1 Egenerklæring (med bekreftelse)

Jeg erklærer herved at besvarelsen som jeg leverer er mitt eget arbeid og

- ikke har vært brukt i en annen eksamen eller vært levert eller publisert ved en annen utdanningsinstitusjon innenlands eller utenlands.
- ikke inneholder andres arbeid uten at dette er oppgitt.
- ikke inneholder eget tidligere arbeid uten at dette er oppgitt.

Jeg er kjent med at brudd på disse bestemmelsene er å betrakte som fusk.
Dersom du er usikker på om du kan stille deg bak erklæringen, se retningslinjer for bruk av kilder i skriftlige arbeider ved Universitetet i Bergen og eventuelt ta kontakt med veileder/emneansvarlig.
Alle innleveringene dine ved UiB kan bli sendt til elektronisk plagiatkontroll.

Merk: Det er ikke anledning til å levere besvarelser som ikke oppfyller kravene i egenerklæringen.

Velg ett alternativ

- ○ Jeg bekrefter at jeg har lest egenerklæringen og at besvarelsen er mitt eget arbeid

---

Maks poeng: 0

ℹ Kontaktinformasjon under eksamen:

For kontakt under eksamen, se kunngjøringen i INF101 MittUiB:
https://mitt.uib.no/courses/21613/discussion_topics/169894

ℹ Denne Eksamen består av 4 seksjoner som gir poeng

	Tema	maks poeng
1	Avkryssning (envalgsoppgaver)	14
2	Spørsmål angående semesteroppgave 2	20
3	Oppgaver med implementering	36
4	Poeng overført fra semesteroppgave	30

Du kan få totalt 100 poeng, 30 av disse poengene er poeng som overføres fra semesteroppgavene, det vil si du kan få maks 70 poeng fra det du gjør på denne eksamen.
Eksamen varer 5 timer så du har ca. 4 minutter per poeng.

Skaff deg god oversikt over hva som må gjøres og planlegg tiden slik at du får gjort mest mulig.

Seksjon 1 er avkryssing, når du ikke vet svaret lønner det seg alltid å gjette. Det er ikke minus poeng for feil svar.

Lykke til!

2 Hvilket navn passer best for denne metoden?

```
public double function1(ArrayList<Number> input) {
    double sum = 0.0;
    for(Number number : input) {
        sum += number.doubleValue();
    }
    return sum/input.size();
}
```

Dokumentasjon for klassen Number kan finnes her:
https://docs.oracle.com/en/java/javase/13/docs/api/java.base/java/lang/Number.html

Velg et alternativ:

- toDouble
- sum
- average
- doubleSum

Maks poeng: 2

3 Hvorfor er det lurt å definere klasser og lage objekter?

Velg ett alternativ:

- Jo flere klasser du har jo lettere blir det å forstå koden.
- Med flere klasser blir det mindre kode å skrive og dermed lettere å unngå feil (bugs) i koden.
- Det viktigste med klasser er å organisere koden, de samler variabler og metoder som hører sammen slik at det blir lettere å finne frem i koden.
- Med flere klasser blir det mindre behov for dokumentasjon.

Maks poeng: 2

4 Det er viktig å teste koden din, til det bruker vi ofte JUnit tester. Det er lurt å tenke at de testene du skriver skal kunne brukes videre selv om deler av koden din endres i fremtiden.

For at koden din skal kunne testes med JUnit tester og at testene skal være nyttige selv når koden utvikler seg videre er følgende viktig når du skriver koden:

Velg ett alternativ:

- Bruke GIT.
- Skrive Javadoc kommentarer
- Bruke modularitet (modularity)
- Skrive lesbar kode

Maks poeng: 2

5 Her er 3 klasser, hva printes når main metoden kjøres?

```
public class Thanks {

    public String toString(){
        return "Thanks";
    }

}

public class ManyThanks extends Thanks {

    public String toString(){
        return "Many "+super.toString();
    }
}

public class SayThanks {

    public static void main(String[] args) {
        Thanks thank = new ManyThanks();
        System.out.println(thank);
    }

}
```

Velg ett alternativ:

- Many Thanks
- ManyThanks@2f92e0f4 (forskjellig tall hver gang man kjører programmet)
- Thanks
- Vil få Exception fordi det ikke er implementert noen konstruktører.

Maks poeng: 2

6 Jeg har prøvd å gå foran med et godt eksempel og skrevet god kode med utfyllende kommentarer på alle forelesningene. Når dere kommer ut i jobb vil mange av dere møte på noen som ikke tenker på lesbarhet når de programmerer :-(

Det må vi bare leve med og prøve å forstå hva koden gjør.

Her har noen skrevet en metode med dårlig metodenavn "function3" og ingen kommentarer. Det er heller ikke skrevet på den mest lettleselige måten.

Forstå hva koden gjør og velg det beste metodenavnet.

```
private static int function3(int n, int digit) {
    if(digit<0 || digit>9) {
        throw new IllegalArgumentException("This is not a valid digit");
    }
    if(n<0) {
        return function3(-n, digit);
    }
    if(n<10) {
        if(n==digit)
            return 1;
        else
            return 0;
    }

    int r = n%10;
    n = n/10;
    if(digit==r) {
        return function3(n,digit) + 1;
    }
    return function3(n,digit);
}
```

Velg ett alternativ:

- digitSum (betyr tverrsum på norsk)
- countMatchingDigits
- shiftDigits
- indexOfDigit

Maks poeng: 3

7 Denne enkle koden virker ikke som den skal. Meningen var å ha en liste over alle bøkene et bibliotek har og søke igjennom denne listen for å finne ut om biblioteket har en bestemt bok. Koden vår skriver ut "Oh no, I will not be an Java expert!". Hva er galt?

```
public class Library{
    ArrayList<Book> books = new ArrayList<Book>();

    Library(){
        //add books to the library
        books.add(new Book("Effective Java", "Joshua Bloch"));
        books.add(new Book("Clean Code", "Robert Martin"));
        books.add(new Book("Head First Design Patterns", "Elisabeth Robson"));
    }

    /**
     * Checks if the library has the book and prints a message
     * @param book
     */
    public void hasBook(Book book) {
        if(books.contains(book))
            System.out.println("Yes I can read this over summer!");
        else
            System.out.println("Oh no, I will not be an Java expert!");
    }

    public static void main(String[] args) {
        Library UiBLib = new Library();
        Book wanted = new Book("Clean Code", "Robert Martin");
        UiBLib.hasBook(wanted);
    }
}
```

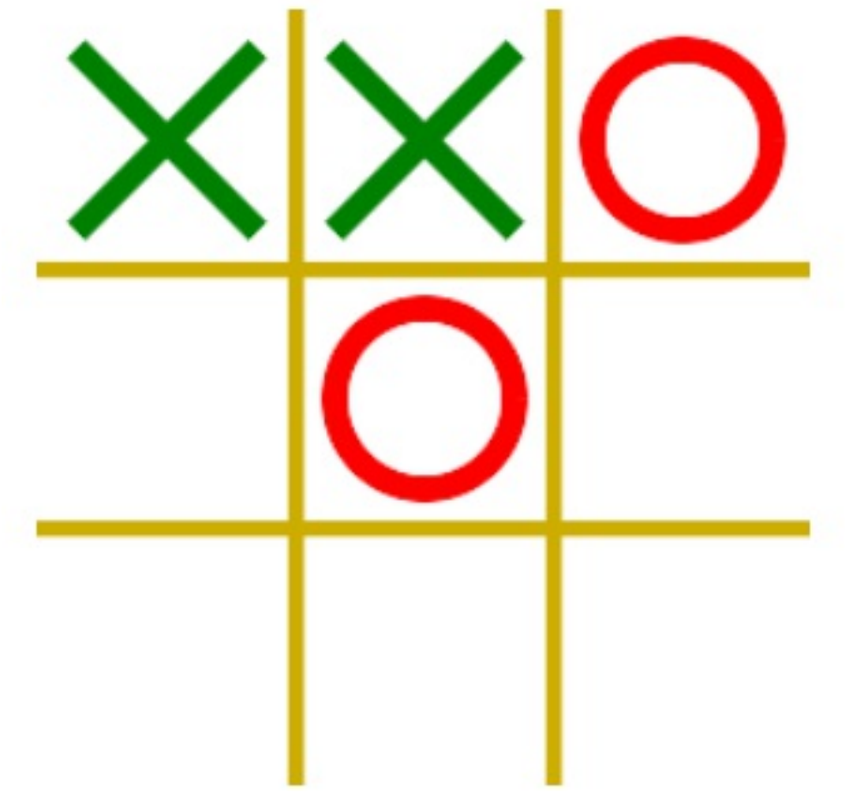
```
public class Book {

    String title;
    String author;

    public Book(String title, String author) {
        this.title = title;
        this.author = author;
    }
}
```

Velg ett alternativ:

- Vi må overlaste (override) equals metoden for å sjekke om to bøker er like.
- contains metoden fungerer bare på String og primitive variabler.
- Book må implementere Comparable
- Vi skulle hatt et ! (for negasjon) forran books.contains(book)

Maks poeng: 3

Denne seksjonen handler om Semesteroppgave 2 og mer spesifikt er spørsmålene relatert til løsningsforslaget vi har laget til dere for noen uker siden og gikk igjennom med videoforelesninger.



Denne seksjonen tester om du har forstått løsningsforslaget til semesteroppgave 2 og om du har forstått noen grunnleggende konsepter innen objektorientert programmering.

Løsningsforslaget på semesteroppgave 2 finner dere her:
https://retting.ii.uib.no/inf101.v20.oppgaver/inf101.v20.sem2.losning
https://mitt.uib.no/courses/21613/files/2486903/download

Skriv presise svar, hvis du klarer å trekke frem de rette punktene så er 5-10 linjer tekst per oppgave nok, du får ikke mer enn 4 poeng uansett om du skriver 3 sider.

8 Se på løsningsforslaget til Semesteroppgave 2. Der har vi valgt å implementere en klasse som heter AbstractPlayer. Hvorfor har vi gjort dette?

Skriv ditt svar her

Maks poeng: 4

9 Se på løsningsforslaget til Semesteroppgave 2. Beskriv et sted hvor vi overkjører (override) en metode som er implementert i superklassen og forklar hvorfor vi gjør dette.

Skriv ditt svar her

Maks poeng: 4

10 Se på løsningsforslaget til Semesteroppgave 2. Beskriv et eksempel fra løsningsforslaget hvor det brukes polymorfisme. Beskriv både hvor i løsningsforslaget (hvilket metodekall) polymorfisme brukes og hvorfor det er polymorfisme.

Her er link til Oracle tutorials som forklarer polymorfi.

Skriv ditt svar her

Maks poeng: 4

11 Se på løsningsforslaget til Semesteroppgave 2. Beskriv et godt eksempel fra løsningsforslaget hvor innkapsling (encapsulation) er brukt og hvordan dette gjør koden bedre.

Skriv ditt svar her

Maks poeng: 4

12 Se på løsningsforslaget til Semesteroppgave 2. I klassen GameBoard i metoden count bruker vi en linje

```
for(Location loc : locations()) {
```

Forklar denne kodelinjen,

- hva gjør koden?
- hvilken kode er linjen avhengig av? List opp metoder/klasser i løsningsforslaget som blir kallt av denne linjen.

Skriv ditt svar her

Maks poeng: 4

Denne seksjonen inneholder 4 oppgaver fordelt på 3 tema der du skal implementere kode.

- Alle oppgavene kommer med noe kode, to av dem kommer med interface du skal implementere og en kommer med ferdig kode som du skal teste og fikse.
- Dere velger selv om dere skriver rett inn i Inspera eller om dere programmerer i annen editor og kopierer inn svaret
- Interfacene og koden gitt i denne seksjonen er ferdig med javadoc, du trenger ikke skrive noen javadoc kommentarer.
- Det kan lønne seg å skrive noen kommentarer, særlig hvis du er usikker på om du har gjort rett.
- God kodestil og lesbar kode gir poeng, gode kommentarer kan veie opp for at koden din ikke er så lesbar.
- Hvis du vet hva du skal gjøre men ikke får til å skrive koden kan du få noen poeng for å forklare hva du mente å gjøre.
- Dere blir bedømt både på at koden er korrekt og på kodekvalitet, e.g. variabelnavn.
- Vi trekker ikke poeng for små syntax feil som glemt semikolon på slutten av linjen.

13 En handleliste er en liste med varer. Vi har implementert ShoppingItem som representerer varer og et interface IShoppingList. Du må skrive en klasse ShoppingList som implementerer IShoppingList. Kommentarene i IShoppingList hjelper deg å finne ut hva du må gjøre.

IShoppingList.java

ShoppingItem.java

1. Klassen må ha en konstruktør. Du velger selv hvilke input parametre denne konstruktøren skal ha, velg det som er lettest for deg. Du får ikke ekstra poeng for fancy kostruktører.
2. Det er 5 metoder i interfacet, implementer alle disse metodene.

Skriv ditt svar her

```
1
```

Maks poeng: 10

14 Coronaviruset har gitt oss utfordringer på mange måter, men det gir også noen nye muligheter. Websider som worldometers.info har fått mange nye besøkende som følger med på statistikken. For å få til en slik webside ligger det en del programmering bak.

Vedlagt til denne oppgaven er et interface kallt ICoronaData som dere skal jobbe med i denne oppgaven.
ICoronaData.java

Dere skal implementere følgende to metoder: cumulativeDeaths() og deathsPerMillion().
Enten lag en abstrakt en klasse som implementerer interfacet ICoronaData, eller implementer default metoder i ICoronaData.
Hvis du trenger å implementere andre metoder er det greit, men det er de to metodene nevnt over du får poeng for.

Skriv ditt svar her...

```
1
```

Maks poeng: 15

15 Spillet Scrabble skal lages. Spillet går ut på å omorganisere noen av de 7 gitte bokstavene til et ord. Vi må sjekke at det ordet spilleren ønsker å lage faktisk går an å lage med de bokstavene spilleren har. Vi skiller ikke på små og store bokstaver, vi ønsker å godta input som inneholder en blanding av små og store bokstaver og gjør om til store bokstaver inne i metoden. (For denne oppgaven antar vi at det kun brukes bokstaver fra det engelske alfabetet).
Scrabble har en blank brikke som brukes som 'wildcard' I denne oppgaven ser vi bort fra dette, men i INF102 vil dere lære det.

Vi er kommet godt i gang med spillet og har skrevet følgende metode med tilhørende JUnit tester. Denne metoden skal kun sjekke om det er mulig å lage ordet fra bokstavene, ikke om ordet er lov å legge ut (finnes i ordlisten og passer med de eksisterende ordene).
Vi ønsker at metoden fungerer uansett hvor lang "letters" og "word" er.

Scrabble.java

```java
/**
 * This method is used by the game Scrabble where the goal is to rearrange
 * some of the given letters into a word.
 * A letter in Scrabble is always upper case, this method accepts both upper and lower case letters
 * and considers e.g. a lower case 'a' equal to an upper case 'A'
 * You may assume that all letters in input are valid letters in the English alphabet.
 *
 * @param letters - the letters you have to your disposal
 * @param word - the word you are trying to form
 * @return true if it is possible to form the word by rearranging the letters, false otherwise
 */
public static boolean canMake(String letters, String word) {
    letters = letters.toUpperCase();
    word = word.toUpperCase();

    for(Character c:word.toCharArray()) {
        if(!letters.contains(c.toString()))
            return false;
    }
    return true;
}
```

Vi skrev to JUnit tester og begge testene passerer, og vi får ingen Exception når vi kjører spillet.
ScrabbleTest

```java
@Test
void testCanMakeEksamen(){
    assertTrue(Scrabble.canMake("SKAMENE", "EKSAMEN"));
    assertTrue(Scrabble.canMake("skaMENE", "EKSAMEN"));
    assertTrue(Scrabble.canMake("SKAMENE", "eksAMEN"));
}

@Test
void testCanNotMakeFerie(){
    assertFalse(Scrabble.canMake("SKAMENE", "FERIE"));
}
```

Men det virker ikke slik som vi forventet (en logisk feil gjør at vi av og til får lov til å legge ut ord som ikke går an å lage) og vi er sikker på at det er metoden canMake som er feil. Du må hjelpe med å finne feilen.

Finn feilen og skriv en test som fanger opp den feilen som er i koden, det vil si en test som feiler slik koden er nå, men vil passere når feilen er rettet opp.

Skriv kode for test her

```
1
```

Maks poeng: 5

16 Forrige oppgave (ScrabbleTest) var å skrive en test som feiler. Skriv om metoden canMake for å fikse feilen.
Skriv ditt svar her

```
1
```

Maks poeng: 6

17 Denne oppgaven skal du ikke gjøre noe på. Her får du poeng du for det du har gjort på semesteroppgavene.
Du er nå ferdig med eksamen :-)
Vennlig hilsen
Martin
Skriv en hyggelig melding til foreleser ;-)

Maks poeng: 30