Sensorveiledning

IN1000 vår 2026

Felles for seksjon 2, 3, 4, 5

Små skrive/slurvefeil overses (eks: mangler _ i et instansvariabelnavn eller en self-parameter), men gjennomgående feil som viser manglende forståelse skal gi trekk (ingen instansvariabler markert som non-public, alle/ de fleste metoder mangler self-parameter). Manglende lesbarhet gir trekk der det er uklart hva studenten faktisk kan. Begrens trekk for misforståelse av oppgaven om ellers god kode.

OBS: Syntaks på Oppgave 4

På hele oppgave 4 var støtte for Python-syntaks ikke skrudd på. Syntaksfeil som fargekoding av reserverte ord kunne forhindret må dermed godtas uten trekk. Det samme gjelder innrykksfeil.

Læringsmål/ aspekter i seksjon 2 og 5

I oppgave 3 og 5 vurderes spesielt de siste 3 punktene fra Læringsmålene for IN1000. Poeng i deloppgavene på 3 og 5 fordeles på følgende aspekter i henhold til tabellen nedenfor:

- **PRO**: løse PROblemer i python (fornuftig logikk/ fremgangsmåte) og
- **PRE**: PREsis implementasjon av fremgangsmåten

Oversikt oppgave 2 og 5: Poeng fordelt på deloppgave og aspekt

LUB / Deloppgave	Løse PROblemer i python	PREsis implementasjon av fremgangsmåten
2	**3**	**2**
5a	**2**	**1**
5b	**3**	**2**
Sum aspekt	**8**	**5**

Læringsmål/ aspekter i seksjon 3 og 4

Det er definert 3 aspekter på tvers av deloppgavene i oppgave 4.

Klassedefinisjon med konstruktør og metode («Definisjon»)

Definerer og initialiserer instansvariabler (inkl lister og ordbøker) som non-public med _, har med parameter self. Håndterer initialisering av instansvariabler både direkte fra parametere og konstanter, og ved hjelp av operasjoner som for eksempel å bygge opp en ordbok (*Hvordan* ordboken bygges opp vurderes under problemløsing og/ eller OO).

Definerer metoder og bruker parametere og returverdier riktig og på rett sted, inkl self. Angir eventuelle interne hjelpemetoder som non-public med _. __str__-metoder returnerer en streng, __len__-metoder returnerer et heltall, __iter__-metoder gir ut flere verdier med yield. Bruke riktig navn på magiske metoder.

Forstår prinsippene for objektorientert programmering og kan benytte disse til å skrive enklere objektorienterte programmer («OO»)

Bygger opp og benytter datastrukturen slik at den virker iht. oppgaven. Oppretter objekter på rett sted med argumenter av rett type, setter objekter riktig inn i datastrukturen. Skiller på strenger og referanser til objekter. Kaller korrekt på eksisterende metoder i andre objekter når dette er hensiktsmessig (også dersom disse ikke er implementert). Bruker delegering. Bevarer innkapsling. Oppfyller grensesnittet gitt i oppgaven (eventuelle ekstra public metoder/instansvariabler kan aksepteres uten trekk dersom studenten gir en rimelig begrunnelse for hvorfor de har gjort det.)

Problemløsing («Problem»)

Finner fremgangsmåter for å løse større og mindre algoritmiske utfordringer i en objektorientert system, og implementerer disse korrekt i Python. For eksempel oppbygging, aksess og traversering av lister og ordbøker, manipulering av indekser. Fil-innlesing og håndtering av input. Streng-manipulering og konvertering mellom typer. Riktige returverdier i ulike situasjoner. Trekk for feil og mangler som ikke faller inn under Def eller OO, men gjør at oppgaven ikke er fullstendig/ korrekt løst, kan foretas her.

Begrens trekk ved misforståelse av datastrukturen der det ellers skrives fornuftig kode som oppfyller tydelige krav.

Oversikt oppgave 3 og 4: Poeng fordelt på deloppgave og aspekt

Kort oppsummert: Den poengsummen en kandidat får på en deloppgave uttrykker hvor godt oppgaven er løst. Poengsum per aspekt på tvers av deloppgavene brukes i begrunnelser for å indikere sterke og svake sider hos kandidaten.

LUB / Deloppgave	Definere klasser	Problemløsning i et større (oo) system	OO - Objektorientert programmering
3a	4		
3b		1	1
3c	2	1	
3d		3	3
3e		2	3
4a	3	2	2
4b	3	2	
4c	5		2
4d	1	4	3
4e	1		6
4f			10
4g		2	4
Sum aspekt	19	17	34