Leren programmeren - Opdrachten

Bart Duisters

Formele logica

Geldige of ongeldige redenering

- Bewering 1: Alle mensen geboren in 1991 zijn oud.
- Bewering 2: Mijn tweelingsbroer is geboren op 13 augustus 1991.
- Conclusie: Ik ben oud.

- Bewering 1: Een cursist die zichzelf heeft ingeschreven op een maandag, haalt minsten 90% op alle opdrachten.
- Bewering 2: 90% van alle cursisten zijn ingeschreven op een maandag.
- Conclusie: 10% van alle cursisten behaalt 80% op alle opdrachten.

- Bewering 1: Een leugenaar spreekt nooit de waarheid.
- Bewering 2: Brigitte is geen leugenaar.
- Conclusie: Brigitte spreekt altijd de waarheid.

- Bewering 1: Het samenvoegen van de verfkleuren rood en geel, vormen de kleur paars.
- Bewering 2: Het samenvoegen van de verfkleuren groen en geel, vormen de kleur paars.
- Bewering 2: De school op het einde van de straat is paars.
- Conclusie: Het paars is bekomen door het samenvoegen van ofwel rood en geel, ofwel groen en geel.

Geef een geldige conclusie die afgeleid kan worden uit de beweringen

- Bewering 1: Alle voertuigen hebben minstens drie wielen.
- Bewering 2: Ik heb een motor.
- Conclusie: ...

- Bewering 1: Alle pizza's zijn vierkant.
- Bewering 2: Alle pizza's zijn blauw.
- Bewering 3: Ik heb een pizza besteld om 18u00.
- Bewering 4: De pizza wordt geleverd om 19u00.
- Conclusie: ...

Deductieve en inductieve logica

Geef aan of het een geldige of ongeldige conclusie is. Geef aan of het om deductieve of

inductieve beweringen gaat. Indien het om inductieve beweringen gaat, geef dan ook aan of het om zwakke of sterke inductieve beweringen gaat.

- Bewering 1: Alle katten zijn dieren.
- Bewering 2: Alle dieren hebben zes poten.
- Conclusie: Alle katten hebben zes poten.

- Bewering 1: Alle voorgaande jaren zijn alle cursisten altijd geslaagd op de eindtesten.
- Bewering 2: Dit jaar zijn er 100 cursisten.
- Conclusie: Er zullen dit jaar met zekerheid 100 geslaagde cursisten zijn.

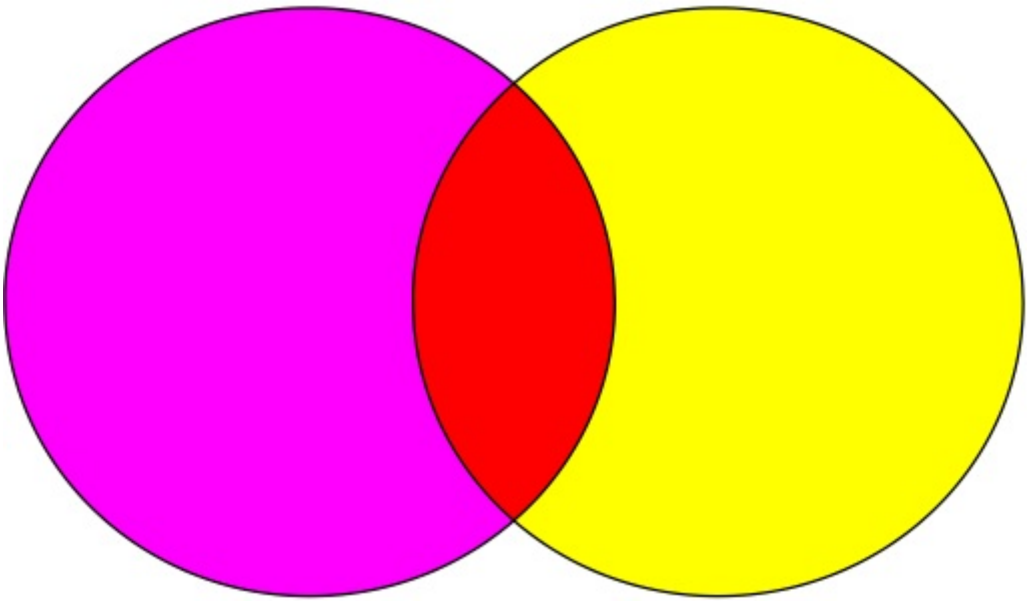
- Bewering 1: Alle voorgaande jaren zijn alle cursisten altijd geslaagd op de eindtesten.
- Bewering 2: Dit jaar zijn er 100 cursisten.
- Conclusie: Er zullen dit jaar waarschijnlijk 100 geslaagde cursisten zijn.

Categoriale logica

Er staat bij elke opdracht waarvoor een cirkel staat. Vul de gegeven zinnen aan.

Opdracht 1

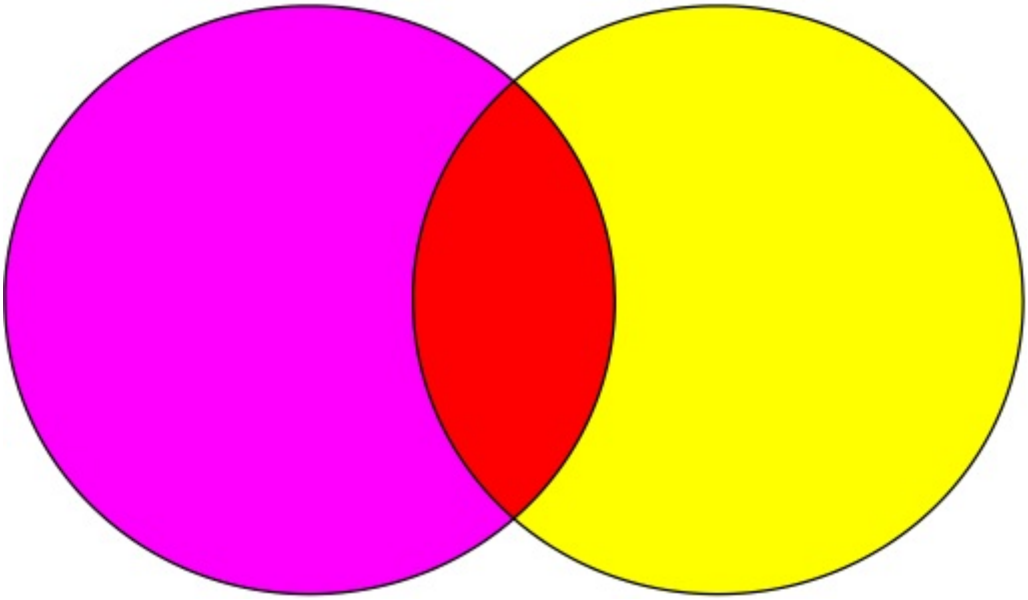
Roze cirkel: Alle cursisten Gele cirkel: Alle docenten



- Het roze gedeelte bevat ...
- Het gele gedeelte bevat ...
- Het rode gedeelte bevat ...

Opdracht 2

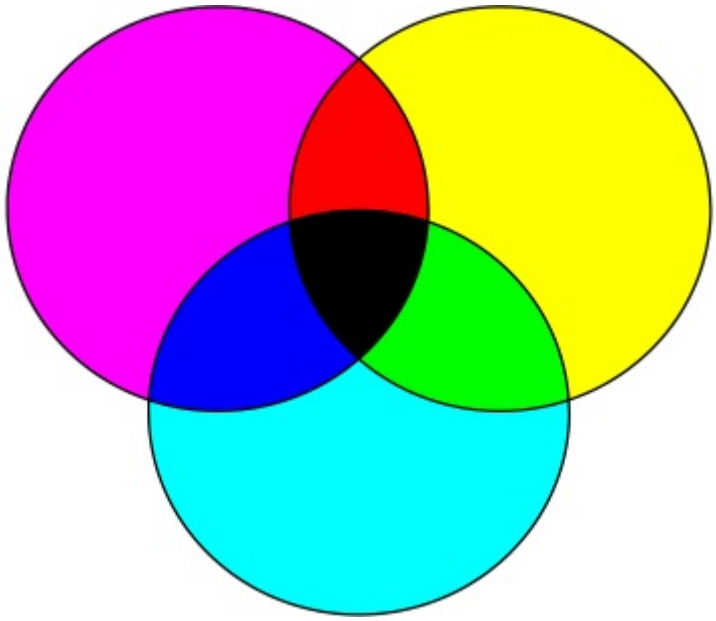
Roze cirkel: Alle wagens Gele cirkel: Alle blauwe voertuigen



- Het roze gedeelte bevat ...
- Het gele gedeelte bevat ...
- Het rode gedeelte bevat ...
- Het roze gedeelte gecombineerd met het rode gedeelte bevat ...

Opdracht 3

Roze cirkel: Alle wagens Gele cirkel: Alle elektrische voertuigen Lichtblauwe cirkel: Alle brandstofvoertuigen



- Het roze gedeelte bevat ...
- Het gele gedeelte bevat ...
- Het lichtblauwe gedeelte bevat ...
- Het rode gedeelte bevat ...
- Het groene gedeelte bevat ...
- Het donkerblauwe gedeelte bevat ...
- Het zwarte gedeelte bevat ...
- Het roze gedeelte gecombineerd met het rode en donkerblauwe gedeelte bevat ...

Basis programmeren - Syntax omschrijven

Kopieer de code naar een snippet. Zet in commentaar wat er gebeurd.

Uitgewerkt voorbeeld

```
/*
Onderstaande code bevat vier statements
*/
let x, y, uitkomst; // Statement 1
x = 6; // Statement 2
y = 9; // Statement 3
uitkomst = x + y; // Statement 4

/*
Statement 1 bevat drie declaraties: variabelen x, y en uitkomst worden gedeclareerd.
Er wordt gebruik gemaakt van het keyword let.
Op dit moment is het type van x, y en uitkomst "undefined".
*/
/*
Statement 2 bevat een toekenning van waarde 6 aan variabele x.
Op dit moment is het type van x "number".
*/
/*
Statement 3 bevat een toekenning van waarde 9 aan variabele y.
Op dit moment is het type van y "number".
*/
/*
Statement 4 bevat een berekening. Er wordt gebruik gemaakt van de operator +.
De variabelen x en y worden bij elkaar opgeteld.
Het resultaat van deze berekening wordt toegekend aan de variabele uitkomst.
Op dit moment is het type van uitkomst "number".
Op dit moment is de waarde van uitkomst 15.
*/
```

Opdracht 1

```
const getal1 = 10,
  getal2 = 20;
let uitkomst;

uitkomst = getal1 * getal1;
uitkomst = getal1 * getal2;
uitkomst = uitkomst + getal1;
```

Opdracht 2

```
let x = 2,
  y = 3;
let uitkomst;
uitkomst = x - y;
```

Opdracht 3

```
const a = 3;
a = 4;
```

Opdracht 4

```
let a = 3;
a = 4;
a = a + a;
```

Opdracht 5

```
function som(getal1, getal2) {
  return getal1 + getal2;
}

som(5, 5);
som(5, "5");
```

Opdracht 6

```
let teller = 0;

function som(getal1, getal2) {
  teller = teller + 1;
  return getal1 + getal2;
}

teller = teller + 1;
som(5, 5);
som(5, "5");
teller = teller + 1;
```

Basis programmeren - Syntax schrijven

Schrijf bij elke opdracht code die voldoet aan de omschrijving. Kies bij elke uitwerking voor het best passende keyword om een variabele te declareren.

Belangrijke terminologie: `Initialiseren` betekent `declareren en meteen een waarde toekennen`.

Opdracht 1

Initialiseer twee variabelen genaamd `voornaam` en `achternaam` met jouw voornaam en achternaam. Declareer een variabele genaamd `volledigeNaam`. Concateneer de `voornaam`, een spatie en de `achternaam` en ken het resultaat toe aan de variabele `volledigeNaam`.

Print de waarde van de variabele `volledigeNaam` in de console. Dit kan met `console.log(variabeleNaam)`, waarbij variabeleNaam de naam van een variabele is.

Opdracht 2

Declareer drie variabelen, genaamd `getal1`, `getal2` en `getal3`. Ken respectievelijk de waarden `3`, `6` en `9` toe aan de drie variabelen.

Initialiseer een variabele genaamd `resultaat` met het resultaat van deze expressie: getal2 - getal1 + getal3.

Print de waarde van de variabele `resultaat` in de console.

Opdracht 3

Initialiseer twee variabelen, genaamd `x` en `y` met de waarden `10` en `100`. Declareer een variabele genaamd `resultaat`. Ken het resultaat van de expressie `x > y` toe aan de variabele `resultaat`. Zet onder elke regel de waarde en het type van elke variabele in de lijn boven de lijn met commentaar.

Opdracht 4

- Declareer twee variabelen, genaamd `type1` en `type2`.
- Ken aan `type1` het resultaat van de expressie `typeof {}` toe.
- Ken aan `type2` het resultaat van de expressie `typeof []` toe.
- Print in de console het resultaat van de vergelijking van de waarden `type1` en `type2`.
- Zet onder elke regel de waarde en het type van elke variabele in de lijn boven de lijn met commentaar.
- Zet in commentaar wat er uitgeprint zal worden door console.log

Extra uitleg:

- `{}` is de syntax om een object aan te maken in JavaScript
- `[]` is de syntax om een array aan te maken in JavaScript
- Het keyword `typeof` geeft de waarde van wat erachterkomt terug als string
- Een vergelijking wordt gedaan met `===`

Opdracht 5

- Definieer een functie, genaamd `zetOmNaarString`
- De functie accepteert één parameter, genaamd `parameter`
- In het codeblok van de functie:
    - Zet de doorgegeven parameter om in een string
    - Geef de omgezette parameter terug
- Roep de functie op een cijfer en ken het resultaat toe aan een variabele genaamd `resultaat`.
- Zet in commentaar wat de waarde en het type van de variabele `resultaat` is na het toekennen van het resultaat van de functie.

Herinner: 3 + "3" = "33"

Conditionele statements omschrijven

- Omschrijf zo gedetailleerd mogelijk wat er te zien is bij elke opdracht.
- Vul de console.log met lege string aan met een zinvolle omschrijving.
- Geef bij elk blok code (if, else, case) aan of het blok code wordt uitgevoerd.

Uitgewerkte voorbeelden

```
if (true) {
  // 1.
  console.log("Dit wordt uitgeprint in de console"); // 2.
}

/*
1. Er is een if-statement gedefinieerd met een voorwaarde true.
Dit zal altijd evalueren naar true, de string in het codeblok zal altijd uitgevoerd wo

2. Dit statement zal de tekst "Dit wordt uitgeprint in de console" printen in de conso
*/
```

```
let x; // 1.

if (typeof x === "undefined") {
  // 2.
  console.log("x is undefined"); // 3.
}

/*
1. Er wordt een variabele x gedeclareerd met keyword let. De waarde is undefined.
Het type is "undefined".
2. Er is een if-statement gedefinieerd, de voorwaarde vergelijkt
het type van de variabele x met een string "undefined".
Dit evalueert naar true. Het codeblok van de if-statement zal uitgevoerd worden.
3. Dit statement zal de tekst "x is undefined" in de console printen.
*/
```

Opdracht 1

```
const age = 30;

if (age > 18) {
  console.log("");
}
```

Opdracht 2

```
const age = 30;

if (age < 18) {
  console.log("");
} else {
  console.log("");
}
```

Opdracht 3

```
const age = 30;

if (age <= 18) {
```

```
  console.log("");
} else if (age < 30) {
  console.log("");
} else {
  console.log("");
}
```

Opdracht 4

```
const digit = 3;
let result = "";

switch (digit) {
  case 1:
    result = "Een";
    break;
  case 2:
    result = "Twee";
    break;
  case 3:
    result = "Drie";
    break;
  case 4:
    result = "Vier";
    break;
  case 5:
    result = "Vijf";
    break;
  default:
    result = "Getal niet gekend";
}
```

Conditionele statements schrijven

Opdracht 1

Herschrijf `Opdracht 4` uit de vorige reeks oefeningen. Maak hierbij gebruik van if, else-if en else.

Opdracht 2

Schrijf een functie genaamd `grootsteGetal` die twee parameters accepeert. Zorg ervoor dat de functie de grootste parameter teruggeeft.

De commentaar bij onderstaande code is het verwachte resultaat:

```
console.log("De grootste parameter is: ", grootsteGetal(2, 3));
// De grootste parameter is 3

console.log("De grootste parameter is: ", grootsteGetal(3, 2));
// De grootste parameter is 3

console.log("De grootste parameter is: ", grootsteGetal(2, 2));
// De grootste parameter is 2
```

Opdracht 3

Schrijf een functie genaamd `evenGetal` die één parameter accepteert. Zorg ervoor dat de functie `true` teruggeeft als de parameter even is. Zorg ervoor dat de functie `false` teruggeeft als de parameter oneven is.

Verwachte resultaat:

```
console.log(evenGetal(0)); // true
console.log(evenGetal(1)); // false
console.log(evenGetal(10)); // true
console.log(evenGetal(13)); // false
console.log(evenGetal(69)); // false
```

Opdracht 4

In Finland is het puntensysteem anders dan in België. In Finland worden testen beoordeeld op 5.

0/5 is gelijk aan een 0/100 in België. 1/5 is gelijk aan een 50/100 in België. 5/5 is gelijk aan een 100/100 in België.

2, 3 en 4 op 5 zijn gelijk aan een waarde tussen 50 en 100 in België.

Schrijf een functie genaamd `convertScore` die één parameter accepteert. De parameter is de score in België. De functie print uit wat de score in Finland is.

Verwachte resultaat:

```
convertScore(0); // "In Finland is dit 0 op 5"
convertScore(12); // "In Finland is dit 0 op 5"
convertScore(30); // "In Finland is dit 0 op 5"

convertScore(50); // "In Finland is dit 1, 2, 3 of 4 op 5"
convertScore(99); // "In Finland is dit 1, 2, 3 of 4 op 5"
convertScore(75); // "In Finland is dit 1, 2, 3 of 4 op 5"
convertScore(51); // "In Finland is dit 1, 2, 3 of 4 op 5"

convertScore(100); // "In Finland is dit 5 op 5"
```

Opdracht 5

Maak een functie genaamd `gewerveldeDierenCategorie`. De functie accepteert één parameter (een string). Maak gebruik van een switch-statement.

Parameter: `"Kikker"` `"Salamander"` `"Wormsalamander"` Resultaat: `"Amfibie"`

Parameter: `"Hagedis"` `"Slang"` `"Krokodil"` `"Schildpad"` Resultaat: `"Reptiel"`

Parameter: `"Aal"` `"Kabeljauw"` `"Forel"` Resultaat: `"Vis"`

Parameter: `"Kanarie"` `"Merel"` `"Valk"` Resultaat: `"Vogel"`

Parameter: `"Dolfijn"` `"Walvis"` `"Vogelbekdier"` `"Mens"` Resultaat: `"Zoogdier"`

Iteratieve statements omschrijven

- Omschrijf zo gedetailleerd mogelijk wat er te zien is bij elke opdracht.
- Geef aan hoevaak het codeblok wordt uitgevoerd.

Uitgewerkt voorbeeld

```
let i = 0; // 1.

while (i < 2) {
  // 4.
  // 5.

  console.log("Loop: ", i); // 2.
  i++; // 3.
}

/*
 * 1. Er wordt met keyword let een variabele i geïnitaliseerd met waarde 0
 */
/*
 * 2. Uitprinten van "Loop: " met daarachter de waarde van i
 */
/*
 * 3. Ophogen van i met 1
 */
/*
 * 4. Dit codeblok wordt uitgevoerd wanneer de expressie
 * (i < 2) evalueert naar true
 */
/*
 * 5. Dit codeblok wordt twee keer uitgevoerd
 */
```

Opdracht 1

```
let i = 2;

while (i > 0) {
  console.log("Loop: ", i);
  i--;
}
```

Opdracht 2

```
const resulaat = typeof {} && typeof [];
let i = 0;

do {
  console.log(i);
  i++;
} while (resultaat);
```

Opdracht 3

Extra opdracht: Omschrijf elke iteratie.

```
for (let i = 9; i > 0; i--) {
  console.log("Voor: ", i);
  i--;
  console.log("Na: ", i);
  if (i === 8) {
    break;
  }
}
```

Opdracht 4

Extra opdracht: Omschrijf elke iteratie.

```
for (let i = 9; i > 0; i--) {
  console.log("Voor: ", i);
  i--;
  console.log("Na: ", i);
  if (i === 7) {
    break;
  }
}
```

Iteratieve statements schrijven

Opdracht 1

- Initialiseer een variabele i met waarde 0.
- Maak gebruik van een while loop.
- Elke iteratie wordt i met 1 opghoogd.
- Maak gebruik van een while loop om elk getal dat geheel deelbaar is door 10 uit te printen.
- Zorg dat er 100 iteraties gebeuren.

Opdracht 2

- Gebruik de oplossing van Opdracht 1.
- Maak een functie genaamd `deelbaarDoorTien`.
- De functie accepteert één parameter.
    - Wanneer de functie wordt aangeroepen, zal de parameter het aantal iteraties dat gedaan moet worden bevatten.

Opdracht 3

- Gebruik de oplossing van Opdracht 2.
- Hernoem de functie naar `deelbaarDoorN`.
- Voeg een tweede parameter toe.
    - Wanneer de functie wordt aangeroepen, zal de tweede parameter het getal bevatten dat gebruikt wordt bij de gehele deling.

Objecten en arrays omschrijven

Opdracht 1

```
const cursisten = [
  "Angelique",
  "Arne",
  "Domenico",
  "Ian",
  "Jochen",
  "Jorg",
  "Marco",
  "Marvi",
  "Michelle",
  "Romy",
  "Simone",
  "Sofian",
  "Tony",
  "Yoeri",
];

console.log(cursisten[2]);
```

Opdracht 2

```
const getallen = [3, 1, 3, 2];
const eersteGetal = getallen[0];

getallen[0] = getallen[1];
getallen[1] = eersteGetal;
```

Opdracht 3

```
const fiets = {
  merk: "Specialized",
  type: "Rockhopper",
  kleur: "wit",
  soort: "Mountainbike",
};
```

Opdracht 4

```
const hond = {
  ras: "Bearded Collie",
  naam: "Samson",
  bijnaam: "Bob",
  spreek: () => {
    console.log("Mwoaah seg hé!");
  },
};
```

Objecten en arrays schrijven

Opdracht 1

- Declareer een variabele genaamd `hobby`.

- Declareer een variabele genaamd `ik`.
- Ken een waarde toe aan de variabele `hobby`, de waarde is een object met als properties:
    - naam, de naam bevat een waarde van het type "string" en omschrijft de hobby
    - aantalUren, dit bevat het aantal uren dat gespendeerd wordt aan de hobby per week
- Ken een waarde toe aan de variabele `ik`, de waarde is een object met als properties:
    - voornaam, dit bevat een "string" met als waarde jouw voornaam
    - achternaam, dit bevat een "string" met als waarde jouw achternaam
    - hobby, ken hieraan de variabele `hobby` toe
- Print uit, gebruikmakende van het object `hobby`, hoeveel uren er per week gespendeerd worden.
- Print uit wat de naam is van de hobby, gebruikmakende van de variabele `ik`.

Opdracht 2

- Initialiseer een variabele genaamd `pi`, met als waarde een array. Geef tijdens het toekennen van de array vijf initiële waarden mee aan de array. Zoek op het internet wat de eerste vijf getallen ná de komma zijn van het getal Pi en ken deze toe als initiële waarden.
- Voeg het zesde getal ná de komma toe op index 5.

Opdracht 3

- Initialiseer een variabele genaamd `calculator` en ken er een object aan toe met de volgende properties:

    - `som`, met als waarde een anonieme functie die twee parameters accepteert. Geef de som van de twee parameters terug als resultaat van de functie.
    - `verschil`, met als waarde een anonieme functie die twee parameters accepteert. Geef het verschil van de twee parameters terug als resultaat van de functie.

- Ken een extra property toe aan de variabele `calculator`:

    - `vermenigvuldigen`, met als waarde een anonieme functie die twee parameters accepteert. Geef de vermenigvuldiging van de twee parameters terug als resultaat van de functie.

- Ken een extra property toe aan de variabele `calculator`:

    - `delen`, met als waarde een anonieme functie die twee parameters accepteert. Geef de deling van de twee parameters terug als resultaat van de functie.

- Maak gebruik van het object in de variabele `calculator` om onderstaande berekeningen te doen en print het resultaat er van uit:

    - 2 + 3
    - 128 * 2
    - 1414 - 707
    - 355 / 113

Opdrachten - alle kennis

Extra informatie: Net zoals `console.log()` een functie is om iets in de console uit te printen, zijn er ook nog andere functies.

- `alert()`, met als parameter een string, zal de string in een popup tonen.
- `prompt()`, met als parameter een string, zal de string in een popup tonen én de popup bevat een invoerveld. Wat ingevoerd wordt in het invoerveld, wordt teruggeven als `return` van de functie `prompt()`.

Belangrijk: Dit werkt alleen in de browser. Voer het uit als snippet!

Een voorbeeld:

```
const inputVanGebruiker = prompt("Wat is jouw naam?");
/*
 * Wanneer JavaScript bovenstaande regel uitvoert (in de browser)
 * verschijnt er een popup met een invoerveld.
 * Boven het invoerveld staat "Wat is jouw naam?".
 * Hetgeen de gebruiker invoert, zal in de variabele genaamd inputVanGebruiker
 * gestoken worden.
 */

// Stel dat de gebruiker als input meegeeft "Samson".
console.log(inputVanGebruiker);
// Bovenstaande regel zal "Samson" printen in de console.

alert("Hallo " + inputVanGebruiker);
// Bovenstaande regel zal een popup tonen met "Hallo Samson"
```

Opdracht 1

- Initialiseer een variabele genaamd `voornaam`, maak gebruik van `prompt()` om de input van de gebruiker te vragen, het resultaat is de waarde die toegekend zal worden aan de variabele `voornaam`:
    - Wat er gevraagd moet worden: "wat is jouw voornaam?"
- Initialiseer een variabele genaamd `achternaam`, maak gebruik van `prompt()` om de input van de gebruiker te vragen, het resultaat is de waarde die toegekend zal worden aan de variabele `achternaam`:
    - Wat er gevraagd moet worden: "wat is jouw achternaam?"
- Maak gebruik van `alert()` om de volledige naam aan de gebruiker te tonen.

Opdracht 2

- Definieer een functie genaamd `vraagInfoAanGebruiker`.
- Kopieer de code uit Opdracht 1 en plak deze in de body van de functie.
- Ken twee parameters toe aan de functie, genaamd `vraagVoornaam` en `vraagAchternaam`.
- Indien `vraagVoornaam` de waarde `false` bevat, wordt er niet naar de voornaam gevraagd.
- Indien `vraagAchternaam` de waarde `false` bevat, wordt er niet naar de achternaam gevraagd.
- Indien er een voornaam of een achternaam gekend is, toon de alert uit Opdracht 1. Indien ze beide niet gekend zijn, console.log "Niks om te tonen!".

Test de opdracht uit met:

```
vraagInfoAanGebruiker(true, false);
// Dit zou maar één prompt moeten tonen, die van de voornaam.

vraagInfoAanGebruiker(false, true);
// Dit zou maar één prompt moeten tonen, die van de achternaam.

vraagInfoAanGebruiker(true, true);
// Dit zou maar twee prompts moeten tonen, die van de voornaam én achternaam.

vraagInfoAanGebruiker(false, false);
```

```
// Dit zou géén prompts moeten tonen.
```

Opdracht 3

Fibonacci-nummers werken alsvolgt:

- 1e nummer is een 1
- 2e nummer is een 1
- 3e nummer is de som van 1e en 2e nummer
- 4e nummer is de som van 2e en 3e nummer
- 5e nummer is de som van 3e en 4e nummer
- ...
- Definieer een functie `berekenFibonacci`.
- Ken één parameter toe aan deze functie, genaamd `aantalNummers`.

Verwachte output staat in commentaar achter elke regel:

```
berekenFibonacci(1); // 1
berekenFibonacci(2); // 1 1
berekenFibonacci(5); // 1 1 2 3 5
berekenFibonacci(10); // 1 1 2 3 5 8 13 21 34 55
```

Wat is het honderste getal in de Fibonacci-reeks?

Opdracht 4

Bij het spelletje `blad steen schaar` kiezen twee partijen elk één optie. De regels zijn:

- blad wint van steen
- steen wint van schaar
- schaar wint van blad
- Definieer een functie genaamd `oneerlijkeBladSteenSchaar`.
- De functie accepteert één parameter genaamd `keuze`.
- In het codeblok van de functie wordt er gezorgd dat de functie altijd de verliezende keuze neemt ten opzichte van de parameter.
- Toon een `alert()` met "Gebruiker: [keuze], Computer: [keuze van computer], de gebruiker wint!". Waarbij [keuze] de doorgegeven parameter is en [keuze van computer] de verliezende optie.

Opdracht 5

Ga verder op Opdracht 4.

- Definieer een functie genaamd `speelSpel`.
- Zorg ervoor dat alle code in het codeblok drie keer wordt uitgevoerd.
- Vraag de gebruiker met `prompt()` naar `blad, steen of schaar?`, sla de keuze op in een variabele genaamd `keuzeVanGebruiker`.
- Roep de functie `oneerlijkeBladSteenSchaar` op met `keuzeVanGebruiker` als parameter.

Extra:

- Zorg ervoor dat de functie `speelSpel` één parameter accepteert, genaamd `aantal`.
- Deze parameter bepaalt het aantal keer dat het codeblok wordt uitgevoerd.