Menti

IN1000 gruppetime uke 9

Gruppe 4: Design

0

Menti

Plan for i dag

- Magiske metoder (__eq__, __str__)
- "is", "is not" og "in" når vi sammenligner klasse-objekter
- Eksempler med større strukturer

0

Magiske metoder __str__

- I likhet med **__init__** har vi andre "magiske metoder"
- **__str__** forteller python hva vi egentlig ønsker å printe ut
- Uten **__str__** vil python printe ut adressen til objektet

```python
class Person:
    def __init__(self, navn, alder):
        self._navn = navn
        self._alder = alder

    def __str__(self):
        pen_streng = f"Navn: {self._navn}, Alder: {self._alder}"
        return pen_streng

person = Person("Arkimedes", "9128")

# printer nå "Navn: Arkimedes, Alder: 9128"
print(person)

#Uten __str__ ville de printet ut:
# <__main__.Person object at 0x?????????>
```

Menti

0

Menti

Magiske metoder __eq__

- Klasser kan se så ulike ut at det er vanskelig å vite nøyaktig hva som gjør dem likt nok
- Hvis vi skal sammenligne om to objekter av samme klasse har like verdier må vi definere **__eq__**
- Hva som gjør to objekter av egendefinerte klasser like er opp til oss å bestemme med **__eq__**

```python
class Person:
    def __init__(self, navn, alder):
        self._navn = navn
        self._alder = alder

    def hent_navn(self):
        return self._navn

    def hent_alder(self):
        return self._alder

    def __eq__(self, annen_person):
        return ( self._navn == annen_person.hent_navn() and
                 self._alder == annen_person.hent_alder() )

# To individuelle objekter med like verdier
person1 = Person("Arkimedes", "9128")
person2 = Person("Arkimedes", "9128")

print(person1 == person2) # True

# Dette er egentlig hva som skjer når vi gjør ==
# Men ikke bruk __eq__(), bruk ==
print(person1.__eq__(person2))
```

0

Menti

Sammenligning med objekter av egendefinerte klasser

- Man kan også bruke **is** for å sammenligne objekter
- **is** vil sjekke om referansen er lik. Altså vil det ikke se på selve verdiene, men om det er nøyaktig samme objekt på samme adresse
- **is not** reverserer sannhetsverdien:
  - print(person1 **is not** person2) # True
  - print(person1 **is not** person3) # False

```
class Person:
    def __init__(self, navn, alder):
        self._navn = navn
        self._alder = alder

    def hent_navn(self):
        return self._navn

    def hent_alder(self):
        return self._alder

    def __eq__(self, annen_person):
        return ( self._navn == annen_person.hent_navn() and
                 self._alder == annen_person.hent_alder() )

# To individuelle objekter med like verdier
person1 = Person("Arkimedes", "9128")
person2 = Person("Arkimedes", "9128")

# Variabel som peker på nøyaktig samme objekt som person1
person3 = person1

print(person1 is person2) # False
print(person1 is person3) # True
```

0

Menti

Forskjellen mellom is og ==

- **is** for å sjekke samme referanse
- **==** for å sjekke samme verdi

0

Menti

Andre magiske metoder

- **__ne__**: Not equal, !=
- **__lt__**: Less than, <
- **__le__**: Less than equal, <=
- **__gt__**: Greater than equal, >
- **__ge__**: Greater than equal, >=
- **__add__**: +
- **__sub__**: -
- **__mul__**: *
- **__truediv__**: /

0

La oss "lage" en forenklet monopol

- Snakk med de rundt deg eller tenk høyt. Hva slags klasser ville man trengt for å lage monopol?
- La oss "lage" en forenklet monopol hvor man bare kjøper boliger (ingen andre ruter)
- Hvordan kan vi representere brettet vårt?

Menti
0

Menti

Hvilke klasser trenger vi?

