skop!?

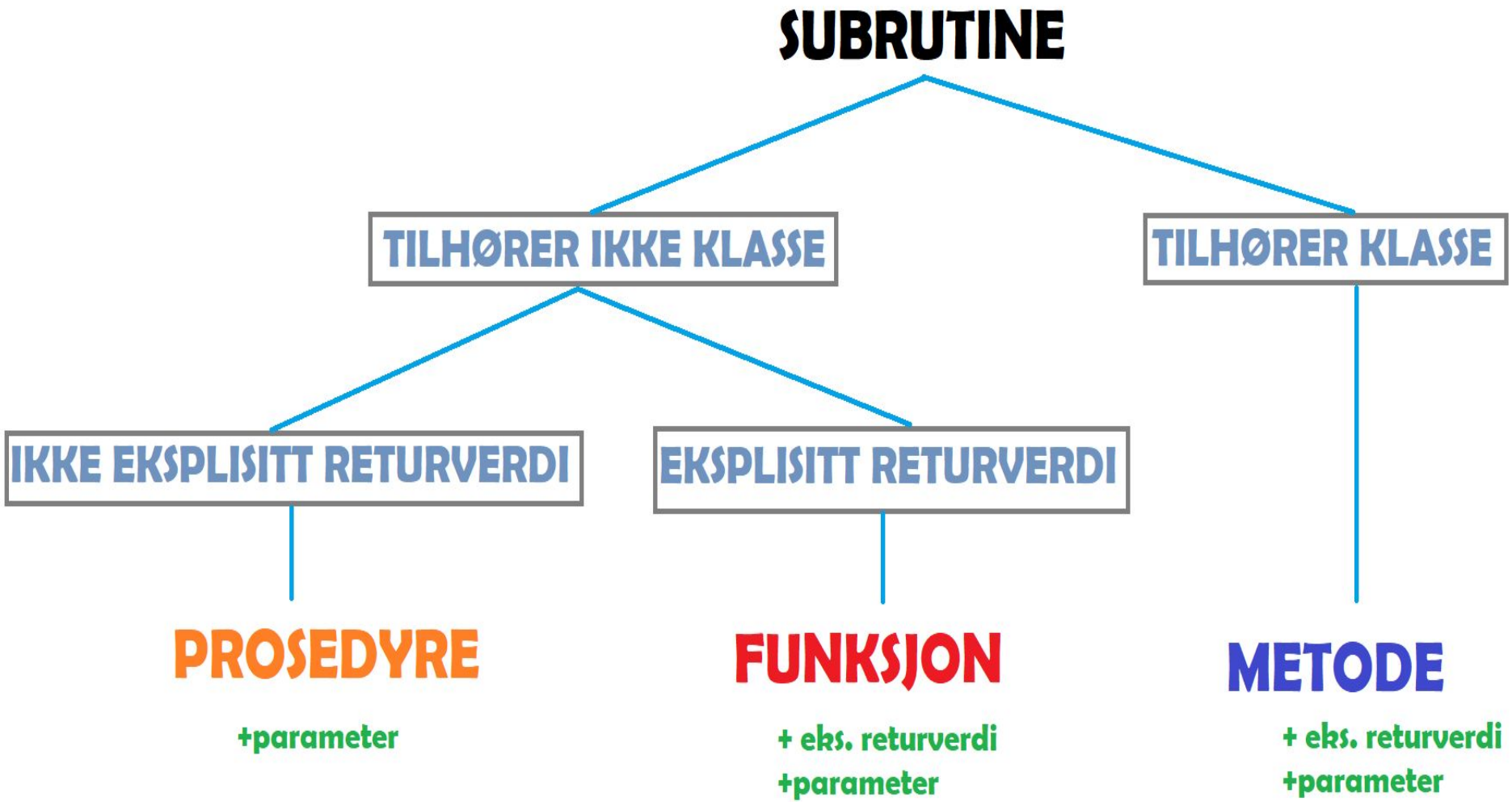
Metoder, funksjoner, programflyt og skop

IN1000, repetisjonskurs, 2017
(Delvis gjentatt fra forrige års foiler)

Kahoot

Hvorfor subrutiner?

- subrutiner brukes for å dele opp oppgaver i mindre biter
- brukes for å unngå å repetere kode
- endrer kontrollflyten
- enklere å holde styr på koden
- innebygde metoder


SUBRUTINE
TILHØRER IKKE KLASSE
TILHØRER KLASSE
IKKE EKSPLISITT RETURVERDI
EKSPLISITT RETURVERDI
PROSEDYRE
+parameter
FUNKSJON
+ eks. returverdi
+parameter
METODE
+ eks. returverdi
+parameter

Prosedyrer

- ...kaller vi subrutiner uten eksplisitt returverdi
- de returnerer allikevel None implisitt
- Syntaks:
  - def X:
- Brukes typisk til hovedprogram, menyer
- Vi kan se at de returnerer None ved å skrive ut kallet, print() er et eksempel.

```
>>> print("hei")
hei
>>> print(print("hei"))
hei
None
```

Eksempel på en prosedyre

```
def hei():
    navn = input("Hva heter du?\n")
    print("Hei, " + navn + "! Hyggelig å hilse på deg.")
```

Hva er et **kall**? Hva skjer hvis vi **kaller** på denne prosedyren?

Funksjoner

- Har en eksplisitt returverdi (return)
- Brukes gjerne når vi vil gjøre noe spesielt med en verdi, som å:
  - utføre matematiske operasjoner (spesielt utover de som er innebygde i Python)
  - jobbe med strenger
  - sjekke ting i lister
  - +++
-

Eksempel på to funksjoner

```
def gange(tall1, tall2):
    return tall1 * tall2
```

```
def fakultet(tall):
    resultat = 1
    for i in range(tall):
        resultat *= i + 1
    return resultat
```

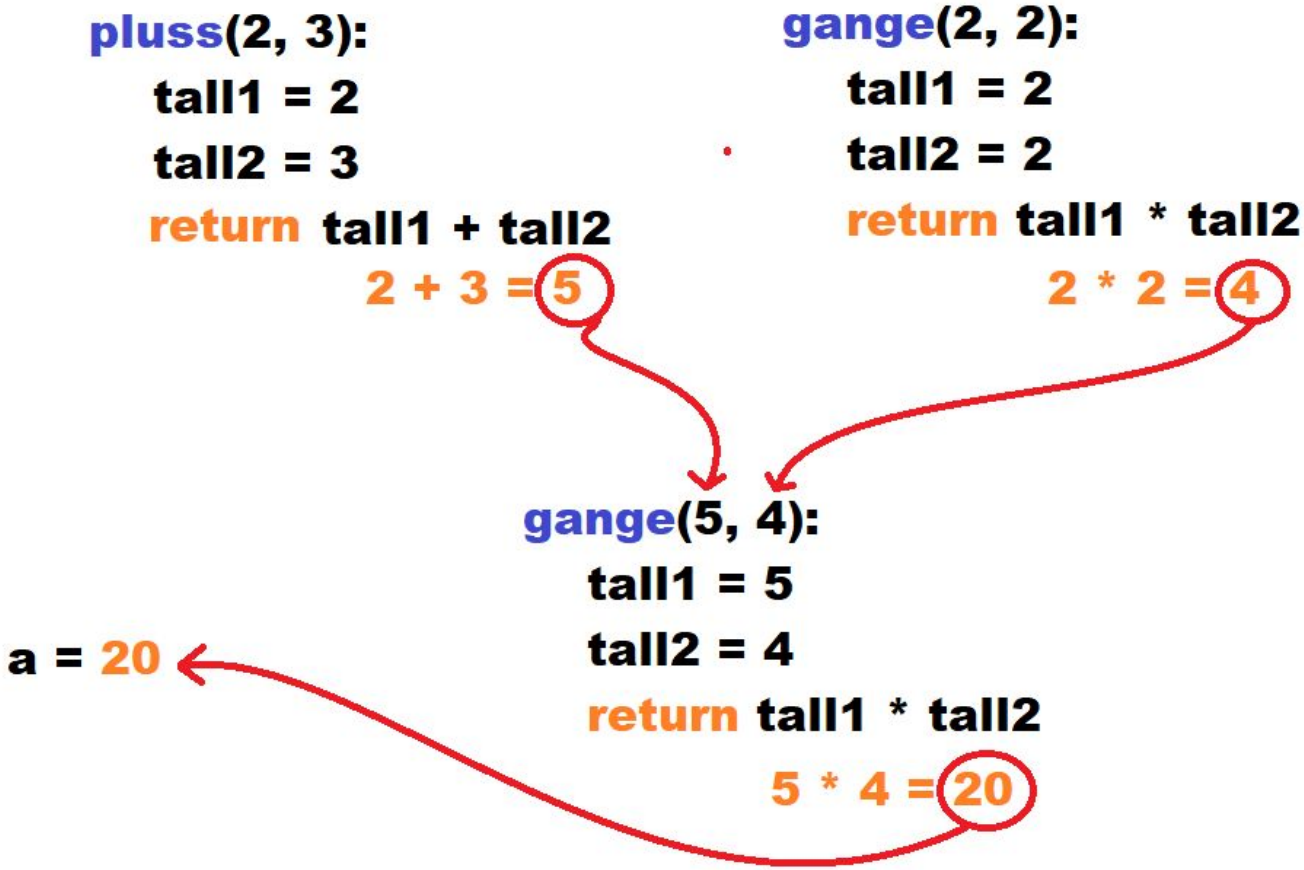
Et kall på en funksjon evalueres til returverdien

Vi må evaluere uttrykkene “innenifra” først.

```
def gange(tall1, tall2):
    return tall1 * tall2
```

```
a = gange(pluss(2,3), gange(2,2))
```

```
def pluss(tall1,tall2):
    return tall1 + tall2
```


pluss(2, 3):
tall1 = 2
tall2 = 3
return tall1 + tall2
2 + 3 = 5
gange(2, 2):
tall1 = 2
tall2 = 2
return tall1 * tall2
2 * 2 = 4
gange(5, 4):
tall1 = 5
tall2 = 4
return tall1 * tall2
5 * 4 = 20
a = 20

Metoder

- Hører til en klasse
- Tar alltid inn **self** som parameter
- Er ellers som prosedyrene og funksjonene vi har sett før

Eksempel

```
class Hund:
    def __init__(self, alder):
        self._alder = alder

    def hentAlder(self):
        return self._alder
```

SKOP

- rekkevidden til en variabel eller en subrutine
- henger i stor grad sammen med indentering (!)
- parametre har skop inne i subrutiner
-

Variabler som brukes inne i en subrutine

```
liste = ['kake','egg','svele']

def spis(mat):
    print("jeg spiste",mat)

print(mat)
```

```
Traceback (most recent call last):
  File "C:/Users/Petter/Documents/partalltest.py", line 20, in <module>
    print(mat)
NameError: name 'mat' is not defined
```

PROGRAMFLYT

- Det er flere måter å kontrollere programflyt på
- if, elif, else
- løkker
- input
- subrutiner

Eksempel

Enten if eller else blir utført, de blir ikke utført begge to.

```
a = 3
b = 4

def partall(tall):
    if tall % 2 == 0:
        return True
    else:
        return False

print(partall(a))

print(partall(b))
```