Het programma bestaat uit een for loop die 25 keer doorlopen wordt omdat er in totaal 25 kruispunten zijn. Vanaf de wagen dus een kruispunt heeft overgestoken, gaan we over naar de volgende iteratie met verschil dat het programma is ingesteld op basis van het volgende kruispunt. We doorlopen dus deze for loop en zetten vanaf het begin ook de ledlamp van de kleursensor uit. Daarnaast gebruiken we een global variabele waarin alles staat om de wagen te kunnen besturen en om informatie over de route te kunnen zien. Hierin staat bijvoorbeeld wat het laatste kruispunt was dat we gepasseerd hebben.

**Manueel**:

Aan het begin van het programma wordt er gekeken of de manuele boolean op TRUE staat. Als dit het geval is, kunnen we het wagentje zelf besturen doordat in de case structure naar het bijhorende paneel wordt gegaan.

Merk nog op dat we het wagentje kunnen laten rijden wanneer er een obstakel gedetecteerd wordt. (De manuele boolean is dus belangrijker dan het stoppen voor een obstakel) Zodra we manueel aanduiden, gaan we over op manueel en wordt met al de rest geen rekening meer gehouden. Dit is goed, want dan kunnen we in elke situatie ervan gebruik maken.

De boolean voor de manuele bediening staat in de global variabele. Vanaf deze boolean wordt indrukt, staat de wagen stil en kan deze worden bestuurd aan de hand van vier andere booleans. Deze zijn ‘rechtdoor’, ‘achteruit’, ‘rechts’ en ‘links’. Hierbij mag wel maar 1 boolean tegelijk aan staan, want anders zal de motor twee verschillende dingen moeten doen wat natuurlijk niet kan. (bv rechtdoor rijden en links draaien) Als we rechtdoor rijden en we willen links afslaan, moeten we dus eerst de boolean ‘rechtdoor’ op FALSE zetten en daarna de boolen ‘links draaien’ op TRUE, anders is er een conflict. Hierbij moet uiteraard de manuele boolean voortdurend op TRUE staan. Wanneer enkel de manuele boolean aanstaat, staat de wagen stil, maar vanaf één van de vier besturingsbooleans op TRUE wordt gezet, is dit uiteraard niet meer het geval.

**Snelheid en richting**

Om de snelheid van de wagen in te stellen, moet een van de getallen 0,2 of 5 worden ingegeven in de output voor beide wielen. Hierbij staat 2 (of lager, maar groter dan 0) voor de traagste stand, terwijl het wagentje bij 5 ( of hoger dan 2) zich met maximale snelheid voortbeweegt. Als het wagentje moet stoppen aan de stopstreep of omdat er een obstakel op de weg gedetecteerd wordt, moet er voor beide wielen de waarde 0 worden ingegeven zodat ze onmiddellijk stoppen met rijden. Een van deze getallen wordt vervolgens omgezet naar een voltage waardoor de motor precies weet hoe snel hij moet werken.

Wanneer voor beide wielen hetzelfde getal wordt ingegeven, rijdt het wagentje mooi rechtdoor. Maar wanneer echter twee verschillende getallen worden doorgegeven, zal het ene wiel een grotere snelheid hebben dan het andere waardoor het wagentje zal draaien. Als het rechter (resp. linker) wiel sneller rijdt dan het linker (resp. rechter) wiel zal het wagentje naar links (resp. rechts) draaien. Op die manier kan een bocht genomen worden (na een kruispunt) of kan het wagentje zich corrigeren om perfect op de lijn te rijden.

**Obstakel**:

Vervolgens wordt er gekeken of er een obstakel op de weg wordt gedetecteerd. Hierbij geeft de digitale input een TRUE wanneer geen obstakel wordt gedetecteerd. Wanneer dit echter wel het geval is, geeft de digitale input een FALSE en staat de wagen stil. Zowel de boolean manueel als de input van de afstandssensor staan in een while loop, waardoor ze constant updaten.

**Digitale code lijn** => vormen van een lijst bestaande uit énen en nullen.

Wanneer de boolean manueel op FALSE staat en wanneer er geen obstakel op de weg gedetecteerd wordt, gaan we over op het controleren van de positie van de wagen ten opzichte van de lijn.

Dit programma creëert een array van énen en nullen aan de hand van de waarden die de lijnsensor teruggeeft. Deze waarden kunnen we inlezen en in een array zetten aan de hand van de ‘analoge inputs’. Daarna gaan we deze array (met 8 elementen) 1 voor 1 controleren en maken we hierbij een nieuwe array met waarden 0 (lijn NIET gedetecteerd) en 1 (lijn WEL gedetecteerd). Dit doen we door te kijken of de waarden van de input array groter zijn dan 0,3 (=>1 en kleiner dan 0,3 =>0), want dan wordt de lijn namelijk gedetecteerd. Wanneer de nieuwe array gemaakt is, controleren we eerst en vooral of alle waarden gelijk zijn aan 1, dit wil namelijk zeggen dat we op de stopstreep staan (omdat alle sensoren de lijn detecteren). Als dit het geval is, gaan we over naar de programmatie voor de kleursensor en indien dit niet het geval is, gaan we over naar de code voor de lijndetectie. De reden dat we al aan het begin controleren of we op de stopstreep staan, is dat we aan de hand van de berekeningen voor het bijsturen van de lijn (zie ‘lijnvolgen’), de waarde nul bekomen. Dit wil zeggen dat we rechtdoor moeten rijden. Aan de stopstreep bekomen we echter ook de waarde 0 (die dus al in gebruik is om rechtdoor te rijden).

De lijnsensor geeft als input een array van 8 analoge spanningen tussen 0V en 5V. Deze moet worden omgezet naar een array met 8 waarden die ofwel 0 ofwel 1 zijn. 0 wil zeggen dat de sensor de lijn niet detecteert en komt overeen met een inputwaarde tussen 0V en 3.3V. Daarnaast kan de sensor bij 1 wel de lijn zien en dit komt overeen met een inputwaarde tussen 3.3V en 5V. Eenmaal de nieuwe array is bepaald, moet deze worden omgezet naar besturing van de wagen.

De analoge inputs worden omgevormd tot een array met 8 waarden die ofwel 1 of 0 zijn.

De inputarray wordt dus omgezet naar nullen en énen.

Daarna wordt gecontroleerd of we op de stopstreep staan.

**Lijnvolgen**

Dit programma zet de array van nullen en énen om naar een getal dat zegt hoe sterk er moet bijgestuurd worden. Hoe meer naar buiten de énen zitten, hoe meer er moet bijgestuurd worden. Hoe meer naar binnen, hoe minder. Afhankelijk van de grootte van de waarde zal er meer of minder worden bijgestuurd. Als de waarde groot is, zal er sterk worden bijgestuurd en bij een kleine waarde zwak. Merk op dat er enkel van de snelheden 0,2 en 5 wordt gebruik gemaakt omdat we maar 2 snelheden kunnen instellen. Dit is vooral goed om een rechte lijn te volgen (en minder voor scherpe bochten) aangezien het parcours toch geen scherpe bochten bevat, maar enkele rechte lijnen. Merk ook nog op dat er vier waarden moeten ingegeven worden om de snelheden van de wielen in te stellen. Dit komt omdat de snelheden van beide wielen wordt bepaald aan de hand van het verschil tussen de twee waarden horende bij de 2 wielen. (de 2 waarden van A horen bij linker wiel en de 2 B-waarden horen bij rechter wiel)

Nu wordt uitleg gegeven voor het bijsturen op de lijn.

De waarden in de gevormde digitale array worden nu één voor één gecontroleerd op een 1 of een 0, maar er wordt ook telkens bijgehouden op welke index van de array we zitten. Er gebeurt niks tot er bij een specifieke index een 1 wordt gezien. Dan wordt namelijk een waarde tussen -7 en 7 overgebracht naar de volgende lus. Afhankelijk van hoever naar buiten de 1 staat (in de array), zal de doorgegeven waarde groter of kleiner zijn. Zo zal de waarde bijvoorbeeld 7 zijn als de eerste index al een 1 bevat en -7 wanneer bij elke index van de array de waarde 0 is behalve bij de laatste. De mogelijke waarden zijn ±7, ±5, ±3, ±1. Nadat er voor de eerste keer een 1 is aangetroffen, zal telkens wanneer opnieuw een 1 wordt tegengekomen, één eenheid worden afgetrokken van de waarde die tot dan toe werd bekomen. Dit wijst erop dat het wagentje minder aan de buitenkant (ten opzichte van de lijn) zit bij een positief getal, of juist meer naar buiten zit bij een negatief getal. Op die manier kan een onderscheid gemaakt worden van hoe ver (grootte van het getal) het wagentje relatief van de lijn is verwijderd en langs welke kant (positief of negatief getal). Wanneer het wagentje perfect op de lijn rijdt, zullen de 4de en 5de index van de array een 1 bevatten en zal de bepaalde waarde nul zijn, wat wil zeggen dat het wagentje niets moet doen. Aan de hand van de twee ingevoerde waarden die zorgen voor de snelheid van de twee wielen kan het wagentje dan naar een bepaalde kant draaien met een bepaalde scherpte. Hoe groter het getal, hoe scherper de draai naar binnen zal zijn.

Daarnaast kan het ook zijn dat de array alleen enen bevat. Dit wil zeggen dat het wagentje op de stopstreep staat (want alle acht sensoren detecteren de lijn). Hiervoor is een apart geval nodig die eerst controleert of alle elementen in de array 1 zijn. Als dit het geval is, zal het wagentje onmiddellijk stoppen. Zo niet dan zal de wagen worden bijgestuurd via de berekende waarde. Dit is nodig omdat bij een array van alleen maar 1’en de berekende waarde gelijk is aan nul (7-7x1) net zoals wanneer het wagentje perfect op de lijn rijdt. Bijgevolg zal het wagentje dan niks doen.

**Kruispunt**

Vanaf de stopstreep gedetecteerd wordt, wil dit zeggen dat we aan een kruispunt staan. De wagen moet dan stoppen en het licht inlezen. De wagen blijft stilstaan zolang het licht op rood staat, er een obstakel wordt gedetecteerd of zolang de manuele boolean op TRUE staat. Vanaf de wagen mag vertrekken, wordt er aan de hand van de voorgeprogrammeerde array gekeken naar welke kant de wagen moet rijden.

De array met de voorgeprogrammeerde route bevat de waarden 1 (links) ,2 (rechtdoor) of 3 (rechts) waarbij we aangeven of de wagen aan het eerstvolgende kruispunt links/rechts moet afslaan of gewoon rechtdoor moet rijden.

Afhankelijk van de route wordt de bijhorende subvi (links.vi/ rechts.vi/ rechtdoor.vi) uitgevoerd. In deze subvi’s staan de specifieke instellingen voor de snelheden van beide wielen. Hierbij is ingesteld hoelang iedere actie moet duren aan de hand van een timer die de verstreken tijd bijhoudt.

Gedurende het oversteken van het kruispunt wordt ook soms gecontroleerd of de manuele boolean aanstaat en of er een obstakel op de weg gedetecteerd wordt.

Wanneer alle specifieke bewegingen zijn uitgevoerd en alle controles zijn uitgevoerd, rijden we rechtdoor tot het wagentje weer de lijn detecteert. Daarna wordt de stopconditie van de while loop op TRUE gezet waardoor deze sluit. Nu wordt overgegaan op de volgende index van de for loop, zodat het programma ingesteld staat op het volgende kruispunt.