Excited

Senter for fremragende IT-utdanning

# Digital eksamen i programmering

## Analyse av oppgavesjangre

*Guttorm Sindre*

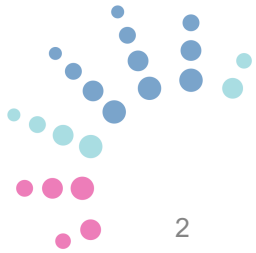
NORD University

Centre for Excellence in Education

NTNU

# Kontekst:
# TDT4127 Programmering og numerikk

- Tas av studenter som begynner i 4.årskurs ved NTNU
- To deler
  - Grunnleggende programmering (Python), 2/3
  - Numerikk, 1/3
- Læringsutbytter
  - Kunnskap om grunnelementer i prosedyreorientert programmering
  - Kunnskap om grunnleggende numeriske metoder
  - Ferdigheter til å løse generelle programmeringsproblem og numeriske problemer med koding
- Forkunnskapskrav
  - Nok matematikk for opptak til 2-årig siv.ing.
  - Derimot **ingen** forkunnskapskrav mhp IT / programmering

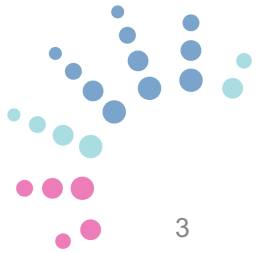


# Mange krav til eksamen / vurderinger

Bør være

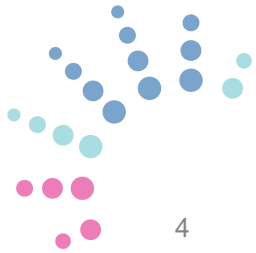
- gyldig
- pålitelig
- transparent
- autentisk
- motiverende for dyp læring
- rettferdig
- formativ
- kjapp, inkrementell
- opprettelig
- krevende
- effektiv, bærekraftig

(Race, Brown & Smith, 2004, pp.2-4)



# Inspera

- Oppgavetype «Programmering»
  - Gir syntaks-farging
  - Men **ikke** mulighet til å kompilere, kjøre, teste koden
  - Manuelt rettet
- Fordeler
  - Mer autentisk enn penn og papir
  - Sensor slipper å tyde håndskrift
- Ulemper
  - Ikke veldig autentisk
  - Tidkrevende for studenter som er
    - trege på tastaturet
    - sliter med å huske syntaks
  - Dårlig dekning på eksamen av kort varighet
  - Tidkrevende å rette
- Interessant å se på miks med autorettede oppgaver



# Relatert forskning

- Klassifisering av eksamensspørsmål i programmering (Sheard et al., 2011)
- Koding vs. Komplettering (Van Merrienboer & De Croock, 1992)
- «Parsons problems» (dra og slipp) (Parsons & Haden, 2006; Denny et al., 2008; Ericson et al., 2017)
- Enkle spørsmål -> gradvis vanskeligere (Zingaro et al., 2012)

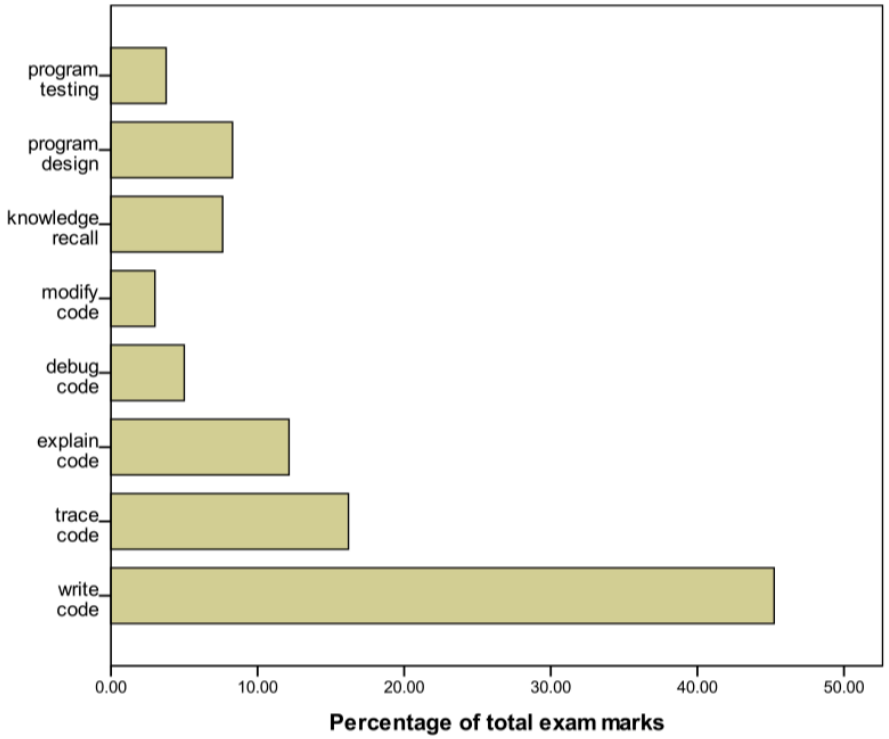
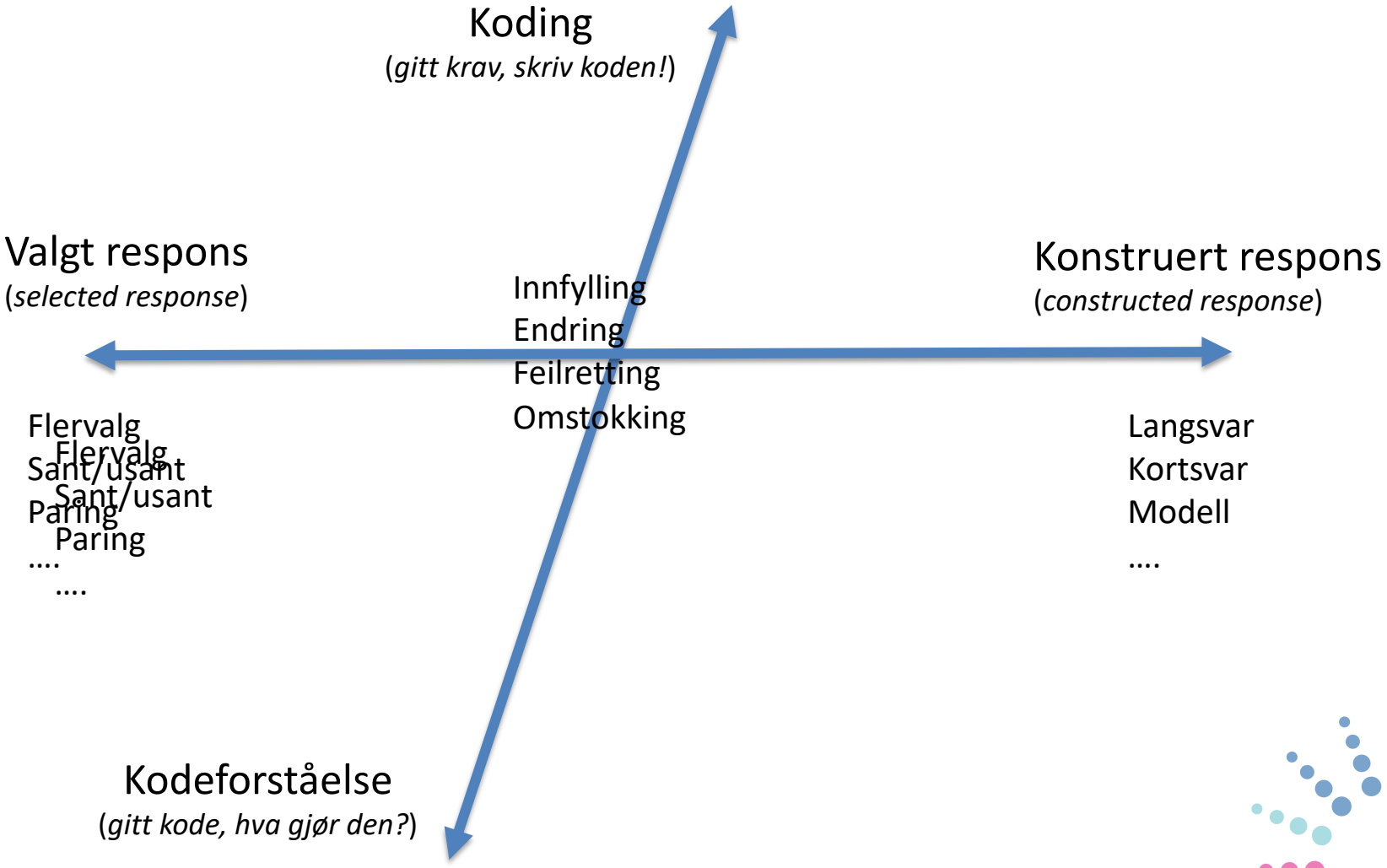



**Figure 1: Skills required to answer questions**

# Oppgavesjangre

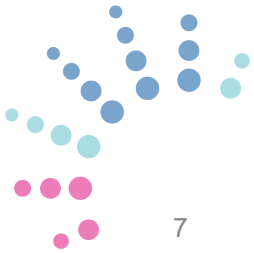


# Oppgaver, TDT4127 (1)

- **Brukt 6 av 10 autorettede sjangre i Inspera**
  - **Samt 2 manuelle**

Tabell 1: Oversikt over oppgaver

| Nr | Vekt% | Sjanger | Pensumdel | Inspera-Type | Retting |
|---|---|---|---|---|---|
| 1 | 5 | Teori | Programmering | Flervalg | Auto |
| 2 | 10 | Teori | Numerikk | Flervalg | Auto |
| 3 | 3 | Forstå kode | Numerikk | Sant/usant | Auto |
| 4 | 3 | Finn feil | Numerikk | Tekstfelt | Manuell |
| 5 | 4 | Finn feil | Programmering | Flervalg | Auto |
| 6 | 5 | Forstå kode | Programmering | Paring | Auto |
| 7 | 5 | Fullfør kode | Programmering | Dra og slipp | Auto |
| 8 | 5 | Fullfør kode | Programmering | Nedtrekk | Auto |
| 9 | 5 | Fullfør kode | Programmering | Nedtrekk | Auto |
| 10 | 5 | Skriv kode | Programmering | Programmering | Manuell |
| 11 | 5 | Fullfør kode | Programmering | Dra og slipp | Auto |
| 12 | 5 | Skriv kode | Numerikk | Programmering | Manuell |
| 13 | 5 | Fullfør kode | Numerikk | Dra og slipp | Auto |
| 14 | 5 | Skriv kode | Numerikk | Programmering | Manuell |
| 15 | 5 | Fullfør kode | Numerikk | Nedtrekk | Auto |
| 16 | 5 | Skriv kode | Numerikk | Programmering | Manuell |
| 17 | 5 | Skriv kode | Numerikk | Programmering | Manuell |
| 18 | 4 | Skriv kode | Programmering | Programmering | Manuell |
| 19 | 3,5 | Fullfør kode | Programmering | Fyll inn tekst | Auto |
| 20 | 7,5 | Skriv kode | Programmering | Programmering | Manuell |



# #6, Kodeforståelse ved paring

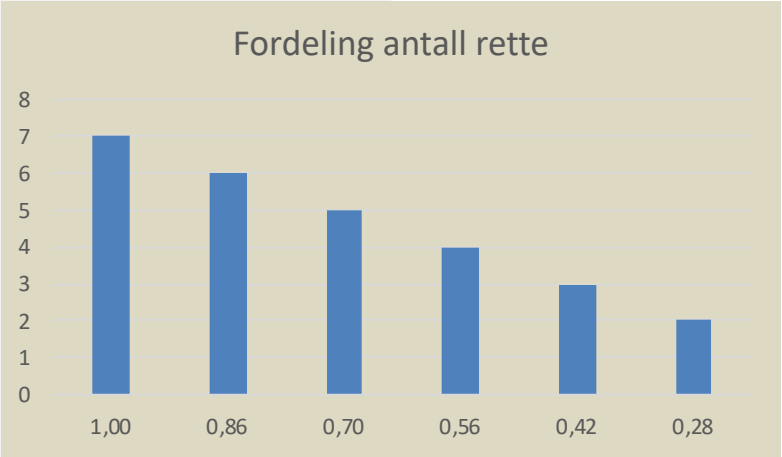
## Lister, mengder, in / Lists, sets, in (5%)

VI har definert tre globale variable:

- A = {0,1,2,3}
- B = {3,4,5}
- C = set(range(8))

Vi har dessuten funksjonen **count_list(lst)** gitt som følger:

```
def count_list(lst):
    result = [0]*5
    for s in lst:
        for i in range(1,6):
            if i in s:
                result[i-1] += 1
    return result
```

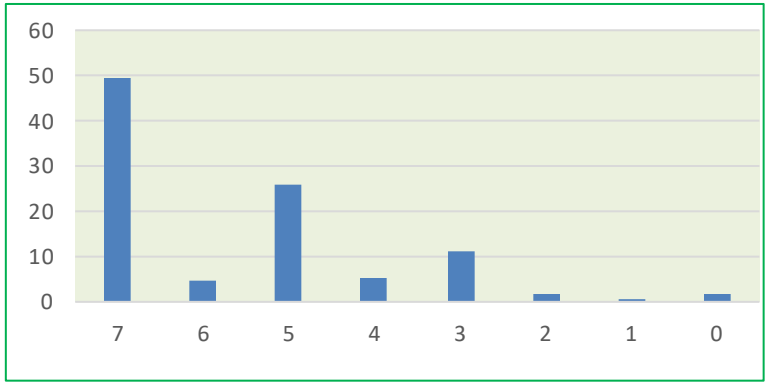



**Hver rad-tittel i tabellen er et mulig argument til funksjonen count_list(). Hver kolonneoverskrift er en mulig returverdi. Marker hvilket argument som fører til hvilken returverdi.**

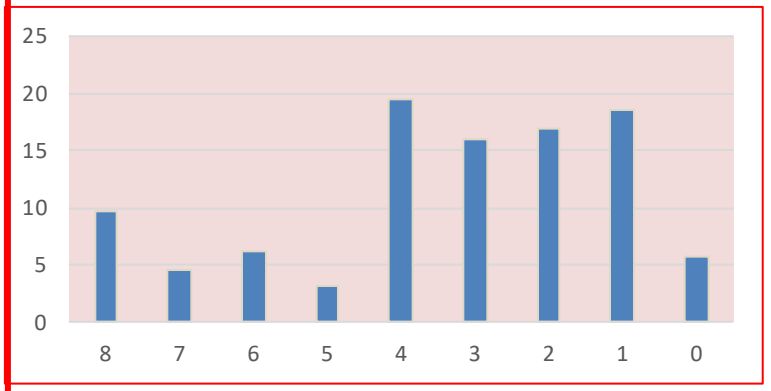
| | [1,1,1,1,1] | [2,2,3,2,2] | [0,0,0,0,0] | [1,1,1,0,0] | [6,6,6,6,6] |
|---|---|---|---|---|---|
| [A] | ○ | ○ | ○ | ○ | ○ |
| [C]*3+[C]*3 | ○ | ○ | ○ | ○ | ○ |
| [A,B,C] | ○ | ○ | ○ | ○ | ○ |
| [A.difference (C)] | ○ | ○ | ○ | ○ | ○ |
| [A.union(B)] | ○ | ○ | ○ | ○ | ○ |

# Omstokking (dra og slipp)

```
exp = len(poly) - 1
result.append(num * exp)
exp -= 1
for num in poly[:-1]:
return result
result = [ ]
def deriv(poly):
```



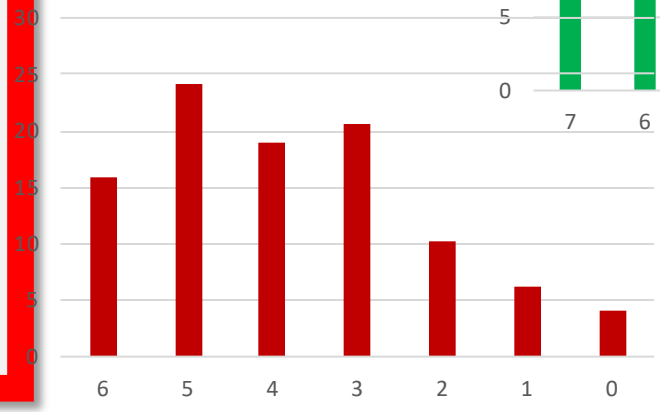

```
def poly_list(expr):
    result = [ ]
    start = 0
result.append(expr[start:len(expr)])
start = i + 1
start = i
if expr[i] == '+':
else:
if expr[i] == '+' or expr[i] == '-':
for i in range(1, len(expr)):
result.append(expr[start:i])
    return result
```

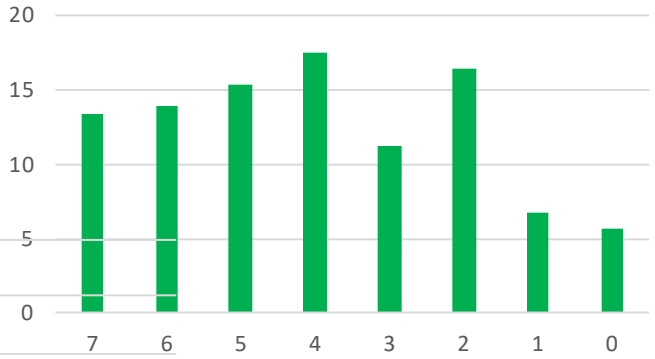


# Nedtrekk og innfylling

```
def poly_part(c, n):
    result = ""
    if c != 0:
        if n == 0:
            result = str(c)
        else:
            if c == -1:
                return "-x**n"
            elif c =
                result = "-x"
                return "-x"
            elif :
                result =
            if n > 1:
    return result
```

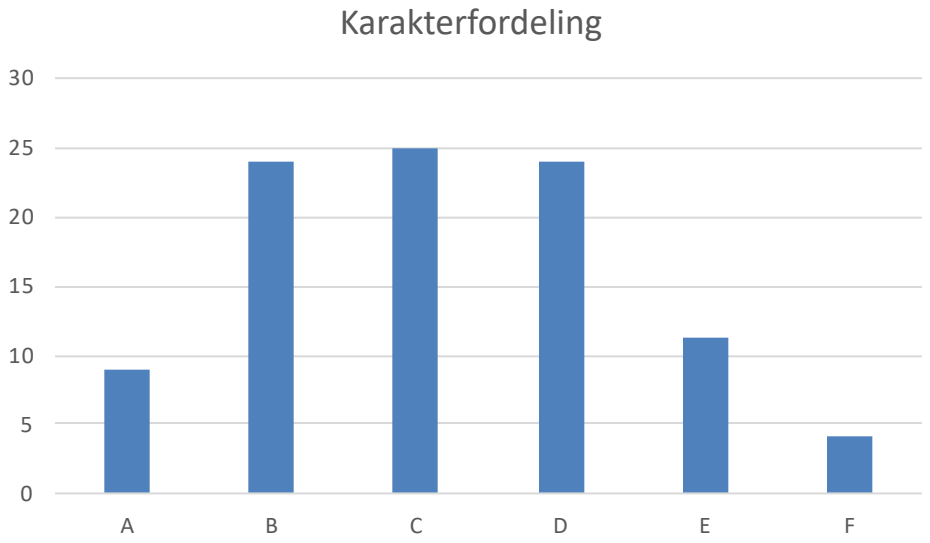
```
def is_unit_matrix(A):
    rows = len( A )
    for i in range(rows):
        if A[ i ] != [0] * i + [1] + [ 0 ] * (rows -i-1 ):
            return False
    return True
```



# Analyse av eksamensresultat (1)

- Karakterfordeling etter justering
  - Forteller ikke så veldig mye
- Korrelasjoner
  - Deloppgave – total
- Hvor godt skiller oppgaven?
  - Diskriminerings-indeks (D)



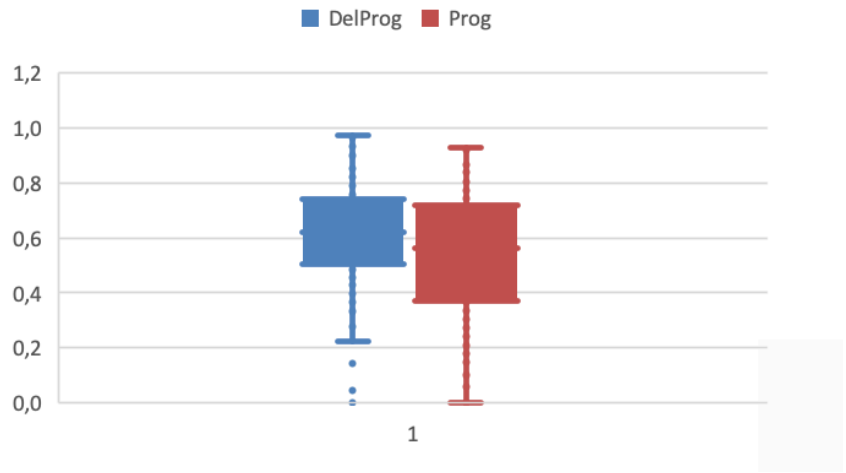

$$\text{Discrimination index} = 2 \times \frac{(H-L)}{N}$$

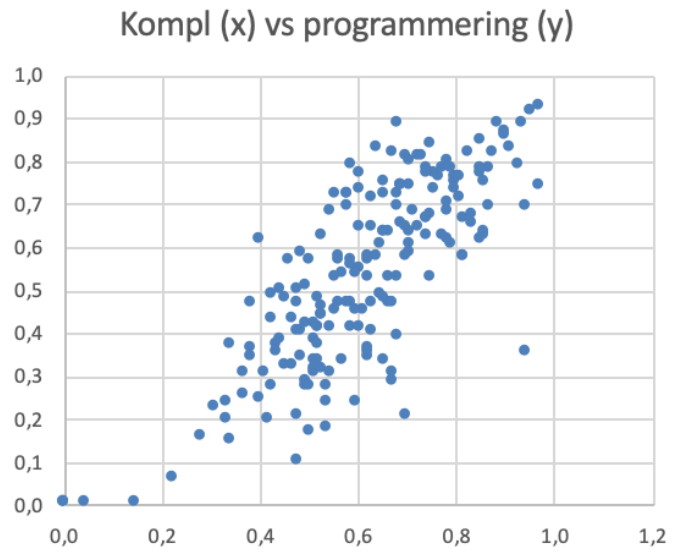
# Analyse av eksamensresultat (2)

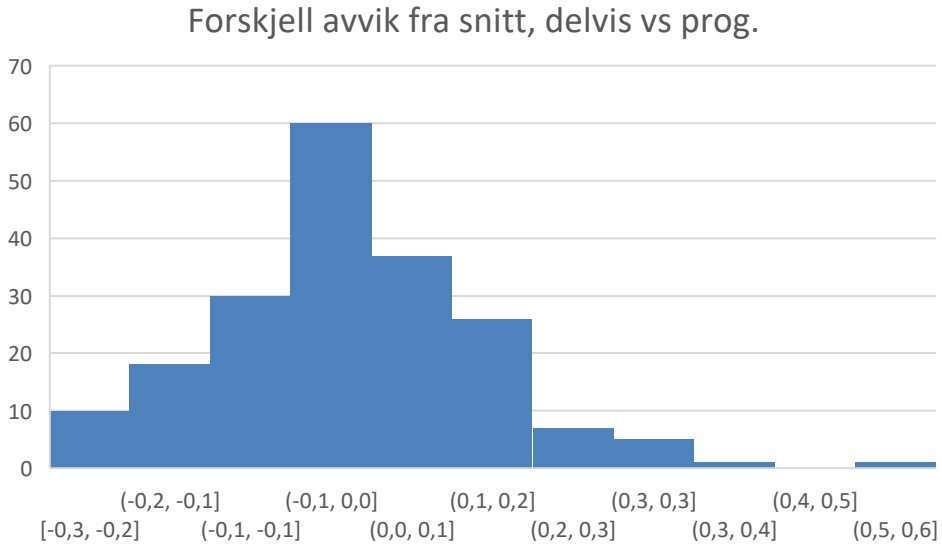
Tabell 2: Oppgavenes vanskegrad og annen statistikk

| Nr | IA-Type | MEAN | STDEV | CORREL | DISC |
|---|---|---|---|---|---|
| 14 | Programmering | 85 | 20 | 0,51 | 0,25 |
| 4 | Kortsvar | 82 | 24 | 0,59 | 0,36 |
| 7 | Dra og slipp | 81 | 22 | 0,58 | 0,33 |
| 2 | Flervalg | 74 | 19 | 0,56 | 0,27 |
| 15 | Nedtrekk | 73 | 20 | 0,54 | 0,26 |
| 13 | Dra og slipp | 71 | 30 | 0,41 | 0,34 |
| 1 | Flervalg | 69 | 20 | 0,33 | 0,16 |
| 8 | Nedtrekk | 65 | 26 | 0,59 | 0,36 |
| 6 | Paring | 62 | 35 | 0,59 | 0,53 |
| 16 | Programmering | 60 | 29 | 0,72 | 0,50 |
| 5 | Flervalg | 60 | 27 | 0,44 | 0,35 |
| 17 | Programmering | 58 | 30 | 0,67 | 0,51 |
| 19 | Fyll inn tekst | 58 | 28 | 0,63 | 0,43 |
| 3 | Sant / usant | 57 | 50 | 0,31 | 0,37 |
| 18 | Programmering | 55 | 31 | 0,74 | 0,57 |
| 10 | Programmering | 50 | 23 | 0,62 | 0,35 |
| 9 | Nedtrekk | 48 | 23 | 0,56 | 0,31 |
| 11 | Dra og slipp | 44 | 28 | 0,60 | 0,38 |
| 12 | Programmering | 43 | 29 | 0,67 | 0,47 |
| 20 | Programmering | 30 | 30 | 0,72 | 0,48 |

# Delvis koding vs koding

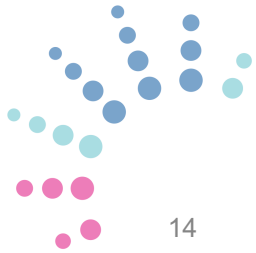


# Konklusjoner

- Autorettede oppgaver
  - Ikke for lette, skilte bra
  - God korrelasjon (0,8) med trad. programmeringsoppgaver
- Spesifikke anbefalinger
  - Fyll-inn: Kun for svært korte innfyllinger
    - Kan trenge kjapp manuell inspeksjon
  - Nedtrekk: bedre enn flervalg mhp å vise kontekst
  - Dra-og-slipp: kanskje mest elegant, men...
    - Tidkrevende å lage
    - Ikke egentlig ment for programmering
      - Noen svakheter i brukerinteraksjonen
    - Bør ta manuell inspeksjon for evt. delvis score
- Viktig at studentene har sett oppgavesjangrene før
  - Prøveeksamen eller lignende
- Ikke spesielt autentisk
  - Men kan passe som del av vurderingen særlig i intro-emner



# Referanser


- Race, P., S. Brown, and B. Smith, *500 tips on assessment*. 2004: Routledge.
- Parsons, D. and P. Haden. *Parson's programming puzzles: a fun and effective learning tool for first programming courses*. in *Proceedings of the 8th Australasian Conference on Computing Education-Volume 52*. 2006. Australian Computer Society, Inc.
- Van Merriënboer, J.J. and M.B. De Croock, *Strategies for computer-based programming instruction: Program completion vs. program generation.* Journal of Educational Computing Research, 1992. **8**(3): p. 365-394.
- Zingaro, D., A. Petersen, and M. Craig. *Stepping up to integrative questions on CS1 exams*. in *Proceedings of the 43rd ACM technical symposium on Computer Science Education*. 2012. ACM.
- Ericson, B.J., L.E. Margulieux, and J. Rick. *Solving parsons problems versus fixing and writing code*. in *Proceedings of the 17th Koli Calling International Conference on Computing Education Research*. 2017. ACM.
- Denny, P., A. Luxton-Reilly, and B. Simon. *Evaluating a new exam question: Parsons problems*. in *Proceedings of the fourth international workshop on computing education research*. 2008. ACM.
- Sheard, J., et al. "Exploring programming assessment instruments: a classification scheme for examination questions." *Proceedings of the seventh international workshop on Computing education research*. ACM, 2011.