1. Leg uit wat het verschil is tussen een klasse en een object

Een object is alles wat we willen onderscheiden als een betekenisvolle eenheid. Dat kan iets zijn uit de fysieke werkelijkheid zoals een auto, een hotel, een kamer, een televisietoestel of een mens, of iets abstracts zoals een reservering, een vonnis, een schaakpartij of een contract. Objecten worden gekenmerkt door een toestand en door gedragsmogelijkheden. Een klasse specificeert een groep van objecten met overeenkomstige attributen, overeenkomstige relaties en overeenkomstig gedrag

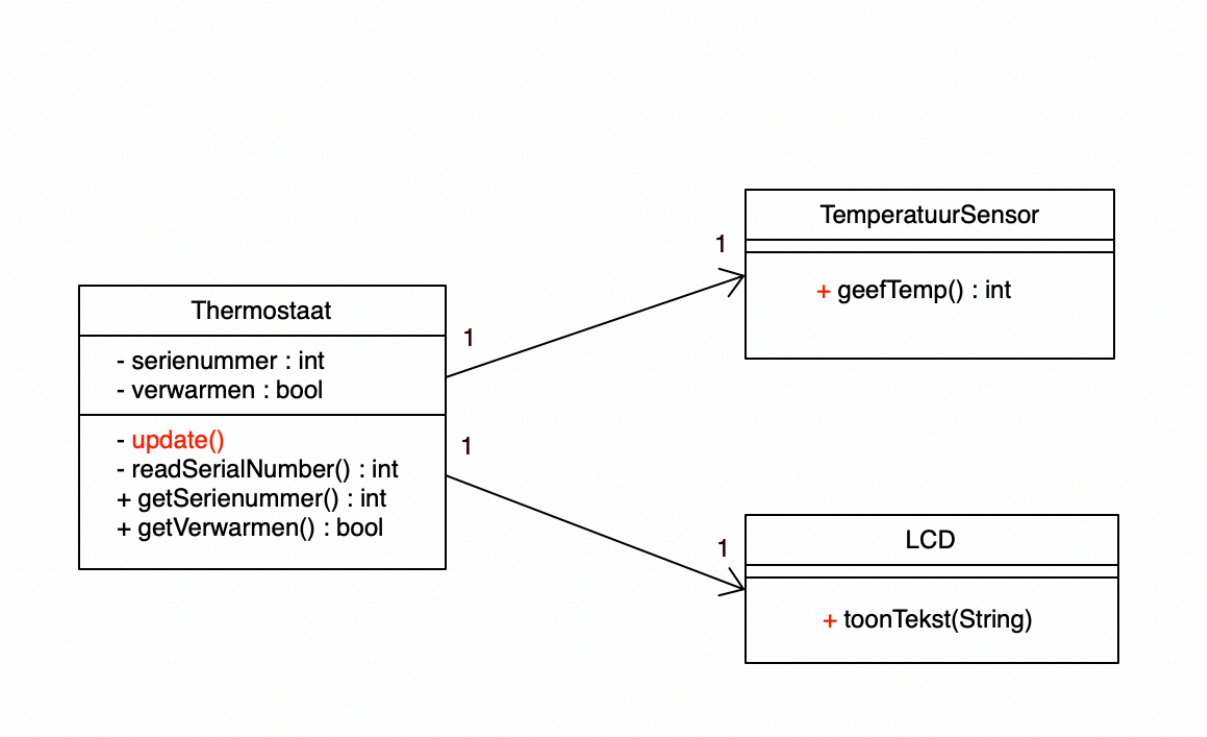
2. Leg uit wat bij object georiënteerd programmeren inkapseling inhoudt. Geef daarnaast een voorbeeld.
Inkapseling houdt in dat (sommige) attributen van een klasse niet toegankelijk zijn voor andere klassen. Hiervoor zijn dan methoden gedefinieerd die die toegang regelen.
Voorbeeld: een attribuut 'id' van een klasse Persoon die mbv de constructor een waarde krijgt, en verder alleen d.m.v. een methode 'getID()' kan worden opgevraagd.

3. Leg uit wat bij object georiënteerd programmeren overerving inhoudt. Geef daarnaast een voorbeeld.
Een hiërarchie van klassen en subklassen, waarbij een subklasse attributen en methoden erft van superklassen. Bijvoorbeeld een klasse 'Leerling' die de attributen 'voornaam' en 'achternaam' en de methode 'getID()' erft van een superklasse 'Persoon'.

4. Hieronder staat de code voor een JavaScript klasse waarin gegevens voor een persoon kunnen worden opgeslagen. Alle attributen zijn nu public. Welke attribu(u)t(en) zou(den) beter private kunnen zijn? Leg per attribuut uit waarom.

Het zo het veiligst zijn om alle attributen private te maken en via methoden uit te vragen of te bewerken (=> inkapseling). Het zou voor voornaam en achternaam niet onoverkomelijk zijn om deze public te houden. De attributen 'geboortedatum' en 'leeftijd' moeten wel private zijn. De waarde van dit attribuut wordt op basis van de aan de constructor meegeven geboortedatum berekend. Het moet voorkomen worden dat deze waarde of die van geboortedatum door een andere klasse veranderd kan worden en zo de 'verhouding' tussen deze twee verbroken wordt.
Een methode 'setGeboortedatum(date)' en getLeeftijd() zijn hierbij nodig.

5. Hieronder zie je de code van meerdere C++ klassen. Teken het bijbehorende klassendiagram



6. Hieronder zie je een klassendiagram. Maak de bijbehorende JavaScript-programmeercode.

```
class Persoon {
  voornaam;
  achternaam;
  email;

  constructor(_voornaam, _achternaam, _email) {
    this.voornaam = _voornaam;
    this.achternaam = _achternaam;
    this.email = _email;
  }
}

class Mentor extends Persoon {
  afkorting;

  constructor(_voornaam, _achternaam, _email, _afkorting) {
    super(_voornaam, _achternaam, _email);
    this.afkorting = _afkorting;
  }
}

class Leerling extends Persoon {
  leerlingnummer;

  constructor(_voornaam, _achternaam, _email, _leerlingnummer) {
    super(_voornaam, _achternaam, _email);
    this.leerlingnummer = _leerlingnummer;
  }
}

class Klas {
  mentor;
  leerlingen;

  // _mentor moet een object zijn van de klasse Mentor
  // _leerlingen moet een array zijn van leerling-objecten
  constructor(_mentor, _leerlingen) {
    this.mentor = _mentor;
    this.leerlingen = _leerlingen;
  }
}
```

7. Hieronder zie je JavaScript-code van een ‘spel’ waarin twee ballen tegen de wanden stuiteren. Teken een objectdiagram voor beide ballen voor de toestand waarin ze verkeren nadat de draw-methode tweemaal is uitgevoerd.