Oblig 1 – Teori

Oppgave 1.1

For å kjøre et Java-applikasjon må man først kompilere Java-koden om til bytekode.
Java Runtime Environment, forkortet JRE, er et kjøresystem eller en programvare som er nødvendig for å kjøre bytekoden ved hjelp av JVM (Java Virtual Machine).
Java Development Kit, forkortet JDK, er et utviklingsverktøy som er nødvendig for å lage og feilsøke java-applikasjoner.

Oppgave 1.2

Java-applikasjoner er bygd opp av tre hovedkomponenter: Javakode, JDK og JRE. Utviklingen starter med å skrive Java-kode i en tekseditor eller et IDE, som InelliJ. Koden lagres i filer med filtypen .java, som inneholder syntaks, datatyper og objektorienterte programmeringskonsepter som klasser og objekter.

For at datamaskinen skal kunne forstå Java-koden, kompileres den med javac som er et verktøy fra JDK. Den gjør lesbar kode for mennesker, til bytecode som maskinen kan forstå. Bytecode lagres i .class-filer.

Etter at koden er kompilert til bytecode, kan den kjøres gjennom maskiner som har en JRE. JRE inneholder JVM, og er nødvendig for å kjøre Java-applikasjoner. JVM tolker bytecode og utfører den på maskinen.

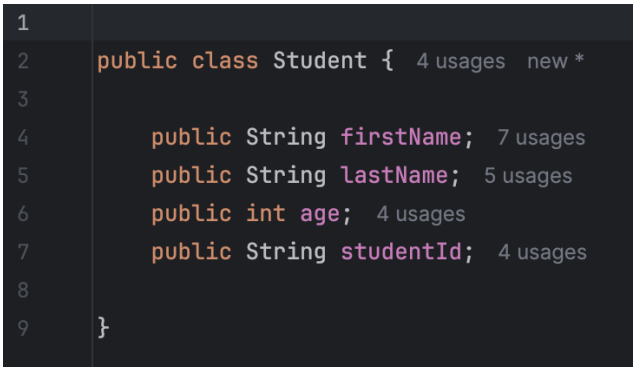
Oppgave 1.3

Compile time errors oppstår når det er en feil i syntax og semantikk når programmet kompileres. Eksempler på feil kan være syntaksfeil som ugyldig nøkkelord, at man mangler et semikolon eller feilplassering av parenteser. En annen feil er hvis man skriver en string til en int.

Runtime errors oppstår når programmet er kompilert, men det er skjer en feil under kjøringen. Feilen oppstår når noe uventet skjer som programmet ikke håndterer. Eksempler på slike feil kan være exceptions som NullPointerException eller prøve å dele noe på 0.

Oppgave 1.4

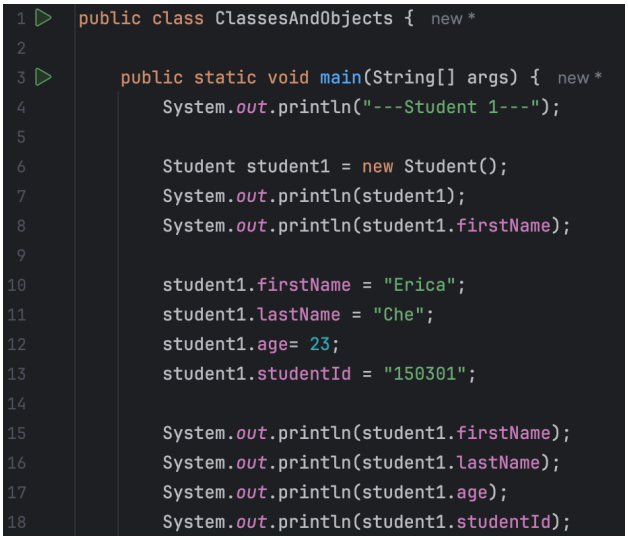
En klasse er en mal eller blueprint og beskriver hvordan objekter skal se ut og oppføre seg. En klasse består av et hode og en kropp. Hodet inneholder public class MediaPlayer, og koden skrives i kroppen. Kodeinnholdet av en klasse er bygd opp av variabler (egenskaper) og metoder (funksjoner) som objektene vil ha.



```
1
2   public class Student {  4 usages  new *
3
4       public String firstName;  7 usages
5       public String lastName;  5 usages
6       public int age;  4 usages
7       public String studentId;  4 usages
8
9   }
```

Her defineres klassen Student.
I kroppen skrives koden, og i eksemplet er det egenskapene objektene skal ha.

Et objekt er en instans av en klasse. Ved å definere en klasse, kan man opprette flere objekter fra denne klassen. Hvert objekt har sitt eget data med tilgang til klassens metoder.



```
1   public class ClassesAndObjects {  new *
2
3       public static void main(String[] args) {  new *
4           System.out.println("---Student 1---");
5
6           Student student1 = new Student();
7           System.out.println(student1);
8           System.out.println(student1.firstName);
9
10          student1.firstName = "Erica";
11          student1.lastName = "Che";
12          student1.age= 23;
13          student1.studentId = "150301";
14
15          System.out.println(student1.firstName);
16          System.out.println(student1.lastName);
17          System.out.println(student1.age);
18          System.out.println(student1.studentId);
```

Fra linje 6-8 opprettes et objekt av klassen Student.

Fra linje 10-13 settes verdiene for objektet.