# Programmeringsoppgaver IDAT1001 – Programmering 1

## Innhold

## Beskrivelse av arbeidskrav

Det vil bli gitt 11 øvinger i emnet, hvorav **9** må være godkjent for å gå opp til eksamen. De 9 øvingene som godkjennes må dekke et bredt spekter av øvingene etter følgende krav: Obligatoriske enkeltøvinger: **10 og 11.** I tillegg 2 øvinger i hver av følgende grupper: (1-3), (4-6) og (7-9). Obligatorisk oppmøte på lab for å få godkjent øvingene.

## Øving 1 Variabler, datatyper og uttrykk

### Oppgave 1

Skriv et program som regner om fra tommer til centimeter. En tomme er 2,54 centimeter. Sett opp testdatasett og prøv ut programmet.

### Oppgave 2

Skriv et program som regner om timer, minutter og sekunder til totalt antall sekunder. Sett opp testdatasett og prøv ut programmet.

### Oppgave 3

Skriv et program som leser inn et antall sekunder og beregner hvor mange timer, minutter og sekunder dette er. (Hint: Bruk heltallsdivisjon.) Sett opp testdatasett og prøv ut programmet.

---

## Øving 2 Kontrollstrukturen valg

### Oppgave 1

Et år er skuddår dersom det er delelig med 4. Unntaket er hundreårene, de må være delelig med 400.

Tegn aktivitetsdiagram som viser algoritmen for å finne ut om et år er skuddår. Årstallet skal leses inn fra brukeren. Sett opp testdata. Lag og prøv ut programmet.

### Oppgave 2

Lag et program som hjelper oss i forhold til følgende problemstilling: Kjøttdeig av merke A koster kr 35,90 for 450 gram, mens kjøttdeig av merke B koster kr 39,50 for 500 gram. Hvilket merke er billigst?

---

## Øving 3 Kontrollstrukturen Løkke

Du skal i alle oppgavene lese data fra brukeren. Løs først oppgavene uten kontroll av inndata. Du kan så, der det er relevant, legge inn kontroll av inndata med enkel bruk av do-setningen

(teknikk 1, programliste 4,3). Merk at denne teknikken ikke omfatter det tilfellet at brukeren skriver inn en tekst der et tall er forventet. Denne feilen gir NumberFormatException, og du må ha kunnskap om unntakshåndtering for å behandle feil av denne typen.

### Oppgave 1:

Lag et program som skriver ut en del av multiplikasjonstabellen, for eksempel fra 13-15. Da skal utskriften se omtrent slik ut (prikkene skal erstattes med regnestykker).

13-gangen:

13 x 1 = 13

13 x 2 = 26

…

13 x 10 = 130

14-gangen:

14 x 1 = 14

14 x 2 = 28

…

14 x 10 = 140

15-gangen:

15 x 1 = 15

15 x 2 = 30

…

15 x 10 = 150

Brukeren skal velge hvilken del av tabellen som skal skrives ut.

### Oppgave 2

Skriv et program som finner ut om et tall er et primtall. Et primtall er et tall som kun kan deles med 1 og med seg selv uten å få rest. Les inn tallet fra brukeren og la programmet repetere slik at flere tall kan analyseres.

---

# Øving 4 OOP, Klasser

Lag klassediagram for alle klasser i denne øvingen og aktivitetsdiagram for klientprogrammet i oppgave 1.

## Oppgave 1

Lag en klasse Valuta med minst en konstruktør. Klassen skal ha metoder for å regne fra og til norske kroner.

Lag et klientprogram som oppretter minst tre objekter som representerer forskjellige valutaer. Brukeren skal få tilbud om å regne om flere ulike beløp i de forskjellige valutaene til norske kroner.

Programmet må altså presentere en meny for brukeren. Den kan for eksempel se sånn ut:

Velg valuta:

1: dollar

2: euro

3: svenske kroner

4: avslutt

Brukeren skriver inn ett av tallene 1, 2 eller 3 (eller 4 for avslutt). Dette skal styre programflyten slik at riktig valutaobjekt blir brukt.

## Oppgave 2

Du skal programmere terningspillet 100 med to spillere, A og B. Reglene er som følger:

A og B kaster terningen annenhver gang. Antall øyne på terningen er antall poeng oppnådd i denne runden. Poengene summeres fortløpende. Dersom en spiller kaster 1, settes summen tilbake til 0. Den som først passerer 100 poeng, har vunnet spillet.

Lag en klasse som simulerer en spiller. Den kan han sumPoeng som objektvariabel og metoder som heter getSumPoeng(), kastTerningen() og erFerdig().

For å simulere terningen skal du bruke klassen java.util.Random fra java-API-et. Denne klassen brukes til å lage rekker med tilfeldige tall. La et objekt av klassen være objektvariabel:

Java.util.Random tering= new java.util.Random(); // vi lager en tilfeldigtallgenerator

I metoden kastTerningen() henter du et tilfeldig tall i intervallet (0, 5) på følgende måte:

Int terningkast = terning.nextint(6);

Legg til 1 for å få et gyldig terningkast.

Lag et klientprogram som oppretter to objekter av klassen, ett for hver av spillerne. Lag en løkke som kjører inntil en av spillerne har vunnet. I hvert gjennomløp av løkken skal hver spiller kaste terningen en gang. Skriv ut rundenummer og poengsummene til hver av spillerne i hvert gjennomløp. Det er sannsynligvis mest praktisk å skrive til kommandovinduet her.

En raffinering av spillet kan være å kreve at man skal komme akkurat til 100. Dersom denne grensa er passert, skal neste kast trekkes fra poengsummen. Hvis man nå havner akkurat på 100, er man ferdig. Ellers legges neste kast til, og slik holder man på og legger til og trekker fra inntil man kommer akkurat til 100.

---

# Øving 5 Klasser som byggeklosser

## Oppgave 1

Lag en klasse for å regne med brøk.

Klassen skal ha to konstruktører:

- Den ene konstruktøren tar teller og nevner som argument. Hvis nevneren er 0, skal et unntaksobjekt av typen IllegalArgumentException kastes.
- Den andre konstruktøren tar kun telleren som argument. Da skal nevneren settes lik 1.

Klassen skal ha get-metoder, men ikke set-metoder.

Klassen skal ha metoder for å summere, subtrahere, multiplisere og dividere en brøk (this) med en annen brøk (parameter til metoden). Metodene har ikke returverdi, men etter at operasjonen er utført, ligger resultatet i this.

Du kan se bort fra at resultatet av operasjonen bør forkortes.

Lag testklient som del av klassen.

*Ekstraoppgave:* Sørg for å forkorte brøkene.

## Oppgave 2

I oppgave 4 i kapittel 5 brukte du klassen java.util.Random. Metoden nextInt() er laget slikt at den gir et heltall i intervallet fra og med 0 og opp til den tallverdien du sender inn som

argument. Klassen tilbyr også en metode nextDouble() som gir deg et desimaltall uniformt fordelt i intervallet (0,0, 1,0>[15]. Denne metoden tar ingen argumenter.

Lag en klasse MinRandom som tilbyr følgende metoder:

public int nesteHeltall(int nedre, int ovre) // intervallet (nedre, ovre)

public double nesteDesimaltall(double nedre, double ovre) // intervallet >nedre, ovre>

[15]. Denne notasjonen betyr at 0,0 er med, mens 1,0 ikke er med.

Klassen skal ha et objekt av klassen java.util.Random som objektvariabel. Metodene foran skal implementeres ved at man bruker dette objektet til å generere neste tilfeldige tall. Resultatet skal så transformeres til det ønskede intervallet.

Prøv klassen ved å lage mange tilfeldige tall av begge typer og forsikre deg om at de ligger i de oppgitte intervallene.

---

# Øving 6 Tabeller av primitive datatyper

## Oppgave 1

Fra programmeringsoppgaver kjenner du kanskje nå klassen java.util.Random.

Lag et objekt av klassen:

Java.util.Random random = new java.util.Random(); // vi lager en tilfeldigtallgenerator

Setningen

Int tall = random.nextInt(10);

gir en verdi i intervallet (0, 9).

Opprett en tabell

Int antall = new int(10);

Lag et program som går i løkke for eksempel 1000 ganger og henter ut tilfeldige tall. I tabellen antall skal du lagre antall ganger hvert enkelt av tallene 0, 1, …, 9 er hentet ut.

Til slutt skriver du ut en liste over antall forekomster av hvert enkelt av de ti tallene. Prøv gjerne også programmet med 5000 og 10 000 gjennomløp av løkken.

*Ekstraoppgave:*

Skriv ut resultatene med tall og med stjerner, der en stjerne representerer 1/100 av antall ganger et bestemt tall er trukket. Med 1000 løkkegjennomløp og tallet 5 trukket 86 ganger vil linjen for tallet 5 kunne se slik ut (86 blir avrundet til 90, og dermed 9 stjerner)

5  86 *********

## Oppgave 2

I denne oppgaven skal du lage en klasse for tekstanalyse. Teksten er et objekt av klassen String, og du får tak i tegnet på en bestemt posisjon ved å bruke metoden charAt(). Lengden av teksten er gitt ved metoden length(). Følgende kodebit illustrerer bruken av metodene charAt() og length() og gir også tips om hvordan resten av oppgaven kan løses (a-z og A-Z ligger rett etter hverandre i Unicode-tegnsettet).

Som objektvariabel i klassen Tekstanalyse skal du ha en tabell av int:

Int antallTegn = new int(30)

I denne tabellen gjelder at indeks 0-28 inneholder antall forekomster i en tekst av bokstavene a-å, mens indeks 29 inneholder antall forekomster av alle andre tegn enn bokstaver, for eksempel mellomrom, komma og punktum. Vi skiller ikke mellom store og små utgaver av en bokstav.

Konstruktøren skal ta den aktuelle teksten som argument og fylle opp tabellen antallTegn med antall tegn av hver type. Etter at tabellen antallTegn er fylt opp, skal en klient kunne få utført følgende tjenester:

- Finn antall forskjellige bokstaver i teksten. Her ser vi altså bort fra alle andre tegn enn bokstaver.
- Finn totalt antall bokstaver i teksten.
- Hvor stor del av teksten (prosent) er ikke bokstaver?
- Finn antall forekomster av en bestemt bokstav. Bokstaven skal være parameter til metoden.
- Hvilken – eller hvilke, om det er flere som forekommer like ofte – bokstaver forekommer oftest i teksten?

Lag et klientprogram som går i løkke. I hvert løkkegjennomløp skal klienten skrive inn en tekst, og resultatet av alle analysene foran skal skrives ut.

## Oppgave 3

Lag en klasse Temperaturer. Den skal inneholde en todimensjonal tabell med temperaturer for hver time i døgnet i en måned.

Klassen skal tilby metoder som gjør det mulig å finne

a) Middeltemperaturen for hver dag i måneden
b) Middeltemperaturen for hver time i døgnet i måneden
c) Middeltemperaturen for hele måneden
d) Antall døgn med middeltemperatur innen følgende grupper: mindre enn -5 grader, mellom -5 og 0, mellom 0 og 5, mellom 5 og 10 og 10 og over 10 grader.

Tips: Metodene i oppgavene a, b og d skal returnere referanser til tabeller som inneholder de resultatene det spørres om.

Lag et enkelt klientprogram som kan brukes til å teste klassen. La antall timer i døgnet og antall døgn i måneden være lite under testing.

Oppgave 3:

Lag en klasse Matrise. Den skal inneholde en todimensjonal tabell som svarer til en matrise i matematikken. Lag metoder for å

- Addere en matrise til denne matrisen
- Multiplisere denne matrisen med en annen matrise
- Transponere matrisen.

Klassen skal være immutabel. Det vil si at alle metodene må lage nye metoder som returneres. Dersom operasjonene er umulige, skal metodene returnere null.

Lag et enkelt klientprogram som kan brukes til å teste klassen.

---

# Øving 7 Java API

## Oppgave 1

Klassen String inneholder mange metoder. Vi skal nå lage noen flere.

Lag en klasse som heter NyString. Den skal ha en objektvariabel av klassen String. Klassen NyString skal være immutabel og tilby følgende tjenester:

- Konstruktør som tar et objekt av klassen String som argument.
- Forkorting. Forkortingen skal skje ved at første tegn i hvert ord plukkes ut. Bruk mellomrom som skille mellom ordene. Eksempel: «denne setningen kan forkortes» skal forkortes til «dskf». Tips: Bruk String-metoden split(), som returnerer en tabell med ordene som teksten består av: String() ord = tekst.split()
- Fjerning av tegn. Eksempel: Dersom tegnet 'e' fjernes fra teksten «denne setningen kan forkortes», står vi igjen med «dnn stningn kan forkorts». Tips: Bruk en while-løkke og for eksempel String-metodene indexOf() og substring().

## Oppgave 2

Lag en klasse for enkel tekstbehandling. Konstruktøren skal ta en tekst som argument. Klassen skal tilby følgende tjenester:

- Å finne antall ord i teksten.
- Å finne gjennomsnittlig ordlengde. Skilletegn er ikke en del av ordene.
- Å finne gjennomsnittlig antall ord per periode. Bruk punktum, utropstegn, kolon og spørretegn som skilletegn mellom periodene. Anta at teksten er feilfri, slik at ikke to eller flere slike tegn følger etter hverandre.
- Å skifte ut et ord med et annet gjennom hele teksten. For eksempel kan en ønske å skifte ut ordet «finnes» med «fins».
- Å hente ut teksten slik den står, uten endringer.
- Å hente ut teksten, men slik at alle bokstaver er store.

Lag en enkel testklient. Husk å prøve klassen for tekster som inneholder æ, ø og å.

---

# Øving 8 Samarbeid mellom objekter

## Oppgave 1

Lag en klasse Person med attributter fornavn, etternavn og fødselsår. Klassen skal ha en get-metode for hvert attributt og være immutabel.

Lag en klasse ArbTaker med attributter personalia (datatype Person), arbtakernr, ansettelsesår, månedslønn og skatteprosent. I tillegg til get-metoder for alle attributtene og set-metoder for å endre attributter som det er naturlig at bør kunne forandres, skal klassen tilby operasjoner som finner ut

- Hvor mange kroner arbeidstakeren trekkes i skatt per måned;
- Bruttolønn per år;
- Skattetrekk per år. Husk at juni er skattefri og i desember betales halv skatt;
- Navn på formen: etternavn, fornavn, eksempel: Johnsen, Berit;
- Alder;
- Antall år ansatt i bedriften;
- Om personen har vært ansatt mer enn et gitt antall år (parameter);

Finn ut i hvilke av disse tilfellene at ArbTaker-objektet må samarbeide med personalia-objektet for å løse oppgaven. Tegn sekvensdiagram for disse operasjonene.

Inneværende år får du ut med følgende kodelinjer:

*Java.util.GregorianCalendar kalender = new java.util.GregorianCalendar();*

*Int år = kalender.get(java.util.Calendar.YEAR);*

Lag et enkelt klientprogram som legger inn data i et objekt av klassen ArbTaker og kaller alle objektmetodene du har laget. Kontroller at resultatene blir riktige.

Lag et menystyrt program som gir brukeren muligheten til å forandre på datainnholdet i objektet. La programmet gå i løkke slik at flere forandringer kan gjøres. For hvert løkkegjennomløp skal programmet sende passende get-meldinger til objektet og skrive ut resultatet av disse, slik at det er mulig å kontrollere at endringen er blitt utført. Lag eventuelt toStrin()-metoder og bruk disse.

---

# Øving 9 Tabeller av objekter

## Oppgave 1

Lag en klasse Student.  Den skal ha to objektvariabler:

```
private String navn;    // entydig(!)

private int antOppg;
```

antOppg holder orden på hvor mange av de oppgavene studenten har levert inn, som er godkjent. Klassen skal tilby følgende operasjoner:

```
public String getNAvn()
```

public int getAntOppg()

public void økAntOppg(int økning)

public String toString()

Lag et enkelt testprogram for klassen.

Lag en klasse Oppgaveoversikt som holder orden på hvor mange oppgaver hver enkelt student har fått godkjent. Denne klassen skal ha to objektvariabler:

private Student studenter;  // tabellen opprettes i konstruktøren

private int antStud = 0;   // økes med 1 for hver ny student

Du skal lage metoder for følgende operasjoner (gjerne også flere, dersom du finner at det er behov for det):

- Finn antall studenter registrert
- Finn antall oppgaver som en bestemt student har løst
- Registrer en ny student
- Øk antall oppgaver for en bestemt student.

I tillegg skal du lage toString()-metode

Lag et enkelt klientprogram.

---

# Øving 10 - Tabellister

## Oppgave 1

Kommunens turistkontor trenger et program for å vedlikeholde et register over arrangementer som kan være av interesse for turister og andre. For hvert arrangement lagrer vi et entydig nummer, navn, sted, arrangør, type (konsert, barneteater, foredrag osv) og tidspunkt (dato og klokkeslett). Lagre tidspunktene som heltall, eksempel: 200210301800 (kl 18 den 30.oktober 2002) slik at du ved å sammenligne tallene kan finne ut om et arrangement er før eller etter et annet.

Du trenger klassene Arrangement og ArrangementRegister. Den siste klassen skal tilby følgende operasjoner:

- Å registrere et nytt arrangement
- Å finne alle arrangementer på et gitt sted
- Å finne alle arrangementer på en gitt dato
- Å finne alle arrangementer innenfor et tidsintervall gitt ved to datoer. Listen skal være sortert på tid.

- Å lage lister over alle arrangementer, sortert etter henholdsvis sted, type og tidspunkt.

(Du trenger ikke legge inn kontroll på at et gitt tidspunkt er gyldig)

Lag et menystyrt klientprogram.

### Oppgave 2

Lag et register over menyer for en restaurantkjede. En meny består av flere retter. Hver rett kan inngå i flere menyer. Hver rett har navn type (forrett, hovedrett, dessert osv) og pris. Oppskriften ligger lagret som en streng for hver rett. Du kan anta at navnet identifiserer retten entydig.

Klassen MenyRegister skal tilby følgende operasjoner:

- Å registrere en ny rett.
- Å finne en rett, gitt navnet.
- Å finne alle retter av en gitt type.
- Å registrere en ny meny som består av ett sett med retter.
- Å finne alle menyer med totalpris innenfor et gitt intervall.

Finn selv ut hvilke operasjoner klassene Meny og Rett bør tilby for at menyregisteret på enklest mulig måte skal kunne utføre sine oppgaver.

Tegn UML-diagram. Programmer klassene, og lag en enkel testklient.

---

## Øving 11 – Eksamensoppgave

Kommer

---