lik <=
  def __le__(self, annen_ansatt):
    if self < annen_ansatt or self == annen_ansatt:
      return True
    return False
```

```
# Er lik ==
  def __eq__(self, annen_ansatt):
    return self._lønn == annen_ansatt._lønn

  def __str__(self):
    info = "Navn: " + self._navn
    info += "\nBrukernavn: " + self._brnavn
    info += "\nTittel: " + self._tittel
    info += "\nLønn: " + str(self._lønn)
    return info
```

```
uio = Universitet("Universitetet i Oslo")

tast = ""
while tast != "STOPP":
    print("Menyvalg:")
    print("1. legg til student")
    print("2. legg til ansatt")
    print("3. søk etter person")
    print("STOPP for å avslutte")
    tast = input("Ditt valg: ").strip()
    if tast == "1":
        navn = input("Navn: ")
        brnavn = input("Brukernavn: ")
        program = input("Studieprogram: ")
        uio.legg_til_student(navn, brnavn, program)
    elif tast == "2":
        navn = input("Navn: ")
        brnavn = input("Brukernavn: ")
        stilling = input("Stilling: ")
        lønn = int(input("Lønn: "))
        uio.legg_til_ansatt(navn, brnavn, stilling, lønn)
    elif tast == "3":
        brnavn = input("Hvem søker du etter? Tast brukernavn: ")
        print(uio.hent_info(brnavn))
    print()
```