



Uke 14 - Repetisjonstime

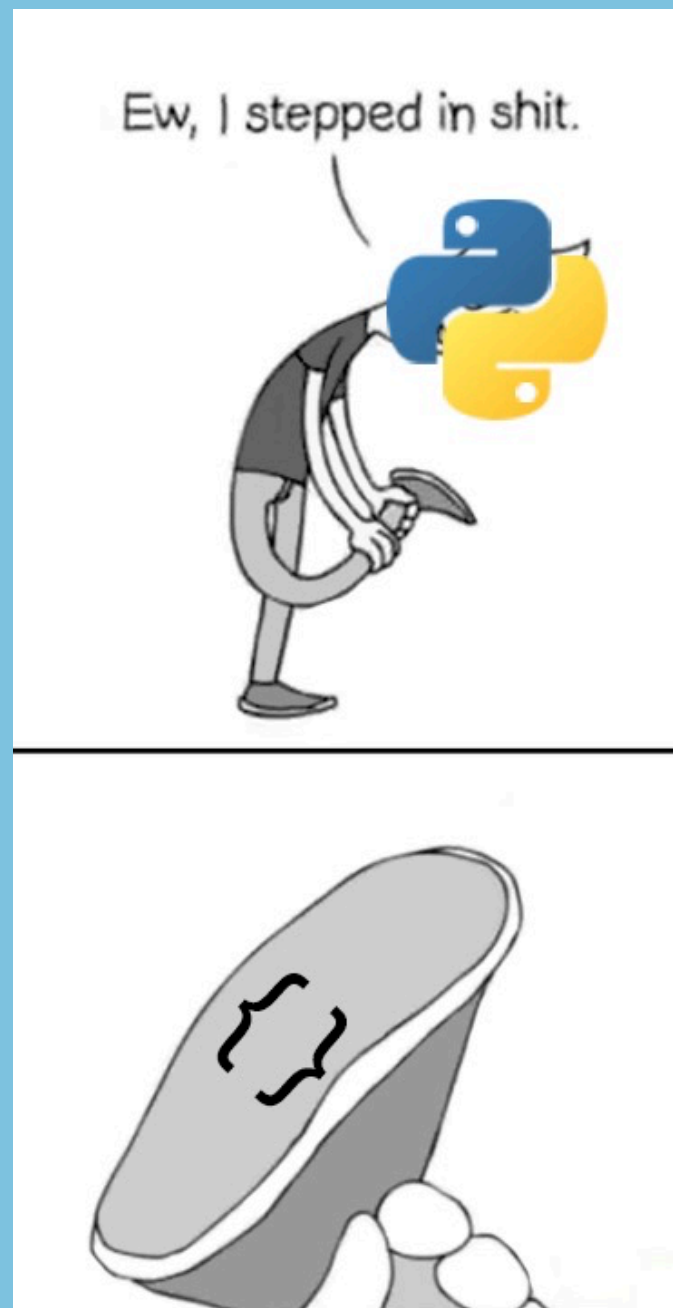
Prosedyrer, funksjoner og programflyt

IN1000 Høst 2025

JONAS



Agenda



Prosedyrer

funksjoner

programflyt

oppgaver i lag

Prosedyrer

Prosedyrer

En bit med kode som kjøres når den kalles på

Prosedyrer

En bit med kode som kjøres når den kalles på

Skriver koden her, så man kan gjenbruke det
istedenfor å skrive det flere ganger 🧠💥

Prosedyrer

En bit med kode som kjøres når den kalles på

Skriver koden her, så man kan gjenbruke det
istedenfor å skrive det flere ganger 🧠💡

På samme måte som vi lagrer verdier i en variabel,
så lagrer vi kode til en prosedyre

Prosedyrer og funksjoner

prosedyre og funksjoner
hva er forskjellen?

```
def prosedyre_med_parameter(en, to, tre):  
    print(en)  
    print(to)  
    print(tre)  
  
def prosedyre_uten_parameter():  
    print("Har ingen parameter å jobbe med")  
  
def funksjon_med_parameter(en):  
    return en  
  
def funksjoner_uten_parameter():  
    return "returverdi"
```

Prosedyrer

prosedyrer

```
def prosedyre_med_parameter(en, to, tre):  
    print(en)  
    print(to)  
    print(tre)  
  
def prosedyre_uten_parameter():  
    print("Har ingen parameter å jobbe med")
```


Prosedyrer

prosedyrer

returnerer ingenting

```
def prosedyre_med_parameter(en, to, tre):  
    print(en)  
    print(to)  
    print(tre)  
  
def prosedyre_uten_parameter():  
    print("Har ingen parameter å jobbe med")
```

funksjoner

funksjoner

```
def funksjon_med_parameter(en):  
    return en  
  
def funksjoner_uten_parameter():  
    return "returverdi"
```

funksjoner

funksjoner

metode med returverdi

```
def funksjon_med_parameter(en):  
    return en  
  
def funksjoner_uten_parameter():  
    return "returverdi"
```

funksjoner

funksjoner

metode med returverdi

for samme input, returneres samme
output, hver gang

```
def funksjon_med_parameter(en):  
    return en  
  
def funksjoner_uten_parameter():  
    return "returverdi"
```

funksjoner

funksjoner

metode med returverdi

for samme input, returneres samme
output, hver gang

(Hvis ikke sier vi at funksjonen har
sideeffekter)

```
def funksjon_med_parameter(en):  
    return en  
  
def funksjoner_uten_parameter():  
    return "returverdi"
```

parameter og argumenter



parameter og argumenter

```
def funksjon_med_parameter(liste, student):  
    liste.append(student)  
    return liste  
  
student_liste = ["Jonas", "Martin"]  
ny_student = "Pål"  
  
ny_liste = funksjon_med_parameter(student_liste, ny_student)
```

parameter og argumenter

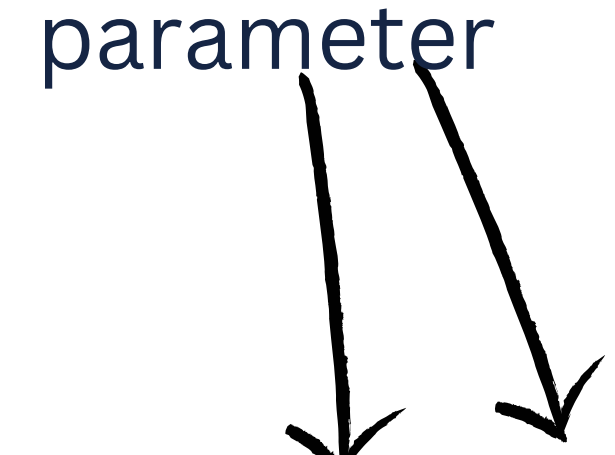
Hva er egentlig parameter og hva er argument her?

```
def funksjon_med_parameter(liste, student):  
    liste.append(student)  
    return liste  
  
student_liste = ["Jonas", "Martin"]  
ny_student = "Pål"  
  
ny_liste = funksjon_med_parameter(student_liste, ny_student)
```


parameter og argumenter

Hva er egentlig parameter og hva er argument her?

parameter



```
def funksjon_med_parameter(liste, student):  
    liste.append(student)  
    return liste  
  
student_liste = ["Jonas", "Martin"]  
ny_student = "Pål"  
  
ny_liste = funksjon_med_parameter(student_liste, ny_student)
```

parameter og argumenter

Hva er egentlig parameter og hva er argument her?

```
def funksjon_med_parameter(liste, student):  
    liste.append(student)  
    return liste  
  
student_liste = ["Jonas", "Martin"]  
ny_student = "Pål"  
  
ny_liste = funksjon_med_parameter(student_liste, ny_student)
```

parameter

argument

parameter og argumenter

Hva er egentlig parameter og hva er argument her?

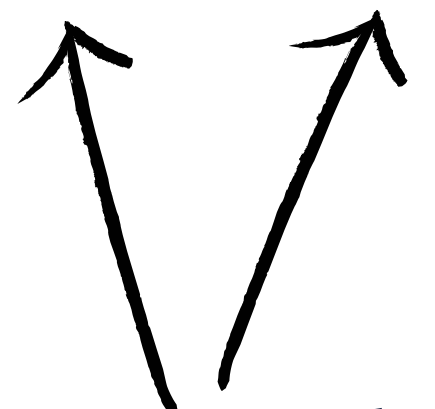
parameter = navnet i funksjonsdefinisjonen

```
def funksjon_med_parameter(liste, student):  
    liste.append(student)  
    return liste  
  
student_liste = ["Jonas", "Martin"]  
ny_student = "Pål"  
  
ny_liste = funksjon_med_parameter(student_liste, ny_student)
```

parameter



argument



parameter og argumenter

Hva er egentlig parameter og hva er argument her?

parameter = navnet i funksjonsdefinisjonen

argument = verdien du sender inn når du kaller funksjonen



The diagram illustrates the relationship between parameters and arguments in a Python function. At the top, the word "parameter" has two arrows pointing down to the parameters "liste" and "student" in the function definition. At the bottom, the word "argument" has two arrows pointing up to the arguments "student_liste" and "ny_student" in the function call. The code is as follows:

```
def funksjon_med_parameter(liste, student):  
    liste.append(student)  
    return liste  
  
student_liste = ["Jonas", "Martin"]  
ny_student = "Pål"  
  
ny_liste = funksjon_med_parameter(student_liste, ny_student)
```

Programflyt

I hvordan rekkefølge kjøres kode?

Fra topp til bunn, linje for linje

Når du kaller en funksjon, så hopper
man inn i den

Funksjoner kan kalle andre funksjoner

