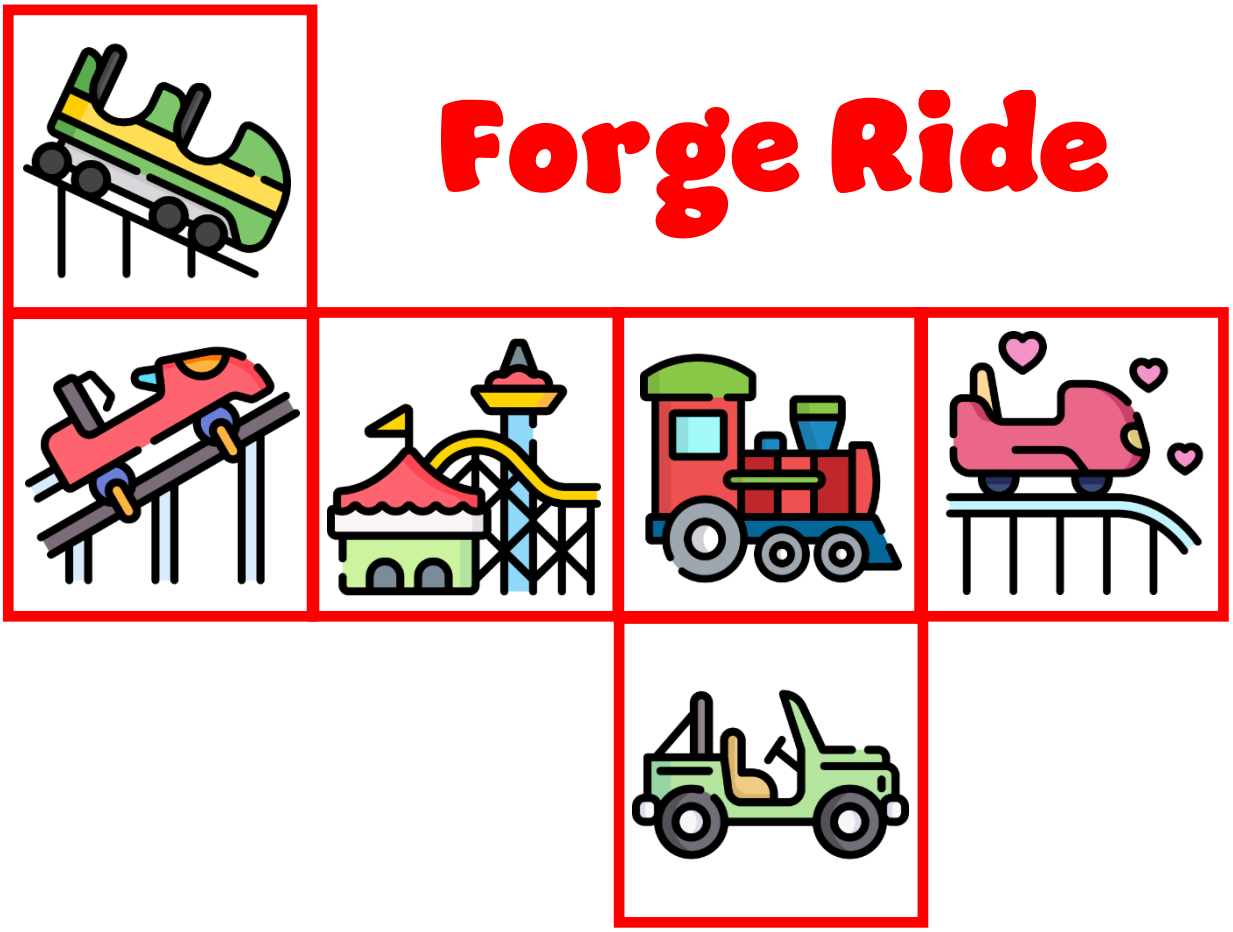


Forge Ride

Handleiding box 1: onderzoek [150 min]

Uitleg computationeel denken: extra informatie voor de leerkracht

Uitleg hardware: extra informatie voor de leerkracht

[15 min] Stap 1: vraaggestuurd gesprek

[5 min] stap 2: verbinding maken

[30 min] Stap 3: functies programmeren

[10 min] Stap 4: de code voor de testrit

[10 min] Stap 5: oei, het werkt niet / oei, het werkt niet goed!

[30 min] Stap 6: de verschillende plattegronden

[30 min] Stap 7: de lijnsensor gebruiken

[15 min] Extra's: snelheid, lichten en lussen

Uitleg computational denken

5 min

Woordenschat

Deze woorden worden gebruikt tijdens het programmeren.
Het is de bedoeling dat jij de woorden juist gebruikt en leerlingen deze oppikken.
We geven onderstaand telkens een voorbeeld van hoe dit in deze les aan bod komt.

Code: de blokjes met instructies voor de robot.
Vb: de software van mico:bit biedt 'bouwblokjes' met codetaal.

Programmeren: het proces van code schrijven, testen en bijsturen.
Vb: de leerlingen voerden de blokjes in maar niet alles zal meteen lukken.
Ze zullen af en toe moeten aanpassen in de code en opnieuw proberen.

Voorwaarde: nadat er een voorwaarde voldaan wordt volgt een actie.
Vb: wanneer de lijnsensor geen lijn detecteer moet hij opschuiven (motor aanzetten).

Invoer: schudden, geluid maken, duwen, … een signaal geven aan de Micro:bit.
Vb: we duwen op de knop 'A' om de robot in actie te laten treden.

Lussen: de handeling een bepaald aantal keer laten herhalen.
Vb: als de robot 3 sec. vooruit moet kan je zetten 'herhaal 3 keer'.

Functie: een reeks met code groeperen om ze niet opnieuw in te moeten voeren.
Vb: de functie vooruit rijden bestaat uit:
beide motoren laten draaien en na een bepaalde tijd stoppen.

Digitale variabele: een waarde die kan variëren.
Analoge variabele: een getal dat zich situeert tussen minimum en maximum.
Vb. de lijnensor meet lichtsterkte en herkent zo de kleur van de ondergrond.
hij geeft dit variabel getal door en dan kan de motor aan of uit schakelen.

De onderstaande woorden maken deel uit van het **computationeel denken**.
Omdat we hier werken met een robot, sensoren en computer spreken we van een **plugged** werkvorm, dit betekent dat er met computer of elektronica gewerkt wordt.

Algoritme: stapjes of herhalingen in de code.

Debuggen: het vinden van fouten in een code en deze aanpassen.
Vb: de robot moet 90° draaien maar draait slechts 45°.
Ze moeten zoeken welk stukje code aangepast moet worden, en hoe dit moet.

Patroon: herhalingen in de code.
Vb. vooruit, rechts draaien, vooruit, rechts draaien = robot beweegt zizag.

Decompositie: het probleem opsplitsen in deelproblemen om op te lossen.
Vb. vooruit rijden, recht rijden, afdraaien, … worden afzonderlijk aangeleerd.
de leerlingen schrijven telkens een kort stukje code wat ze eerst testen.

Abstraheren: enkel de essentiële informatie of onderdelen gebruiken.
Vb. de leerlingen hebben maar enkele categorieën bouwblokken nodig.

Uitleg hardware

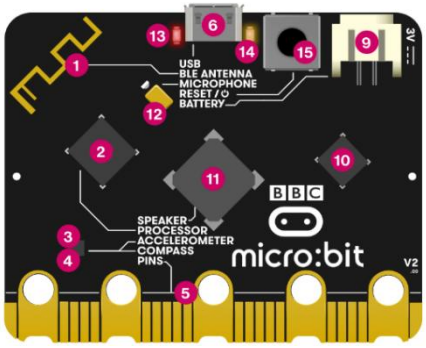
5 min

We gebruiken een kleine computer (Micro:bit) en een robot (Move).
We treden niet in detail maar overlopen de belangrijkste functies voor deze les.

1. De Micro:bit

De software is erg toegankelijk en vind je via: **https://microbit.org/nl**
Het coderen zelf gebeurt via: **https://makecode.microbit.org**

Je kan de Micro:bit bestuderen met de kinderen en de sensoren zoeken.
Het volledige overzicht: **https://microbit.org/get-started/features/overview**
Wij bespreken enkel wat je in de les nodig hebt.



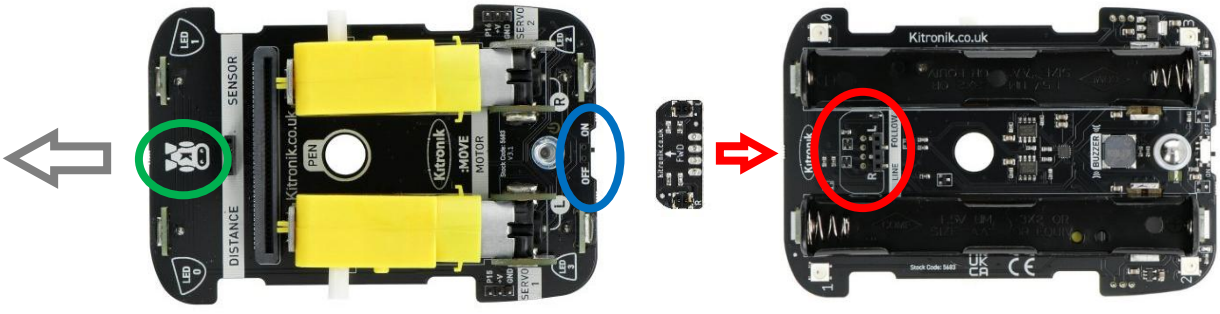
1. **Antenne** voor Bluetooth.
2. De **processor** (waar berekeningen gebeurt).
5. De **pinnen** om te verbinden met de robot.
6. **USB – aansluiting** om te verbinden met de laptop.
13. **Controle LED** brandt als er stroom is.
15. **Reset** knop om programma opnieuw te starten.

2. De Micro – USB kabel

De code downloaden kan via **Bluetooth** of via de **Micro – USB** kabel.
In deze handleiding werken we met het kabeltje.
Dit omdat elke klas andere chrome books of laptops heeft.
Op de website van Micro:bit lees je hoe je via Bluetooth kan werken.

3. De MOVE robot en lijnsensor

Ook hier bespreken we de belangrijkste onderdelen.
Achteraan zit een **ON/OFF** knop.
Deze moet aangezet worden.
De **voorkant** van de Motor herken je aan het popje.
De **lijnsensor** zit onderaan aan de voorkant.



Meest gemaakte fouten door kinderen:
1. Baterijen of Micro:bit worden omgekeerd in de houders gezet.
2. De ON/OFF knop wordt vergeten aan te zetten.
3. Ze kunnen niet goed inschatten wat de voorkant en achterkant is.

Stap 1: vraaggestuurd gesprek

25 min

Er is een inleidend gesprek om de **voorkennis** en **vragen** op te roepen.
Je kan een brainstorm aan het bord opbouwen. Bedenk ook wat je zelf wel / niet weet.
Hieronder vind je een mogelijke brainstorm. Leerlingen en jijzelf zullen op vragen uitkomen waar je niet direct het antwoord op weet. Dat is net de bedoeling en meerwaarde: dit toont aan dat er nood is aan een verkenning van het programmeren.
Vertel dat de vragen mogelijks tijdens de les beantwoord zullen worden.
Indien dit niet het geval is kunnen enkele leerlingen of jijzelf zelf het antwoord op zoeken:
- Vragen aan ouders, collega's of ICT leerkracht
- Opzoeken met de computer

1. Wat weten we over programmeren?

Wie heeft al ervaring met programmeren?
Welk programma (software) gebruikte je?
Welk merk robot (hardware) gebruikte je?
Waar leerde je programmeren? Wat heb je geprogrammeerd?
Wat was het probleem en wat was jouw oplossing?

2. Waar vinden we thuis programma's en codes?

Welke voorwerpen zijn geprogrammeerd, werken enkel bij bepaalde voorwaardes?
Welke foto's herkennen we en hoe zouden deze geprogrammeerd zijn?

3. Waarom zou er in een pretpark geprogrammeerd worden?

Stel je eens voor wat er allemaal in een pretpark is:
winkeltjes, spelletjeshal, attracties, kassa, wegwijzers, spektakel, ...
Waar gaan programmeurs aan de slag?
Tip: je kan het over **manuele** of **digitale** bediening hebben.

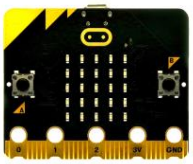
4. Welke sensoren kennen we?

Door de vorige vragen komen we zeker uit op sensoren.
Wat denken de leerlingen hier al over te weten?
Zoek maar eens op internet 'welke sensoren bestaan er'.
Er bestaan druksensoren, temperatuursensoren, infraroodsensoren (lijnsensor robot) of echolocatie (om afstand te bepalen). Of wat dacht je van trillen, bewegen, geluid, ...

Stap 2: verbinding maken

5 min

Gebruikte dia's: dia 5 tot en met dia 14 Werkboekje: pagina 1, 2 en 3



Deel per duo de robots en Micro:bits en USB kabeltjes uit.
Laat even reageren op wat ze herkennen of zien.
Wijs erop **voorzichtig** met elektronica om te gaan.
De chips zijn bewust niet in een omhulsel geplaatst om alle componenten te zien.
Er zijn twee motoren op elke robot.

Stap 1: de software
Start de website https://makecode.microbit.org/ en volg de stappen op de dia's.
De **taal** kan je instellen op Nederlands met het tandwiel rechtsboven.
Gebruik de browsers Google Chrome of Microsoft Edge. Mozilla werkt niet.

Stap 2: de hardware
Vertel de kinderen dat de Micro:bit altijd ofwel in de pc, ofwel in de robot zit. (dia 22)
De Micro:bit mag niet aangesloten worden aan het kabeltje als hij nog in de robot zit.

Steek het micro – USB kabeltje in de Micro:bit en wacht **3 tot 5 seconden**.
Daarna kan je verbinding maken. Lukt het niet? Zie stap 3 hieronder.

Stap 3: functies programmeren

30 min

Gebruikte dia's: dia 15 tot en met 19 Werkboekje: pagina 4 en 5

Vertel ook dat de twee **motoren** mogelijks een beetje **afwijken** van elkaar.
Mensen zijn immers ook niet hetzelfde, er zijn steeds kleine verschillen.
Daarom zal de robot mogelijks niet 100 % recht rijden.
Dit kan achteraf opgelost worden met de Fun Forge **optimalisaties**. (werkboekje p. 10)

Je kan ervoor kiezen om **klassikaal** te werken (aangeraden bij een eerste keer) of de leerlingen **zelfstandig** met het boekje te laten werken.
Het is best om zelf als leraar even alle stappen vooraf door te nemen.
In het programmeerboekje en op de powerpoint worden alle stappen uitgelegd.

Wat als het downloaden niet lukt?
In een uitzonderlijk geval is er een probleem met de verbinding.
Duw dan op de 3 puntjes en klik op 'verbreek de verbinding'.
Daarna kan je proberen om opnieuw te verbinden.
Lukt het nog steeds niet, probeer dan een andere kabel of Micro:bit.

Welke fouten maken de kinderen?
- Ze wisselen de woorden rechts of links.
- Het getal voor de snelheid of duurtijd klopt niet.
- Ze vergeten de code te downloaden na een wijziging.
- Soms valt de robot en zetten ze de batterijen er fout terug in.

Stap 4: de code voor de testrit

10 min

Gebruikte dia's: 19 tot en met 23 Werkboekje: pagina 6, 7 en 8

Tip: de leerlingen kunnen rechtsonder op de min duwen om uit te zoomen.

We oefenen klassikaal een code voor een rit.
Gebruik geknutselde speelgoedboompjes of andere objecten op de plaats van de bomen.
Deze worden gebruikt om te navigeren en **ruimtelijk inzicht** te krijgen.

1. De code schrijven

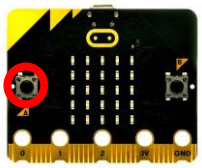
Plaats de functies (vooruit en draaien) zoals voorzien in de powerpoint in de code..

2. De code downloaden

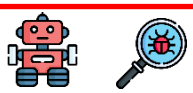
De code wordt gedownload op de Micro:bit en deze wordt in de robot gestopt.

3. Het parcours klaarzetten en testen

Laat eerst het parcours nabouwen. (boompjes, start – en stopkaart)
De robot wordt op de startkaart gezet en met het knopje A geactiveerd.
De kinderen vinden dit knopje vaak niet, toon klassikaal waar dit zit.



Stap 5: oei, het werkt niet!



10 min

Gebruikte dia's: 24, 25 en 26 Werkboekje: pagina 5

Naar alle waarschijnlijkheid, vooral bij groepen die zelden programmeren, zal er een probleem zijn. Hier komen we aan het lesdoel: optimaliseren. Vertel dat dit **debuggen** heet. Vraag voor de aandacht van alle leerlingen, ook bij wie alles werkt.

Vertel het verhaaltje:
De term "debuggen" komt oorspronkelijk van een incident in 1947. Tijdens het werken aan de Harvard Mark II computer ontdekte de Amerikaanse programmeur Grace Hopper dat de computer niet goed functioneerde. Bij nader onderzoek bleek een echte insect, een mot, vast te zitten in een van de schakelingen van de machine. Het insect werd uit de computer verwijderd en op een stuk papier geplakt met de aantekening: "First actual case of bug being found." Sindsdien werd de term "bug" gebruikt om technische problemen of fouten in een computerprogramma aan te duiden, en "debuggen" werd de term voor het proces van het oplossen van die fouten.

Vertel de kinderen dat ze nu op zoek moeten naar hun 'bug':
° Is de Micro:bit goed (niet omgekeerd) aangesloten, zitten de batterijen er goed in?
° Staat de robot aan en brandt er een groen lampje?
° Klopt het aantal seconden in de vakjes?
° Staan de juiste woorden in de groene hokjes?
Controleer stap per stap. Dit heet in het computationele denken **decompositie**.

Stap 5: oei, het werkt niet goed!

10 min

Gebruikte dia's: 24, 25 en 26 Werkboekje: pagina 9 en / of pagina 10

Een ander scenario is dit waarbij de robot wel rijdt maar de draai niet goed neemt.
Wijs de kinderen erop dat elke motor uniek is en mogelijks bij productie een kleine **afwijking** heeft. Net zoals alle mensen wat verschillen van elkaar.
Daardoor kan het zijn dat de motor net iets sneller of trager draait.
Dit kan ook zijn omdat de batterijen niet meer de volle **capaciteit** bezitten.

De draaisnelheid kan aangepast worden.
Draait de robot minder dan 90°? Dan neem je een groter getal.
Draait de robot meer dan 90°? Dan neem je een minder groot getal.

We stellen voor om **kleine aanpassingen** te doen van 50 of 100 ms.
Daarna kan de nieuwe code gedownload worden (niet vergeten!) en getest worden.

Als een groepje klaar is en de robot het parcours kan laten rijden,
kan je overgaan naar de volgende lesfase. Deze kunnen ze nu normaal zelfstandig.

Als de robot niet goed recht rijdt (dit zal je vooral opvallen bij langere ritten) kan je ook hier optimaliseren. We raden dit omwille van de extra stappen aan nu pas te vertellen aan de kinderen. Deze stap kunnen ze vinden in hun boekje op pagina 10.

Stap 6: de verschillende plattegronden

30 min

Voor deze stap werken de leerlingen zelfstandig in duo's.
Je kan ze elk een verschillende plattegrond geven.
Wie slaagt erin om de robot het parcours foutloos te laten maken.
Mogelijks zal er met links of rechts gemist worden vanuit het perspectief van de robot.

Begeleiding door leerkracht:
- Waarom is het goed gelukt?
- Wat moet er anders?
- Waar is een fout in de code?
- Heb je moeten debuggen in de code?
- Wat heb je moeten optimaliseren?
- Wat heb je geleerd?
- Hoeveel graden draait de robot?
- Beweegt de robot in een rechte lijn of is er een afwijking?
- Hoe kan ik dit corrigeren?

demonstratieles

Zorg ook voor veel **bevestiging**. Programmeren is (zeker de eerste keer) niet makkelijk.

Stap 7 en de **extra's** kan je weglaten als je enkel box 5 gebruikt.

Stap 7: de lijnsensor gebruiken

30 min

Gebruikte dia's: 29 tot 38 Werkboekje: pagina 14 tot en met 17

Met de lijnsensor kunnen we de robot eenzelfde parcours steeds opnieuw laten rijden.
Zo kunnen we in een latere stap karretjes, tunnels, hellingen of bruggen maken.
Doordat we met een lijn werken hebben we één keer wat meer werk aan de code.
Daarna is het makkelijker want dan wijzigen we het parcours zelf.
Dit is dus een vorm van **abstraheren**.
In plaats van telkens een nieuwe code te schrijven bedenken we een andere oplossing.

1. De lijnsensor installeren

Kijk of de lijnsensor (correct) bevestigd werd onder aan de robot.
Is dit niet het geval, dan kan je dat zelf doen.
Opgepast: de pootjes kunnen plooien dus doe dat voorzichtig.
Vertel dat de lijnsensor 2 infrarood sensoren heeft.
Deze kunnen 'zien' of ze op of naast de zwarte lijn rijden.
De robot heeft er 2 nodig om te zien of hij links of rechts van de lijn rijdt.

2. De code schrijven

Dit kan klassikaal (met de powerpoint) of in duo's.
De meest gemaakte fout is het wisselen van de > in een <.
Een andere fout is het kiezen van het rode blokje 'verander in' in plaats van 'stel in'.
We verwijzen naar het programmeerboekje voor deze stap.

3. De code downloaden en testen

De robot zal in kleine **schokken** vooruit gaan.
Dit komt omdat hij telkens links of rechts van de lijn komt en dus bijstuurt.

Je kan zwarte isolatietape gebruiken want deze is makkelijk te verwijderen.
Een dikke zwarte streep (stift of gekleurd papier) op een lichte ondergrond werkt ook.

4. Onderzoek voeren met de lijnen

Stel een **hypothese** hoe je best een bocht of draai maakt.
Waarom werkt een scherpe, rechte, stompe bocht?
Waarom werkt een gebogen draai wel of niet?
Hoe kan je de plakband kleven zodat de robot niet afwijkt van het parcours?
Wat gebeurt er als je een splitsing of kruispunt kleeft?
Hoe zou dat komen?
Hoe snel kan de robot dit parcours afleggen zonder uit de bocht te vliegen?

Extra: lichten, lussen en snelheid

30 min

Deze stap kan overgelaten worden.
Wil je meer bijleren over werken met lussen en aanpassen aan een nieuwe snelheid?
Of wil je extra wiskunde integreren?
Misschien wil je een rit in het donker maken en de klaslichten uitzetten?
Lees dan snel verder want de robot heeft extra opties!

1. Snelheid berekenen [pagina 11 werkboekje]

Deze oefening sluit mooi aan bij **snelheid berekenen** en **recht evenredig**.
Laat de robot 1 seconde vooruit rijden en meet de afstand.
Dit kan je dan omrekenen van cm / sec naar m / min of naar km / uur.

Ook kan je de snelheid van de robot laten aanpassen.
Let op, de functies draaien naar links en rechts zullen dan ook aangepast moeten worden.
Een **snelle robot** heeft **minder tijd** nodig om te draaien (omgekeerd evenredig).

Wil je met een snellere robot het grondplan laten rijden?
Dan zal je waarschijnlijk ook **of** de tijd moeten aanpassen **of** de bomen verder zetten.

2. Lichten aan en uit zetten [pagina 12 werkboekje]

Je kan 2 of 4 LED onderaan laten schijnen.
Als je ze uit wil kan dit gemakkelijk door 'toon kleur zwart' te kiezen.
Zwart is immers geen kleur omdat het alle kleur opslorpt.

3. Werken met lussen [pagina 13 werkboekje]

Als de leerlingen het grondplan bekijken kunnen ze patronen zoeken.
Wanneer er gelijke functies zijn (vb. rechts, links, rechts, links) heb je een **patroon**.
Binnen de programmeercode spreken we dan van een **lus**.

Zelf snuisteren

Je kan de leerlingen zelf ook enkele uitdagingen geven waar ze mee aan de slag kunnen.
Zo kan de Micro:bit bijvoorbeeld zorgen dat er een pictogram op het karretje komt.
Je kan ook een toeter of sirene aanzetten.
Of misschien kan je coderen dat de robot vertrekt als je in je handen klapt?

Maar dat brengt ons natuurlijk te ver.
Op de site van Kitronic en Micro:bit vind je heel wat extra ideetjes.
Zelf experimenteren kan natuurlijk ook.
Wie kan ervoor zorgen dat lichten enkel aan gaan in een donkere tunnel / klas?

Doelstellingen programmeren

Eindtermen niet – levende natuur:

1.2 De leerlingen kunnen, onder begeleiding, minstens één natuurlijk verschijnsel dat ze waarnemen via een eenvoudig onderzoek toetsen aan een hypothese.

Eindtermen techniek:

2.3 De leerlingen kunnen onderzoeken hoe het komt dat een zelf gebruikt technisch systeem niet of slecht functioneert.

2.5 De leerlingen kunnen illustreren dat technische systemen evolueren en verbeteren;

2.7 De leerlingen kunnen in concrete ervaringen stappen van het technisch proces herkennen (het probleem stellen, oplossingen ontwikkelen, maken, in gebruik nemen, evalueren).

2.8 De leerlingen kunnen technische systemen, het technisch proces, hulpmiddelen en keuzen herkennen binnen verschillende toepassingsgebieden van techniek.

2.9 De leerlingen kunnen een probleem, ontstaan vanuit een behoefte, technisch oplossen door verschillende stappen van het technisch proces te doorlopen.

2.10 De leerlingen kunnen bepalen aan welke vereisten het technisch systeem dat ze willen gebruiken of realiseren, moet voldoen.

2.15 De leerlingen kunnen technische systemen in verschillende toepassingsgebieden van techniek gebruiken en/of realiseren.

Eindtermen wiskunde:

3.7 De leerlingen zijn in staat:

- zich ruimtelijk te oriënteren op basis van plattegronden, kaarten, foto's en gegevens over afstand en richting.
- zich in de ruimte mentaal te verplaatsen en te verwoorden wat ze dan zien.

Strategieën en probleemoplossende vaardigheden:

4.1 De leerlingen kunnen met concrete voorbeelden aantonen dat er voor hetzelfde wiskundig probleem met betrekking tot getallen, meten, meetkunde en ruimtelijke oriëntatie, soms meerdere oplossingswegen zijn en soms zelfs meerdere oplossingen mogelijk zijn afhankelijk van de wijze waarop het probleem wordt opgevat.

4.2 De leerlingen zijn in staat om de geleerde begrippen, inzichten, procedures, met betrekking tot getallen, meten en meetkunde, zoals in de respectievelijke eindtermen vermeld, efficiënt te hanteren in betekenisvolle toepassingssituaties, zowel binnen als buiten de klas.

4.3 De leerlingen kunnen met concrete voorbeelden uit hun leefwereld aangeven welke de rol en het praktisch nut van wiskunde is in de maatschappij.

Eindtermen Nederlands:

3.1 De leerlingen kunnen (verwerkingsniveau = beschrijven) de informatie achterhalen in: voor hen bestemde instructies voor handelingen van gevarieerde aard.

3.2 De leerlingen kunnen (verwerkingsniveau = beschrijven) de informatie achterhalen in: de gegevens in schema's en tabellen ten dienste van het publiek.