🪙 Myntkast og sannsynlighet – del 3: Utforsk og analyser

LØSNINGSFORSLAG

6.–8. trinn · 35 min · SkapLab – Python

Hva du skal lære

Bruke simuleringen til å utforske sannsynlighetsbegrepet og store talls lov: jo flere kast, desto nærmere 50 % nærmer vi oss.

Kodeeksempel – stor talls lov

```
import random

for antall in [10, 100, 1000, 10000, 100000]:
    kron = sum(random.randint(0, 1) == 0 for _ in range(antall))
    prosent = kron / antall * 100
    avvik   = abs(prosent - 50)
    print(f"{antall:>8} kast: {prosent:5.1f} % kron  (avvik: {avvik:.1f} %)")
```

Oppgaver

1. Kjør koden. Nærmer avviket seg 0 med flere kast?
2. Kjør den 5 ganger. Er det alltid samme mønster?
3. Lag en ordentlig simulering av Lotto 7 av 34: trekk 7 tilfeldige tall.
4. Beregn: hvor mange kast trengs i gjennomsnitt for å få 5 kron på rad?
5. Lag en funksjon som simulerer terningkast og finner gjennomsnittet.
6. Utfordring: simuler sannsynligheten for å kaste minst én sekser på tre terninger.

Løsningsforslag – terningkast-gjennomsnitt

```
import random

def simuler_terning(antall_kast):
    total = sum(random.randint(1, 6) for _ in range(antall_kast))
    return total / antall_kast

for n in [100, 1000, 100000]:
    snitt = simuler_terning(n)
    print(f"{n:>8} kast: snitt = {snitt:.3f}  (teoretisk: 3.500)")
```

```
# Sannsynlighet for minst én sekser på tre terninger
treff = 0
forsøk = 100000
for _ in range(forsøk):
    if any(random.randint(1, 6) == 6 for _ in range(3)):
        treff += 1
print(f"\nSjanse for minst én sekser på 3 terninger: {treff/forsøk*100:.1f} %")
print("Teoretisk: 42.1 %")
```

> *any() returnerer True hvis minst ett element i en sekvens er True. sum() med en generator er en kompakt måte å summere opp mange verdier på.*