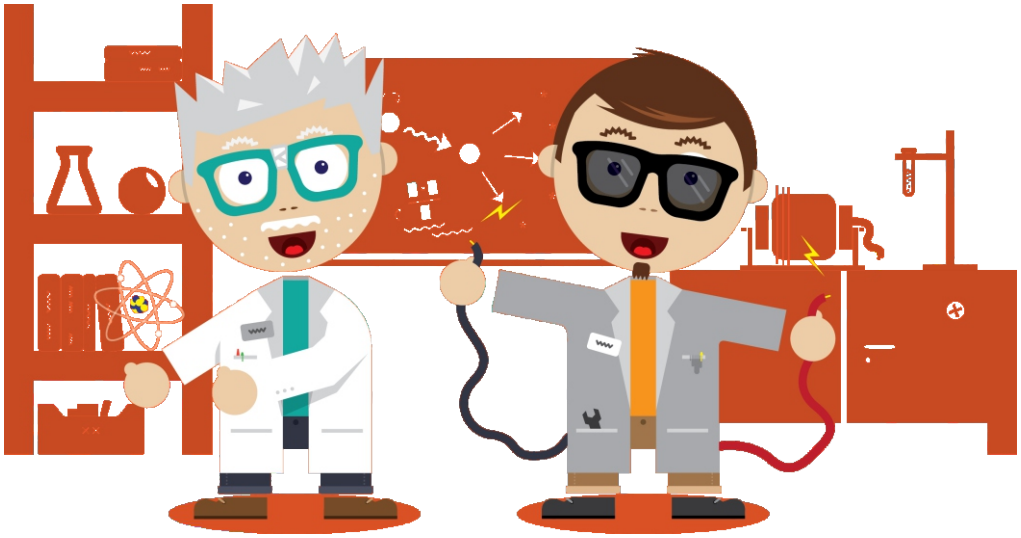


ROBOTYKA

SKRYPT DLA UCZNIA

Z FIZYKĄ I TECHNIKĄ ZA PAN BRAT!

Interdyscyplinarny program nauczania fizyki i techniki

GIMNAZJUM

Skrypt dla ucznia do Techniki stanowi obudowę dydaktyczną do Programu Nauczania Fizyki i Techniki „Z FIZYKA I TECHNIKĄ ZA PAN BRAT!" dla części Technika.

Autor: Ewa Bednarek

Projekt pt. „Z FIZYKĄ I TECHNIKĄ ZA PAN BRAT!"

Beneficjent: Gmina Strzelno

Numer projektu: POKL.03.03.04-00-238/12

Okres realizacji projektu: 01.02.2013 – 30.06.2015

Priorytet III. Wysoka jakość systemu oświaty
Działanie 3.3. Poprawa jakości kształcenia
Poddziałanie 3.3.4. Modernizacja treści i metod kształcenia – projekty konkursowe

Program Operacyjny Kapitał Ludzki

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

SPIS TREŚCI

Temat 1: KRÓTKI WSTĘP DO ROBOTYKI

1. Co to jest robot?

Robot to mechaniczne urządzenie wykonujące automatycznie określone zadania.

Skąd nazwa?

Słowo ROBOT pochodzi od czeskiego słowa robota, oznaczającego ciężką pracę, wysiłek.

Czy roboty obowiązują jakieś zasady? (3 prawa robotyki)

1. Robot nie może skrzywdzić człowieka, ani przez zaniechanie działania dopuścić, aby człowiek doznał krzywdy.
2. Robot musi być posłuszny rozkazom człowieka, chyba że stoją one w sprzeczności z Pierwszym Prawem.
3. Robot musi chronić sam siebie, jeśli tylko nie stoi to w sprzeczności z Pierwszym lub Drugim Prawem.

2. Jak robot widzi świat – opis czujników

A. Czujnik dotyku

Budowa

Wewnątrz obudowy czujnika dotyku znajduje się płytka elektroniczna z zamontowanym przełącznikiem, rezystorem oraz gniazdem połączeniowym. Zamontowany rezystor zabezpiecza przez bezpośrednim zwarciem linii zasilania i masy.

Zasada działania

Naciśnięcie przycisku powoduje zamknięcie obwodu elektrycznego. Czujnik dotyku możemy porównać do włącznika światła. Naciskając włącznik zapalamy światło, naciskając raz jeszcze gasimy światło.

Zastosowanie czujników dotyku w robotyce

- Wykrywanie przedmiotów.
- Sterowanie procesem w zależności od załączenia czujnika dotyku.
- Dojazd robota do przeszkody do czasu załączenia czujnika.
- Wykrycie kolizji.

Zadanie 1. Sprawdzenie działania czujnika dotyku

Podłącz czujnik do jednego z portów (1-4) kostki EV3, następnie wybierz opcję Port View i odnajdujemy port do jakiego jest podłączony czujnik.
Co powoduje naciśnięcie przycisku?

B. Czujnik podczerwieni i pilot IR

Połączenie czujnika podczerwieni (odbiornika) i pilota IR (podczerwieni) pozwala zbudować zdalnie sterowanego robota.

Budowa

Czujnik odległości składa się z emitera promieniowania w zakresie ściśle określonej długości fali promieniowania podczerwieni (IR) oraz z dwóch odbiorników zdolnych rozróżniać poszczególne długości fal.

Pilot IR składa się tylko z emitera fal promieniowa podczerwonego o zmiennej długości fali w zależności jaki przycisk lub kombinacja przycisków zostanie wduszona (czujnik rozróżnia 10 kombinacji oraz brak wciśniętego przycisku). Dodatkowo pilot posiada cztery różne kanały, dzięki czemu ilość rozróżnialnych kombinacji rośnie do 40.

Czujnik i pilot posiadają następujące funkcje:

1. Pomiar odległości pomiędzy przeszkodą, a czujnikiem

Działanie czujnika odległości porównać można do sposobu wykorzystywanego przez nietoperze czy delfiny - echolokację. Czujnik wysyła impulsy podczerwone i mierzy czas po jakim wracają odbite od danego obiektu. Zintegrowana elektronika automatycznie dokonuje pomiaru i oblicza odległość. Czujnik mierzy odległość od przeszkody. Odległość mierzona jest poprawnie jeśli powierzchnia odbicia fali jest dość duża, gładka i jasnego koloru (gorzej w przypadku mniejszych lub owalnych przedmiotów o ciemnych kolorach – kolor czarny w większym stopniu pochłania promieniowanie podczerwone).

Zastosowanie czujników odległości w robotyce

- Wykrywanie obiektów.
- Pomiar długości/wysokości obiektów, np. drewna, poziomu cieczy.
- Systemy antykolizyjne.
- Detektor ruchu.
- Zliczanie obiektów na przenośniku taśmowym.

2. **Pomiar odległości pomiędzy pilotem IR, a czujnikiem**
   Czujnik pozwala określić odległość

3. **Wykrycie wciśnięcia przycisku na pilocie IR**
   Czujnik i pilot mogą nadawać/odbierać na różnych falach, dzięki czemu możliwe jest rozpoznanie, który przycisk został wciśnięty.

4. **Pomiar położenia (kąta) pomiędzy pilotem IR, a czujnikiem.**

   We wnętrzu czujnika znajdują się dwa detektory promieniowania umieszczone w pewnej odległości od siebie. Pilot emituje promieniowanie podczerwone które po dotarciu do poszczególnych detektorów ma inne cechy (inną falę). Na tej podstawie układ elektroniczny określa położenie pilota.

Zadanie 2. Sprawdzenie działania czujnika podczerwieni

Podłącz czujnik do jednego z portów (1-4) kostki EV3. Odczyt czujnika sprawdzamy wykorzystując w kostce opcję Port View i odnajdujemy port, do którego jest podłączony czujnik. Do wyboru mamy trzy opcje pomiaru:

- IR-PROX – pomiar odległości (pomiar określany w jednostkach PCT od 0 do 100 – nie jest to zależność liniowa). Naszym zadaniem jest ustalenie dla jakiej odległości od przeszkody (w cm) wyświetlana wartość osiągnie 100.
- IR-SEEK – pomiar położenia pilota względem czujnika (pomiar określany w jednostkach PCT od -25 do 25).
- IR-REMOTE – określa jaki przycisk lub kombinacja przycisków jest wciśnięta na pilocie.

Naszym zadaniem jest przyporządkowanie numeru ID wyświetlanego na kostce do numerów wciśniętego przycisk, lub kombinacji przycisków. Przycisk nr 9 powoduje, że pilot działa ciągle.

0 –	4 –	8 –
1 –	5 –	9 –
2 –	6 –	10 –
3 –	7 –	11 –

3. Czujnik koloru

Budowa

Wewnątrz czujnika znajdziemy płytkę elektroniczną z 3 diodami: diodę odbiorczą, podczerwoną diodę nadawczą oraz diodę RGB.

Zasada działania

W trybie czujnika światła działają tylko dioda nadawcza emitująca światło i dioda odbiorcza mierząca odbite od danej powierzchni promienie poczerwieni. Natomiast w przypadku czujnika koloru dioda RGB emituje po kolei kolory podstawowe i na podstawie odczytu z diody odbiorczej ustalany jest kolor badanej powierzchni.

Zastosowanie czujników koloru w robotyce

- Zliczanie obiektów danego koloru (rodzaju) na przenośniku taśmowym.
- Badanie koloru przedmiotu - sprawdzanie jego poprawności.
- Sortowanie przedmiotów na podstawie jego koloru – np. sortowanie butelek.

Zadanie 3. Sprawdzenie działania czujnika koloru

Działanie czujnika sprawdzamy podłączając czujnik do jednego z portów (1-4) kostki NXT. Następnie wybieramy opcję Port View i odnajdujemy port, do którego jest podłączony czujnik. Zbliż czujnik do powierzchni o różnych kolorach. Czujnik powinien znajdować się w odległości około 0,5 cm od badanej powierzchni. Czujnik rozróżnia następujące kolory: czarny, biały, czerwony, zielony, niebieski, żółty, brązowy oraz brak koloru. Każdemu kolorowi odpowiada inny numer ID wyświetlany na kostce.

Przyporządkuj odpowiednie numery do odpowiednich kolorów.

Biały -	Żółty -
Czerwony -	Czarny -
Zielony -	Brązowy -
Niebieski -	Brak koloru -

4. Inne czujniki stosowane w robotyce

- Czujnik temperatury — pomiar temperatury
- Kompas – badanie kierunku względem osi magnetycznych ziemi
- Akcelerometr – badanie przyspieszenia kątowego
- Żyroskop – badanie położenia kątowego
- Kamera – rozpoznawanie twarzy
- Dynamometr – pomiar siły

5. Silniki LEGO Mindstorms EV3

Budowa

Silnik dołączony do zestawu Lego Mindstorms EV3 jest silnikiem prądu stałego. W obudowie silnika znajduje się zestaw kół zębatych tworzących przekładnie mechaniczną. Zastosowanie przekładni mechanicznej jest konieczne w celu zmniejszenia prędkości obrotowej silnika i zwiększenia momentu obrotowego. Więcej na temat przekładni mechanicznych w kolejnych lekcjach. Silnik został wyposażony w enkoder (czujnik pozwalający określić obrót silnika), dzięki czemu sterowanie silnikiem jest bardzo precyzyjne. Dokładność pozycjonowania silnika wynosi 1 stopień. Silnik jako czujnik obrotu może znaleźć zastosowanie np. przy konstrukcji sejfu, w którym cyfry będziemy wpisywać na podstawie obrotu silnika.

Zastosowanie w robotyce

- Napęd robotów mobilnych,
- Napęd linii produkcyjnych,
- Napęd maszyn kroczących.

Zadanie 4. Sprawdzenie działania silnika

Silnik podłączmy do jednego z portów (A, B, C, D) kostki NXT. Wybieramy opcję Port View, następnie odnajdujemy port, do którego podłączyliśmy silnik.. Kręcąc osią silnika sprawdzamy, kiedy silnik kręci się do przodu, a kiedy do tyłu.

6. Kostka EV3

Najważniejszym elementem naszego robota jest kostka EV3- mózg robota. Kostka EV3 wyposażona została w dwa procesory, które sterują pracą podłączonych podzespołów. Komputerek EV3 posiada 8 wejść/wyjść. Porty 1, 2, 3, 4 służą do podłączenia czujników, do portów A, B, C, D podłączamy silniki, kostka posiada również wejście USB, slot kart micro SD. Komputerek zasilamy za pomocą 6 baterii AA (1,5V lub akumulatorków 1,2V). Kostka posiada ciekłokrystaliczny ekran, na którym wyświetlane są informacje o programach, pomiary czujników, itp.

Zadanie 5. * Sprawdzenie, w jaki sposób słońce wpływa na pracę czujnika odległości?

Zadanie 6. * Kalibracja czujnika odległości

Naszym zadaniem jest ustalenie jakiej wartości PCT wyświetlanej na kostce będzie odpowiadała rzeczywista wartość odległości czujnika od przeszkody. Dokonujemy pomiarów odległości czujnika od przeszkody przy pomocy miary i sprawdzamy jaka wartość jest wyświetlana na kostce. Na podstawie uzyskanych wyników tworzymy wykres przy pomocy programu komputerowego.

Temat 2: NASZ PIERWSZY ROBOT

Pierwszą konstrukcją, jaką zbudujemy będzie robot poruszający się pomiędzy dwoma liniami. Na początek zastanówmy się, w jaki sposób nasz robot może skręcać?

1. W JAKI SPOSÓB RÓŻNE POJAZDY SKRĘCAJĄ?

Samochód używa do skręcania systemu skrętnych kół.

Czołg do skręcania wykorzystuje różnicę prędkości kół (w przypadku czołgu różna prędkość gąsienic). Jeżeli koła będą obracać się w przeciwnym kierunku z tą samą prędkością wtedy nasz robot będzie skręcał w miejscu. Jeżeli tylko jedno koło będzie się obracać nasz robot będzie skręcał po łuku. Promień skrętu zależy od różnicy prędkości pomiędzy silnikami.

Skręt w miejscu	Skręt po łuku w lewo	Skręt po łuku w lewo
Silniki obracają się z taką samą prędkością, ale w przeciwnych kierunkach.	Oba silniki kręcą się w tym samym kierunku, ale prawy silnik kręci się szybciej, niż lewy.	Prawy silnik się obraca, a lewy jest zatrzymany.

2. WIEDZĄC JUŻ, W JAKI SPOSÓB NASZ ROBOT BĘDZIE SIĘ PORUSZAŁ MOŻEMY PRZYSTĄPIĆ DO BUDOWY.

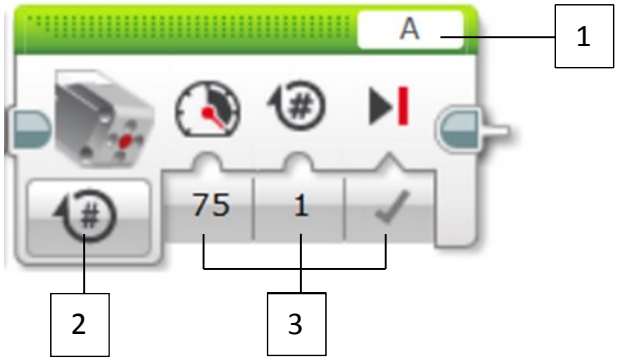
Nasza konstrukcja będzie posiadała dwa symetrycznie umieszczone silniki. Budujemy robota według instrukcji „1robot.pdf". W lewym górnym rogu instrukcji znajduje się spis elementów, które są nam potrzebna w danym kroku budowy. Natomiast poniżej znajduje się sposób, w jaki należy połączyć te elementy.

3. PIERWSZE KROKI W PROGRAMOWANIU EV3

Zanim przystąpimy do pisania programu musimy zapoznać się z oprogramowaniem. Uruchamiamy program Lego Mindstorms EV3 Home Edition. W lewym górnym rogu panelu klikamy zakładkę **File**, następnie **New Project.** W celu otwarcia już istniejącego programu klikamy zakładkę **File**, następnie **Open Project...** i otwieramy program, który nas interesuje. Na pierwszych zajęciach

zapoznamy się z działaniem bloków dotyczących pracy silników: Medium Motor, Large Motor, Move Steerling znajdujących się w zielonej zakładce (bloki działań). Bloki przeciągamy przytrzymując lewy przycisk myszy i przyczepiamy do zielonego przycisku start znajdującym się już na ekranie. Możemy dodać drugi program przeciągając jeszcze jeden przycisk start z pomarańczowej zakładki (bloki przepływu).

A. Obsługa bloku sterowania średnim silnikiem (Medium Motor)

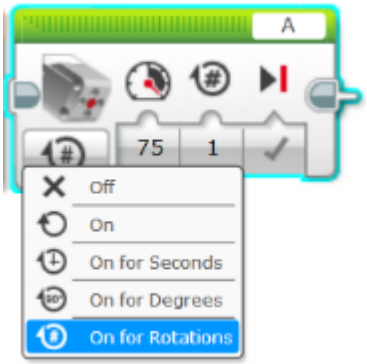


1 – Wybór portu, który będzie kontrolowany przez blok

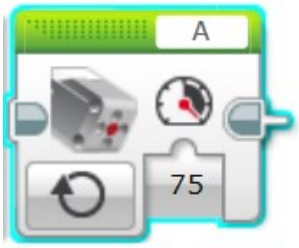
2 – Wybór trybu działania

3 – Wejścia – zmieniają się w zależności od wybranego trybu działania

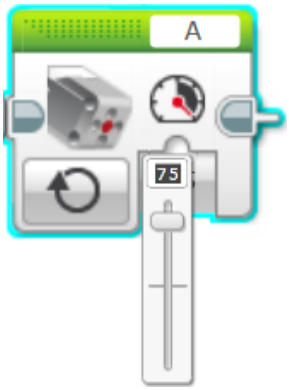
TRYBY DZIAŁANIA



ON – uruchamia silnik, następnie przechodzi do kolejnego bloku programu



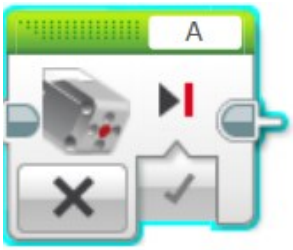
W trybie ON steruje się prędkością i kierunkiem napędu wykorzystując wejście Mocy (**Power**). Silnik będzie się kręcił, dopóki nie zostanie zatrzymany albo zmieniony przez inny, późniejszy blok w programie, lub aż do końca programu.



Wejście Mocy (Power) przyjmuje wartości od -100 do 100. Wartości dodatnie umożliwiają obrót zgodnie z kierunkiem ruchu wskazówek zegara, a ujemne przeciwny zwrot.

- kierunek ruchu silnika dla wartości od 1 do 100

- kierunek ruchu silnika dla wartości od -100 do -1



OFF – wyłącza silnik, tryb stosowany zazwyczaj do zatrzymania silnika uruchomionego wcześniej trybem ON

Wartością, którą się steruje jest Hamowanie końcowe (**Brake at End**). Przyjmuje ona wartości logiczne Prawda (**True - Brake**)/Fałsz (**False - Coast**).

Gdy jest wybrana opcja **Brake**, wówczas silnik jest natychmiast zatrzymywany i znajduje się w tej pozycji, dopóki w programie nie pojawi się bloczek w trybie ON lub program się nie skończy.

Gdy wybierze się **Coast** wyłączane jest zasilanie silnika. Będzie się on obracał, aż do wytracenia energii obrotu, bądź do innego bloku uruchomiającego silnik.

ON FOR SECONDS – obraca silnik na czas odliczenia sekund podanych na wejściu, a następnie wyłącza go.

Blok będzie oczekiwał dopóki nie upłynie wyznaczony czas, zanim przejdzie do następnego bloku.

Tryb ten umożliwia sterowanie prędkością i kierunkiem obrotu silnika dzięki parametrom wejścia.

Wartości czasu podawane są w sekundach i przyjmuje wartości ≥ 0, łącznie z wartościami ułamków dziesiętnych.

Jeśli opcja hamowania będzie ustawiona na True silnik zatrzyma się, natychmiast po upływie czasu podanego na wejściu.

ON FOR DEGREES – obraca silnik o liczbę stopni obrotu podawanych na wejściu, a następnie wyłącza go.

Blok umożliwia sterowanie prędkością i kierunkiem obrotu silnika.

Ustawienie Hamowania na True powoduje zatrzymanie silnika natychmiast po wykonaniu obrotu o zadaną ilość stopni.

Nie ma ograniczeń, jeśli chodzi o podawanie wartości wejściowej stopni obrotu. Należy jednak pamiętać, że jeden pełen obrót to 360 stopni.

Wykorzystując ten tryb, blok będzie czekał aż silnik obróci się o dokładną liczbę stopni, podaną na wejściu. Dopiero po osiągnięciu zadanej pozycji program przejdzie do następnego kroku. W przypadku zablokowania (np. zła konstrukcja mechaniczna) program zatrzyma się i będzie oczekiwał na ukończenie obrotów.

ON FOR ROTATION – obraca silnik o liczbę obrotów podaną na wejściu, a następnie wyłącza silnik.

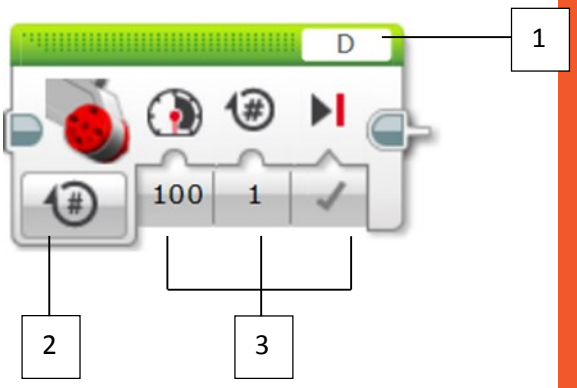
Blok umożliwia sterowanie prędkością i kierunkiem obrotów silnika. Ustawienie hamowania na True spowoduje, że zaraz po wykonaniu obrotów silnik się zatrzyma.

B. Obsługa bloku sterowania dużym silnikiem (Large Motor)

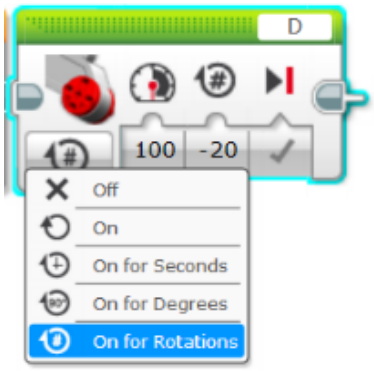
1 – Wybór portu

2 – Wybór trybu

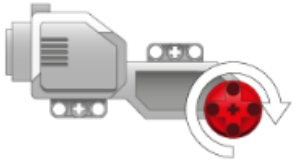
3 – Wejścia – ulegają zmianie w zależności od wybranego trybu



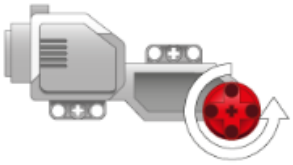
TRYBY DZIAŁANIA



Obsługa jest niemal identyczna, jak w przypadku bloku sterowania Średnim Silnikiem (**Medium Motor**). Różni się jedynie kierunkami obrotów.



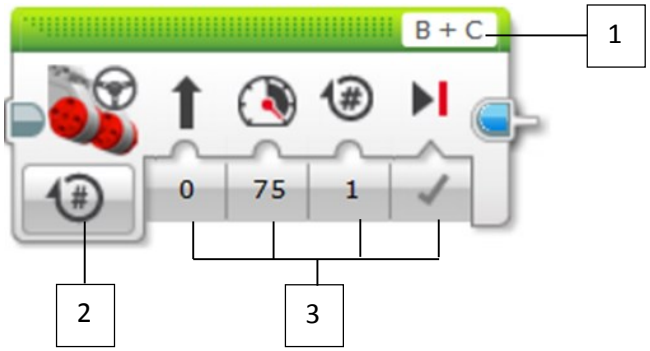
W przypadku wartości **mniejszych od 0** silnik kręci się w przód.



W przypadku wartości **mniejszych od 0** silnik kręci się w tył.

C. Obsługa bloku sterowania dwoma silnikami (Move Steering)

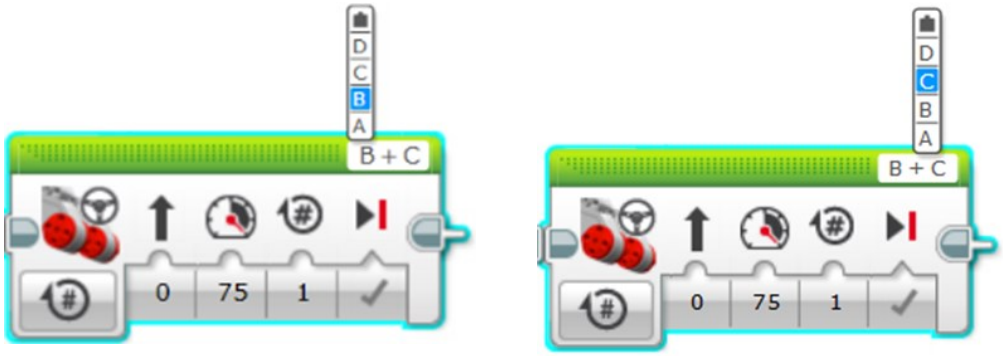
Sterowanie (**Move Steering**) to złożenie dwóch dużych silników (**Large Motor**), przy czym jeden steruje lewą stroną pojazdu, a drugi prawą. Blok ten umożliwia sterowanie dwoma silnikami równocześnie.



1 – Wybór portów

2 – Wybór trybu działania

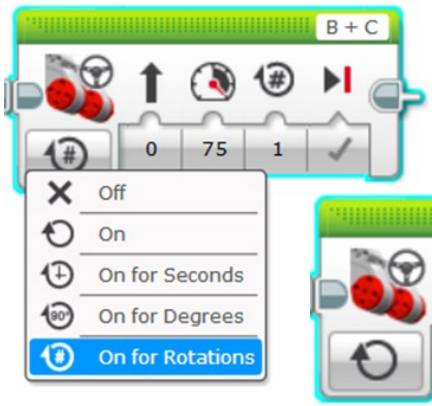
3 – Wejścia – zmienne w zależności od trybu



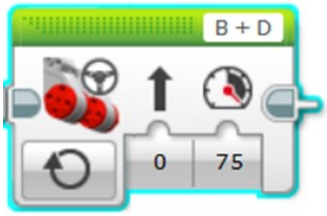
W celu wybrania portów należy kliknąć w pole wyboru portu z lewej i z prawej strony. Należy upewnić się, że **lewy silnik** jest **pierwszy**. W przeciwnym wypadku robot będzie poruszał się w przeciwną stronę.

TRYBY DZIAŁANIA

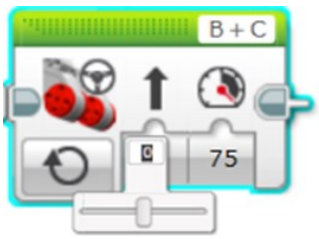
ON – uruchamia oba silniki natychmiastowo i program przechodzi do następnego kroku.



Silniki będą działać dopóki nie zostaną zatrzymane, bądź zmienione przez inny blok wykorzystany w późniejszym kroku programu, lub program się skończy.

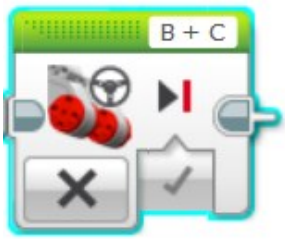


Blok umożliwia sterowanie prędkością oraz kierunkiem ruchu silników po przez wykorzystanie wejść bloku Mocy (**Power**) i Kierunku (**Steering**)



Kierunek podaje się liczbą z zakresu od -100 do 100.
0 zapewni jazdę prosto robota, powyżej 0 (dodatnie wartości) spowoduje skręt w prawo, a

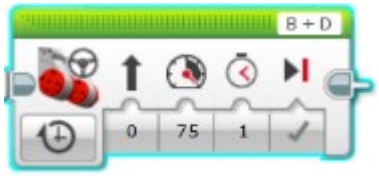
OFF – wyłącza oba silniki.



Zmienną wejściową jest Hamowanie (**Brake at End**). Jeśli jest **True**, napędy zatrzymują się natychmiastowo i utrzymują pozycję.

Jeśli jest **False**, zasilanie silnika jest wyłączane a napędy wytracają swoją prędkość.

ON FOR SECONDS – uruchamia silniki na określony z góry czas, a po jego upłynięciu wyłącza je.



Blok umożliwia sterowanie prędkością i kierunkiem ruchu robota.

Wartość czasu można podawać również w postaci ułamka dziesiętnego (np. *3,5 s*).

ON FOR DEGREES – silniki uruchamiane są dopóki jeden z nich nie osiągnie wymaganej liczby stopni obrotu, a następnie wyłącza je.

Blok umożliwia sterowanie prędkością i kierunkiem ruchu robota.

Jeden pełen obrót wynosi 360 stopni.

Odległość, którą pokona robot zależy od stopni podawanych w danych wejściowych. Jednakże odległość zależy od średnicy koła robota.

ON FOR ROTATIONS – uruchamia silniki i oczekuje na wykonanie przez nie podanej na wejściu liczby obrotów.

Blok umożliwia sterowanie prędkością i kierunkiem ruchu robota.

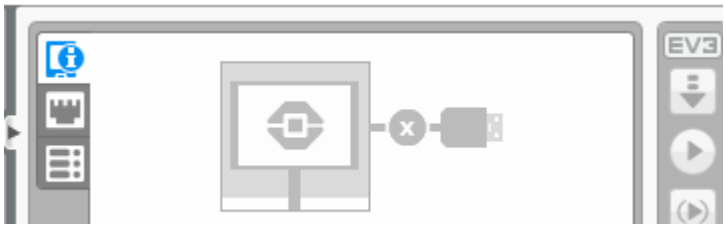
Kierunki obrotów silnika takie jak przedstawione w sekcji dot. Dużego Silnika (Large Motor).

D. Obsługa bloku startu (Start Block)

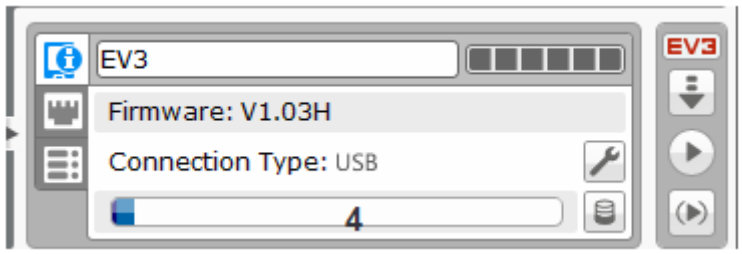
Blok startu stanowi początek sekwencji każdego programu. W programie może znaleźć się więcej niż jeden blok startowy. Wszystkie sekwencje połączone z tym blokiem zostaną automatycznie uruchomione w tym samym czasie.

4. ZGRYWANIE PROGRAMU NA KOSTKĘ

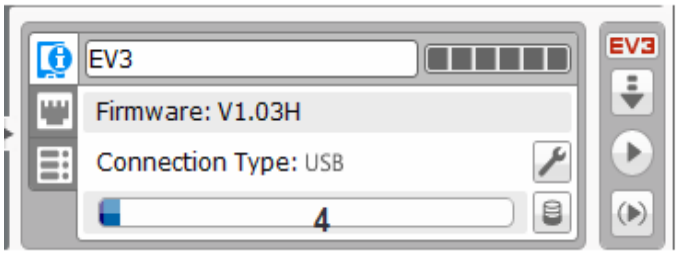
W celu zgrania programu na kostkę należy podłączyć kostkę do komputera przy pomocy kabla USB. Kostka musi być włączona. W prawym dolnym rogu uaktywni się okienko:



- kostka nie podłączona



- kostka podłączona



1 - Zgranie programu na kostkę

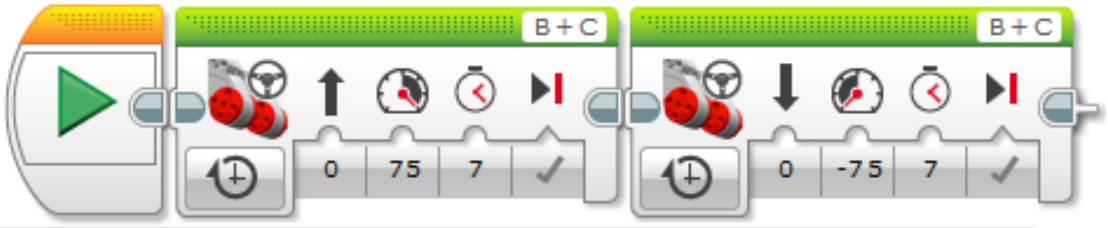
2 – Zgranie programu na kostkę i jego uruchomienie

3 – Uruchomienie programu

4 – Stan pamięci

Zadanie 1. Jazda robota od linii do linii

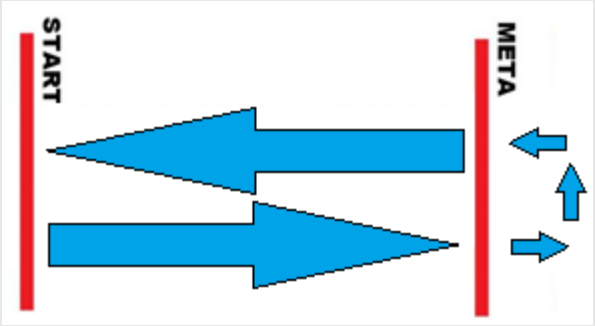
Przyklejamy na podłodze dwie linie w odległości około 2-3m. Należy zaprogramować robota tak aby dojechał od linii do linii i z powrotem .



Program: 1robot.ev3

Zadanie 2. Jazda od linii do linii z zawracaniem

Zadaniem robota jest jazda od linii do linii. W jedną stronę robot jedzie z ustawieniem silników na obroty (rotations), natomiast z powrotem z ustawieniem na czas (seconds). Na linii mety robot musi zawrócić o 180 stopni.



Program: 1robot.ev3 Program2

5. Obliczenia fizyczne

Zadanie 3. Pomiar prędkości średniej

Zmierz czas, jaki potrzebuje robot potrzebuje na przejechanie 6, 4, 2m.

Na podstawie uzyskanych wyników oblicz prędkość średnią ze wzoru:

$$v = \frac{s}{t}$$

gdzie:

v – prędkość średnia [m/s],

s – droga [m],

t – czas [s];

Pytania do uzyskanych wyników:
Czy wszystkie prędkości średnie mają zbliżoną wartość?

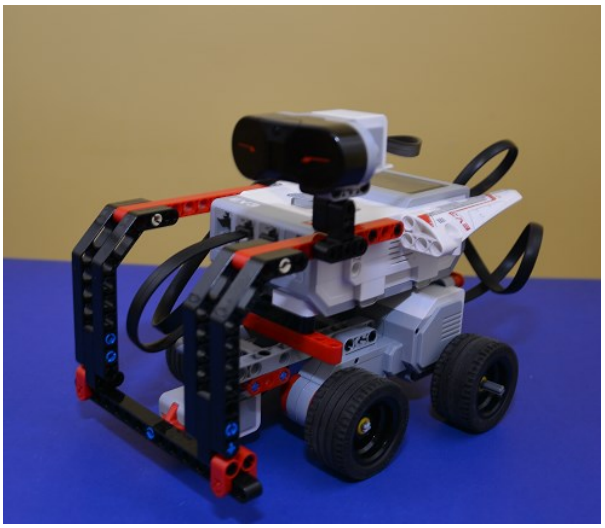
Jakim ruchem poruszał się robot?

Co należy zrobić, aby określić dokładnie, jakim ruchem poruszał się robot w danej chwili czasu t?

Zadanie 4. * Budowa robota poruszającego się slalomem, po ósemce lub w labiryncie.

Temat 3: EASYBOT

Z pierwszych zajęć wiemy, w jaki sposób działa czujnik podczerwieni i czujnik dotyku. W czasie tych zajęć wykorzystamy je do budowy robota zdolnego wykrywać przeszkody. W tym celu musimy poznać dwie nowe funkcje programistyczne: pętlę, która sprawi, że robot będzie wykonywał daną czynność określoną ilość razy oraz funkcję decyzyjną „switch", która sprawi, że robot będzie reagował na sygnały zewnętrzne.

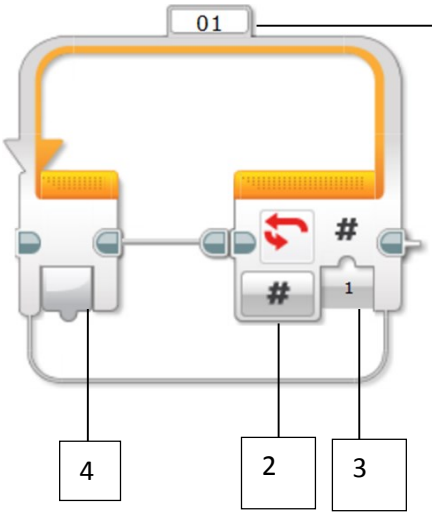


Ale zanim przystąpimy do programowania musimy skonstruować naszego robota. Budujemy robota według instrukcji „easybot.pdf".

1. Programowanie robota

A. Obsługa bloku pętli (Loop Block)

Pętla jest tzw. kontenerem, który może przechowywać sekwencje bloków programowych. Sprawia, że sekwencje znajdujące się wewnątrz są powtarzane. Użytkownik ma możliwość ustawiania ilości powtórzeń. Po zakończeniu pętli, program realizuje bloki znajdujące się poza pętlą.



1 – Nazwa pętli

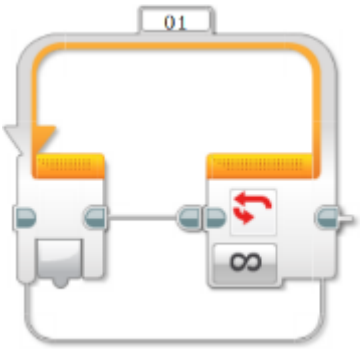
2 – Wybór trybu działania

3 – Sygnały wejściowe

4 – Sygnał wyjścia

TRYBY DZIAŁANIA

Unlimited – pętla niekończąca się.

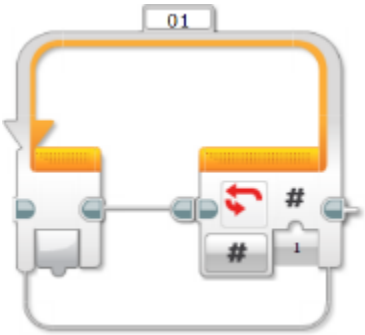


Wszystkie sekwencje bloków umieszczone wewnątrz będą nieustannie odtwarzane.

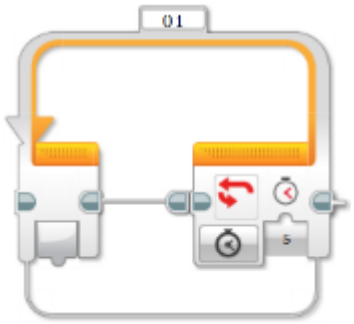
Wszystkie bloki znajdujące się za pętlą nigdy nie zostaną wykonane.

W celu zatrzymania programu należy nacisnąć klawisz Powrotu (**Back**) na klocku EV3.

Count – pętla powtarzająca program zawarty w niej określoną liczbę razy.



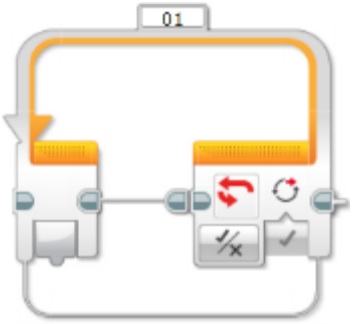
Time – pętla działający przez określony przez użytkownika czas [s].



Czas trwania pętli odliczany jest od momentu wejścia programu głównego do bloku pętli.

Limit czasu jest testowany jedynie pod koniec sekwencji pętli. Sekwencja ta będzie zawsze wykonana, chociaż raz, a pętla będzie kontynuowana od początku jedynie, jeśli miniony czas trwania pętli jest mniejszy od czasu trwania pętli podanego w parametrach bloku.

Logic – pętla powtarzana, dopóki wejście warunku jest spełnione (**True**).

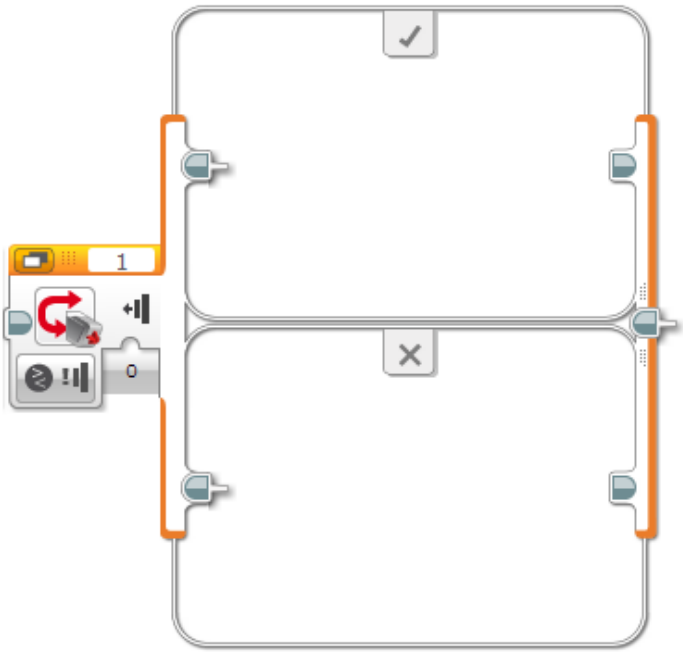


Zawartość pętli zostanie wykonana, chociaż raz i warunek wejścia jest testowany na koniec każdej iteracji pętli.

B. Obsługa bloku przełącznika (Switch Block)

Blok ten jest kontenerem, który może przechowywać dwie lub więcej sekwencji bloków programowych. Każda sekwencja nazywana jest **Case** (pl. przypadek, sytuacja).

Na początku wejścia do bloku Przełącznika (Switch) sprawdzany jest, który przypadek będzie realizowany. Za każdym wejściem na ten blok może być, bowiem realizowana tylko jedna sekwencja.



1 – Wybór portu

2 – Wybór trybu działania

3 – Sygnały wejściowe, różne w zależności od wybranego trybu

4 – Warunek, jaki musi być spełniony

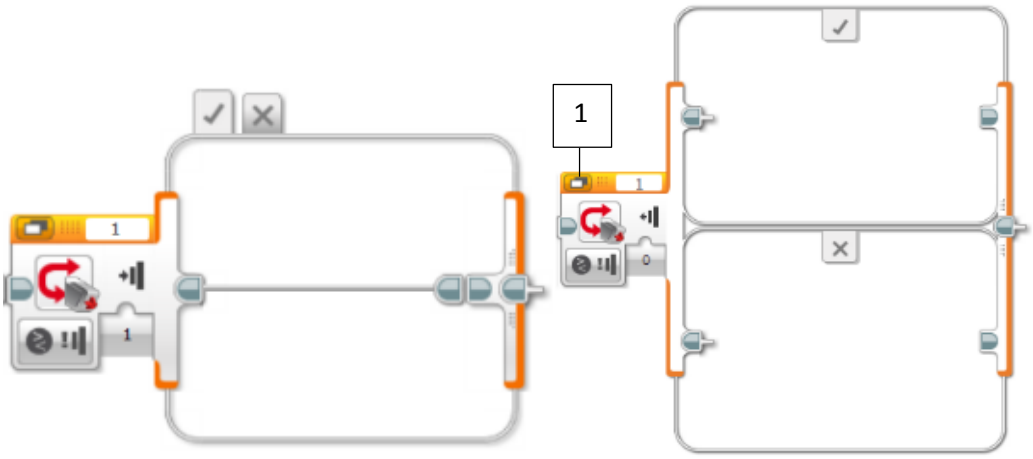
5 – Sekwencja, gdy warunek jest spełniony

6 – Sekwencja, gdy warunek nie jest spełniony

Blok ten nie czeka, aż warunek zostanie spełniony. Sprawdza aktualną wartość i realizuje zadania zgodne z wynikiem (gdy się zgadza lub nie).

Widok

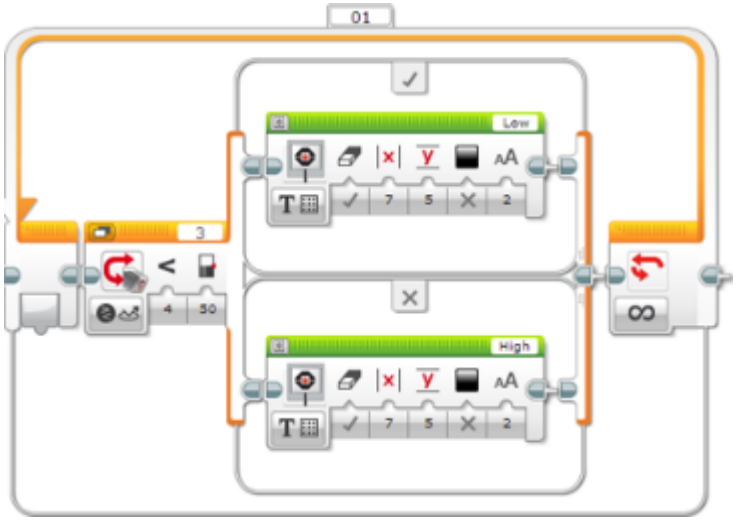
Okno bloku może być wyświetlane na dwa sposoby: w postaci płaskiego (A) lub zakładkowego widoku (B). 1 – Wybór widoku okna



Rys. A Rys. B

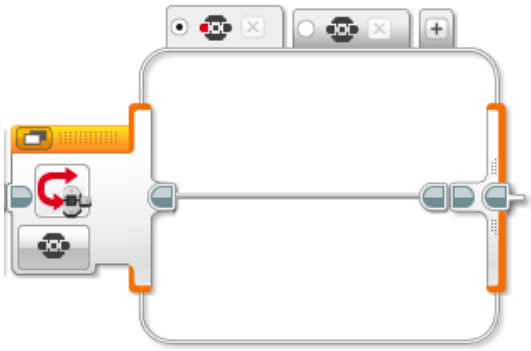
TRYBY DZIAŁANIA

W zależności od wybranego trybu działania porównuje się wartości sensorów do wartości progowej (np. czujnik odległości), lub sprawdza się konkretne wartości czujników |(np. czujnik dotyku).



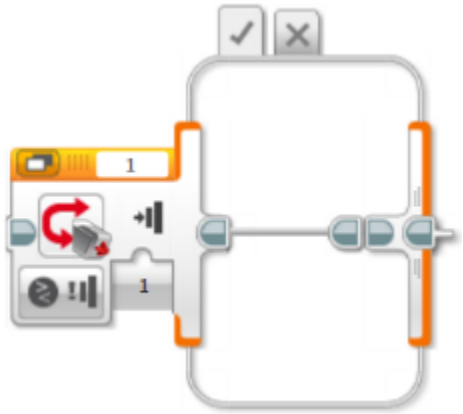
Sprawdzanie progu czujnika – zwracany wynik przyjmuje wartość logiczną Prawda/Fałsz (True/False)

Blok Switch sczyta wartość z czujnika i porówna ją z wartością progową. W zależności od wyniku będzie realizowana odpowiednia sekwencja bloków.



Sprawdzanie konkretnych wartości czujnika – umożliwia sprawdzanie kilku wartości czujnika. Umożliwia tworzenie dwóch lub większej liczby przypadków (**Case**) zależnych od różnych wartości, dla których ma być program testowany.

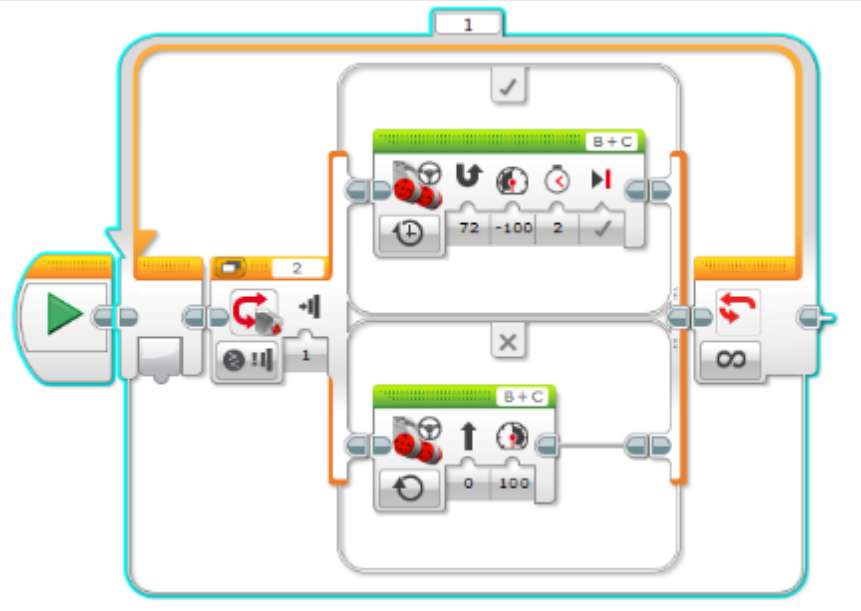
W zależności, który przycisk zostanie naciśnięty można utworzyć kilka różnych sekwencji programu.



Sprawdzenie czy czujnik dotyku został naciśnięty.

Zadanie 1. Napisz program wykorzystujący czujnik dotyku

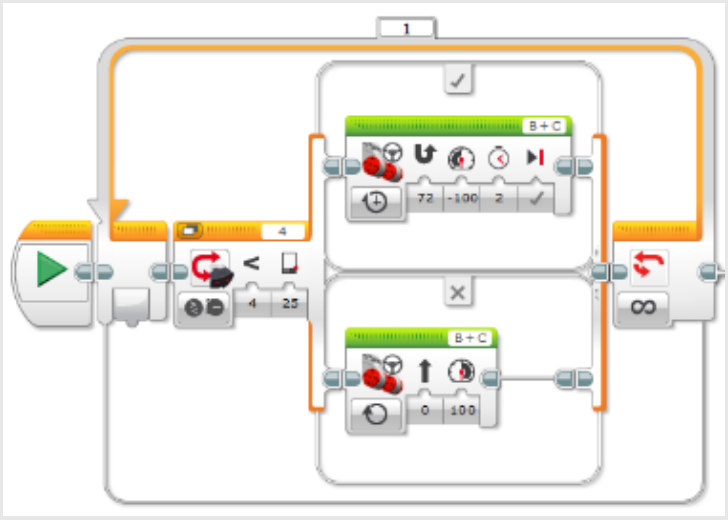
Robot jedzie do przodu do momentu wciśnięcia czujnika dotyku podłączonego do portu 2. Silniki są podłączone do portów B i C. Następnie robot się wycofuje i skręca. Wykorzystanie pętli pozwala na powtarzanie jazdy roboty.



Pogram: EasyBot.ev3 Program

Zadanie 2. Napisz program wykorzystujący czujnik odległości

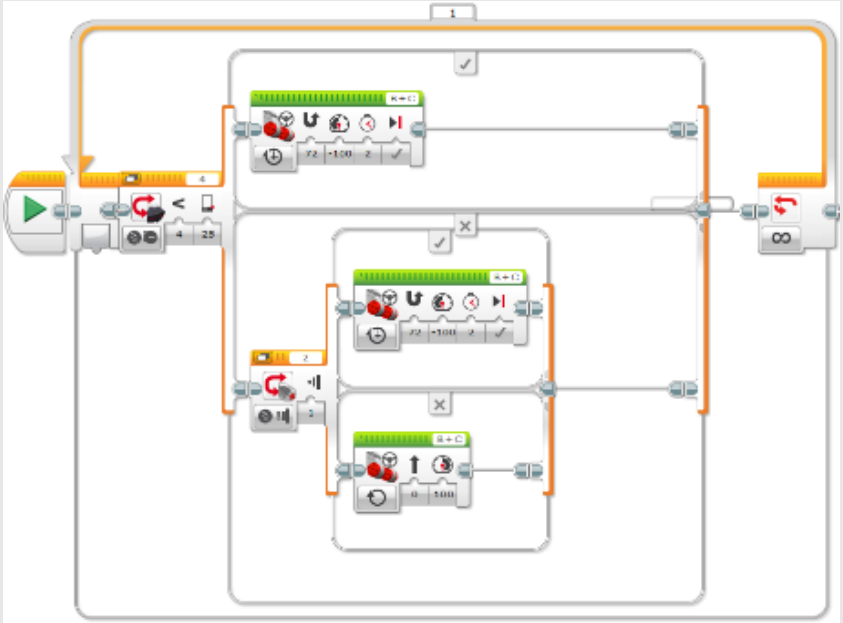
Robot jedzie do przodu do momentu wykrycia przeszkody w określonej odległości przez czujnik podczerwieni podłączony do portu 4. Następnie robot się wycofuje i skręca. Silniki podłączone są do portów B i C. Wykorzystanie pętli pozwala na powtarzanie jazdy robota.



Program: EasyBot.ev3 Program2

Zadanie 3. Napisz program wykorzystujący oba czujniki

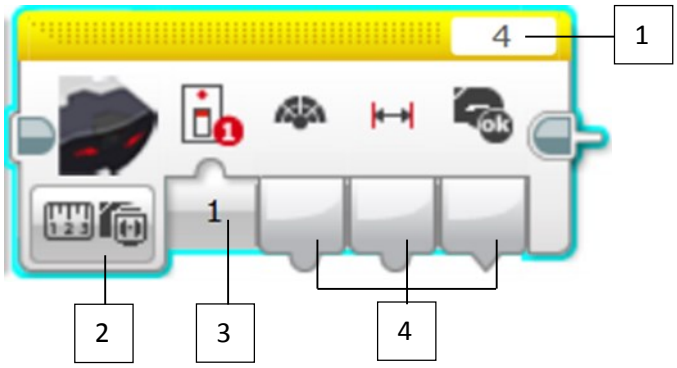
Czasami może się zdarzyć, że czujnik odległości nie zawsze zauważy przeszkodę. Dlatego nie głupim pomysłem będzie wykorzystanie obu czujników. Robot jedzie do przodu aż do momentu wykrycia przeszkody. Co jeśli przeszkoda będzie koloru czarnego? Czujnik jej nie wykryje. Na szczęście mamy czujnik dotyku, który na pewno zadziała.



Program: EasyBot.ev3 Program3

2. Obsługa bloku czujników

A. Obsługa bloku czujnika podczerwieni (Infrared Sensor Block)



1 – Wybór portu

2 – Wybór trybu działania

3 – Sygnał wejściowy, inny w zależności od wybranego trybu

4 – Sygnały wyjścia, inne w zależności od wybranego trybu

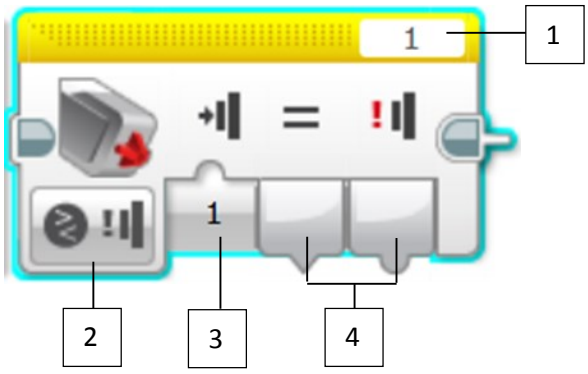
TRYBY DZIAŁANIA

Measure – Proximity

Wykorzystuje czujnik podczerwieni do określenia rzeczywistej odległości do obiektu. Odległość podawana jest w postaci numerycznej. Przyjmuje wartości od 0 – 100, przy czym 0 oznacza bardzo blisko, a 100 bardzo daleko.

B. Obsługa bloku czujnika dotykowego (Touch Sensor Block)

Blok ten umożliwia zbieranie danych za każdym razem, gdy czujnik jest naciskany (**Pressed**), odpuszczany (**Released**) lub wciskany (**Bumped**).



1 – Wybór portu

2 – Wybór trybu działania

3 – Sygnał wejścia, zależny od trybu działania

4 – Sygnał wyjścia, zależny od trybu działania

TRYBY DZIAŁANIA

Measure – State

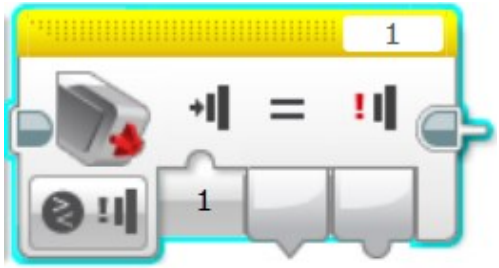
Blok, który na wyjście przekazuje stan przycisku:

0 – Odpuszczony (**Released**)

1 – Naciśnięty (**Pressed**)

2 – Wciskany (**Bumped**)

Compare – State



Użytkownik wybiera, jaki stan przycisku ma być porównywany ze stanami odczytywanymi z czujnika.

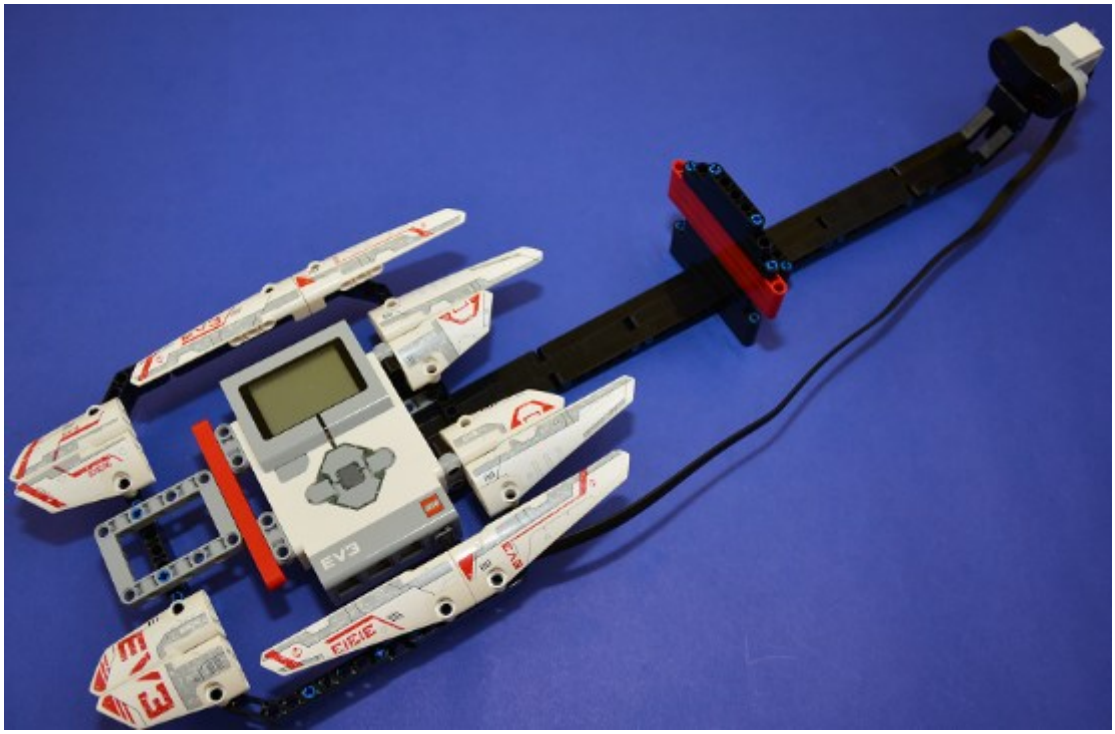
Na wyjściu podawany jest stan porównania i aktualny stan przycisku.

Temat 4: GITARA ELEKTRYCZNA

Nasza gitara będzie wydawała różny dźwięk w zależności od tego w jakiej odległości od czujnika podczerwieni znajdzie się przeszkoda.

Konstrukcja gitary elektrycznej musi posiadać gryf, czujnik odległości oraz poprzeczkę.

Gitarę budujemy według instrukcji ”gitara.pdf”

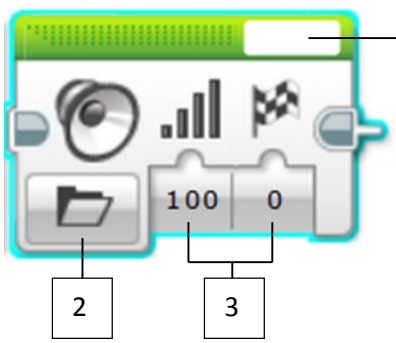


1. PROGRAM

Program sterujący gitarą składa się głównie z instrukcji warunkowej od czujnika podczerwieni oraz bloku dźwięku. W zależności od wykrywanej odległości wydawany jest inny dźwięk. Odległość jest badana, co pewną określoną wartość. Całość programu objęta została pętlą.

2. Obsługa bloku dźwięku (Sound)

Blok dźwięku wykorzystuje głośniki z klocka EV3 do wydobycia dźwięku. Można odgrywać dźwięki z pliku lub określone nuty muzyczne bądź tony.



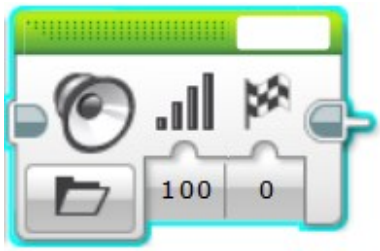
1 – Nazwa pliku dźwiękowego

2 – Wybór trybu

3 – Sygnały wejściowe, inne w zależności od wybranego trybu

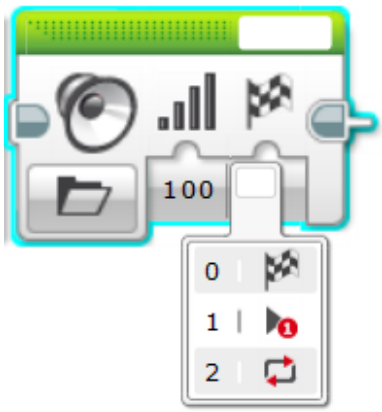
TRYBY DZIAŁANIA

Play File – odtwarza plik dźwiękowy



Użytkownik ma możliwość sterowania głośnością oraz typem odtwarzania.

Głośność (**Volume**) jest wartością procentową dźwięku 0 – 100 %.



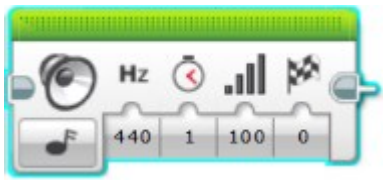
Typ odtwarzania przyjmuje 3 wartości:

0 – dźwięk jest odtwarzany raz, program czeka aż cały dźwięk zostanie odegrany zanim przejdzie do następnego kroku programu

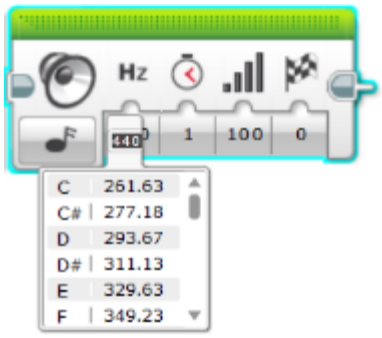
1 – dźwięk jest odtwarzany raz, ale program jest kontynuowany zanim dźwięk dobiegnie końca

2 – dźwięk będzie powtarzany nieustannie, program od razu przechodzi do następnego kroku.

Play Tone – odgrywa tony o określonej częstotliwości. Częstotliwość tonu kontroluje wysokość dźwięku (jak niski lub wysoki jest).



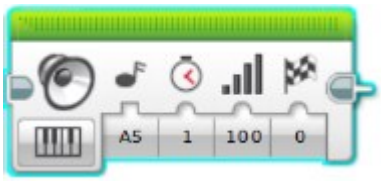
Blok umożliwia sterowanie częstotliwością, czasem trwania dźwięku w sekundach, głośnością oraz typem odtwarzania.



Wejście Częstotliwości (**Hz**) pobiera częstotliwość tonu. Można wprowadzić wartość w postaci liczby z zakresu 300 – 10000, bądź wybrać z listy standardowych częstotliwości muzycznych.

Czas trwania (**Duration**) steruje jak długo będzie trwał dźwięk. Wartość może być podawana w postaci ułamka dziesiętnego.

Play Note – tryb odgrywania nut.



Daje możliwość sterowania opcjami, jak w poprzednich trybach, dając możliwość wyboru dźwięku w postaci nut muzycznych.

Opcja ta daje możliwość programiście samodzielnego wybrania dźwięku prosto z pianina lub ręcznego wpisania wartości

A – G są nazwami nut muzycznych

4 – 6 są numerami oktawy

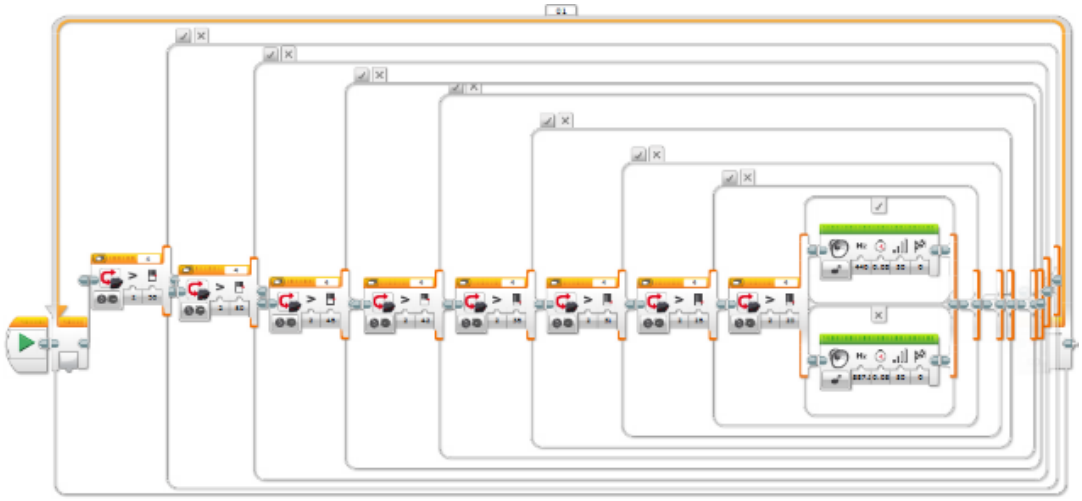
oznacza podniesienie dźwięku o półtonu

Dźwięki szeregu półtonowego

A – G	C	D	E	F	G	A	B
Do, Re, Mi	Do	Re	Mi	Fa	Sol	La	Si

Stop – zatrzymuje każdy dźwięk, który jest aktualnie odgrywany w klocku EV3.

Gotowy program powinna wyglądać tak:



Program: gitara.ev3 Program

Zadanie 1. **W jaki sposób zmiana czasu trwania poszczególnych dźwięków wpłynie, na jakość powstającej muzyki?**

Zadanie 2. **Zamień częstotliwość tonu podaną w Hz na nuty muzyczne.**

Temat 5: ROBOT WYŚCIGOWY – ZASADA DZIAŁANIA PRZEKŁADNI

Co zrobić, aby nasz robot jechał szybciej?

Możemy zastosować system kół zębatych, które sprawią, że koła naszego robota będą się obracać zdecydowanie szybciej niż silnik. Mechanizm taki nosi nazwę przekładni. Jest to układ, który przenosi „ruch" z jednego elementu na drugi. Przy przenoszeniu ruchu ulegają zmianie takie parametry jak prędkość i siła.

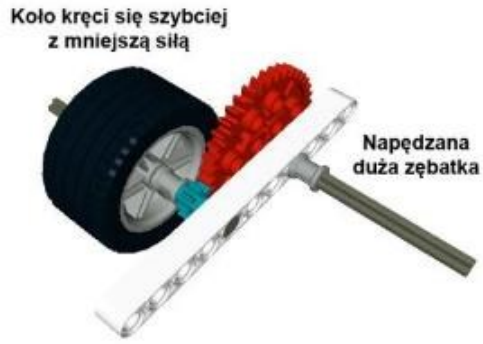
Najbardziej znanym z życia codziennego przykładem przekładni są przerzutki w rowerze.

Chcąc wjechać pod górę należy ustawić z przodu małą zębatkę z tyłu dużą zębatkę. Takie ustawienie zębatek pozwala na uzyskanie dużej siły przy równoczesnym zmniejszeniu prędkości. Jedziemy wolno, ale bez problemu podjedziemy po górę.

Chcąc jechać szybko należy ustawić z przodu dużą zębatkę z tyłu małą zębatkę. Takie ustawienie natomiast pozwala uzyskać dużą prędkość przy jednoczesnym zmniejszeniu siły.

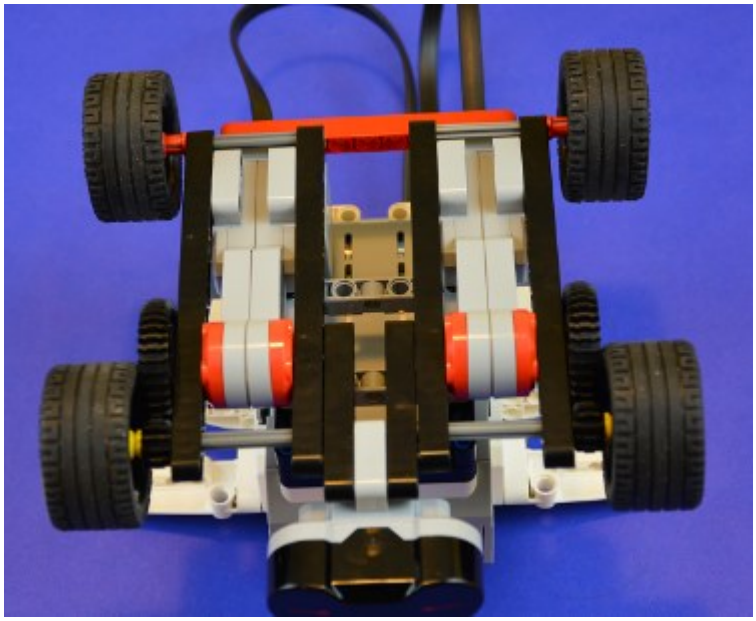
Przekładnia na moc (siłę)

Przekładnia na prędkość



Jaki typ przekładni wykorzystamy w konstrukcji naszego robota? Oczywiście, że przekładni na prędkość, chcemy przecież, żeby nasza wyścigówka jechała jak najszybciej.

Nasza konstrukcja będzie posiadała dwa symetrycznie umieszczone silniki oraz dwa kola zębate na każdy silnik (w sumie 4). Naszego robota budujemy według instrukcji „wyscigowka.pdf".



Następnym krokiem na dzisiejszych zajęciach jest napisanie programu przy pomocy, której nasz robot będzie jechał. Chcemy, aby poruszał się przez ściśle określony czas, w tym celu ustawiamy czas działania silników na sekundy (On for seconds).

Program: wyscigowka.ev3 Program

Program umożliwia w bardzo prosty sposób zmianę czasu poruszania się robota. Silniki podłączamy do portów B i C.

Zadanie 1. Badamy ruch, jakim porusza się nasz robot

Dokonujemy jak największej ilości pomiarów przebytej drogi w zależności od czasu działania silników.

Najlepiej jak każdy zespół w klasie dokona pomiarów dla różnych czasów działania silnika (np. czas jazdy robota zostaje ustawiony kolejno na 0,25; 0,5; 0,75; 1; 1,25; 1,5; 1,75; 2; 2,25; 2,5; 2,75; 3; 3,25; 3,5; 4; 4,5; 5; 5,5; 6s), dla każdego czasu zostaje określona odległość, jaką przebył robot. Na podstawie tych wyników zostaje sporządzony wykres zależności przebytej drogi od czasu.

Pomiarów można dokonać kilkukrotnie.

UWAGA!
Wszystkie roboty muszą mieć w pełni naładowanie baterię, gdyż może to wpłynąć na uzyskiwaną moc silników.

Moc silników ustawiona przy pomocy programu NXT-G dla wszystkich robotów musi być identyczna.

Wszystkie roboty muszą być w 100% zgodne z instrukcją.

Zadanie 2. Analiza uzyskanego wykresu

a) Jakim ruchem porusza się robot?

b) Czy jest to ruch jednostajnie przyspieszony? Jak to sprawdzić?

c) Czy można obliczyć przyspieszenie w początkowej fazie ruchu na podstawie uzyskanego wykresu?

Zadanie 3. *Zależność średniej prędkości od czasu

Sporządź wykres zależności średniej prędkości od czasu na podstawie uzyskanych wyników. Dla każdego odstępu czasu wyznaczamy drogę Δsx pokonanego w czasie Δt=0,25 s i na podstawie wzoru:

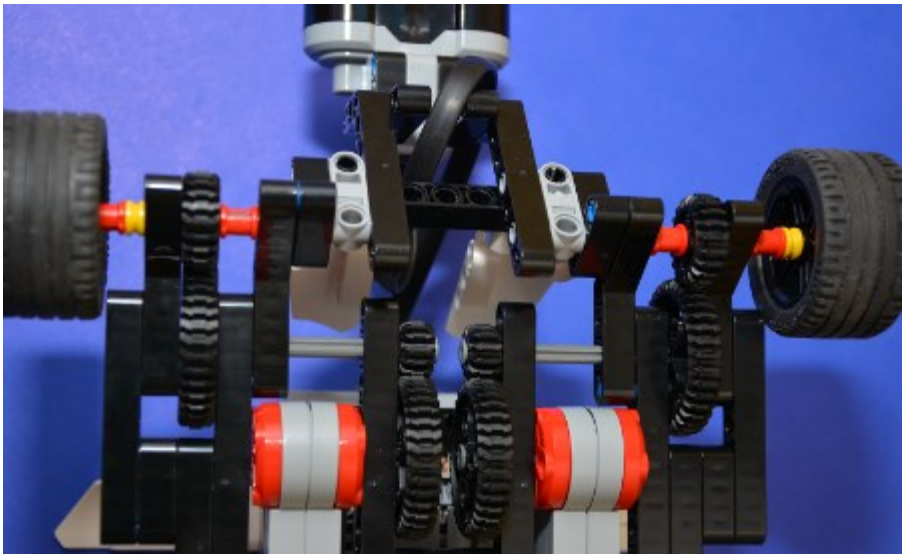
$$V_{śr} = \Delta s_x / \Delta t,$$

Obliczenie $V_{śr}$ po upływie określonego czasu od początku pomiaru.

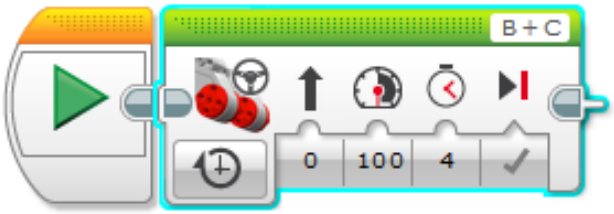
Analiza uzyskanego wykresu.

Temat 6: CO ZROBIĆ, ABY NASZA WYŚCIGÓWKA JECHAŁA JESZCZE SZYBCIEJ?

Tym razem zbudujmy robota, który będzie posiadał podwójną przekładnie na prędkość. Naszego robota budujemy według instrukcji „wyscigowka2.pdf".



Program sterujący naszą wyścigówką będzie prawie identyczny jak na poprzednich zajęciach Tym razem czas pracy silnika pozostanie stały. Zmianie będzie ulegała moc silnika. Należy pamiętać o tym aby czas pracy silnika był ustawiony w sekundach. Moc przyjmuje wartości dodatnie, ponieważ podwójna przekładnia nie zmienia kierunku obracania się kół względem silnika.



Program: wyscigowka.ev3 Program2

Zadanie 1. Badamy zależność przebytej drogi od mocy silnika

Tak jak no poprzednich zajęciach dokonujemy serii pomiarów. W tym wypadku zmiennym parametrem jest moc silnika (np. moc 5, 10, 15, 20, 25, …., 100), dla każdego czasu zostaje określona odległość, jaką przebył robot. Na podstawie tych wyników zostaje sporządzony wykres zależności przebytej drogi od mocy.

Zadanie 2. Analiza uzyskanego wykresu

Jaki charakter ma funkcja? Czy jest to zależność liniowa?

Zadanie 3. *Dokonanie jak największej ilości pomiarów czasu potrzebnego na przebycie określonej drogi w zależności od mocy silnika.

W tym przypadku droga, jaką ma pokonać wyścigówka jest ściśle określona, dokonujemy pomiarów czasu, jakiego potrzebowała wyścigówka na pokonanie tego dystansu w zależności od mocy silnika. Na tej podstawie sporządzamy wykres zależności czasu od mocy.

Zadanie 4. * Analiza uzyskanego wykresu

Jaki charakter ma funkcja? Czy jest to zależność liniowa?

Zadanie 5. * Oszacowanie przyspieszenia w początkowej fazie ruchu w taki sposób jak dla wyścigówki z 1 przekładnią

Temat 7: WYŚCIGI

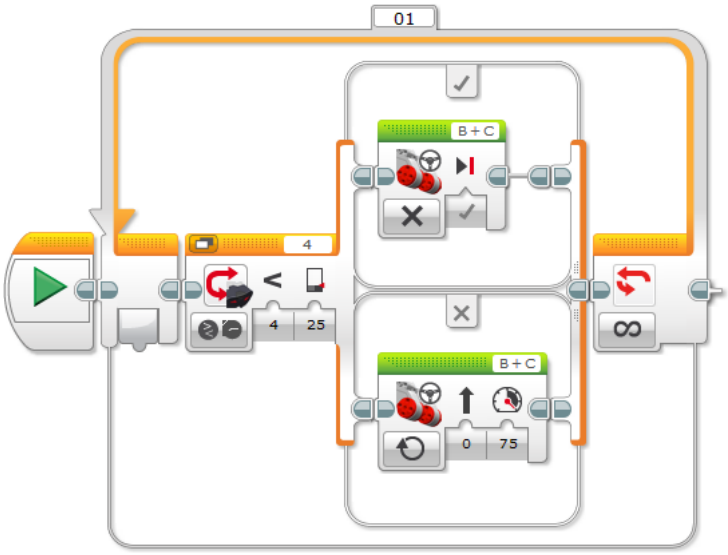
1. Budowa robota wyścigowego według własnego pomysłu i jego programowanie

Budowa robota wyścigowego. Staramy się zbudować robota wyścigowego według własnego pomysłu z zachowaniem kilku wytycznych:

- robot na dwóch silnikach
- konstrukcja robota nie musi umożliwiać skręcania
- zastosowanie przekładni mechanicznej na prędkość
- z przodu robota czujnik odległości zabezpieczający przed uderzeniem robota w ścianę
- robot musi być zbudowany symetrycznie. W przeciwnym razie robot będzie skręcać

2. Program zabezpieczający przed uderzeniem w ścianę.

W tym przypadku znów musimy skorzystać z pętli oraz switcha. Robot pojedzie tak długo aż nie zbliży się do ściany.



Program: wyscigowka.ev3 Program3

3. Wyścigi robotów

Przyklejamy na podłodze dwie linie (start i meta). Wszystkie roboty są ustawiane na linii startu, wygrywa ten robot, który przejedzie pierwszy linię mety. W przypadku dużej liczby robotów można przeprowadzić zawody systemem każdy z każdym.

4. Który robot wygrał? Dyskusja

Zadanie 1. Jakie czynniki decydujące o wygranej?

Zadanie 2. Czy masa robota ma wpływ na wygraną?

Porównajmy ze sobą dwa roboty z identycznymi przekładniami, oraz ustawioną taką samą mocą silników, czy czas przejazdu jest identyczny? Porównajmy masę obu robotów. W razie identycznych wyników (bardzo zbliżonej wagi) zwiększmy ciężar jednego z nich po przez dołożenie klocków. Czy uzyskiwane wyniki ulegną zmianie?

Zadanie 3. * Przyspieszenie robotów

Dla poszczególnych robotów liczymy przyspieszenie w początkowej fazie ruchu.

Temat 8: KALIBRACJA – KONSTRUKCJA WAGI

1. Co to jest kalibracja?

Kalibracja jest to ogół czynności służących do wzorcowania przyrządu pomiarowego. Celem kalibracji jest przyporządkowanie wielkości wskazywanych przez dany przyrząd pomiarowy do konkretnych wartości wielkości fizycznych

2. Dynamometr

Miernik siły firmy HiTechnic pozwala w łatwy sposób zmierzyć siłę nacisku wywieranego przez wsadzoną do niego oś LEGO Technics. Jest to rozbudowana wersja czujnika dotyku. W odróżnieniu od niego mierzy on siłę nacisku i rejestruje ją w skali od 0 do 4080. Otrzymany wynik nie przekłada się bezpośrednio na wartość masy przedmiotów, dlatego do prawidłowego działania czujnika potrzebna jest jego odpowiednia kalibracja.

Czujnik jest w stanie zmierzyć siłę aż do 1 kilograma (9,81 N).

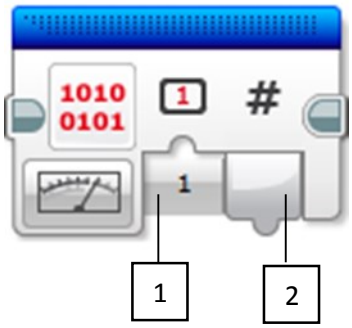
3. Budowa wagi z wykorzystaniem czujnika NXT (dynamometru) według instrukcji

W celu odpowiedniej kalibracji czujnika musimy skonstruować przyrząd, który wyeliminuje wpływ odchylania się osi LEGO Technics, co wpływa na niedokładność pomiaru. Wagę budujemy według instrukcji „waga.pdf".

W celu zaprogramowania wagi musimy zastosować dwa nowe bloki. Pierwszy Raw Sensor Value służy do bezpośredniego odczytywania wartości z czujnika nacisku, drugi Display Block służy do wyświetlania tej wartości na kostce NXT. Poniżej znajduje się ich dokładny opis.

4. PROGRAMOWANIE WAGI

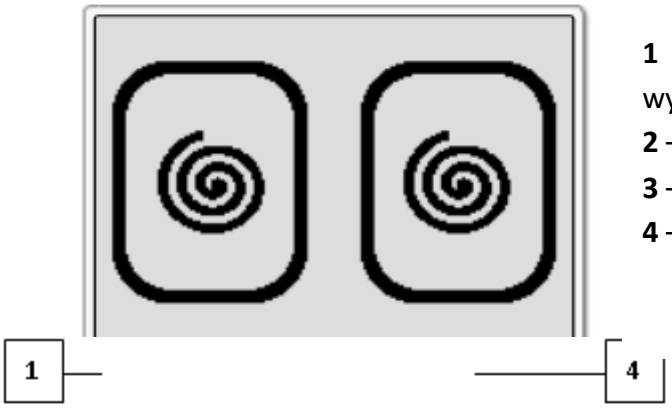
A. Obsługa bloku bezpośredniego odczytywania wartości z czujnika (Raw Sensor Value)



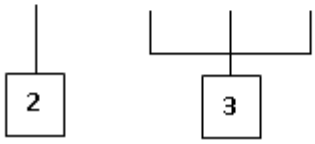
1 – numer portu, do którego jest podłączony czujnik

2 – sygnał wyjściowy w postaci liczby od 0 do 4080

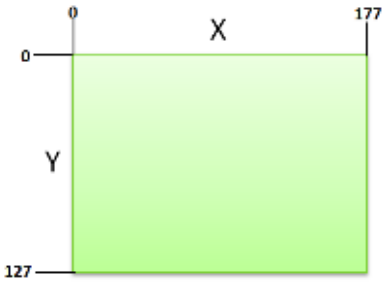
B. Obsługa bloku wyświetlacza (Display Block)



1 – Przycisk podglądu wyświetlacza
2 – Wybór trybu działania
3 – Sygnały wejściowe
4 – Pole tekstowe bloku



Współrzędne wyświetlacza



Blok wyświetlacza (Display Block) wykorzystuje współrzędne do określenia, w którym miejscu rysować obiekty. Wyświetlacz kostki EV3 wykorzystuje w tym celu w współrzędne pozycji w pikselach i ma rozmiar 178 pikseli szerokości i 128 pikseli wysokości.

Rozmieszczenie pikseli na ekranie przedstawia poniższy rysunek.

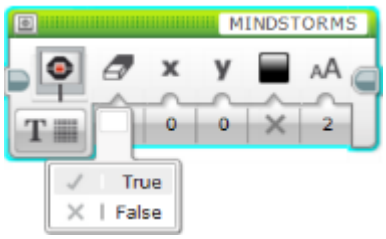
TRYBY DZIAŁANIA

TEXT – Pixels – umożliwia wyświetlanie tekstu w dowolnej lokacji wyświetlacza.

Miejsce, w którym wprowadza się tekst do wyświetlania na ekranie

X – zakres zmiennych od 0 do 177.

Y – zakres zmiennych od 0 do 127.

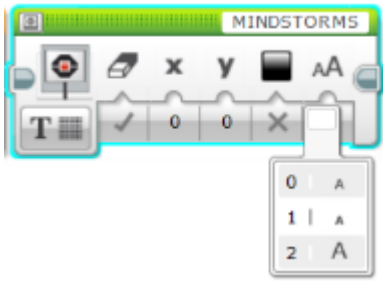


Czyszczenie ekranu (Clear Screen) – przyjmuje wartości logiczne True/False.

Jeśli przyjmie wartość True wyświetlacz zostanie wyczyszczony przed wyświetleniem tekstu.

Kolor (Color) – zmienna logiczna True/False. Odpowiada za kolor tekstu.

Gdy ustawione jest True litery są białe, a gdy False – czarne.



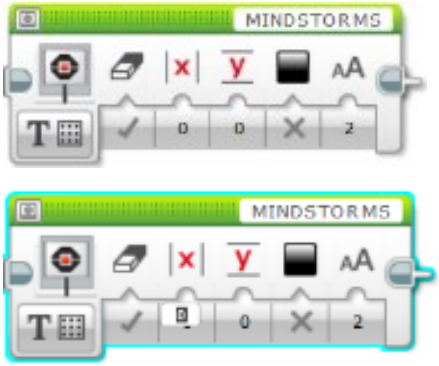
Czcionka (Font) – umożliwia wybór stylu czcionki:

0 – normalna (Normal),

1 – pogrubiona (Bold)

2 – powiększona (Large).

TEXT – Grid – umożliwia wyświetlanie tekstu wyrównanego do siatki rzędów i kolumn. Ułatwia wyświetlanie kilku linijek tekstu.



Umożliwia czyszczenie ekranu, ustawienie koloru i własności tekstu oraz podawanie pozycje w rzędach i kolumnach.

Wartość wpisywana w pole Kolumny (Column) wyznacza początek kolumny, w której wpisany zostanie tekst.

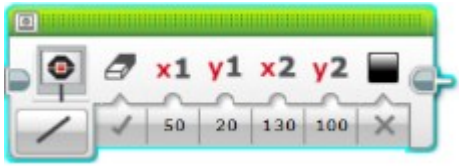
Każda kolumna ma 8px szerokości i są numerowane od 0 do 21 od lewej do prawej.

Wartość Rzędu (Row) bądź numer linii dla tekstu.

Każdy rząd ma 10px wysokości i jest numerowany od 0 do 11 od góry do dołu wyświetlacza.

Szerokość kolumn jest taka sama jak szerokość znaku czcionki normalnej i pogrubionej, natomiast powiększenie czcionki spowoduje, że znak będzie większy zarówno w pionie jak i w poziomie, więc zajmie dwie kolumny i dwa rzędy.

SHAPES – Line – rysuje proste linie pomiędzy dwoma punktami wyświetlacza.

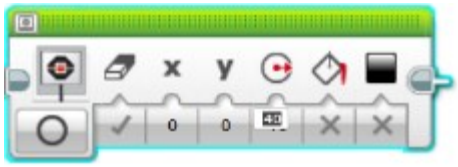


Własności bloku są takie same jak w poprzednich przypadkach. Różni się jedynie potrzebą wpisania współrzędnych dwóch punktów A(x_1,y_1) i B(x_2, y_2).

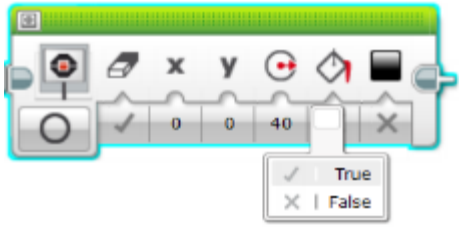
Wartości, które mogą przyjmować są w pikselach odpowiednio
x: 0 – 177, y: 0 – 127.

SHAPES – Circle – umożliwia rysowanie okręgów na wyświetlaczu.

X (0-177) i Y (0-127) przyjmują wartości określające środek okręgu.



Promień (Radius) przyjmuje wartości dodatnie, ≥0 i określa jak duży będzie okrąg.

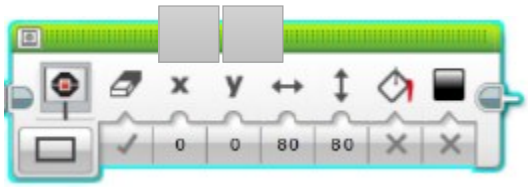


Wypełnienie (Fill) przyjmuje wartości logiczne: True/False.

Jeśli True – rysowany kształt zostanie wypełniony kolorem.

Jeśli False – kolorem zostanie wypełniony obwód kształtu.

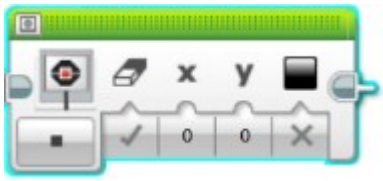
SHAPES – Rectangle – umożliwia rysowanie prostokątów na wyświetlaczu.



X,Y określają współrzędne lewego górnego rogu prostokąta.

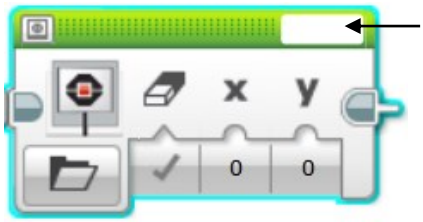
Szerokość (Width) A i wysokość (Height) B określają rozmiar prostokąta w pikselach.

SHAPES – Point – rysuje pojedynczy piksel wyświetlacza.



Blok umożliwia, jak we wcześniej omówionych trybach możliwość czyszczenia ekranu, podania współrzędnych punktu w pikselach oraz określenie koloru punktu.

IMAGE – umożliwia wyświetlanie obrazu graficznego z pliku.



Nazwa pliku (**File Name**) umożliwia wybranie obrazu z listy plików.

X, Y wskazują na współrzędne górnego lewego rogu obrazu.

Zaleca się, żeby **X** = 0, **Y** = 0.

RESET SCREEN

Tryb ten powraca do wyświetlania normalnego ekranu z informacją o przebiegu programu. Ekran ukazuje nazwę programu i informacje zwrotne.

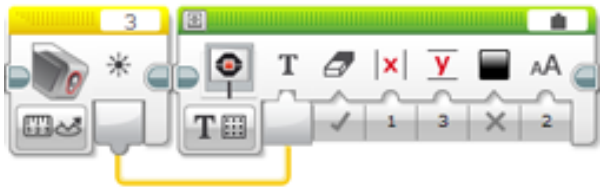
C. Wyświetlanie wielu elementów równocześnie!

Jeśli jest potrzeba wyświetlenia kilku elementów na wyświetlaczu w tym samym czasie, ważne jest, aby nie czyścić ekranu pomiędzy wyświetlaniem ich (**Clear Screen - False**).

Wyświetlanie liczb

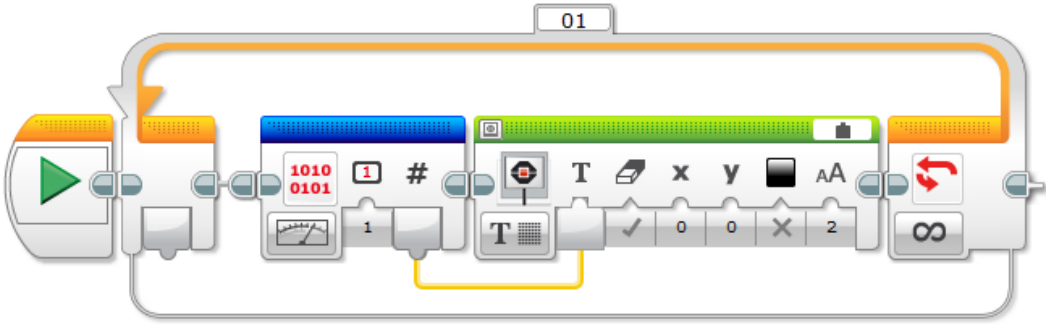
W celu wyświetlenia wartości liczbowych na wyświetlaczu należy połączyć daną „przewodem” do wejścia tekstowego. Liczbowa dana zostanie automatycznie

konwertowana na tekst dzięki odpowiedniemu przewodowi. Przykładowe połączenie na zdjęciu poniżej.



D. Programowanie

Po poznaniu podstawowych funkcji bloków z czytujących dane i wyświetlających je na ekranie kostki EV3 możemy przystąpić do napisania programu. Przykładowy program został przedstawiony poniżej.



Program: waga.ev3 Program

Zadanie 1. Kalibracja czujnika

Program służy do wyświetlania wartości siły nacisku wywieranej na czujnik. Jej rzeczywistą wartość można określić przy pomocy wagi kuchennej. Kalibracja będzie polegała na przyporządkowywaniu wyświetlanej wartości na kostce do rzeczywistej wartości siły nacisku (masy).

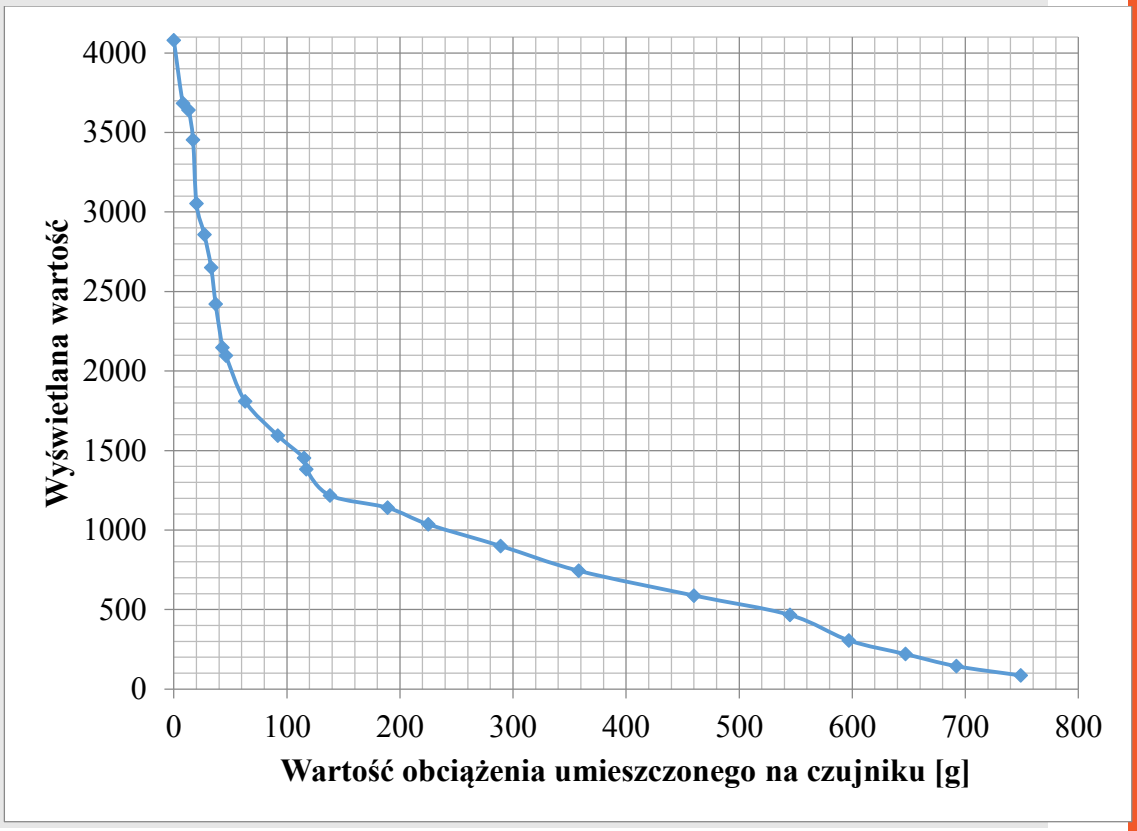
Nacisk jest mierzony po przez umieszczanie na platformie przedmiotu o coraz większej masie (siła nacisku jest powiązana relacją z masą po przez wzór F= mg; gdzie F – siła nacisk, m – masa przedmiotu, g – przyspieszenie ziemskie, wartość siły nacisku jest proporcjonalna do wartości masy przedmiotu)

Na początku należy dokonać pomiaru ciężaru platformy służącej do umieszczania obciążenia, przy pomocy wagi laboratoryjnej, wywiera ona również nacisk na czujnik.

Kilkukrotne odczytujemy wartości wyświetlanej na kostce dla samej platformy, liczymy wartość średnią, wprowadzamy ją do programu komputerowego np. Excel,

Powtórzenie tej samej czynności dla zwiększającego się obciążenia platformy (np. o 25, 50, 100g), aż do uzyskania łącznej wagi razem z platformą 800-1000 g,

Przy pomocy programu sporządzamy wykres zależności wartości wyświetlanej na ekranie od rzeczywistej wartości masy wywierającej nacisk na czujnik. Przykładowy wykres został umieszczony na rysunku poniżej.



Zadanie 2. **Przeprowadź dyskusja na temat czynników wpływających na niepewności w czasie naszego pomiaru.**

Zadanie 3. ***Napisanie programu, który zamiast wartości liczbowych od 0 do 4080 będzie wyświetlał na kostce wartość siły nacisku w niutonach lub w gramach.**

Temat 9: ROBOSIŁACZ

Na dzisiejszych zajęciach zbudujemy robota, który będzie w stanie przewieź jak największy ciężar.

Jaką przekładnię należy zastosować, aby robot mógł przenieś jak największy ciężar?

Nasza konstrukcja będzie posiadała dwa symetrycznie umieszczone silniki oraz dwa kola zębate na każdy silnik (w sumie 4). Naszego robota budujemy według instrukcji „robosilacz.pdf".

Następnym krokiem w dzisiejszych zajęciach jest napisanie programu, przy pomocy którego robot będzie się poruszał. Chcemy, aby poruszał się przez ściśle określony czas, w tym celu ustawiamy czas działania silników na sekundy (On for seconds). Możemy wykorzystać program, który był stworzony do sterowania wyścigówką.

Program: wyscigowka.ev3 Program

Program umożliwia w bardzo prosty sposób zmianę czasu, przez który ma jechać robot. Silniki podłączamy do portów B i C.

Zadanie 1. Dokonanie pomiarów zależność drogi przebytej w określonym czasie przez robota od jego masy.

Poszczególne kroki:

- ustawiamy czas przejazdu np. na 10s,
- dokonujemy pomiaru masy samego robota,
- mierzymy odległość jaką pokona robot w czasie 10s.
- stopniowo zwiększamy masę robota o określony ciężar (np. 100, 200, 400, 600, 800, ….. g)
- dla każdej masy dokonujemy pomiaru odległości, jaką pokona robot w czasie 10s.
- na podstawie uzyskanych wyników sporządzamy wykres zależności przebytej drogi od całkowitej masy robota.

Zadanie 2. Analiza uzyskanego wykresu

a) W jaki sposób masa robota wpływa na przejechany dystans?

b) Jaki jest maksymalny ciężar, jaki jest w stanie przetransportować robot?

Zadanie 3. *Oblicz, po jakim czasie robot osiąga maksymalną prędkość i porównaj ją z czasem, w którym wyścigówka osiąga maksymalną prędkość

Temat 10: WYŚCIGI ŻÓŁWI

Dzisiaj znów urządzimy sobie zawody, lecz tym razem po za czasem przejazdu liczyć się będzie siła naszych robotów. Budujemy pojazd zdolny przetransportować jak największy ciężar w jak najkrótszym czasie na daną odległość. Możemy skorzystać z poprzedniej instrukcji. Możemy ją przerobić lub zbudować coś według własnego pomysłu.

Jakie cechy powinien posiadać taki robot?

- 1 lub 2 przekładnie na moc,
- nacisk skierowany na koła (koła pod platformą nośną),

Zawody będą się składały z 6 oddzielnych wyścigów, w każdym z nich masa robota będzie stopniowo zwiększana od 0kg do 4kg. Wygra ten robot, który zdobędzie najwięcej punktów. Start odbywa się ze startu wspólnego.

Punktacja:

1 miejsce – 4 pkt,

2 miejsce – 3 pkt,

3 miejsce – 2 pkt,

4 miejsce – 1 pkt.

Jeżeli robot nie ruszy z miejsca 0 punktów.

Następnym krokiem jest napisanie oraz zgranie programu na kostkę. W zależności od zastosowanej liczby przekładni i od ustawienia silników musimy ustawić odpowiednią moc silnik (100 lub -100).

Poza wyścigami każda z drużyn dokonuje pomiarów indywidualnie dla własnego robota.

Zadanie 1. * Zadanie specjalne
Pudowa robota ze skrzynią biegów.

Zadanie 2. Pomiar czasu potrzebnego na przebycie określonej drogi w zależności od masy robota.
Poszczególne kroki:

- droga jaką ma pokonać robot zostaje ustalona na dystans od 4 do 6m,
- dokonanie pomiaru masy samego robota,
- pomiar czasu w jakim sam robot pokonuje wyznaczony dystans, porównanie tej wielkości w zależności od masy samego robota i rodzaju zastosowanej przekładni
- stopniowe zwiększanie masy robota o określoną masę (np.0,5; 1; 2; 3; 4),

- dla każdej masy przeprowadzenie oddzielnego wyścigu, dokonanie pomiaru czasu jaki zajmuje danemu robotowi pokonanie ustalonej odległości,
- przydzielenie punktów za poszczególne miejsca i ustalenie klasyfikacji końcowej.
- na podstawie uzyskanych wyników sporządzenie wykresu zależności czasu od całkowitej masy robota.

Zadanie 3. Analiza uzyskanego wykresu.

Jaki charakter ma funkcja? Czy jest to zależność liniowa?

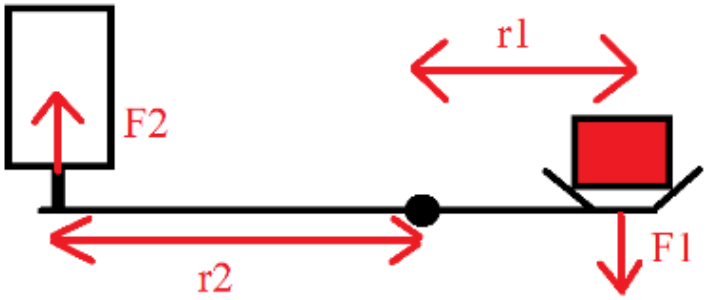
Temat 11: ROBOT DŹWIGNIA

Na dzisiejszych zajęciach zbudujemy robota wykorzystującego zasadę działania dźwigni dwustronnej.

1. Co to jest dźwignia? Rodzaje dźwigni

Dźwignia - jedna z maszyn prostych, których zadaniem jest uzyskanie działania większej siły przez zastosowanie siły mniejszej. Możemy wyróżnić dźwignie dwustronną i dźwignię jednostronna.

Dźwignia dwustronna zbudowana jest ze sztywnej belki zawieszonej na osi.



Wzór na przełożenie dźwigni (wzrost siły) jest przedstawiony po przez **rów. 1.**

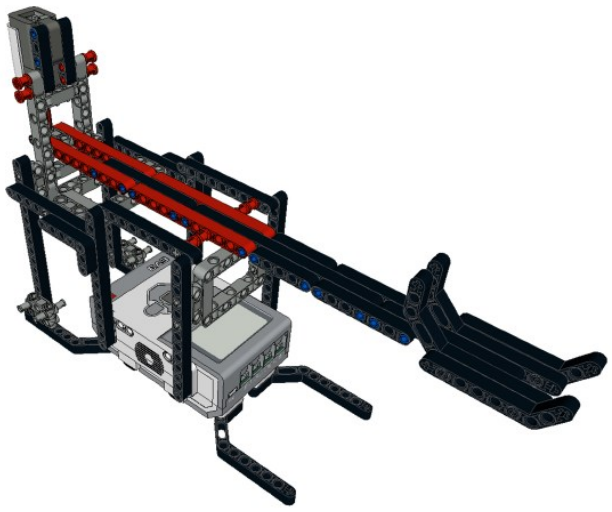
$$\frac{F_1}{F_2} = \frac{r_2}{r_1}$$

gdzie:

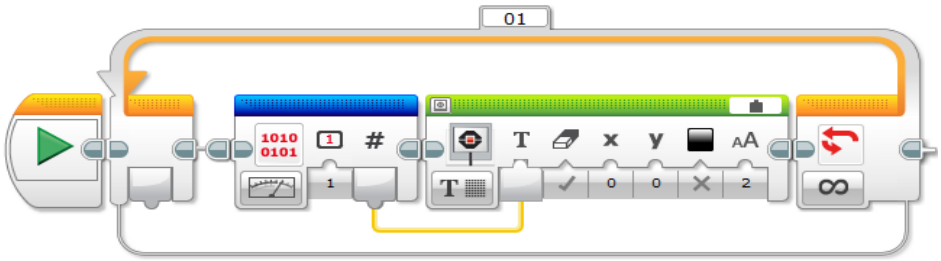
r_1 i r_2 – długości ramion dźwigni [m],

F_1 i F_2 – siły działające po przeciwnych osiach obrotu [N].

Po przypomnieniu sobie podstawowych informacji o maszynach prostych możemy przystąpić do budowy naszego robota według instrukcji „dzwignia.pdf".



napisanie programu, przy pomocy
będzie wyświetlał na wyświetlaczu
ıacisku wywieraną na dynamometr.
ıskorzystać z programu stworzonego dla
ɔracji tego czujnika:



Program: waga.ev3 Program

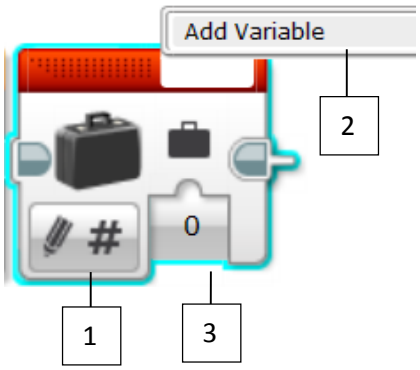
Czy nie będzie za proste rozwiązanie? Program ten służy do wyświetlania aktualnej siły wywieranej na czujnik. Ulega ona ciągłym zmianom. W celu uzyskania dokładnych wyników musieliśmy liczyć średnią z kilku pomiarów. Zastanówmy się czy robot nie mógłby przeprowadzić tej czynności za nas? Na przykład dokonać pomiaru 4 000 razy, obliczyć wartości średniej i wyświetlić ją na kostce? Nic trudnego, wystarczy, że poznamy dwa dodatkowe bloki w programie Lego Mindstorms i nauczymy się jego obsługiwać – blok zmiennych i blok operacji matematycznych.

2. Obsługa bloku zmiennych (Variable Block)

Blok ten umożliwia odczyt i zapis zmiennych w programie, oraz tworzenie nowych zmiennych i nazywanie ich. Zmienne umieszczane są w pamięci klocka EV3, który ma możliwość przechowywania danych. Każda zmienna ma swój typ i nazwę.

TRYBY DZIAŁANIA

Dodawanie nowej zmiennej i jej wartości

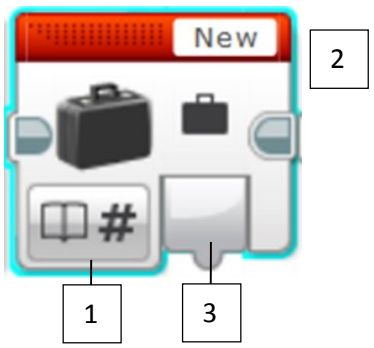


1 – Wybór trybu – zapis (write) – i typu zmiennej

2 – Dodawanie zmiennej/ wybór zmiennej, wg typu określonego w trybie działania.

3 – Wprowadzenie wartość dla zmiennej.

Odczyt ze zmiennej

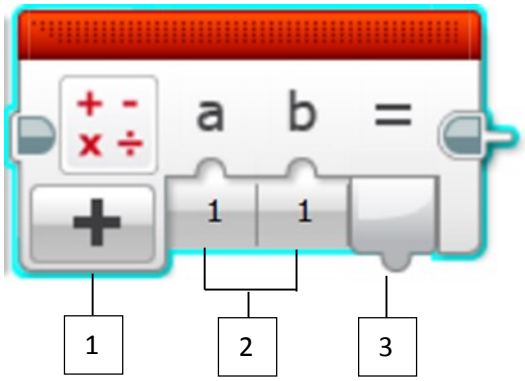


1 – Wybór trybu – odczyt (read) – oraz typu zmiennej

2 – Wybór zmiennej w prawym górnym rogu bloku

3 – Podłączenie zmiennej do bloku, który będzie ją wykorzystywał.

3. Obsługa bloku operacji matematycznych (MATH OPERATIONS BLOCK)

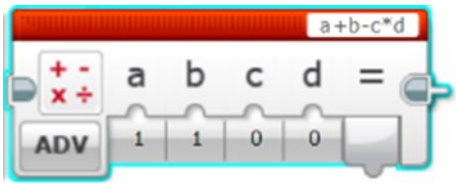


1 – Wybór operacji matematycznej:

- Dodawanie
- Odejmowanie
- Mnożenie
- Dzielenie
- Wartość absolutna
- Pierwiastkowanie
- Potęgowanie

Opcje zaawansowane

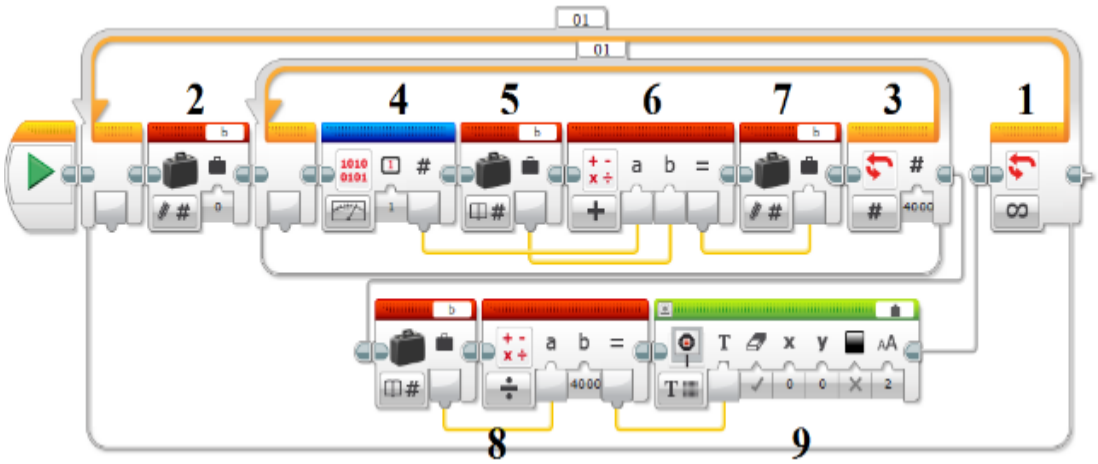
Wybór równania (z możliwością wpisania własnego)



2 – Wprowadzanie zmiennych

3 – Wynik operacji

4. Jak napisać program krok po kroku:



Program: dzwignia.ev3 Program

1. Cały program ma działać cały czas, więc umieszczamy go w pętli działającej nieskończoną ilość razy.
2. Definiujemy zmienną b i przyporządkowujemy jej wartość 0.
3. Wewnątrz pętli tworzymy drugą pętlę, która będzie działała określoną ilość razy np. 4000 w celu sumowania poszczególnych odczytów czujnika.
4. Wewnątrz drugiej pętli umieszczamy bloczek sczytujący bezpośrednio wartość z czujnika.

5. Umieszczamy bloczek wczytujący zmienna b.
6. Zmienną b i wartość sczytaną z czujnika sumujemy przy pomocy Math Operations Block.
7. Otrzymany wynik zostaje zapisany, jako zmienna b, operacja ta zostaje powtórzona 4000 razy.
8. Ostatecznie b stanowi wynik sumowania 4000 odczytów z czujnika, w celu obliczenia wartości średniej musimy podzielić otrzymaną wartość przez 4000 za pomocy Math Operations Block.
9. Ostateczny wynik będzie widoczny na wyświetlaczu kostki tak długo aż nie dokona się następne obliczenie wartości średniej.

Jeżeli chcemy zwiększyć ilość pomiarów należy zmienić zarówno liczbę powtórzeń wewnętrznej pętli oraz liczbę, przez którą dzielimy wartość b.

Zadanie 1. Określenie prawdziwości rów. 1.

Przebieg:

- odczytanie wartości z kostki EV3,
- zamiana tej wartości na siłę nacisku na podstawie wykresu kalibracji stworzonego na 7 zajęciach,
- na podstawie rów. 1 obliczenie siły nacisku wywieranej na krótsze ramię,
- porównanie otrzymanego wyniku z rzeczywistą masą przedmiotu zmierzonego przy pomocy wagi,
- powtórzenie pomiarów dla innych obciążników,

Pytania:

- Czy wyniki są zgodne z rzeczywistą masą przedmiotów?
- Dyskusja na temat niepewności pomiarowych

Zadanie 2. * Napisanie programu, który zamiast wartości liczbowych od 0 do 4080 będzie wyświetlał na kostce wartość siły nacisku w niutonach lub w gramach. Wykorzystanie rów. 1.

Temat 12: III ZASADA DYNAMIKI NEWTONA, A ROBOT STRZELAJĄCY KULKAMI

1. III ZASAD DYNAMIKI NEWTONA

Odziaływania ciał są zawsze wzajemne. Siły oddziaływania dwóch ciał mają takie same wartości, taki sam kierunek, przeciwne zwroty i różne punkty przyłożenia (każda działa na inne ciało).

2. ODRZUT

Zjawisko powstawania siły zwanej siłą odrzutu. Powstaje w wyniku oddziaływania jednego ciała na drugie. Dane ciało działając siłą na inne ciało nadaje mu prędkość (rzuca to ciało). Powstawanie tej siły wynika z III zasady dynamiki Newtona. Obie siły mają takie same wartości, taki sam kierunek, przeciwne zwroty i różne punkty przyłożenia

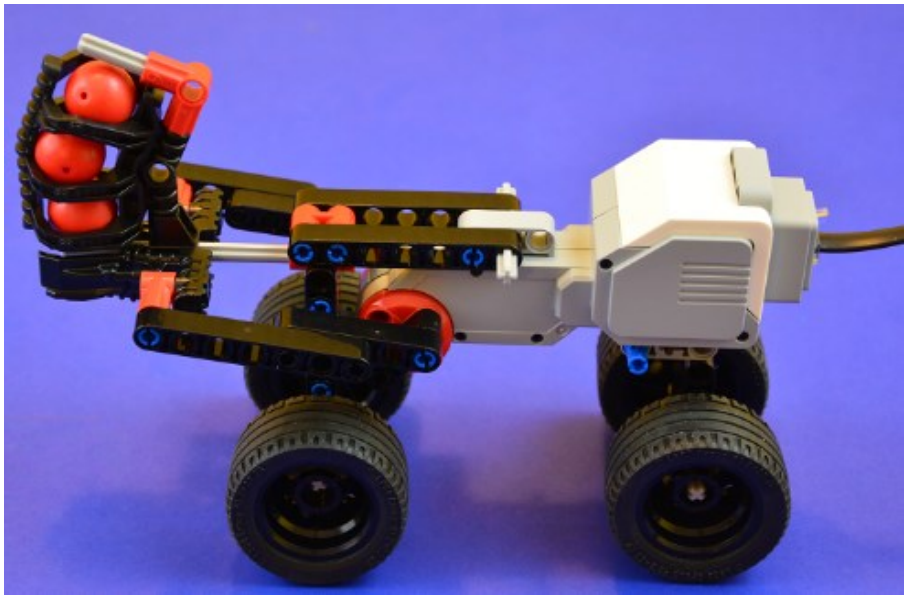
Zjawisko odrzutu jest powszechnie spotykane w przyrodzie i wykorzystywane w technice:

- silnik odrzutowy,
- odrzut broni palnej,
- odrzut jądrowy,
- poruszanie się niektórych zwierząt np. meduza.

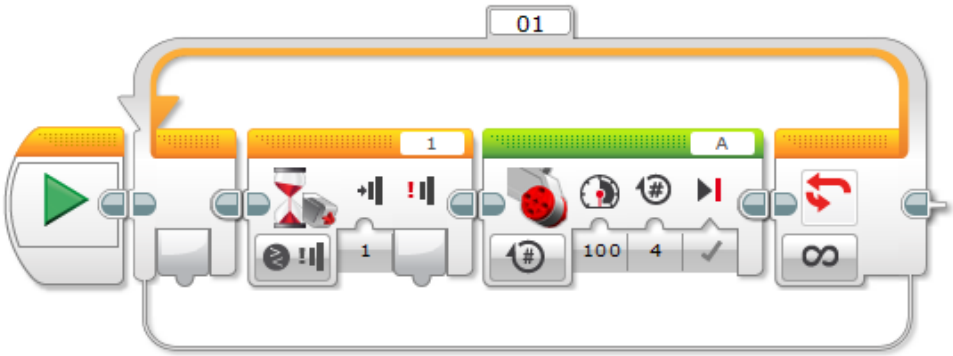
3. CO TO MA WSPÓLNEGO Z ROBOTYKĄ?

Na zajęciach zbudujemy robota, który będzie strzelał kulkami.. Zgodnie z III zasadą dynamiki Newtona wystrzeliwana kulka będzie powodować odrzut robota strzelającego. Zobaczymy czy powstająca siła jest na tyle duża, aby spowodować ruch pojazdu w przeciwną stronę.

Nasza konstrukcja będzie posiadała jeden silnik umieszczony na dwóch swobodnie obracających się osiach. Kostki EV3 nie mocujemy do naszego robota w celu minimalizacji masy pojazdu. Korzystamy z instrukcji „bezwladnosc.pdf".

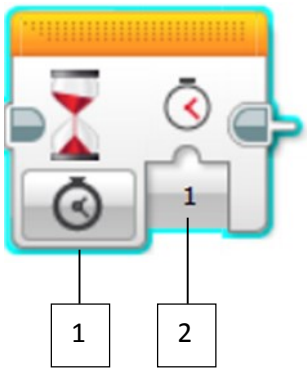


Następnym krokiem w dzisiejszych zajęciach jest napisanie programu, przy pomocy którego nasz robot będzie strzelał. Sygnałem powodujący rozpoczęcie strzelania będzie naciśnięcie czujnika dotyku podłączonego do portu 1. Silnik podłączamy do portu A. Czas działania silnika ustawiamy na cztery obroty z maksymalną pocą. Dokładny opis bloku oczekiwania (wait) znajduje się poniżej zdjęcia programu.



Program: bezwladnosc.ev3 Program

4. Obsługa bloku oczekiwania (Wait Block)



1 – Wybór trybu działania

2 – Sygnały wejść, zależne od wybranego trybu działania

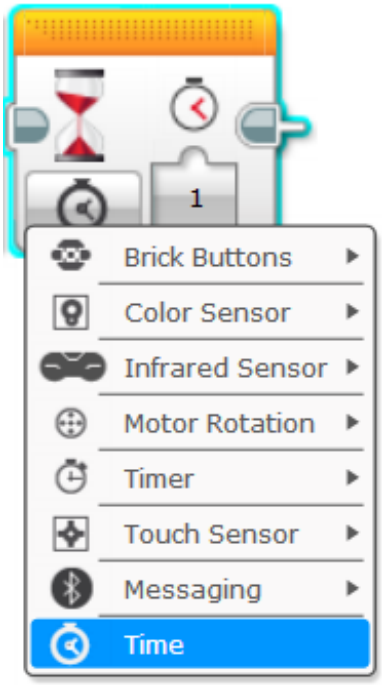
TRYBY DZIAŁANIA

Time – umożliwia zawieszenie realizacji programu na określony w programie czas.

Czas oczekiwania (Seconds) podaje się w sekundach. Może przyjmować wartości po przecinku.

Czas odliczany jest od momentu uruchomienia bloku oczekiwania.

Tryby porównujące dane z czujników



Każdy typ czujnika posiada jeden lub więcej trybów działania. Tryby te wykorzystują odczytane dane z czujników i oczekują, by osiągnąć wymagany próg lub wartość odgórnie wprowadzaną.

Typ porównania wybiera się z rozwijanej listy.

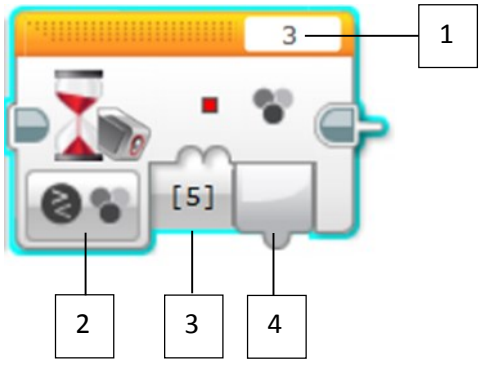
Wartość progowa podawana jest w postaci numerycznej i porównywana jest z danymi otrzymywanymi z czujnika.

Blok wstrzymuje działanie programu do czasu, aż porównanie wartości progowej do danych z czujnika będzie spełnione.

Jeśli warunek będzie spełniony od początku wejścia programu w blok oczekiwania, natychmiast wykonywany jest kolejny krok.

Na wyjściu bloku otrzymuje się końcową wartość czujnika.

Color Sensor – Compare – Color – oczekiwanie na wykrycie jednego lub większej liczby określonych barw.

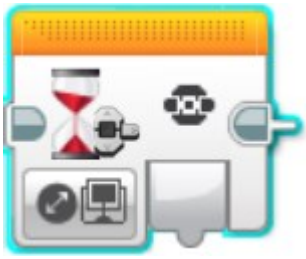


1 – Wybór portu
2 – Wybór trybu
3 – Sygnały wejściowe
4 – Sygnał wyjścia

Touch Sensor – Compare – State – oczekiwanie, aż czujnik dotyku zostanie naciśnięty, odciśnięty lub wciskany.

TRYBY OCZEKUJĄCE NA ZMIANĘ DANYCH Z CZUJNIKÓW

Każdy z czujników w bloku oczekiwania może pracować w kilku trybach (Change modes). Tryb ten będzie nieustannie sczytywać dane z czujnika i oczekiwać na nie, by zmienić ją na inną wartość, lub zamienić na wartość podaną przez programistę. Przykładowe możliwości ustawienia:



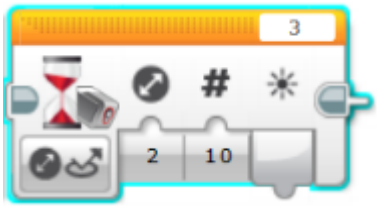
brick buttons – change – brick buttons – oczekuje na dowolną zmianę przycisków na klocku EV3

Na wyjście wysyłane jest ID przycisku, który został naciśnięty lub odpuszczony.

Color Sensor – Change – Color

Tryb oczekujący na zmianę numer koloru wykrytego przez czujnik koloru (0-7). Na wyjściu przekazywany jest wykryty kolor.

Color Sensor – Change – Reflected Light Intensity



Tryb oczekujący na odbicie intensywności światła by zmienić pewną wielkość.

Zadanie 1. **Obserwacja wielkości odrzutu robota w zależności od rodzaju podłoża, posiadania ogumienia, naprężenia kabla łączącego robota z kostką EV3.**

Zadanie 2. **Dyskusja na temat czynników wpływających na wielkość odrzutu oraz związku robota z III zasadą dynamiki Newtona i zasadą zachowania pędu. Wymień przykłady z najbliższego otoczenia i filmów.**

Zadanie 3. *** Obliczenie prędkości, z jaką zostaje wystrzelona kulka.**

Prędkość tą można obliczyć na dwa sposoby:

a) na podstawie nagrania przy pomocy aparatu fotograficznego lub kamery.

$$V = \frac{s}{t}$$

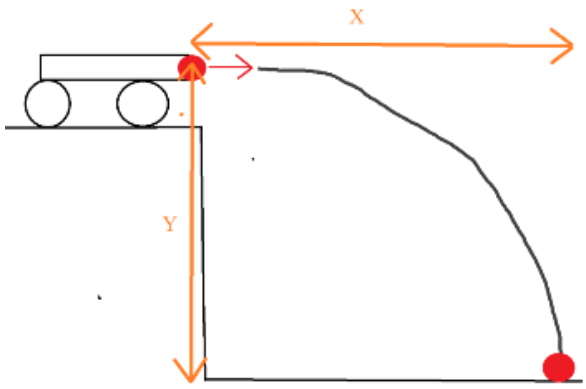
gdzie:

v – prędkość kulki [m/s],

s – pozioma odległość przebyta w czasie t [m],

t – czas pomiędzy zapisem poszczególnych klatek nagrania[s].

b) na podstawie odległości, jaką przeleciała kulka i wysokości, z jakiej była wystrzeliwana. Ruch kulki traktujemy jak rzut poziomy.



Czas spadku swobodnego (t):

$$t = \sqrt{\frac{2Y}{g}}$$

g = 10 m/s^2

Y – wysokość [m]

X -odległość, na którą doleciała kulka

Odległość: X = V ·t

V – prędkość kulki

Czas spadku swobodnego i poziomego lotu kulki są identyczne $V = \frac{x}{t} = x\sqrt{\frac{g}{2Y}}$

Zadanie 4. *** Z jaką prędkością poruszałaby się wyrzutnia, gdyby nie było żadnych oporów ruchu?**

Można to obliczyć z zasady zachowania pędów, pęd jest związany z prędkością i masą. Pęd kulki i wyrzutni są ze sobą związane relacją:

$$v_1 m_1 = v_2 m_2$$

gdzie:

m_1 - masa kulki

m_2 - masa wyrzutni

v_1 – prędkość kulki

v_2 – prędkość wyrzutni

Temat 13: PRZYSPIESZENIE ZIEMSKIE

Na dzisiejszych zajęciach spróbujemy przeprowadzić doświadczenie, przy pomocy którego wyznaczymy przyspieszenie ziemskie. Zastanówmy się jak wyznaczyć przyspieszenie ziemskie korzystając z klocków Lego?

1. Przyspieszenie ziemskie

Przyspieszenie ziemskie wyznaczone przy pomocy bardzo dużej precyzji wynosi:

$$g = 9{,}80665\ m/s^2$$

Takiej wartości używamy do obliczeń fizycznych. Zobaczymy jak dokładne są klocki Lego. Przyspieszenie ziemskie najłatwiej obliczyć ze spadku swobodnego. Wzór na spadek swobodny ma postać:

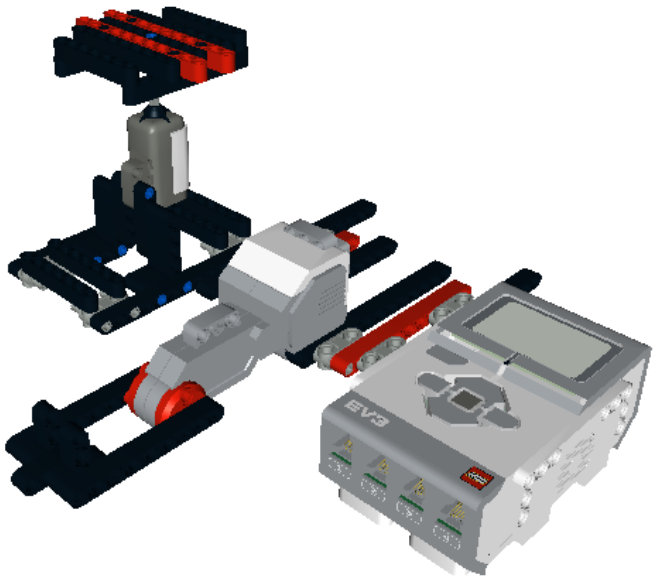
$$s = \frac{gt^2}{2}$$

z czego:

$$g = \frac{t^2}{2s}$$

W celu wyznaczenia przyspieszenia „wystarczy" skonstruować przyrząd, który będzie mierzył czas spadku przedmiotu. Po zmierzeniu odległości, jaką przebyło ciało możemy wyznaczyć przyspieszenie ziemskie.

Naszym spadającym ciałem będzie kulka z zestawu Lego Mindstorms EV3. Do zarejestrowania upadku kuliki użyjemy bardzo czułego czujnika nacisku – dynamometru. Robota budujemy według instrukcji „przyspieszenie.pdf".

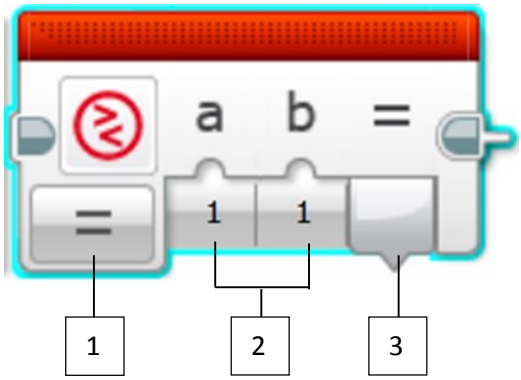


Do podłączenia silnika i czujnika należy użyć jak najdłuższych przewodów. Zapewni to zwiększenie drogi, jaką będzie pokonywać kulka, co wpłynie na dokładniejsze wyniki.

Kulka zwalniana jest przy pomocy silniczka. Zwolnienie kulki rozpoczyna pomiar czasu. W momencie przekroczenia granicznej siły (ustawienie dynamometru) pomiar czasu zostaje zatrzymany i wynik wyświetlony na ekraniku kostki. Na podstawie czasu i wysokości spadku można obliczyć przyspieszenie ziemskie.

Do zaprogramowania robota użyjemy dotąd niestosowanego bloku porównania. Jego zadaniem jest porównanie dwóch wartości ze sobą. Wynikiem porównania może być prawda lub fałsz. Użyjemy go w celu określenia momentu, w którym kulka uderza w platformę.

2. Obsługa bloku porównania (Compare Block)



1 – Wybór trybu porównania

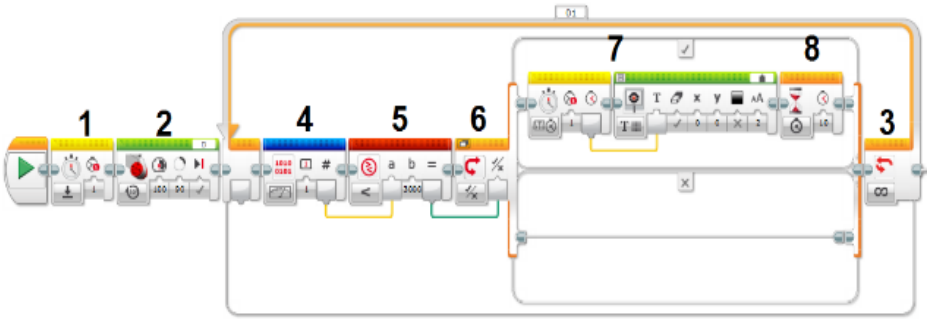
2 – Wartości porównywane

3 – Wynik operacji w postaci logicznej True/False

TRYBY DZIAŁANIA

TRYB		WEJŚCIA	WYNIK PRAWDZIWY, GDY
=	RÓWNE (EQUAL TO)	A, B	A = B
≠	RÓŻNE (NOT EQUAL TO)	A, B	A ≠ B
>	WIĘKSZE NIŻ (GREATER THAN)	A, B	A > B
<	MNIEJSZE NIŻ (LESS THEN)	A, B	A < B
≥	WIĘKSZE LUB RÓWNE (GREATER THAN OR EQUAL TO)	A, B	A ≥ B
≤	MNIEJSZE LUB RÓWNE (LESS THEN OR EQUAL TO)	A, B	A ≤ B

Gotowy program powinna wyglądać tak:



Program: przyspieszenie.ev3 Program

3. Jak napisać program krok po kroku

1. Resetowanie zegara – w celu pomiaru czasu w chwili działania silnika.
2. Obrót silnika o kąt 90 stopni z maksymalną mocą.
3. Następnie dodajemy blok pętli, który będzie powtarzany w nieskończoność (do momentu uderzenia kulki w platformę).
4. Wewnątrz pętli umieszczamy blok sczytujący wartość numeryczną z dynamometru.
5. Wartość ta jest porównywana z określoną wartością progową przy pomocy bloku porównania.
6. Następnie wstawiamy blok switch, który sprawdza czy została osiągnięta wartość progowa. Jeżeli nie to nić się nie dzieje i pętla jest znowu powtarzana (pętla jest powtarzana kilka tysięcy razy w ciągu 1 sekundy).
7. Gdy wartość progowa jest osiągnięta prawie natychmiast na wyświetlaczu zostaje wyświetlona wartość czasu zarejestrowania przekroczenia wartości progowej.
8. W celu chwilowego zatrzymania programu jest ustawiony blok oczekiwania umożliwiający sczytanie wyniku.

Zadanie 1. **Dokonaj pomiaru przyspieszenia ziemskiego dla różnych odległości silnika od platformy. Porównanie wyników. Określenie czynników wpływających na dokładność pomiarów.**

Zadanie 2. *** Napisanie programu, który będzie zaczynał pomiar czasu w chwili, w której to my sami będziemy zrzucać kulkę z większej wysokości. W chwili spuszczenia kulki będziemy równocześnie przyciskać czujnik dotyku, co będzie powodować rozpoczęcie pomiaru czasu. Czy wartość przyspieszenia różni się od wcześniej wyznaczonej? W jaki sposób? Co na to wpłynęło?**

Temat 14: ROBOT PCHAJĄCY

Na zajęciach zbudujemy robota, który będzie pchał jakiś ciężar, ale w tym wypadku nie będziemy stosować żadnej przekładni. Na początku zastanówmy się, od czego będzie zależało, jaki maksymalny ciężar przepchnie robot?

Skoro nie stosujemy przekładni maksymalny ciężar będzie zależał od dwóch czynników:

- masy samego robota,
- zastosowanego środka napędowego (opony, gąsienice brak ogumienia).

Te oba czynniki mają wpływ na parametr fizyczny, który określamy mianem tarcia.

Tarcie

Tarcie to całość zjawisk fizycznych towarzyszących przemieszczaniu się względem siebie dwóch ciał fizycznych lub elementów tego samego ciała i powodujących rozpraszanie energii podczas ruchu.

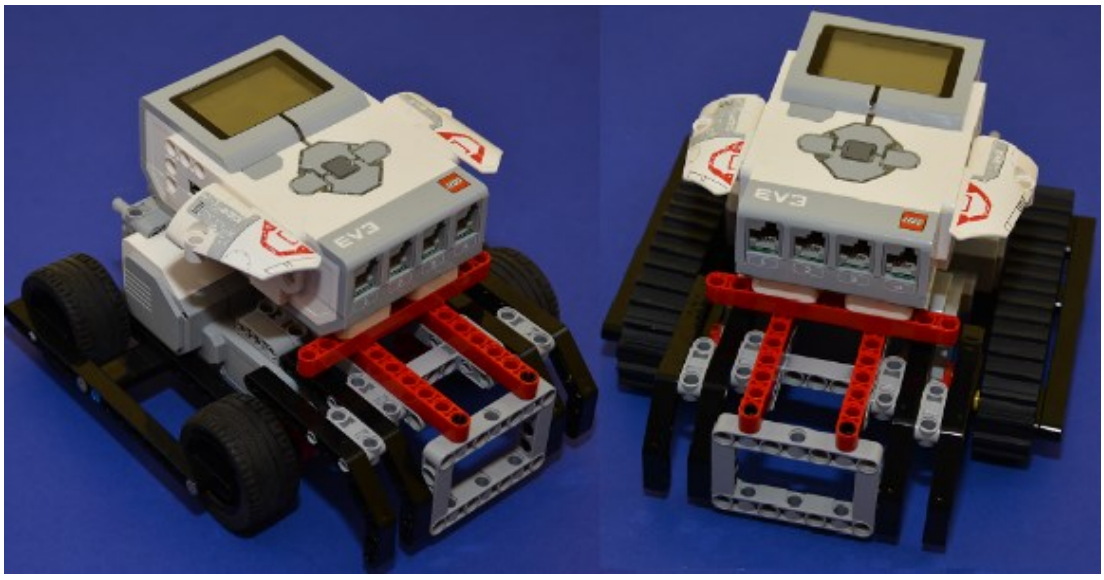
Siłę tarcia można opisać przy pomocy wzoru:

$$T = \mu N$$

gdzie:

μ – współczynnik tarcia zależny od rodzaju powierzchni stykających się ciał,

N – siła nacisku prostopadła do powierzchni styku ciał.

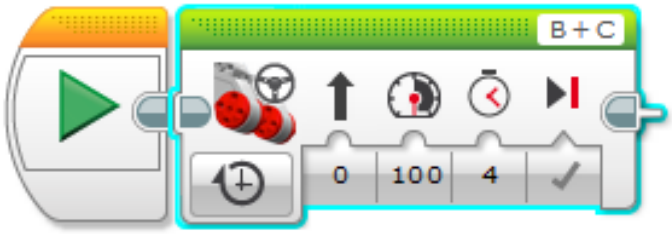


Skonstruowany przez nas pojazd pchający będzie miał w przybliżeniu stałą masę, przez co siła nacisku N, jaką będzie wywierał na podłożę będzie taka sama. Siła tarcia będzie zależała od współczynnika tarcia pomiędzy robotem, a podłożem.

Nasz pojazd budujemy według instrukcji „spychacz.pdf".

Konstrukcja umożliwia w łatwy sposób zamianę gąsienic na opony, lub całkowite ich usunięcie.

Program do sterowania robotem może wyglądać tak samo jak dla wyścigówki z dwiema przekładniami:



Program: wyscigowka.ev3 Program2

Zadanie 3. **Sprawdzenie, jaki maksymalny ciężar jest w stanie przepchnąć robot przy zastosowaniu opon, ich braku lub użyciu gąsienic. W którym z tych wypadków tarcie o podłoże będzie największe? Jak zwiększyć tarcie kół o podłoże?**

Zadanie 4. *** Co się stanie jeżeli zmienimy podłoże?**

Temat 15: RÓWNIA POCHYŁA

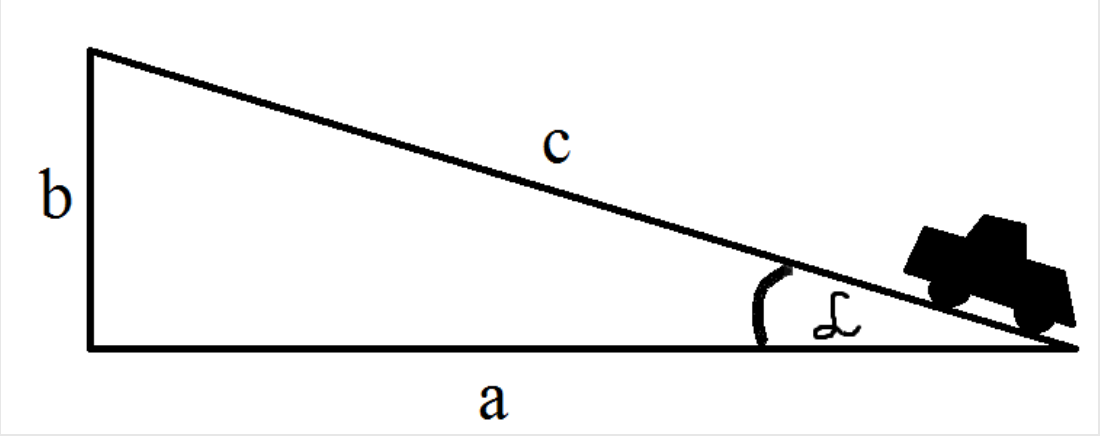
Pamiętając informacje o tarciu oraz stosowaniu przekładni różnego typu w dniu dzisiejszym zostaną przeprowadzone zawody.

Naszym zadaniem będzie zbudowanie robota, który będzie w stanie podjechać pod jak najbardziej stromą równie pochyłom. Wraz ze wzrostem kąta nachylenia będzie malał nacisk, co będzie skutkowało spadkiem siły tarcia. Robota można zbudować według instrukcji lub według własnego pomysłu. Od nas zależy czy zastosujemy przekładnię na moc, gąsienicę czy opony.

Wygra ten robot który będzie w stanie wjechać jak najwyżej przy zastosowaniu tego samego podłoża (deski o długości 1,5m). Na budowę robota mamy 70 min. W tym czasie możemy dokonywać nielimitowaną ilość prób podjazdu i modyfikacji konstrukcji.

Zadanie 1. W jaki sposób zmiana podłoża wpływa na maksymalny kąt nachylenia równi pochyłej, po której jest wstanie podjechać robot (zastosowanie 1,5m deski wyszlifowanej lub pokrytej lakierem).

Zadanie 2. * Czy zwiększenie masy robota ułatwi mu w każdym przypadku podjazd?



$$F_t = \mu mg \cos \alpha$$

μ – współczynnik tarcia

W tym przypadku mamy do czynienia z tarciem tocznym umożliwiającym podjazd robota pod górę. Jest ono proporcjonalne do wartości siły nacisku.

$$F_t \sim mg \cos \alpha$$

Składowa siły grawitacji działająca wzdłuż osi podjazdu:

$$F_g = mg \sin \alpha$$

Dla ciała nieruchomego (dla ruchomego należałoby jeszcze uwzględnić tarcie dynamiczne) maksymalny kąt, dla którego równoległe składowe siły grawitacji i tarcia się równoważą wynosi:

$$\mu mg \cos \alpha = mg \sin \alpha$$

$$\mu \cos \alpha = \sin \alpha$$

$$\sin \alpha = b/c \; ; \cos \alpha = a/c$$

$$b = \mu a$$

Tarcie nie zależy od masy robota. Jest to łatwy sposób wyznaczania współczynnika tarcia statycznego.

Zadanie 3. *** Czy zwiększenie masy robota ułatwi mu w każdym przypadku podjazd? Zwiększamy masę robota o 100% i sprawdzamy jak wpłynie to na maksymalny kat, pod który jest w stanie podjechać robot.**

Zadanie 4. *** Zastosuj inną powierzchnię zamiast deski (np. polakierowana deska, deska obita materiałem itp.).**

Jak rodzaj materiału wpływa na maksymalny kąt nachylenia, pod który jest w stanie podjechać robot?

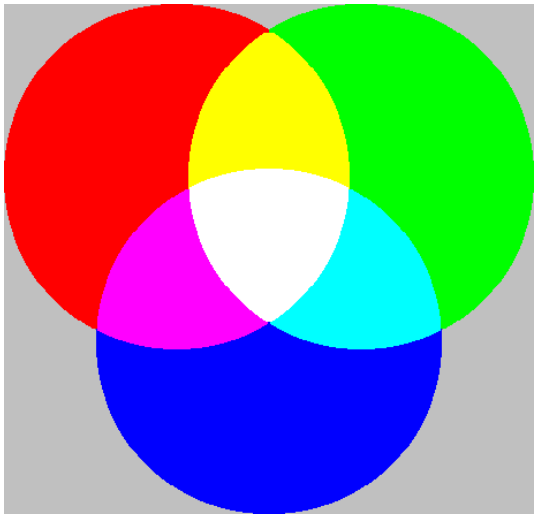
Temat 16: MIESZANIE KOLORÓW

Światło białe jest w rzeczywistości mieszaniną różnych kolorów. Na dzisiejszych zajęciach zbudujemy robota, który umożliwi nam mieszanie kolorów. Na początek powtórzymy sobie wiadomości z przedszkola na temat mieszania farb. Jeżeli połączymy ze sobą dwie farby to możemy otrzymać całkiem inny kolor, niż kolory składowe.

Możemy wyróżnić dwa różne sposoby mieszania się kolorów.

ADDYTYWNE MIESZANIE SIĘ BARW, czyli mieszanie świateł o różnych kolorach w wyniku czego powstaje zawsze kolor jaśniejszy

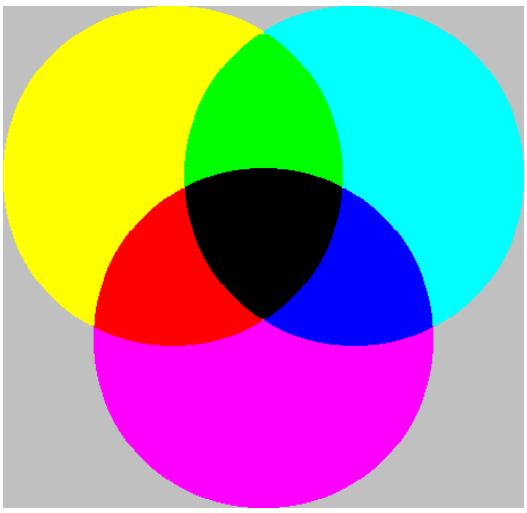
Trzy podstawowe kolory widma światła widzialnego to czerwony, niebieski i żółty. Po ich zmieszaniu otrzymujemy kolor biały. Z połączenia poszczególnych kolorów składowych ze sobą otrzymujemy barwy dopełniające. W wyniku ich połączenia powstanie kolor biały.



Ten typ mieszania się barw jest wykorzystywany głównie w wyświetlaczach. Kolor czarny jest wynikiem braku emisji światła.

SUBTRAKTYWNE MIESZANIE SIĘ BARW – mieszanie farb o różnych kolorach. Wynik mieszania będzie zawsze ciemniejszy niż kolory składowe.

W przypadku subtraktywnego mieszania się barw kolorami podstawowymi będą żółty, błękitny i fioletowo-czerwony.

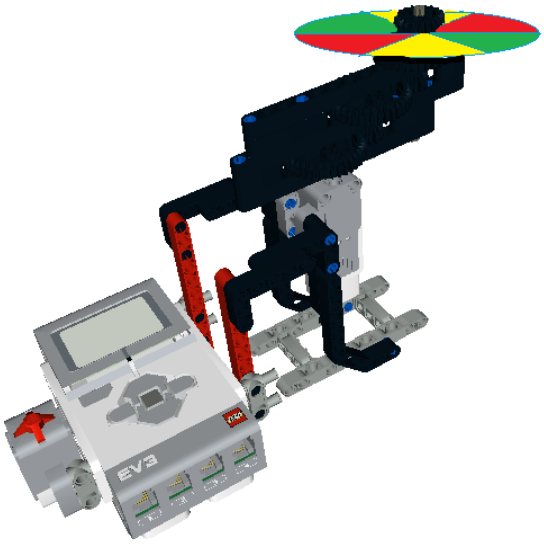


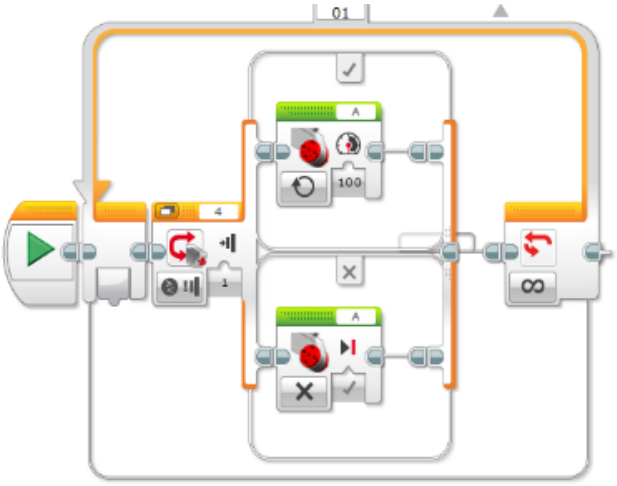
Ten typ mieszania się barw jest wykorzystywany głównie do mieszania pigmentów. Znalazł zastosowanie w drukarkach, w których przy pomocy trzech barwników można otrzymać dowolny kolor. W wyniku połączenia wszystkich trzech powstaje kolor czarny.

Na dzisiejszych zajęciach nie zbudujemy robota, który będzie dla nas mieszał farby, lecz zbudujemy robota, który będzie mieszał barwy, ale nie w sposób dosłowny. Mieszanie będzie wywołane niedoskonałością

aparatu wzrokowego człowieka. Wykorzystamy ten sam mechanizm, który jest wykorzystywany przy kręceniu filmów. Przy zastosowaniu 16 klatek/s w filmach powstaje już wrażenie płynności obrazu. Zjawisko to wykorzystamy do budowy robota z szybko obracającą się tarczą składającą się z różnych kolorów. Zastosujemy także podwójną przekładnię na prędkość oraz czujnik dotyku, który będzie uruchamiał silnik.

Naszego robota budujemy według instrukcji "baczek.pdf".





Program będzie składał się pętli oraz switcha sprawdzającego czy czujnik dotyku jest wciśnięty. Jeżeli tak silnik będzie się obracał, jeżeli nie to się zatrzyma.

Program: baczek.ev3 Program

Zadanie 1. **Z jakim rodzajem mieszania się barw mamy w tym przypadku do czynienia?**

Zadanie 2. **Jaka jest minimalna moc silnika, dla której nie będą rozróżniane poszczególne kolory?**

Zadanie 3. *** Napisz program, który umożliwi zmianę mocy silnika (szybkości obracania się tarczy) przy pomocy strzałek na kosce.**

Temat 17: EKO-ROBOT

Na dzisiejszych zajęciach nie użyjemy kostki EV3. Pomówimy trochę o formach energii i zamianie jednej formy w drugą. Zbudujemy robota napędzanego siłą naszych mięśni, zbadamy sprawność silników oraz jeżeli wystarczy na to czasu urządzimy wyścigi.

Rodzaje energii:

- energia mechaniczna (suma energii potencjalnej i energii kinetycznej),
- energia chemiczna,
- energia cieplna,
- energia elektryczna,
- energia jądrowa,
- energia słoneczna,

Dzięki prawom fizyki możliwa jest zamiana jednej formy energii w drugą. Np. w turbinach elektrowni energia mechaniczna jest zamieniana na energię elektryczną. Natomiast w silniku elektrycznym energia elektryczna jest zamieniana na energię mechaniczną.

Silniki EV3 mogą produkować prąd elektryczny na zasadzie działania prądnicy, lub zamieniać energię elektryczną na mechaniczną (obracać się na skutek przyłożonego napięcia). Na dzisiejszych zajęciach zbudujemy układ, który będzie zamieniał energię mechaniczną na energię elektryczną i z powrotem na energię mechaniczną.

Sprawność – jest to stosunek wielkości energii wydawanej przez układ do wielkości energii pobieranej przez ten sam układ. Zamiany pomiędzy poszczególnymi typami energii nigdy nieodbywaną się bez strat energii. Energia oddana jest mniejsza niż energia pobrana.

$$\eta = \frac{E_{wy}}{E_{we}} \cdot 100\%$$

gdzie:

η – sprawność [%]

E_{we} – energia pobrana przez układ

E_{wy} – energia oddana przez układ

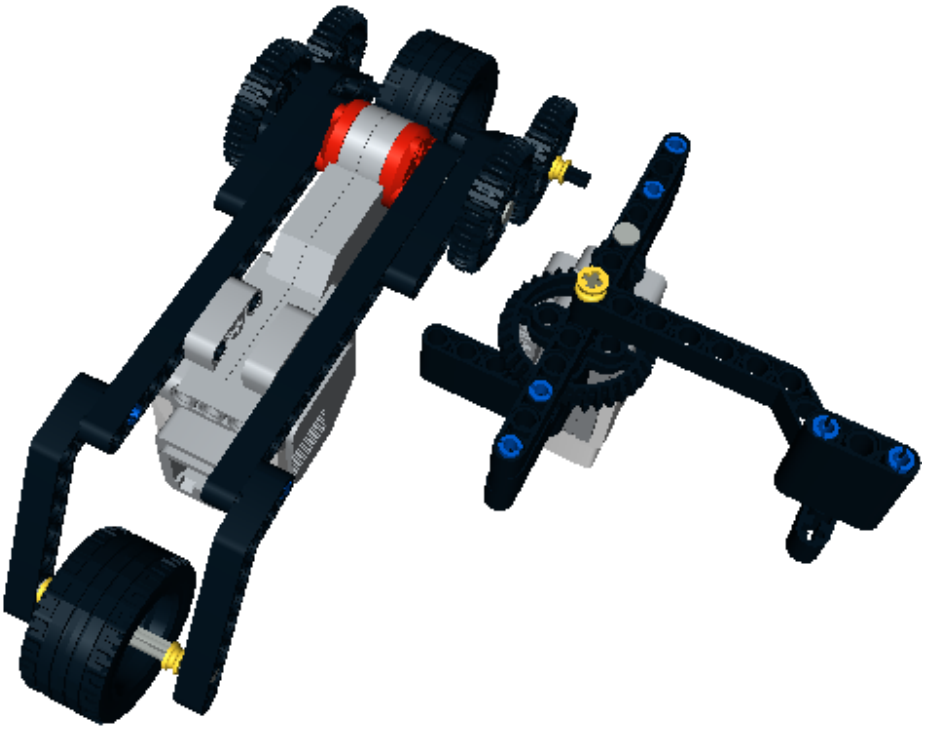
W naszym wypadku miarą energii wejściowej i wyjściowej będzie liczba obrotów silnika.

Zadanie 1. Zbadaj sprawność dwóch połączonych ze sobą silników.

Łączymy dwa silniki przy pomocy przewodu i obracamy pierwszym silnikiem. Liczy ile razy obróci się drugi silnik – np. pierwszy silnik obracamy 7 razy w tym samym czasie drugi silnik obrócił się 6 razy. Czy szybkość obracania pierwszym silnikiem ma wpływ na liczbę obrotów?

Znając efektywność pracy silnika możemy przystąpić do budowy naszego ekologicznego pojazdu. Przy budowie robota możemy skorzystać z instrukcji "ekorobot.pdf".

Zastosowano tutaj dwa silniki połączone długim przewodem oraz przekładnię zwiększającą liczbę obrotów silnikiem napędzającym.



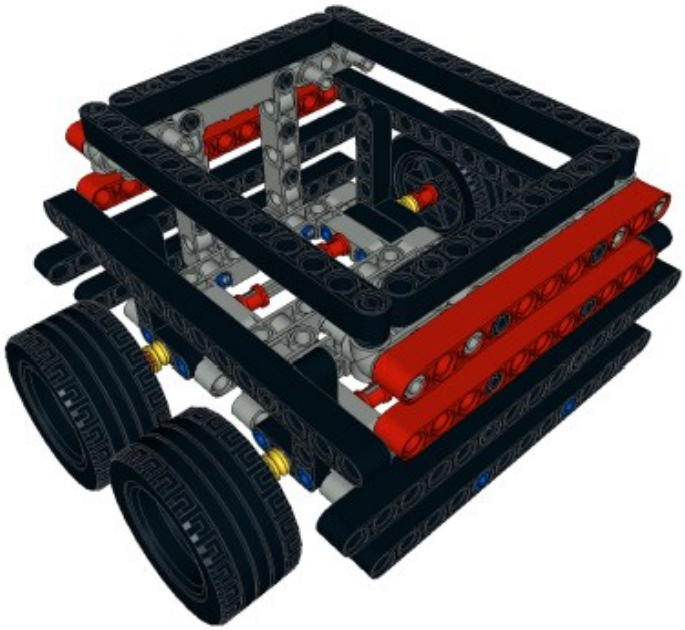
Zadanie 2. * Jakie maksymalne napięcie jest w stanie wytworzyć silnik z zestawu EV3 oraz rotacyjny? Czy istnieje jakaś górna graniczna wartość?

Temat 18: ZDERZENIA

Na dzisiejszych zajęciach przypomnimy sobie wiedzę na temat III zasady dynamiki Newtona. Zajmiemy się badaniem zderzeń ciał.

Oddziaływania ciał są zawsze wzajemne. Siły wzajemnego oddziaływania dwóch ciał mają takie same wartości, taki sam kierunek, przeciwne zwroty i różne punkty przyłożenia (każda działa na inne ciało).

W tym celu musimy skonstruować pojazdy, które będą się zderzać. W celu wyeliminowania wpływu hamowania silników po zderzeniu zbudujmy dwa „wózki", które będą w siebie uderzać. Można skorzystać z instrukcji "wozek.pdf". Zastosowanie takiej konstrukcji umożliwia w bardzo prosty i szybki sposób zmianę masy wózka.



Do nadania prędkości możemy użyć jednej z wcześniejszych konstrukcji lub równi pochyłej.

1. W JAKI SPOSÓB ZMIENIĆ PRĘDKOŚĆ WÓZKA?

Najprościej po przez zastosowanie równi pochyłej i przyjęciu założenia, że całość energii potencjalnej jest zamieniana na energię kinetyczną.

$$mgh = \frac{mv^2}{2}$$

$$v = \sqrt{2gh}$$

Zwiększając wysokość h równi pochyłej czterokrotnie zwiększymy prędkość wózka dwukrotnie. Zmiana masy nie wpływa na uzyskaną prędkość wózka.

Do analizy możemy użyć także poza programowego wzoru na zasadę zachowania pędu:

$$mv = m_1v_1 + m_2v_2 = \text{const},$$

całkowity moment pędów układu jest zachowany.

2. Typy zderzeń

a) sprężyste, np.

- uderzenie piłki w ścianę
- zderzenie kul bilardowych

b) niesprężyste, np.

- uderzenie kropli wody w szybę
- zderzenie doniczki z ziemią
- uderzenie samochodu w drzewo.

W obydwu przypadkach spełniona jest zasad zachowania pędu. Różnica polega na tym, że w przypadku zderzeń sprężystych energia jest zachowana, zaś w przypadku zderzeń niesprężystych nie jest.

Zadanie 1. **Jaki typ zderzeń obserwujemy w przypadku pustych wózków?**

Zadanie 2. **W jaki sposób będą zachowywały się wózki, jeżeli masa jednego z nich będzie dwukrotnie większa?**

Zadanie 3. **W jaki sposób będą zachowywały się wózki po zderzeniu, jeżeli prędkość jednego z nich będzie dwa razy większa niż drugiego?**

Zadanie 4. **W jaki sposób będą zachowywały się wózki po zderzeniu, jeżeli prędkość jednego z nich będzie dwa razy większa niż drugiego, a wolniejszy wózek będzie dwa razy lżejszy?**

Temat 19: LINEFOLLOWER

Niedawno mówiliśmy o kolorach. Ciekawe czy pamiętacie, w jaki sposób robot może rozróżniać kolory? Używa do tego specjalnego czujnika, który jest opisany w pierwszym rozdziale tego podręcznika.

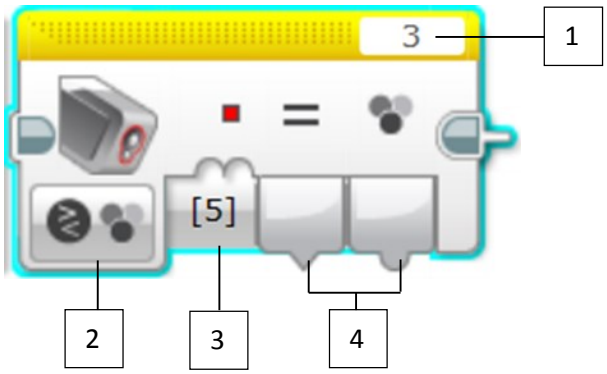
Jak możecie się domyślić po temacie dzisiejszego rozdziału zbudujemy robota poruszającego się po linii.

Roboty poruszające się po wyznaczonej trajektorii mają bardzo szerokie zastosowanie w przemyśle. Najliczniejszą grupą są automatyczne wózki AGV. Wózki AGV poruszają się po trasie wyznaczonej poprzez linię odpowiedniego koloru.

Na początku zapoznajmy się z funkcjami czujnika koloru.

1. Obsługa bloku czujnika koloru (COLOR SENSOR BLOCK)

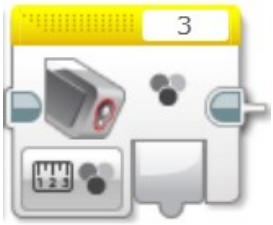
Umożliwia odczyt danych z czujnika, pomiar koloru, intensywności światła i wysyła na wyjście wartość numeryczną. Dodatkowo można porównywać dane wejściowe z wartością progową, dając na wyjściu wartość logiczną (**True/False**).



1 – Wybór portu
2 – Wybór trybu działania
3 – Sygnały wejścia, zależne od wybranego trybu
4 – Sygnały wyjścia, zależne od wybranego trybu

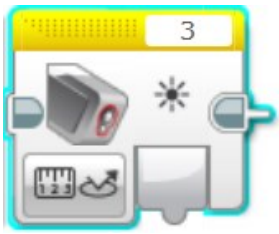
TRYBY DZIAŁANIA

Measure – Color



Umożliwia wykrywanie kolorów, które w postaci numerycznej przesyłane są w postaci sygnału wyjściowego.

Measure – Reflected Light Intensity



Blok przesyła na wyjście wartość numeryczną intensywności odbitego światła.

Measure – Ambient Light Intensity

Na wyjściu otrzymywana jest wartość numeryczna rozproszonego światła.

Compare – Color

Umożliwia porównanie koloru wybranego przez użytkownika z kolorem odbieranym przez czujnik, wysyłając na wyjście informacje o stanie porównania Prawda/Fałsz (True/False) oraz numer koloru odczytanego z czujnika.

Compare – Light Intensity

Porównuje wykrytą intensywność światła do wartości progowej. Na wyjście przekazywany jest wynik porównania wartości logicznej oraz wartość zmierzonej intensywności światła.

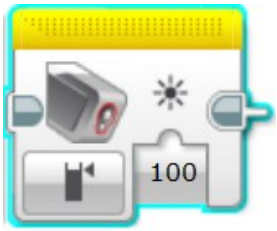
TRYBY KALIBRACYJNE

Tryby te umożliwiają kalibrację czujnika koloru z poziomu programu. Można również ręcznie wpisać wartość minimalną i maksymalną, którą powinien wykorzystać sensor.

Calibrate – Minimum

Umożliwia ustawienie wartości minimalnej intensywności światła. Przyjmuje wartości numeryczne z zakresu 0 – 100.

Calibrate – Maximum



Umożliwia ustawienie wartości maksymalnej intensywności światła. Przyjmuje wartości numeryczne z zakresu 0 – 100.

Calibrate – Reset

Blok ten przywraca ustawienia początkowe czujnika koloru.

Naszego robota budujemy według instrukcji ”linefollower.pdf”. Czujnik koloru w celu prawidłowego działania powinien być umieszczony ok.. 0,5 cm nad podłożem. Konstrukcja umożliwia łatwe skręcanie. Z tyłu została zamontowana sama felga, przez co tarcie jest znacznie mniejsze

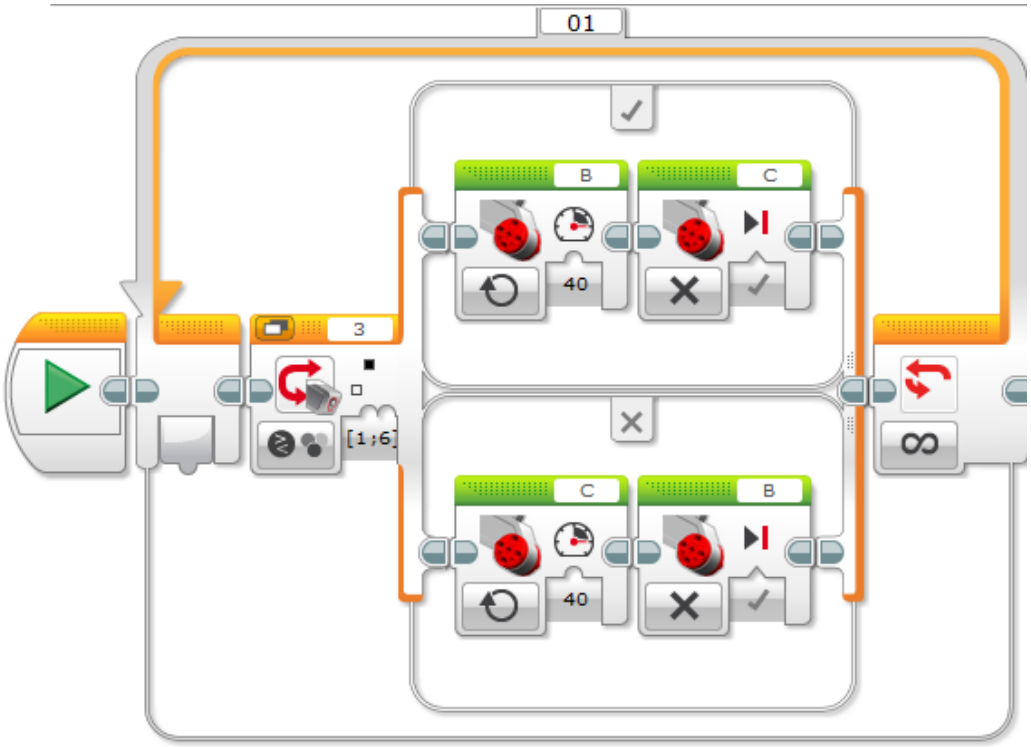
2. W JAKI SPOSÓB SPRAWIĆ, ABY ROBOT JECHAŁ PO WYZNACZONEJ TRASIE?

A. Algorytm poruszania robota typu linefollower

Robot będzie się poruszał według następującego algorytmu:

- w przypadku wykrywania czarnej linii robot skręca w prawo aż wykryje kolor biały,
- w przypadku wykrywania białej linii robot skręca w lewo aż wykryje kolor czarny

B. Programowanie



Program: linnefollower.ev3 Program

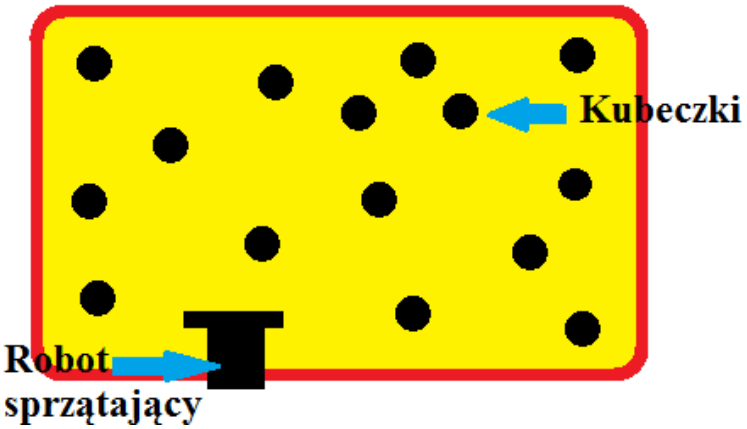
Cały program zostaje umieszczony w pętli. Wykorzystujemy instrukcję warunkową dla czujnika koloru. Czujnik wykrywa kolor czarny. W przypadku wykrycia koloru czarnego robot wykonuje instrukcje z górnej linii. Silnik B się obraca, a C kręci się do przodu. Robot będzie skręcał w lewo. Gdy zjedzie z czarnej linii na białą zacznie wykonywać dolny wiersz poleceń. Silnik B się zatrzyma, a zacznie się obracać silnik C. Robot będzie skręcał w lewo tak długo, aż znowu nie wykryje czarnej linii i tak w koło. Należy pamiętać o odpowiednim podłączeniu silników. Robot jedzie „zygzakiem” raz na linii białej raz na czarnej.

Zadanie 1. Pokonanie wyznaczonej trasy.

Naklejamy na podłodze czarną i białą taśmę izolacyjną, początek powinien być stosunkowo łatwy, następnie robimy coraz ostrzejsze zakręty. Można urządzić wyścigi który robot dojedzie najdalej lub pokona całą trasę najszybciej.

Temat 20: ROBOT SPRZĄTAJĄCY

Dzisiaj rozbudujemy linefollowera w taki sposób, aby stał się robotem sprzątającym. W tym celu musimy rozbudować dotychczasową konstrukcję poprzez dołączenie zderzaka zdolnego wypchnąć obiekty (np. kubeczki jednorazowe) z powierzchni ograniczonej czerwoną linią. Cała konstrukcja robota musi się mieścić na kartce formatu A4.



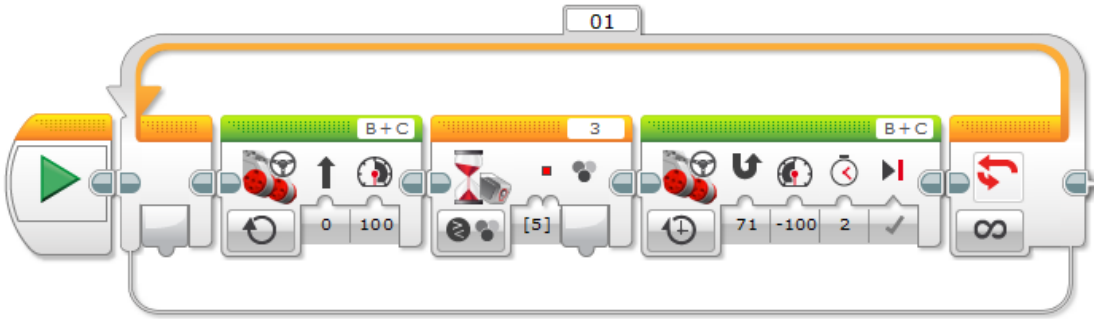
Zderzak montujemy przed czujnikiem koloru. Robot w pierwszej kolejności wypycha kubek poza linię, następnie czujnikiem koloru wykrywa linię i się wycofuje.

Zadanie 1. Sprzątnięcie planszy.

Zadaniem robota jest wypchnięcie jak największej liczby kubeczków poza planszę w określonym czasie (np. 60s). Na podłodze rozkładamy planszę jasnego koloru z brzegiem koloru czerwonego. Ważne, żeby brzeg planszy był innego koloru.

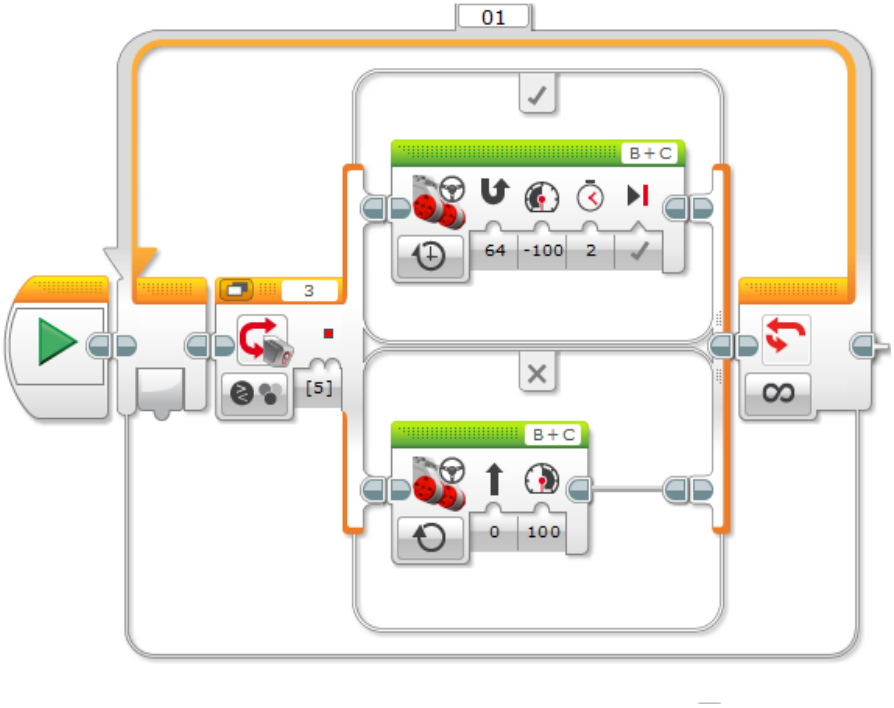
Program

Jazda robota po planszy. Robot jedzie do przodu do momentu wykrycia czerwonej linii. Wykrycie linii powoduje wycofanie się robota oraz jego obrót. Proces zostaje powtórzony dzięki zastosowaniu w programie pętli. Pętla w naszym przykładzie działa do momentu wyłączenia programu (naciśnięcia szarego przycisku na kostce EV3).



Program: sprzatajacy.ev3 Program2

Taki sam program można napisać z wykorzystaniem instrukcji warunkowej.



Program: sprzatajacy.ev3 Program

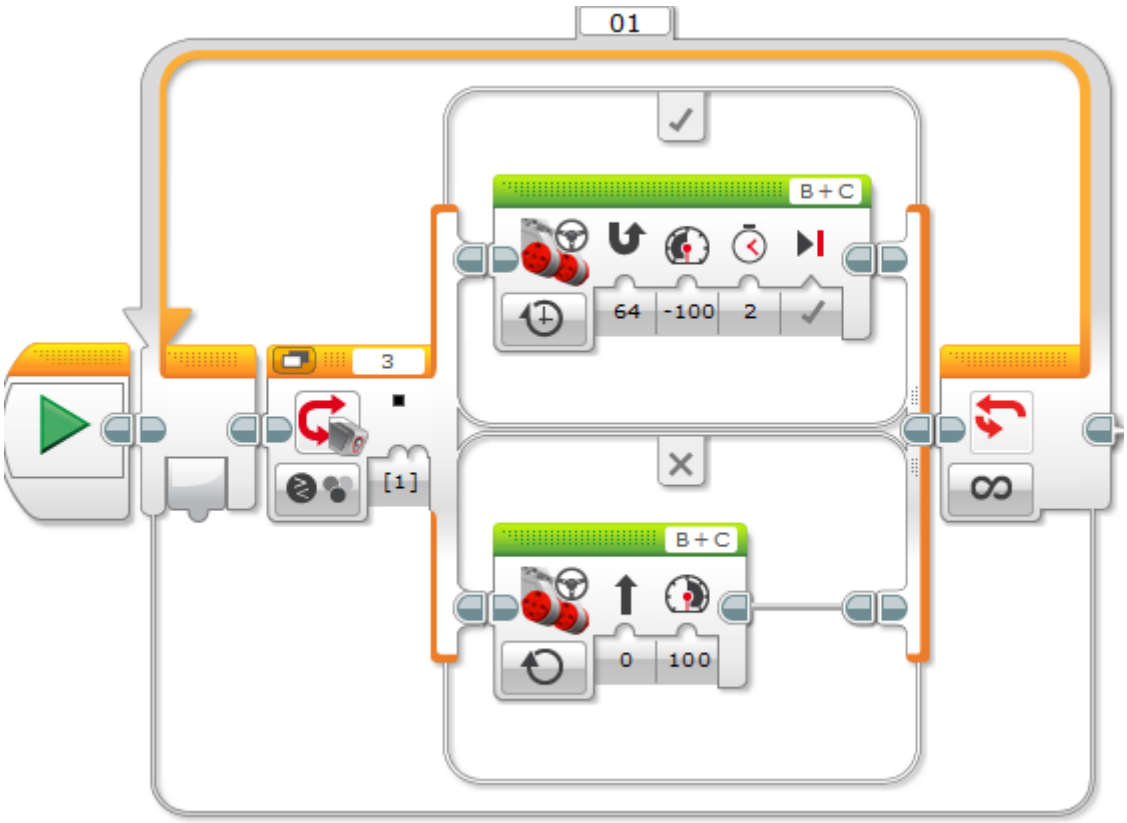
Temat 21: SUMO

Na dzisiejszych zajęciach wykorzystamy całą wiedzę, jaką zdobyliśmy do tej pory na temat przekładni, tarcia, wykrywania przeszkód, programowania. Przeprowadzimy zawody sumo robotów. Zobaczymy, kto najlepiej opanował sztukę konstruowania robotów i ich programowania.

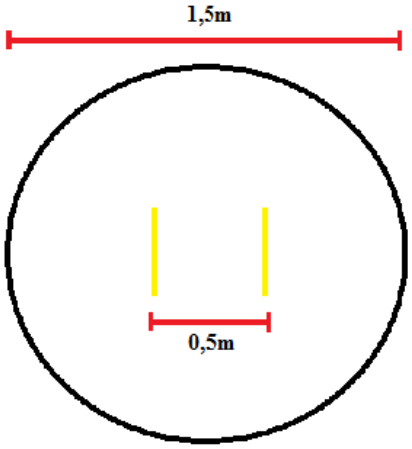
Naszego robota możemy budować według instrukcji „sumo.pdf" lub samodzielnie.

Może ktoś zastosuje jakiś element, który umożliwi przewrócenie przeciwnika? Jedynym ograniczeniem, jakie musi spełnić nasz robot jest to, że musi zmieścić się całkowicie na kartce A4. Żadna z jego części nie może wystawać poza nią. Zawody będą odbywać się na ringu w kształcie koła. Krawędź koła musi być innego koloru niż środek. Robot musi wiedzieć, kiedy dojeżdża do skraju ringu w tym celu należy zastosować czujnik koloru. Robot także może „szukać" przeciwnika przy pomocy czujnika odległości i w momencie jego zauważenia zaatakować go. Nasz robot może charakteryzować się znaczną siłą lub dużą szybkością na skutek zastosowania przekładni. Można zastosować także 3 silnik przy pomocy, którego nasz robot np. przewróci przeciwnika.

Program powinien wykorzystywać wszystkie możliwości robota. Poniżej znajduje się przykładowy program, którego zadaniem jest utrzymanie się robota w polu walki. Można również wykorzystać program od robota sprzątającego. Wystarczy ustawić czujnik koloru na detekcję takiego koloru, jakiego jest krawędź koła.



Program: sprzatajacy.ev3 Program4



1. ZASADY

Zawody są rozgrywane na dohyō – okrągłym ringu o średnicy ok 1,5m. Roboty są ustawiana naprzeciwko siebie, na żółtych liniach w odległości 0,5m od siebie. Na komendę START oba roboty zostają uruchomione. Zawody trwają 1min. Systemem każdy z każdym. W przypadku zwycięstwa otrzymuje się 2 punkty, remisu 1 punkt przegranej 0 punktów.

2. WYGRANA

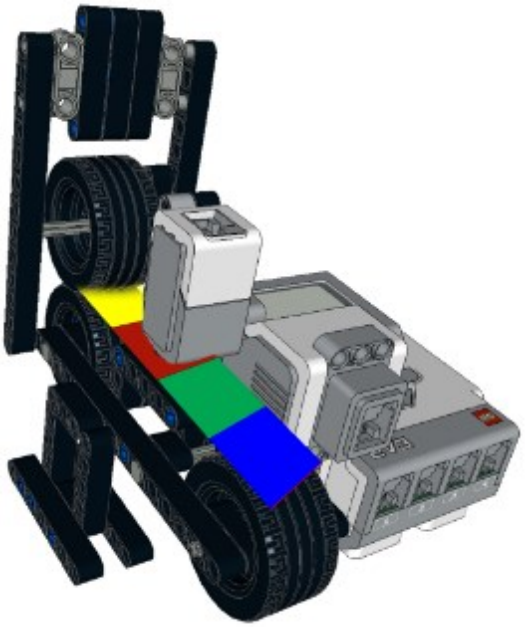
- gdy czujnik koloru któregoś robota wyjedzie (zostanie wypchnięty) poza czarny okrąg wygrywa robot, który pozostał na ringu,
- gdy z robota odpadnie jakakolwiek część wygrywa robot przeciwnika, gdy nie można określić, od którego robota odpadła dana część walka jest kontynuowana
- gdy robot przewróci się, ale pozostaje na ringu walka jest kontynuowana, jeżeli po upływie 1 min pozostaje nadal przewrócony, a drugi robot nie - przegrywa.

3. Remis

- gdy oba roboty opuszczą jednocześnie pole walki,
- gdy od robotów odpadną równocześnie jakiekolwiek części,
- gdy po upływie minuty oba roboty są przewrócone i pozostają na ringu,
- gdy po upływie minuty oba roboty pozostają na dohyō.

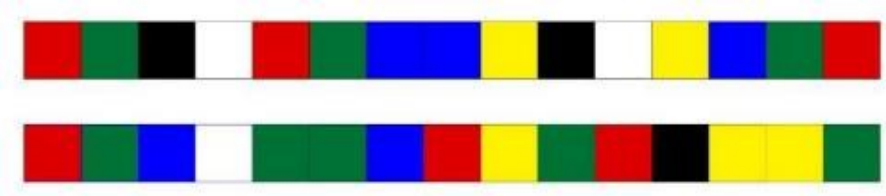
4. Dyskusja na temat tego, który robot wygrał i dlaczego?

Temat 22: POZYTYWKA



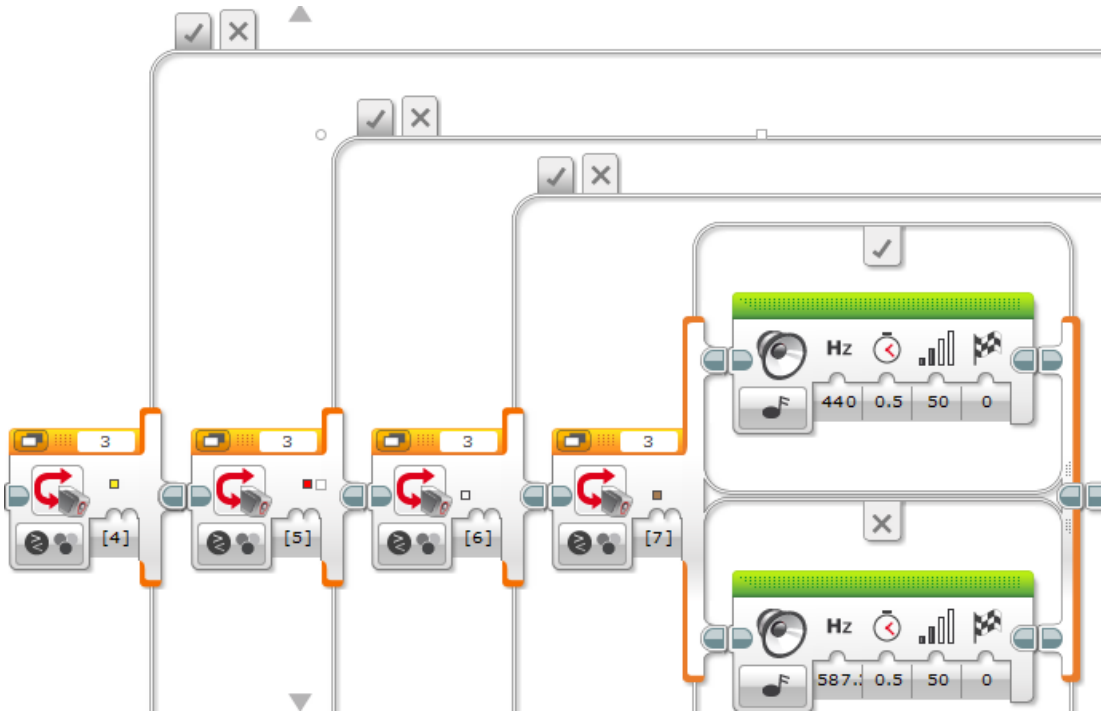
Dzisiaj zbudujemy podobną konstrukcję do gitary. Z tą różnicą, że będzie to pozytywka, która będzie odtwarzała dźwięki przy użyciu czujnika koloru. Robot w zależności od koloru wydaje inny dźwięk. Kolorowy pasek przesuwany jest przy pomocy koła z oponą. Zaraz za mechanizmem przesuwającym znajduje się czujnik koloru do rozpoznawania aktualnego koloru paska. Dzięki zastosowaniu 3 opon pasek może obracać się w kółko, przez co dźwięk jest ciągły. Prędkość dźwięku można regulować przy pomocy prędkości obrotowej silnika.

Instrukcja: „pozytywka.pdf”



1. Program

Program składa się głównie z instrukcji warunkowej od czujnika koloru oraz bloczka dźwięku. W zależności od wykrywanego koloru wydawany jest inny dźwięk. Całość programu objęta została pętlą.

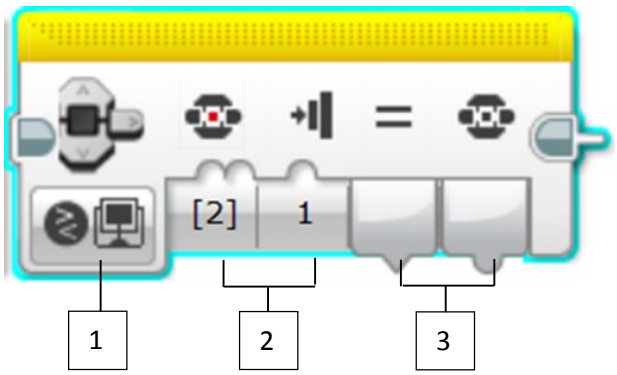


Program: Pozytywka.ev3 Program

Zadanie 1. *** Napisz program, który będzie zmniejszał lub zwiększał moc silnika i wyświetlał dodatkowo wartość mocy na kostce.**

2. Obsługa bloku przycisków kostki EV3 (BRICK BUTTONS BLOCK)

Blok ten sczytuje dane z kostki EV3, a konkretnie z pięciu przycisków. Umożliwia sprawdzenie, które klawisze zostały wciśnięte.



1 – Wybór trybu

2 – Sygnały wejścia, zależne od wybranego trybu

3 – Sygnały wyjścia, zależne od wybranego trybu

TRYBY DZIAŁANIA

Measure – wyświetla, który klawisz został naciśnięty.

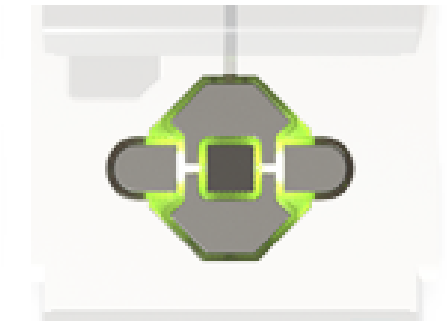
Compare – blok porównuje, czy wciśnięty został wybrany przez użytkownika klawisz oraz sposób wciśnięcia.

Na początku podaję się ID klawiszy oraz sposób naciśnięcia.

Na wyjściu przekazywany jest wynik operacji:

Porównywania (**Compare Result**) oraz Id przycisku (**Button ID**).

Numery ID klawiszy na klocku EV3



Temat 23: PIESEK

Na zajęciach zbudujemy robota, który będzie lokalizował pilota na sali i jechał w jego stronę. Kiedy się do niego zbliży zatrzyma się i zacznie wydawać dźwięk szczekania. Robota budujemy według instrukcji „piesek.pdf".

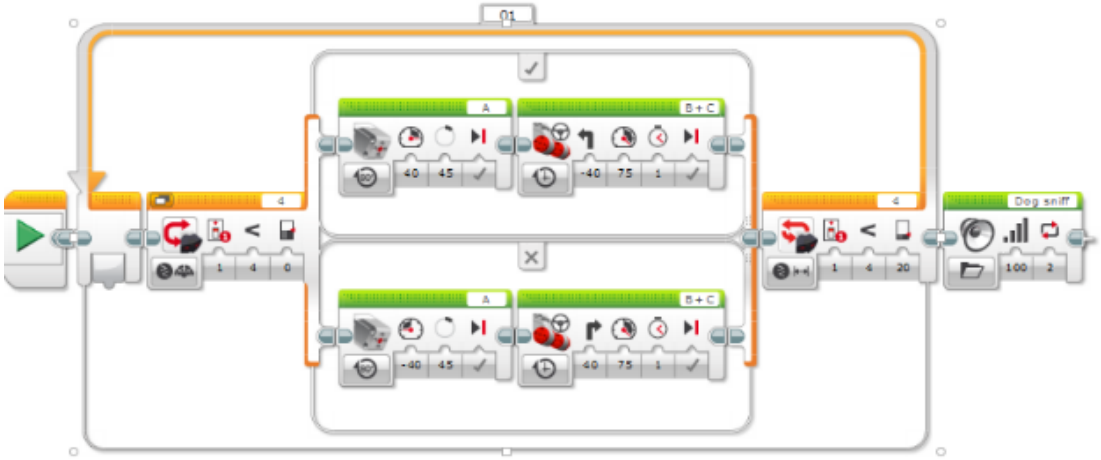
Skorzystamy ze zdolności wykrywania i określania położenia pilota względem czujnika podczerwieni oraz określania odległości pomiędzy czujnikiem podczerwieni, a pilotem.

Infrared Sensor – Compare – Beacon Heading – oczekiwanie, aż pilot zostanie wykryty i czujnik osiągnie określoną wartość (od -25 do 25, gdzie 0 jest to pozycja na wprost czujnika).

Marker czujnika podczerwieni ma opcje wyboru kanału (**Channel Selector**), co umożliwia wybór jednego z czterech kanałów do przesyłu sygnału.

Infrared Sensor – Compare – Beacon Proxiity – oczekiwanie na wykrycie pilota i na osiągnięcie przybliżonej odległości.

Program składa się z pętli. Jeżeli pilot znajduje się po lewej stronie czujnika podczerwieni robot skręca w lewo, jeżeli po prawej stronie robot skręca w prawo. Pętla jest powtarzana tak długo, aż nie zostanie osiągnięta dana odległość pomiędzy czujnikiem podczerwieni a robotem.



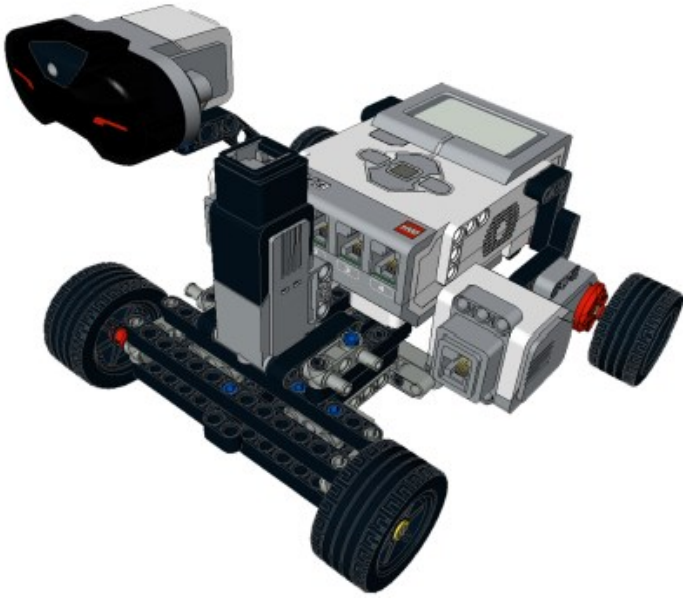
Program: Piesek.ev3 Program

Zadanie 1. Modyfikacja programu w taki sposób, aby robot dojechał jak najszybciej do pilota.

Temat 24: SKRĘTNE KOŁA”

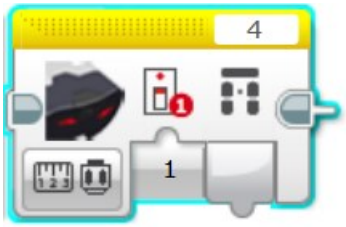
Dzisiaj zbudujemy robota sterowanego przy pomocy pilota. Dodatkowo będzie on posiadał skrętne koła tak jak w samochodzie.

Naszego robota budujemy według instrukcji „skretnekola.pdf”.



Do sterowania wykorzystamy pilot oraz czujnik podczerwieni.

MEASURE – REMOTE

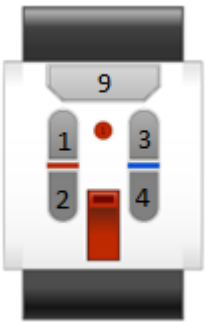


Na wejściu należy podać nr kanału markera.

Na wyjściu podawany jest numer ID przycisku na markerze, który jest w danym momencie naciskany.

INFRARED SENSOR – COMPARE – REMOTE – oczekiwanie na naciśnięcie przycisku/-ów na pilocie czujnika podczerwieni.

W bloczku oczekiwania należy wybrać jeden lub więcej przycisków (**Set of Remote Button IDs**) oraz numer kanału komunikacji z markerem.

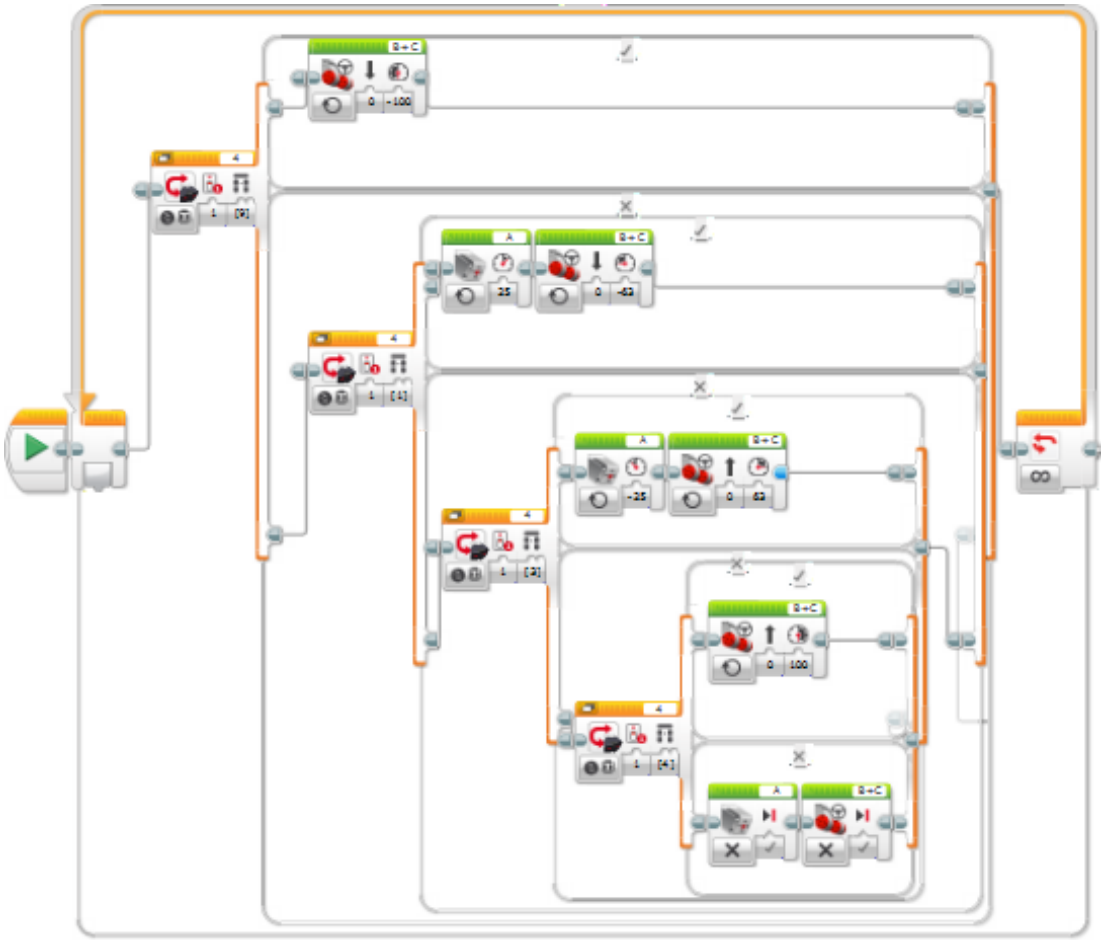


Numer ID przycisków na markerze (**BUTTON ID**) – identyfikuje który przycisk, bądź kombinacja przycisków jest naciśnięta na markerze.

W bloczku przyjęte zostały następujące sekwencje:

0 – Brak	**4** – 4	**8** – 2+4
1 – 1	**5** – 1+3	**9** – Tryb markera włączony
2 – 2	**6** – 1+4	**10** – 1+2
3 – 3	**7** – 2+3	**11** – 3+4

Przykładowy program może wyglądać tak jak ten poniżej. Po naciśnięciu pzycisku 9 robot będzie jechał prosto. Gdy naciśniemy 1 koła zacznią skręcać w lewo, a jak 3 to w prawo. W przypadku w którym rzaden z przycisków nie jest wciśnięty wszystkie silniki się zatrzymują.



Program: skrętnekola.ev3 Program

Zadanie 1. * W jaki sposób można udoskonalić program?

Temat 25: CO SIEDZI W CZUJNIKU

Na dzisiejszych zajęciach dowiemy się na jakiej zasadzie działają czujniki od wewnątrz. Zobaczymy czy jesteśmy w stanie zbudować samodzielnie swój własny czujnik. Zacznijmy od najprostszego czujnika dotyku.

1. Czujnik dotyku

Budowa czujnika dotyku była już omawiana w pierwszym rozdziale tego podręcznika.

Naciśnięcie przycisku powoduje zamknięcie obwodu elektrycznego.

Zadanie 1. Czy można skonstruować czujnik dotyku przy użyciu przeciętego przewodu? Spróbuj odpowiedzieć na pytanie, które przewody muszą być połączone, aby wyświetlana wartość logiczna na kostce wynosiła 1?

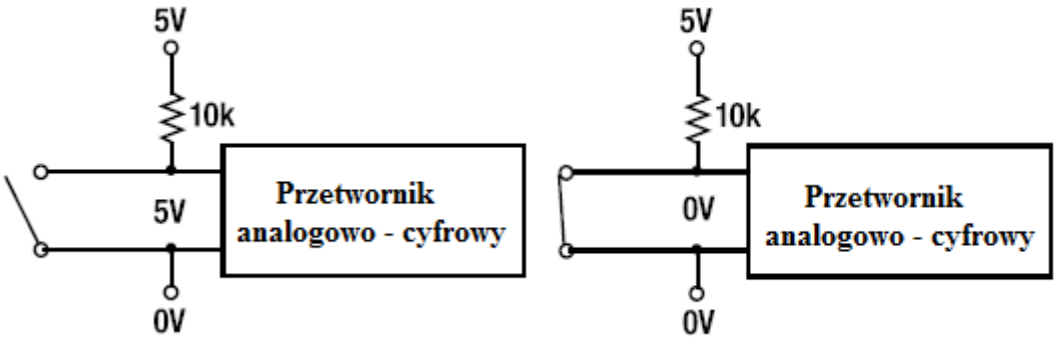
Podłącz czujnik do jednego z portów (1-4) kostki EV3, następnie wybierz opcję Port View i odnajdujemy port, do jakiego jest podłączony przewód. Połączenie, których przewodów powoduje wyświetlenie wartości równej 1?

Zadanie 2. * Jak można użyć zdobytą wiedzę w praktyce?

Rezystor – jest to element elektroniczny, który powoduje spadek napięcia oraz ograniczenie wartości prądu w obwodzie.

2. Czujniki rezystancyjne

Kostka oprócz wykrycia tego czy przełącznik jest włączony czy wyłączony posiada również inną opcję. Kostka EV3 używa przetwornika analogowo-cyfrowego (służy do zamiany wartości odczytywanej przez czujnik na wartość liczbową). Pomiędzy przewodami powstaje napięcie, które przetwornik zamienia na wartości liczbowe od 0 dla 0V do 4080 dla 5V. Jest to zależność liniowa. Biały przewód jest na stałe podpięty do napięcia 5V po przez rezystor 10kΩ, a czarny wraz z czerwonym do masy (uziemienia). Jeżeli złączymy wszystkie trzy przewody mierzone napięcie będzie wynosiło 0V, jeżeli przewody będą rozłączone kostka będzie rejestrować napięcie 5V.



Rejestrowane napięcie będzie równe:

$$V = \frac{R}{10000 + R}5$$

gdzie:

R – opór przedmiotu umieszczonego pomiędzy przewodami.

Jeżeli złączymy bezpośrednio ze sobą oba przewody R=0Ω, wtedy również V=0V

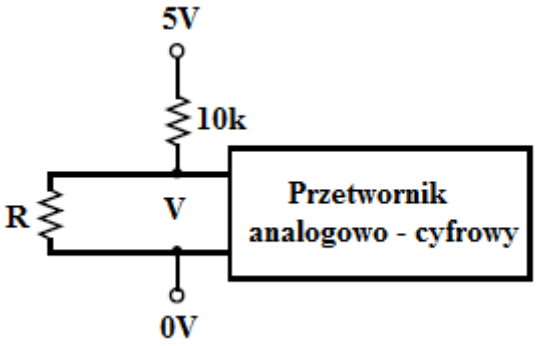
Zastanówmy się czy istnieje jakiś sposób, aby określić opór przedmiotu jaki umieściliśmy pomiędzy przewodami?

Przetwornik analogowo-cyfrowy EV3 zamienia wartość napięcie na wartość liczbową Raw, która może być wyświetlana na kostce. Wartość 4080 wynika z faktu, że konwerter ma 12 bitów dokładności. Największą liczbą, jaką można przedstawić przy pomocy 12 bitów jest 4092. Wartość Raw można obliczyć przy pomocy wzoru:

$$Raw = \frac{4080}{5}V$$

3. Pomiar rezystancji - oporu

Jeżeli umieścimy pomiędzy przewodami przedmiot o skończonej oporności to kostka będzie rejestrować napięcie z zakresu od 0 do 5V i dla tych wartości przyporządkowywać wartości liczbowe od 0 do 4080.



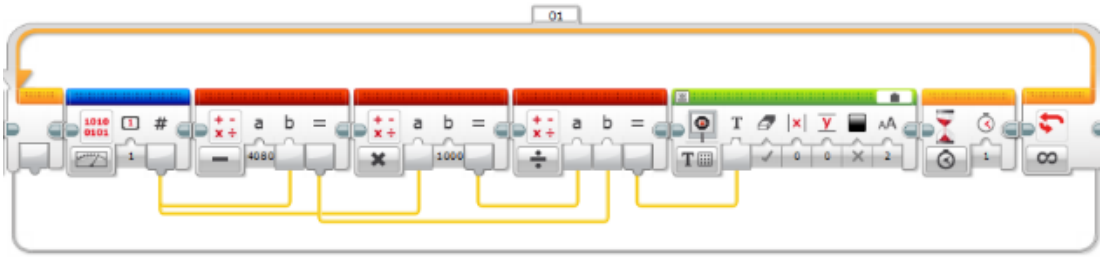
Najmniejszy opór, jaki jest w stanie zmierzyć kostka EV3 wynosi 9Ω, a największa mierzalna wartość wynosi 1.022.000 Ω. Jest to bardzo szeroki zakres.

Opór przedmiotu można wyznaczyć z wzoru:

$$R = \frac{10000Raw}{4080 - Raw}$$

4. Wykorzystanie zdobytej wiedzy do konstrukcji i kalibracji czujników

Znając równanie 3 możemy napisać bardzo prosty program wyświetlający na kostce wartość oporu przedmiotu, który umieścimy pomiędzy białym, a czerwonym i czarnym przewodem kostki.



Program: opor.ev3 Program

Program ten zczytuje wartość Raw z kostki i wykonuje obliczenia zgodnie z rownaniem 3. Mnoży Raw przez tysąć. Od 4080 odejmuje Raw i obie liczby dzieli przez siebie. Otrzymany wynik zostaje wyświetlony na kostce.

W jakich czujnikach jest wykorzystywana zależność od oporu?

W takich gdzie opór zależy od różnych czynników np. od temperatury, natężenia światła czy oraz od innych czynników.

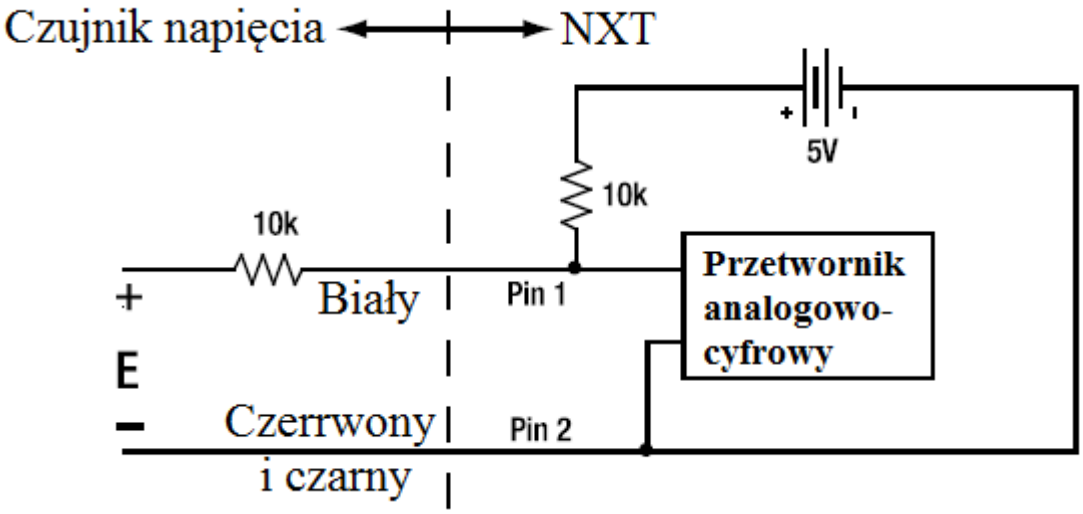
Termistor - opornik półprzewodnikowy lub metalowy, którego rezystancja (opór) zależy od temperatury.

Fotorezystor - opornik półprzewodnikowy lub metalowy, którego rezystancja (opór) zależy od intensywności oświetlenia.

Zadanie 3. * Przeprowadź proces kalibracji czujnika z wykorzystaniem termistora. Sporządź krzywą zależności oporu od temperatury.

Temat 26: JAK ZBUDOWAĆ WOLTOMIERZ

Skoro umiemy mierzyć opór przedmiotów, to może dzisiaj zbudujemy przyrząd do pomiaru napięcia. Jedynym elementem elektrycznym, którego będziemy potrzebować będzie opornik 10kΩ. Poniższy rysunek przedstawia schemat ideowy wejścia EV3 z prostym czujnikiem napięcia. Napięcie mierzone zostało oznaczone jako E.

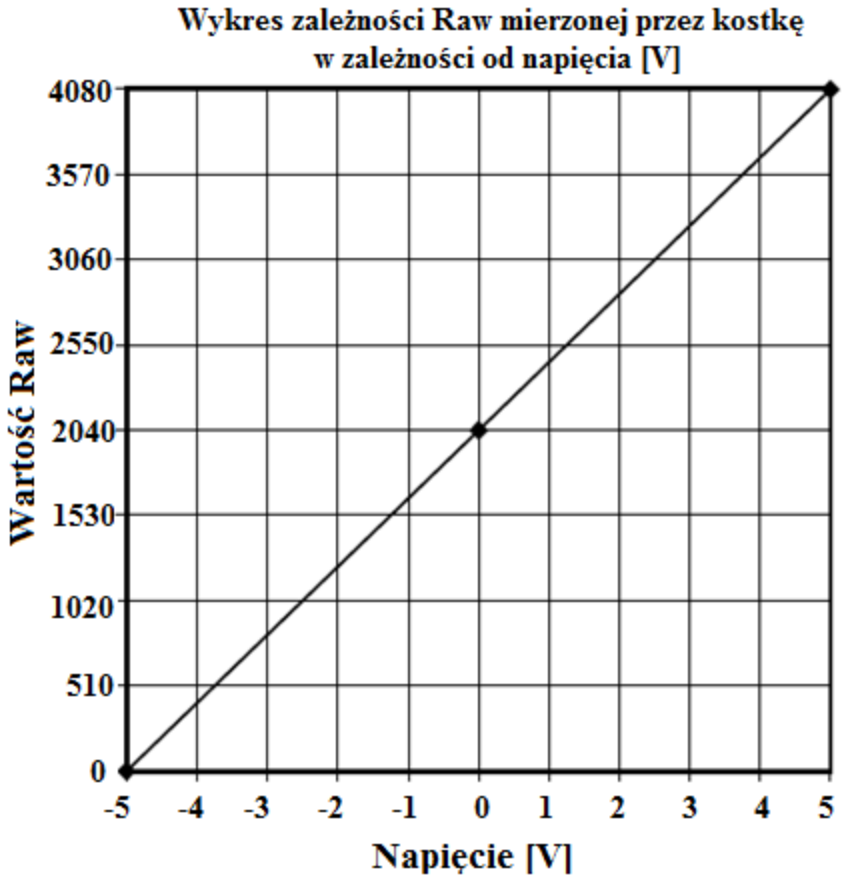


Podłączenie opornika sprawi, że w przybliżeniu uzyskamy liniową zależność Raw od przyłożonego napięcia. Zobaczmy w jaki sposób to osiągniemy.

Jeżeli przyłożymy zewnęczne napięcie E=0V to układ będzie wskazywał tą samą wartość jak byśmy podłączyli opornik 10kV. Robiliśmy to na poprzednich zajęciach. Napięcie zostanie podzielone równomiernie na oba rezystory. Spowoduje to, że na przetworniku analogowo-cyfrowym będzie napięcie 2,5V i zostanie zamieniona na Rew w przybliżeniu równe 2040. Jest to centralny punkt na wykresie.

Teraz załóżmy że pomiędzy przewodami umieścimy napięcie E=5V. Teraz oba napięcia i to z kostki i to z zewnątrz przechodzą przez opornik 10kΩ. Przetwornik analogowo-cyfrowy widzi oba napięcia i nie ma wyboru które prekonwertuje na wartość Rew równą 4080. Jest to górny punkt na rysunku.

Na koniec przymijmy, że przyłożone napięcie wynosi -5V. Przechodząc przez opornik 10kΩ ma odwrotną wartość do napięcia wewnętrznego przechodzącego równierz przez opór 10kΩ. W konsekwencji wypadkowe napięcie na przetworniku analogowo-cyfrowym daje wartość 0V, króre jest zamieniane na wartość Rew=0. Jest to dolny punkt na rysunku.



Na podstawie tego możemy w prosty sposób obliczyć napięcie:

$$E = \left(\frac{2\,Rev}{4080} - 1\right)5 = \frac{10Rev}{4080} - 1$$

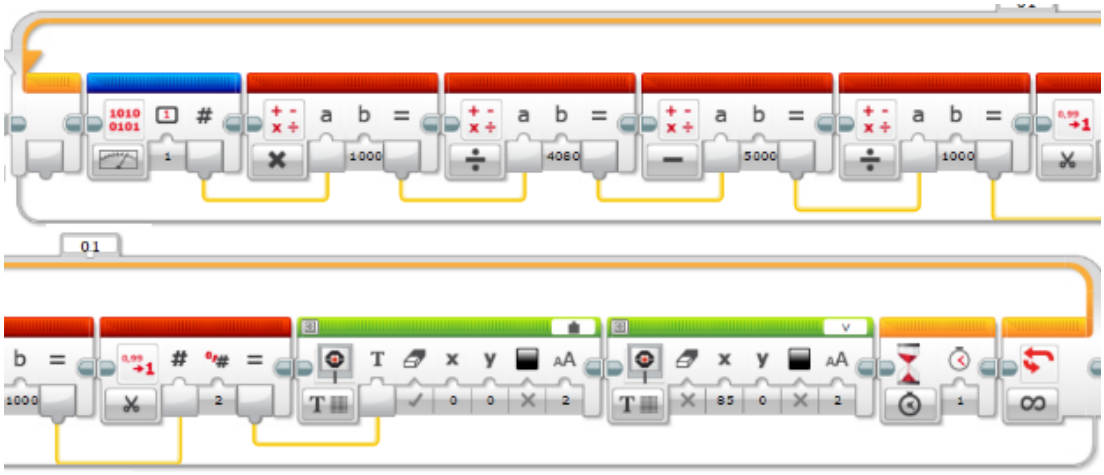
Wcześniejsze wersje kostek Lego Mindstorms nie obsługiwały operacji zmienno-przecinkowych. Powyższy wzór dawałby w takim przypadku wartości 0, 1 lub 2. Byłoby to mało praktyczne. W celu ominięcia tego problemu można pomnożyć cały wzór przez 1000 w efekcie będziemy mierzyć mV.

$$\boldsymbol{E = \frac{10000Rew}{4080} - 5000}$$

W celu otrzymania wyniku w V wystarczy z powrotem cały wzór podzielić przez 1000.

UWAGA – NIE NALEŻY UMIESZCZAĆ PRZEWODÓW W KONTAKCIE, NIE NALEŻY MIERZYĆ NAPIĘCIA WIĘKSZEGO NIŻ 5V TAKI POMIAR MOŻE USZKODZIĆ KOSTKĘ.

Tak jak na ostatnich zajęciach program wykonuje powyższą formułę matematyczną i dzieli otrzymany wynik przez 1000. Ponadto program zaokrągla otrzymany wynik i zaokrągla go do dwóch miejsc po przecinku. Wyświetla jeszcze jednostkę napięcia V. Zwróćmy uwagę, że w programie czyszczenie ekranu jest ustawione tylko dla pierwszego bloku wyświetlacza. Zastosowanie czyszczenia ekranu dla drugiego bloku spowodowałoby zmazanie wartości liczbowej napięcia.



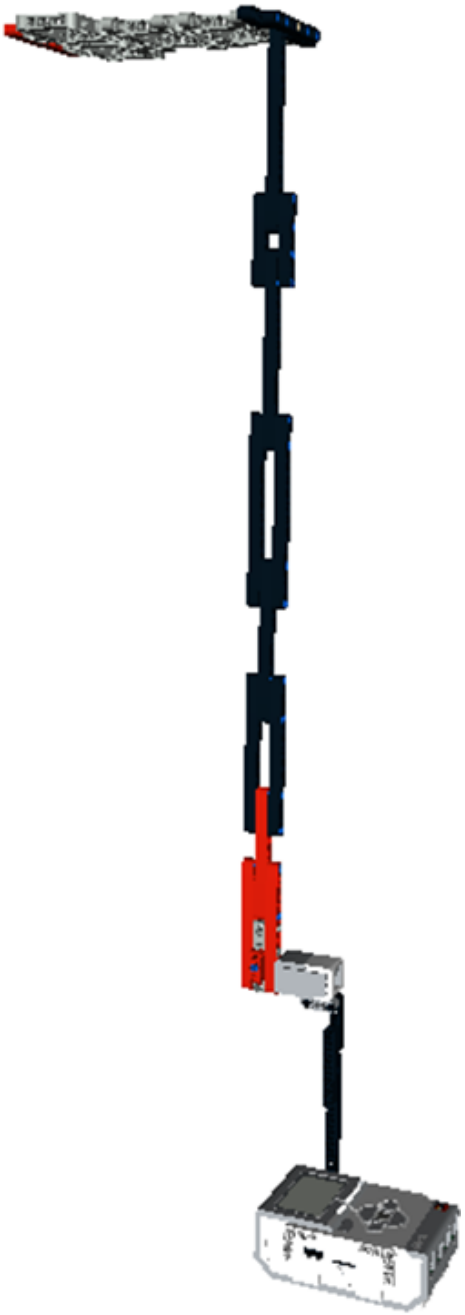
Program: woltomierz.ev3 Program

Zadanie 1. *** Rozbuduj program w taki sposób, aby zaokrąglał uzyskamy wynik.**

Temat 27: WAHADŁO FIZYCZNE

Wahadło – ciało zawieszone nad swoim środkiem ciężkości wykonujące w pionowej płaszczyźnie drgania na skutek działania siły grawitacji.

Rozróżniamy dwa podstawowe typy wahadeł:



- **Wahadło matematyczne** jest to punktowa masa zawieszona na nierozciągliwej, pozbawionej masy nici o długości d. W przypadku małych drgań wahadła matematycznego są one harmoniczne i mogą być opisane wzorem:

$$T = 2\pi\sqrt{\frac{l}{g}}$$

gdzie:

l – długość wahadła [m],

g – przyspieszenie ziemskie [9,81$\frac{m}{s^2}$]

Na podstawie pomiarów okresu drgań wahadła matematycznego można obliczyć przyspieszenie ziemskie.

- **Wahadło fizyczne** to ciało sztywne wykonujące wahania wokół poziomej osi zawieszenia przechodzącej przez nie. Jego drgania są również w przybliżeniu harmoniczne, o okresie:

$$T = 2\pi\sqrt{\frac{I}{mgd}}$$

gdzie:

m – masa ciała [kg],

d – odległość środka ciężkości ciała od osi obrotu [m],

Moment bezwładności wahadła (I) (miara bezwładności ciała w ruchu obrotowym względem określonej, ustalonej osi obrotu). Im większy Momot bezwładności tym trudniej zmienić ruch obrotowy tego ciała [kg·m2],

Nasza konstrukcja będzie to wahadło fizyczne, ale niektóre obliczenia przeprowadzimy jak dla wahadła matematycznego. Naszą konstrukcję budujemy według instrukcji "wahadlo.pdf". Górną część wahadła kładziemy na stole i dociążany książkami w celu ustabilizowania.

Program składa się z dwóch pętli. Zastosowano w programie zmienną, która umożliwia wyświetlanie poszczególnych czasów jednocześnie na kostce w dwóch rzędach.

Program rejestruje moment, w którym dół wahadła przechodzi naprzeciwko czujnika koloru. Bardzo ważne jest odpowiednie ustawienie czujnika koloru w niewielkiej odległości od wahadła. Rejestrowany czas jest to połowa okresu drgań wahadła T.

W celu obliczenia okresu drgań wahadła należy dwa następujące czasy odjąć od siebie i pomnożyć przez dwa.

Program: wahadlo.ev3 Program

Jak już wyznaczymy czas T możemy przystąpić do obliczeń.

Zadanie 1. **Na początek skorzystajmy z znanego z fizyki wzoru na okres drgań wahadła matematycznego i na jego podstawie obliczmy wartość przyspieszenia ziemskiego g.**

$$g = \frac{4\pi^2 l}{T^2}$$

Zadanie 2. **Otrzymana wartość różni się od wartości tablicowej. Spróbujmy skorzystać z wzoru dla wahadła fizycznego.**

$$g' = \frac{4\pi^2 I}{T^2 m d}$$

Przyjmijmy moment bezwładności dla pręta $I = \frac{1}{3}ml^2$ i d w połowie długości $d = \frac{1}{2}l$

Po podstawieniu:

$$g' = \frac{2}{3}\frac{4\pi^2 l}{T^2} = \frac{2}{3}\mathrm{g}$$

Zadanie 3. * Wyznacz rzeczywisty moment bezwładności dla naszego wahadła.

Temat 28: KATAPULTA

Zbudujemy dzisiaj urządzenie wykorzystujące zasadę działania dźwigni. Naszą konstrukcję budujemy Według instrukcji ”katapulta.pdf”.

Profesjonalna nazwa zbudowanej przez nas machiny oblężniczej to trebusz. Jest to specyficzny rodzaj katapulty wykorzystujący zasadę działania dźwigni i momentu obrotowego.

Do krótszego ramienia jest przymocowana kostka. Odblokowanie ramienia przy pomocy silnika rotacyjnego powoduje, że kostka opada z przyspieszeniem ziemskim, powodując gwałtowny ruch obrotowy drugiego, dłuższego ramienia. Posiada ono pojemnik, w którym możemy umieścić nasz pocisk (kawałek zwiniętej foli aluminiowej). Jest on uwalniany na zasadzie działania siły odśrodkowej w chwili zahamowania ruchu dźwigni.

Zbudowana przez nas katapulta może dodatkowo się poruszać dzięki zastosowaniu silników. Program do naszego robota może umożliwiać ruch. Wystrzał pocisku odbywa się w Momocie obrotu silnika rotacyjnego o kąt 90°.

Program: katapulta.ev3 Program

Zadanie 1. *** Dołącz do konstrukcji czujnik podczerwieni i napisz program do sterowania katapultą przy pomocy pilota.**

Temat 29: ROBOT PRZEMYSŁOWY

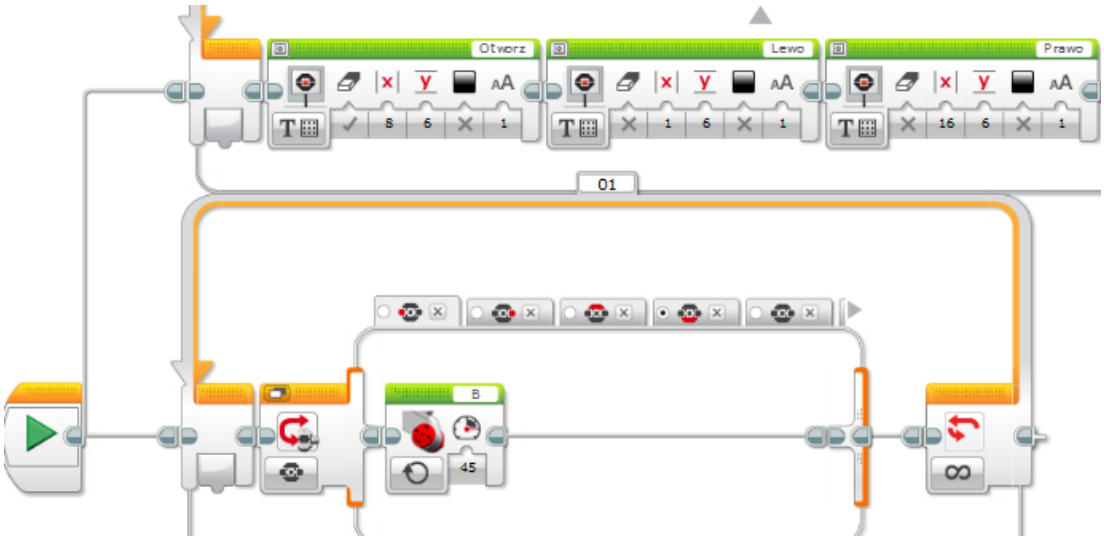
Dzisiaj zbudujemy robota przemysłowego służącego do przenoszenia różnych przedmiotów. Wykorzystamy w tym celu wszystkie trzy silniki, przekładnie na siłę oraz przekładnie zmieniającą kierunek działania siły. Do sterowania naszym robotem wykorzystamy sterowanie przy użyciu przycisków na kostce oraz stworzymy interfejs graficzny wyświetlający funkcje poszczególnych przycisków. Naszego robota budujemy według instrukcji „przemyslowy.pdf"

Program składa się z dwóch pętli.

Pierwsza pętla odpowiada z wyświetlanie interfejsu graficznego oraz jego zmianę w przypadku wciśnięcia środkowego przycisku, który powoduje otwarcie lub zamknięcie chwytaka. Po wciśnięciu przycisku napis na środku kostki się zmienia i obraca się silnik rotacyjny.

Druga pętla odpowiada za ruch ramienia w płaszczyźnie pionowej i poziomej. W środku pętli został zastosowany blok decyzyjny, który obraca silnikiem B lub C w przód lub w tył w zależności od wciśniętego przycisku.

Fragment gotowego programu mógłby wyglądać tak:



Program: Arm.ev3 Program

Zadanie 1. * Samodzielne zaprogramowanie robota.

Temat 30: MASZYNA SORTUJĄCA

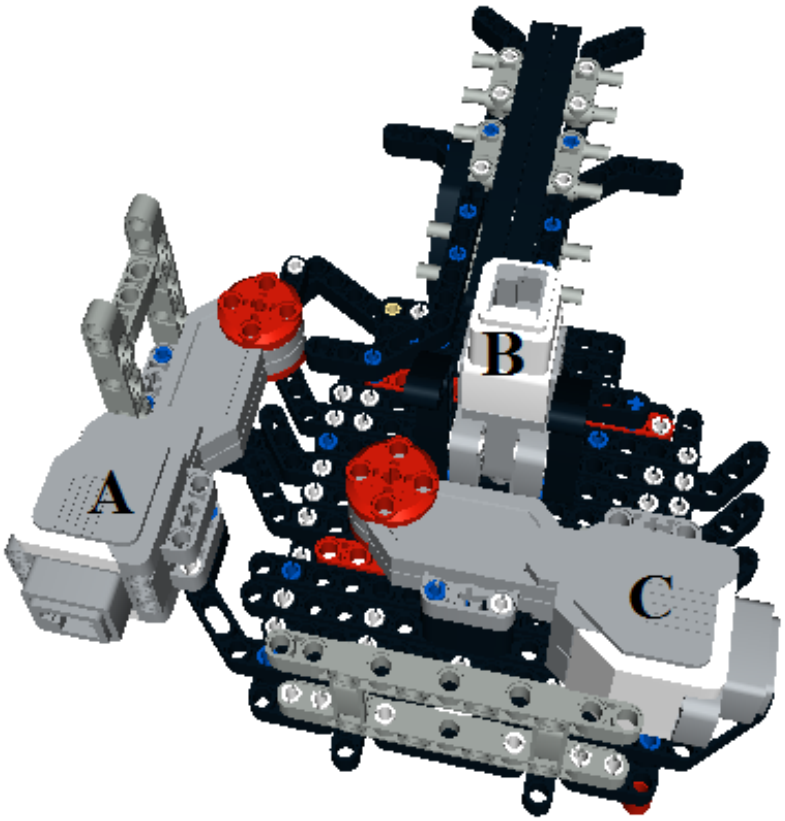
Na dzisiejszych zajęciach zbudujemy robota przemysłowego zdolnego segregować kolorowe kulki.

Podstawowe założenia konstrukcyjne takiego robota to posiadanie:

- czujnika koloru,
- czterech przegródek na kolorowe piłeczki,
- trzech silników.

Takiego robota można zbudować na wiele różnych sposobów.

My zbudujemy robota posiadającego trzy silniki na podstawie instrukcji „sortownica.pdf". Silnik A służy do „dostarczania" piłeczek. Silnik B (rotacyjny) przerzuca piłeczkę na prawo lub lewo w zależności od koloru (np. zielona i żółta na prawo, niebieska i czerwona na lewo). Silnik C wykonuje podobne zadanie jak drugi (np. zielona na prawo, żółta na lewo) tylko oddziela dwa poszczególne kolory.

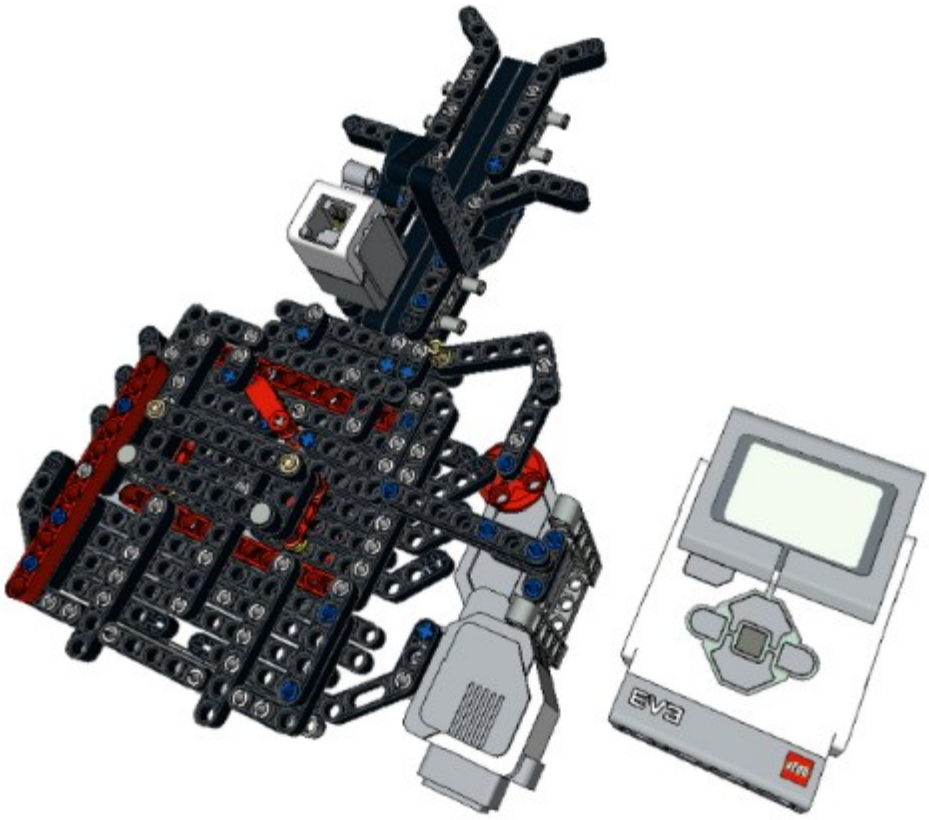


W jaki sposób działa program sterujący sortownicą krok po kroku:

1. Rozpoznaj czujnikiem kolor piłeczki
2. Ustaw silnik B w odpowiednim położeniu w zależności od rozpoznanego koloru (prawo/lewo) - ruch o odpowiedni kąt

3. Ustaw silnik C w odpowiednim położeniu w zależności od rozpoznanego koloru (prawo/lewo) - ruch o odpowiedni kąt
4. Rusz silnikiem A do tyłu, aby z podajnik wypuścić piłeczkę
5. Rusz silnikiem do przodu, aby zamknąć podajnik
6. Ustaw silnik B i C w pozycji startowej (przegródka ustawiona na prawą przegródkę).

Program: Sortownica.ev3 Program



Pamiętać należy o prawidłowym podłączeniu silników i czujnika koloru:

- silnik A należy podłączyć portu A,
- silnik B należy podłączyć do portu B,
- silnik C należy podłączyć do portu C,
- czujnik koloru należy podłączyć do portu pierwszego.

UWAGA!!!

Przy uruchomieniu sortownicy należy zwrócić szczególną uwagę, aby elementy sortujące kulki na lewą i prawą stronę były wychylone maksymalnie w lewą stronę. Ich złe ustawienie może doprowadzić do zacięcia się silników.

Zadanie 1. * Samodzielne zaprogramowanie sortownicy.