NTNU
Kunnskap for en bedre verden

Sensorveiledning IDATx1001

Høst 2022

Om emnet IDATx1001 Programmering 1

Emnebeskrivelse

(https://www.ntnu.no/studier/emner/IDATA1001#tab=omEmnet)

Emnet er felles på tvers av campusene Gjøvik, Ålesund og Trondheim. Emnet har egen emnekode pr campus: IDAT**G**/IDAT**T**/IDAT**A** 1001, vi refererer til emnet som **IDATx1001**.

IDATx1001 Programmering 1 dekker følgende læringsutbytter i OOP:

- Klasser og objekter
- Datatyper, variabler og konstanter
- Metoder med og uten retur og parametre
- Betingelser (if- og switch uttrykk)
- Løkker (for-each, for og while), primært i forbindels emed **samlinger** (ArrayList, HashMap HashSet)
- Samlinger av objekter og klasser for å håndtere disse (ArrayList, HashMap og HashSet)
- Iterator-klassen som hjelp ifm å iterere over en samling
- Designprinsippene **coupling**, **cohesion** og **responsibility driven design**.
- Dokumentasjon, kodestil og robust kodedesign (studentene har krav på seg å kjøre **CheckStyle** for å sikre god kodestil.
- Testing (ikke enhetstesting/JUnit), debugging
- Enkelt tekstbasert/menybasert brukergrensesnitt.
- Grunnleggende

Emnet dekker **ikke**:

- Pakker i Java
- JUnit-testing
- Maven
- Unntakshåndtering (exceptions)
- Arv, polymorfi, interfaces
- Filhåndtering
- Grafisk Brukergrensesnitt

Lærebok og Verktøy/IDE

Lærebok benyttet i Ålesund og Gjøvik er «Objects First with Java» av David J. Burnes og Michael Kölling. Læreboken er utviklet sammen med det pedagogiske IDE'et BlueJ (https://www.bluej.org/).

I Trondheim benyttes annen lærebok.

I løpet av semesteret, går de fleste studentene over til profesjonelle IDE'er som IntelliJ, VSCode, Eclipse, Netbeans etc. (studentene velger selv).

Om Mappen

Mappen som er levert for sensur består av:

- En prosjektoppgave i programmering som kandidaten har jobbet **individuelt** med i 10 uker. Kandidaten har fått mulighet for **tilbakemelding muntlig** på sitt arbeid 3 ganger i løpet av de første 8 ukene, i tillegg til bistand i lab-timene ved behov. Prosjektoppgaven er lastet opp i Inspera sammen med denne sensurveiledningen.
  - Teller 70% av endelig karakter
- En rapport, der kandidaten skal beskrive sin løsning, hvilke valg kandidaten har gjort under utviklingen av løsningen, og hvilke endringer (refaktorering) som er gjort. Kandidaten skal også begrunne sitt design basert på teoriene rundt **coupling**, **cohesion** og **responsibility driven design**, med konkret henvisning til eksempler fra koden.
  - Teller 30% av endelig karakter

Kravspesifikasjonen til prosjektet er levert ut til kandidaten i 3 deler:

- Del 1 i uke 40 - Beskriver prosjektet i sin helhet, med hva man skal utvikle i løpet av prosjektet. For Del 1 skal kandidaten fokusere på å implementere **entitetsklassen** som skal representere en **vare**.
- Del 2 i uke 43 - Fokuserer på **vareregisteret**
- Del 3 i uke 45 - Med fokus på **brukerinteraksjon**. I denne delen åpnes også oppgaven opp betydelig og gir rom for at kandidaten skal foreslå egne utvidelser, egne endringer til opprinnelig kravspek osv. for å sette sitt eget personlige preg på løsningen.

***Prosjektoppgaven*:**

Kandidaten skal utvikle en applikasjon for et **Varehus**. Varehuset har et **vareregister** av **varer**. En **Vare** her representerer en **varetype**, som f.eks. "Laminatgulv bjørk". Antall enheter på lager av denne varen lagres som eget felt i vare-klassen.

Brukergrensesnittet er **tekstbasert**, og kan enten være en **meny** med ulike valg, eller implementert som **kommandoer/kommandolinje**.

Om *Verktøy*/IDE

Vi har ikke stilt krav om ett bestemt IDE. Studentene får selv velge hvilket IDE de ønsker å bruke. Følgende IDE'er benyttes typisk av studentene:

- BlueJ (https://www.bluej.org/)
- IntelliJ
- VSCode
- Netbeans

Dette medfører også at mappestruktur/filstruktur studentene imellom vil variere avhengig av hvilket IDE de har benyttet.

NTNU
Kunnskap for en bedre verden

Karakterskala ved NTNU

Følgende karakterskala benyttes ved NTNU:

Symbol	Betegnelse	Generell, ikke fagspesifikk beskrivelse av vurderingskriterier
A	Fremragende	Fremragende prestasjon som klart utmerker seg. Kandidaten viser svært god vurderingsevne og stor grad av selvstendighet.
B	Meget god	Meget god prestasjon. Kandidaten viser meget god vurderingsevne og selvstendighet.
C	God	Jevnt god prestasjon som er tilfredsstillende på de fleste områder. Kandidaten viser god vurderingsevne og selvstendighet på de viktigste områdene.
D	Nokså god	En akseptabel prestasjon med noen vesentlige mangler. Kandidaten viser en viss grad av vurderingsevne og selvstendighet.
E	Tilstrekkelig	Prestasjonen tilfredsstiller minimumskravene, men heller ikke mer. Kandidaten viser liten vurderingsevne og selvstendighet.
F	Ikke bestått	Prestasjon som ikke tilfredsstiller de faglige minimumskravene. Kandidaten viser både manglende vurderingsevne og selvstendighet.

Sensurskjema

Her følger sensurskjema som er lagt til grunn for sensureringen av mappene ved alle 3 campus (Trondheim, Gjøvik, Ålesund).

Poenger og vekting er **veiledende** og ingen «eksakt matematisk vitenskap». Det vil alltid gjøres en total helhetsvurdering av kandidatens besvarelse før endelig karakter settes.

Sensurskjema er delt inn i 3 deler:

- Del 1 – Funksjonelle krav, og brukervennlighet
- Del 2 – Implementasjon/kode/løsning
- Del 3 – Rapport

NTNU
Kunnskap for en bedre verden

Del 1 - Funksjonelle krav

Funksjonelle krav		**Vekt**
		10%
*Er de **funksjonelle kravene** i kravspesifikasjonen oppfylt (del 1 - 3) ?*	*Poeng:* *- 2p - Svært godt løst* *- 1p - Greit løst*	
Bruker kan liste alle varer i vareregisteret		
Bruker kan legge til ny type vare i vareregisteret		
Bruker kan søke etter vare basert på varenummer		
Bruker kan søke på vare basert på beskrivelse	2p om "fri-søk", beskrivelse inneholder...	
Bruker kan øke og redusere (endre) varebeholdningen til en vare		
Bruker kan slette en varetype fra registeret		
Bruker kan endre detaljer om en vare, som **rabatt**, **pris** og/eller **beskrivelse**		
Brukervennlighet/UI-design		**Vekt**
		5%
Prosjektet skal leveres med et tekstbasert brukergrensesnitt. *Følgende skal vurderes i forhold til brukervennlighet og design av brukergrensesnitt:*	*Poeng:* *- 2p - Svært godt løst* *- 1p - Greit løst*	
Fremstår brukerinteraksjonen som "brukervennlig"/intuitivt?		
Er det greit å forstå informasjonen som presenteres?		
Presenteres vareregisteret på en oversiktlig måte (som en strukturert tabell)?		
Ved registrering av ny vare, håndteres feil inntasting fra bruker på en god måte?		

Del 2 - Implementasjon/Design

KRITERIUM	Ikke bestått (F) 0 - 35p	Svak (D, E) 40 - 45 - 50 - 55p	Middels (C) 60 - 65 - 70 - 75 - 80p	Utmerket (B, A) 85 - 90 - 95 - 100p	Vekt
Grunnleggende OOP					**20 %**
Kan kandidaten deklarere en klasse?	Kandidaten viser betydelig manglende forståelse for hvordan deklarere en klasse	Kandidaten viser noen mangler, f.eks. ved at: • Mangler tilgansmodifikator (access modifier) på felt • Roter med store- og små bokstaver i klasse-, felt- og metodenavn	Kandidaten deklarerer klasser med: • Klassenavn som starter med stor bokstav • Felt som er satt **private** • Minimum **en** konstruktør • Metoder starter med liten bokstav	Kandidaten har i tillegg vist: • Validerer parametere i konstruktør (eller i set-metoder) • Bruk av set-metoder i konstruktør • Implementert flere konstruktører der det er hensiktsmessig (klassen Vare f.eks.)	
Datatyper og variabler		Kandidaten viser noen manglende forståelse for datatyper: • Velger å bruke **String** der det typisk burde vært brukt **int** eller **double** eller en **klasse-type** (spesielt i retur fra metoder)	Kandidaten benytter fornuftige datatyper til felt- og variabler: • Verdier som bør være flyttall implementeres med **double** (eller **float**), som f.eks. Størrelse på på vare • Heltallsverdier håndteres hovedsakelig som **int**	Kandidaten har: • benyttet **Enum** for kategori. • benytte **final** der det er hensiktsmessig. • benytter datatyper som f.eks. **BigDecimal** for å håndtere desimaltall riktigere enn det **double** eller **float** gjør.	
Get- og set-metoder		Kandidaten viser noe manglende forståelse for fornuftig bruk av get- og set metoder: • Har implementert get- og public set metoder for samtlige felt helt ukritisk. Ingen vurdering av hvilke set-metoder som **det er naturlig** gjøres tilgjengelig (public) • Har ikke, eller i svært liten grad validert parametre i set-metodene • Benytter ikke seg av set-metoder i konstruktør.	Kandidaten har en fornuftig bruk av set- og get metoder: • En get-metode pr felt det er naturlig å gjøre tilgjengelig • Kun public set-metoder for felt det er naturlig at skal endres etter at objektet er opprettet. • Noen grad av validering av parametre.	Kandidaten har i tillegg: • Set-metoder for **samtlige** felt i klassen, **MEN** kun et fåtall er satt til **public** • Set-metoder der det **alltid** utføres validering av parameter • Aktiv bruk av set-metoder i konstruktør(ene) for å redusere/forhindre duplisering av kode.	

KRITERIUM	Ikke bestått (F) 0 - 35p	Svak (D, E) 40 - 45 - 50 - 55p	Middels (C) 60 - 65 - 70 - 75 - 80p	Utmerket (B, A) 85 - 90 - 95 - 100p	Vekt
Metoder i klasser		Kandidaten viser manglende forståelse for deklarasjon av metoder: • Metoder starter med stor bokstav • Metoder returnerer feil/ufornuftig datatype (typisk String der det burde vært returnert int eller double osv. • Missbruk av **toString()** - benyttes for å lage en streng som skal presenteres sluttbruker.	Kandidaten viser god forståelse for deklarasjon av metoder i en klasse: • Samtlige metoder starter med liten bokstav • Metoder returnerer **void** der dette er naturlig. • Metoder returnerer fornuftige datatyper	Kandidaten har i tillegg: • I metoder som tar parametre valideres parameteret før øvrig kode i metoden ("guard condition")	
Samlinger (ArrayList, HashMap osv)					**5 %**
Grunnleggende forståelse av **samlinger**		Kandidaten har: • Ikke egen klasse som representerer vareregister, men bruker isteden ArrayList (el.l.) direkte i Main-klassen el.l. • Har brukt **primitiv array** istedenfor en av klassene i **collection-biblioteket.**	Kandidaten viser grunnleggende forståelse for **samlinger:** • Bruker ArrayList som vareregister i egen klasse for vareregister. • Benytter metoder i ArrayList-klassen som: size(), get(index), add() osv.	Kandidaten har i tillegg: • Benyttet HashMap for håndtering av vareregister (unik ID på vare tilsier HashMap) • Eventuelt: kandidaten beskriver at han/hun **har** vurdert HashMap, men konkludert med annen løsning.	
Løkker					**5 %**
For-each og while-løkker		Bruker ikke løkker iht anbefalingene: • Avbryter for/for-each løkke med **break** eller **return** i løkken • I while-løkker: samler ikke alle betingelser for å iterere i selve **while-statmenetet**	Kandiaten viser grunnleggende forsåelse av løkker: • for-each for samlinger - når **hele** samlingen skal itereres over • while-løkke, når det **søkes i en samling** •	Kandidaten har i tillegg: • Konsekvent bruk av kun for-each/for løkker når man skal iterere over samtlige objekter, og ikke ellers. • Bruker **while med iterator** • Kandidaten bruker **streams** og **filters** i stedenfor løkker. Må da gå ut ifra at kandidaten også behersker for- go	

KRITERIUM	Ikke bestått (F) 0 - 35p	Svak (D, E) 40 - 45 - 50 - 55p	Middels (C) 60 - 65 - 70 - 75 - 80p	Utmerket (B, A) 85 - 90 - 95 - 100p	Vekt
		(f.eks. bruker return og/eller break inne i while løkka. • for/while-løkke med index istedenfor den bedre løsningen med Iterator		while-løkke selv om kandidaten ikke bruker dette i sin løsning...	
Kodekvalitet/Design					**15 %**
Er koden godt dokumentert iht JavaDoc-standard?	Ingen dokumentasjon	Kandidaten har: • Sporadisk dokumentasjon av klassene • Sporadisk dokumentasjon av metoder • Dokumentasjon av typen "This methode...", eller dok som beskriver hvordan og ikke hva • Mangelfull/fraværende bruk av "@param", "@return"	Kandidaten viser grunnleggende forståelse for dokumentasjon ved at: • Samtlige klasser er dokumentert med god beskrivelse av rolle/ansvar. • De fleste metoder som er public er godt dokumentert. • Grei "ordlyd" i dokumentasjonen (Ikke "This method...", "Gets the.." osv) • Gjennomgående bruk av "@param" og "@return"	Kandidaten viser i tillegg: • Gjennomført høy kvalitet i dokumentasjon • God formulering som klart dokumenterer klasser og metoder • "@param", "@return" konsekvent brukt • Innslag av HTML-formatering for bedre lesbarhet • CheckStyle rapporterer 0 feil	
Er koden robust (verifiseres parametere før de brukes mm)?	Ingen verifisering	Kandidaten har: • Sporadisk eller ingen sjekk av parameterverdier	Kandidaten viser grunnleggende forståelse av robust kode ved: • Gjennomgående grei verifisering av parameterverdier. • Håndterer ugyldige verdier ved å sette default verdier.	Kandidaten har i tillegg: • Verifiserer samtlige parametre. • Har fornuftig håndtering av situasjoner der parameter har ugyldig verdi (settes til en **default gyldig** verdi f.eks.) • Dersom kandidaten benytter Exceptions er det OK	
Har variabler, metoder og klasser beskrivende navn?		Kandidaten har: • Svært mye bruk av variabelnavn som ikke gjenspeiler data de representerer/holder. • Navn på metoder gjenspeiler i ingen eller liten grad **hva** metoden gjør.	Kandidaten viser grunnleggende forståelse: • Gjennomgående gode variabelnavn som gjenspeiler data de representerer/holder. • Gjennomgående gode navn på metoder som gjenspeiler **hva** metoden gjør. • Klassenavn beskriver godt rollen/ansvaret til klassen	Kandidaten har i tillegg: • Svært gode variabelnavn som gjenspeiler data de representerer/holder. • Svært gode navn på metoder som gjenspeiler **hva** metoden gjør. • Alle klassenavn beskriver godt rollen/ansvaret til klassen	

KRITERIUM	Ikke bestått (F) 0 - 35p	Svak (D, E) 40 - 45 - 50 - 55p	Middels (C) 60 - 65 - 70 - 75 - 80p	Utmerket (B, A) 85 - 90 - 95 - 100p	Vekt
Har koden god struktur, med løse koblinger og høy kohesjon?		Kandidaten viser manglende forståelse: • Det er tett kobling mellom klasser som ikke burde hatt kobling (high coupling) • Mange metoder som utfører flere oppgaver (low cohesion) • Mange klasser med flere roller/ansvar	Kandidaten viser grunnleggende forståelse: • Generelt god løs kobling mellom klasser som ikke burde hatt kobling (low coupling) • Jevnt over de fleste metoder utfører en enkelt oppgave (high cohesion). Der en metode må utføre flere del-oppgaver, benyttes **deligering** til sub-metoder. • De fleste klasser har **en** klart definert rolle/ansvar	Kandidaten har i tillegg: • Gjennomgående svært god løs kobling mellom klasser som ikke burde hatt kobling (low coupling) • Samtlige metoder utfører en enkelt oppgave (high cohesion). Der en metode må utføre flere del-oppgaver, benyttes **deligering** til sub-metoder. • Samtlige klasser har **en** klart definert rolle/ansvar • Returnerer **Iterator** i metoder som skal gi tilgang til samling av varer istedenfor å returnere f.eks. ArrayList. Bidrar til løsere kobling).	
Totalintrykk					**10 %**
Her vurderes totalinntrykket til kandidaten. Dette kriteriet kan benyttes for vurdering som ikke fanges opp av de øvrige kriteriene.		Kandidaten presterer under gjennomsnitt. Løsningen er unødvendig tungvint implementert, eller er mangelfull.	En prestasjon som er midt på treet totalt sett. Kandidaten viser at han/hun har forstått de fleste læringsmål og viser god forståelse.	En fremragende løsning på alle måter. Kandidaten benytter elementer i sin løsning som er ut over det som er forventet.	

Del 3 - Rapport

Kandidatene fikk utlevert en mal for Word. I malen var det gitt en mengde *hjelpetekst* for å hjelpe kandidaten til å forstå hva som er forventet i de ulike kapitlene i rapporten. Malen baserer seg på strukturen ofte benyttet i vitenskapelige artikler og i Bachelor rapporter.

KRITERIUM	**Ikke bestått (F)** **0 - 35p**	**Svak (D, E)** **40 - 45 - 50 - 55**	**Middels (C)** **60 - 65 - 70 - 75**	**Utmerket (B, A)** **80 - 85 - 90 - 95 - 100**	**Vekt**
					30 %
Struktur	Kaos	Elementer mangler eller er plassert på ulogiske steder	Grei struktur, fin rød tråd, ting er beskrevet der de hører hjemme	Ekstra elementer som f.eks. - kryssreferanser - bibliografi	
Kravspesifikasjon (Kapittel "Innledning - Problemstilling")	Mangler	Vanskelig å forstå hva som skal utvikles basert på beskrivelsen.	Har beskrevet de viktigste funksjonelle kravene.	Utfyllende med f.eks • Avgrensninger. • Use case-diagrammer og beskrivelser • andre typer krav (systemkrav, ikke-funksjonelle krav)	
Teoretisk grunnlag Hvilke **teorier** er benyttet ved utviklingen av løsningen?	Mangler	Mangelfullt beskrevet. Beskriver bare deler av teorigrunnlaget.	God beskrivelse av teoriene om god design. Bør minimum ha med **coupling, cohesion**	Har i tillegg inkludert : • Teori om en eller flere av følgende: **responsibility driven design**, **modularisering** • teori rundt **kodestil**, m.a.o. øvrige prinsipper som gir kvalitet i løsningen. • teori rundt god brukerinteraksjon • nevnt SOLID	
Metode Hvilke **verktøy** er benyttet for å løse oppgaven? Hvilken **prosess/prosjektmodell** ble fulgt? I denne mappen har vi f.eks. gitt prosjektet i 3 deler med tilbud om 3 tilbakemeldinger.	Mangler	Mangelfull beskrivelse av IDE og prosess/metode	Beskriver hvilke **verktøy** (IDE) som er brukt.	Beskriver i tillegg greit arbeidsmetoden (hvilken arbeidsmetode kandidaten har fulgt - her bør kandidaten nevne hvordan mappen ble "utporsjonert" i 3 deler, og at de fikk muntlig tilbakemelding på hver del med mulighet til å forbedre løsningen sin)	
Design og Implementasjon (i kapittel "Resultat")	Mangler	Utydelig, manglende diagram. Skriver «dagbok», og ikke en	Godt beskrevet • med bruk av klassediagram	Ekstra utfyllende med f.eks. • UML-diagrammer utover klassediagram som feks	

KRITERIUM	Ikke bestått (F) 0 - 35p	Svak (D, E) 40 - 45 - 50 - 55	Middels (C) 60 - 65 - 70 - 75	Utmerket (B, A) 80 - 85 - 90 - 95 - 100	Vekt
		beskrivelse av rolle- og ansvar til de ulike klassene.	• med god bruk av kodeeksempler og skjermdump • med god beskrivelse av **prosessen**; hvilke **rafakturering** ble utført, og hvorfor?	sekvensdiagram, aktivitetsdiagram for utvalgte funksjoner. • Resultat av gjennomførte **brukertester** • Ryddig og oversiktlig.	
Drøfting og refleksjon, Konklusjon	Mangler	Diskusjon: Knapt beskrevet. Ingen reell *drøfting* av egen løsning i forhold til teoriene beskrevet i Teori-avsnittet.. Konklusjon: Svært kort, der kandidaten ikke peker på de viktigste oppnådde målene i prosjektet	Drøfter greit endelig resultat i forhold til **teoriene** om **coupling, cohesion osv.**, med **god henvisning til eksempler fra egen kode**. God konklusjon der kandidaten oppsummerer prosjektet i forhold til: • Om kandidaten mener prosjektet er iht de prinsipper og standarder adressert i emnet • Om kandidaten har fått vist sine ferdigheter igjennom prosjektet	Drøfter **svært godt** endelig resultat i forhold til **teoriene** om **coupling, cohesion osv.**, med **god henvisning til eksempler fra egen kode**. Konklusjon: Har også fått med f.eks. hva som burde vært gjort annerledes om det skulle gjøres igjen	