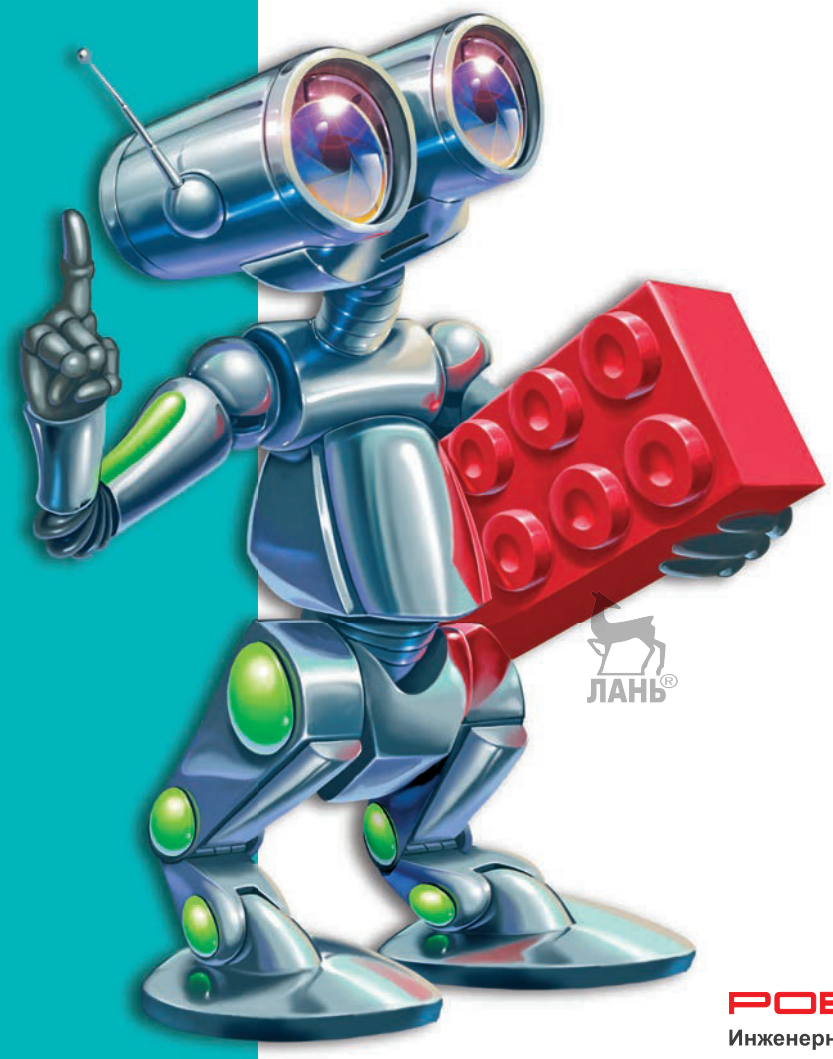
Р • О • Б • О • Ф • И • Ш • К • И

КОНСТРУИРУЕМ РОБОТОВ на LEGO® MINDSTORMS® Education EV3

Крутое пике



Лаборатория ЗНАНИЙ

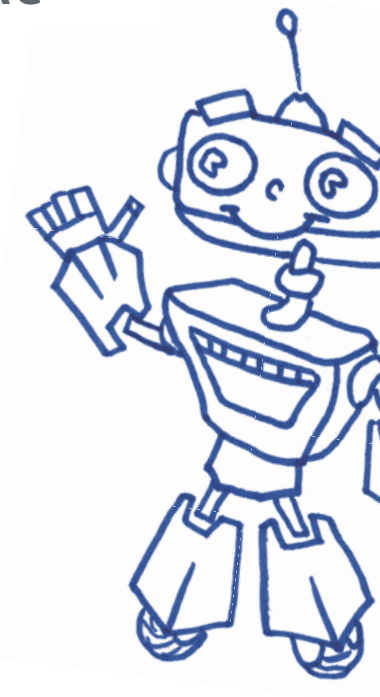
РОБОТОТЕХНИКА
Инженерно-технические кадры инновационной России

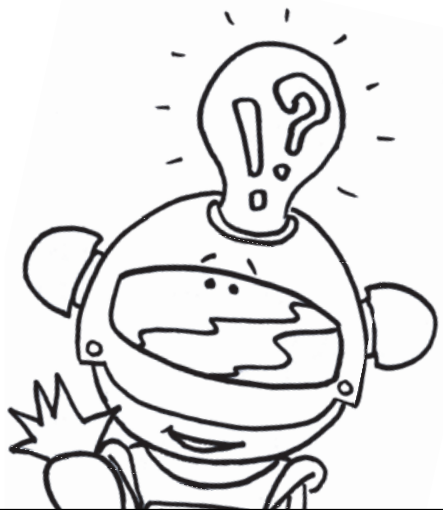
Е. И. Рыжая, В. В. Удалов, В. В. Тарапата

КОНСТРУИРУЕМ РОБОТОВ на LEGO® MINDSTORMS® Education EV3

Крутое пике

3-е издание,
электронное





Лаборатория знаний
Москва
2021

УДК 373.167
ББК 32.97
Р93

Серия основана в 2016 г.
Ведущие редакторы серии *Т. Г. Хохлова, Ю. А. Серова*

Проект подготовлен под руководством В. Н. Халамова

Рыжая Е. И.

Р93 Конструируем роботов на LEGO® MINDSTORMS® Education EV3. Крутое пике / Е. И. Рыжая, В. В. Удалов, В. В. Тарапата. — 3-е изд., электрон. — М. : Лаборатория знаний, 2021. — 97 с. — (РОБОФИШКИ). — Систем. требования: Adobe Reader XI ; экран 10". — Загл. с титул. экрана. — Текст : электронный.

ISBN 978-5-93208-532-5

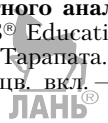
Стать гениальным изобретателем легко! Серия книг «РОБОФИШКИ» поможет вам создавать роботов, учиться и играть вместе с ними.

Вы соберёте из деталей конструктора LEGO® MINDSTORMS® Education EV3 робота, который может имитировать полёт самолёта. Управляя им, вы почувствуете себя настоящим асом!

Для технического творчества в школе и дома, а также на занятиях в робототехнических кружках.

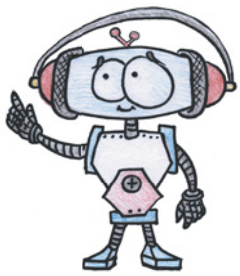
УДК 373.167
ББК 32.97

Деривативное издание на основе печатного аналога: Конструируем роботов на LEGO® MINDSTORMS® Education EV3. Крутое пике / Е. И. Рыжая, В. В. Удалов, В. В. Тарапата. — М. : Лаборатория знаний, 2017. — 92 с. : ил., [4] с. цв. вкл. — (РОБОФИШКИ). — ISBN 978-5-00101-017-3.



ISBN 978-5-93208-532-5

Здравствуйте!



Издание, которое вы держите сейчас в руках, — это не просто описание и практическое руководство по выполнению конкретного увлекательного проекта по робототехнике. И то, что в результате вы самостоятельно сумеете собрать своими руками настоящее работающее устройство, — это, конечно, победа и успех!

Но главное — вы поймёте, что такие ценные качества характера, как терпение, аккуратность, настойчивость и творческое мышление, проявленные при работе над проектом, останутся с вами навсегда, помогут уверенно создавать своё будущее, стать успешным человеком, независимо от того, с какой профессией свяжете жизнь.

Создавать будущее — сложная и ответственная задача. Каждый день становится открытием, если он приносит новые знания, которые затем могут быть превращены в проекты. Особенно это важно для тех, кто выбрал дорогу инженера и технического специалиста. Знания — это база, которая становится основой для свершений.

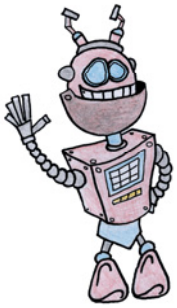
Однако технический прогресс зависит не только от знаний, но и от смелости создавать новое. Всё, что нас окружает сегодня, придумано инженерами. Их любопытство, желание узнавать неизведанное и конструировать то, чего никто до них не делал, и создаёт окружающий мир. Именно от таких людей зависит, каким будет наш завтрашний день. Только идеи, основанные на творческом подходе, прочных знаниях и постоянном стремлении к новаторству, заставляют мир двигаться вперёд.

И сегодня, выполнив этот проект и перейдя к следующим, вы сделаете очередной шаг по этой дороге.

Успехов вам!

Команда Программы «Робототехника:
инженерно-технические кадры инновационной России»
Фонда Олега Дерипаска «Вольное Дело»

Дорогой друг!



Как видно, ты уже совсем не новичок в LEGO, раз добрался до набора LEGO® MINDSTORMS® Education EV3 и, конечно, быстро собрал всё, что там предлагалось!

Внимание!

Ты можешь собрать свои достижения в робототехнике в электронное портфолио! Фотографируй или фиксируй на видео результаты своей работы, чтобы потом представить их для участия в творческих конкурсах. Результаты конкурсов и олимпиад засчитываются при поступлении в профессиональные учебные заведения.

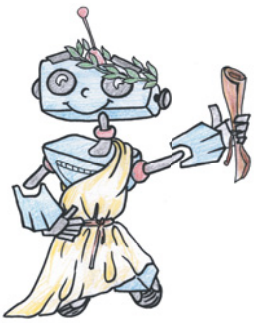
Что же делать теперь? Набор дорогой, выбрасывать жалко, а у младшего братика (если он есть) пока другие игрушки. Не расстраивайся! Мы тебе поможем.

Из этого набора можно собрать ещё много интересных и полезных вещей, например авиасимулятор. Если ты мечтаешь о славе настоящих асов полёта, то этот проект для тебя!

Задумайся над этим!

Фактически за какой-то час работы ты сумеешь пройти многовековой путь изобретателей прошлого! Почему в настоящее время такое стало возможно? Можно ли изобрести что-нибудь новое, не зная, какие машины и механизмы существовали в прошлом? Как интереснее работать — одному или вместе с другом?

Знакомимся с самолётом



Человек всегда хотел летать, как птица. Мечта сбылась — мы с лёгкостью преодолеваем огромные расстояния на самолёте. Может быть, ты мечтаешь управлять этой красивой серебристой птицей — стать пилотом?

Самолёт — аппарат для полётов в воздухе с помощью двигателей и неподвижных крыльев.

Главные части самолёта:

- *крыло* — помогает самолёту оторваться от земли;
- *хвостовое оперение* — поверхности, предназначенные для обеспечения устойчивости и управляемости;
- *фюзеляж* — «тело» самолёта, на нём крепится всё остальное: крыло, оперение, шасси, двигатели; кроме того, в фюзеляже находится кабина пилотов, салон для пассажиров, грузы и оборудование;
- *шасси* — это система опор, необходимых для разбега самолёта при взлёте, пробега при посадке, а также передвижения и стоянки на земле; чаще всего шасси бывают колёсные;
- *силовая установка* — это двигатели;
- *системы бортового оборудования* — это различные приборы и устройства, которые позволяют выполнять полёты при любых погодных условиях. Бортовая электроника является наиболее «умным», сложным и дорогостоящим оборудованием и превосходит по стоимости все остальные части самолёта.

По своему назначению самолёты бывают:

- военные (истребители, перехватчики, бомбардировщики, ракетоносцы, авианосцы, разведчики и пр.);
- гражданские (пассажирские, транспортные, почтовые, сельскохозяйственные, санитарные, спортивные, учебно-тренировочные).

Также самолёты различаются по массе, типу и числу двигателей, по числу и расположению крыльев, по расположению оперения, по размерам и этажности фюзеляжа, по скорости полёта, по типу взлёта и посадки и пр.

Историю создания летательных аппаратов и воздухоплавания можно прочесть в Интернете.

НАВИГАЦИОННЫЕ ПРИБОРЫ

В кабине пилотов на панели управления размещены различные рычаги, штурвалы и приборы для контроля полёта и оборудования самолёта (рис. 1). Это бортовые измерительные приборы. К приборам для контроля параметров полёта относится и искусственный горизонт — авиагоризонт.

Рис. 1. Кабина пилотов Airbus-A320

Авиагоризонт — это навигационный прибор для измерения и сообщения экипажу углов крена и тангажа, соответствующих положению самолёта относительно горизонтальной плоскости (то есть земли).

Крен — поворот самолёта вокруг его продольной оси (рис. 2 и рис. 3).

Тангаж — угловое движение самолёта относительно его поперечной оси (см. рис. 2 и рис. 3): при взлёте носовая часть поднимается вверх, хвостовая опускается вниз, при посадке — наоборот.

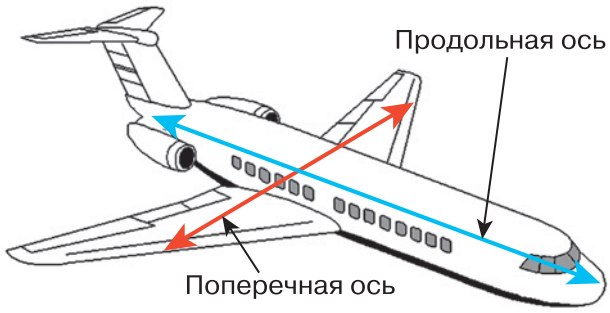


Рис. 2. Оси самолёта

Авиагоризонт (рис. 4) является *гироскопическим прибором*, или *гироскопом*. Он может измерять углы наклона тела, на котором оно установлено (в нашем случае — на самолёте).

Авиагоризонт необходим лётчику для контроля положения самолёта в пространстве: для обеспечения безопасности

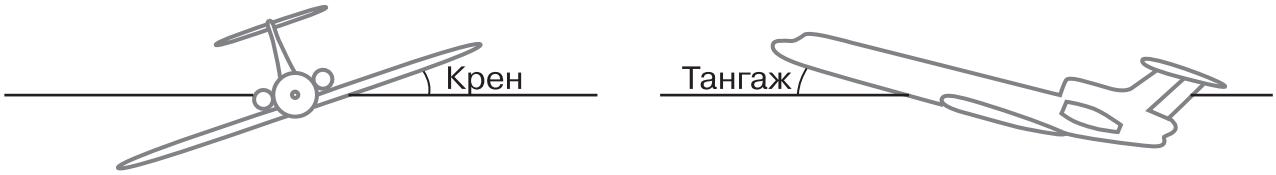


Рис. 3. Взято с сайта по адресу **http://mylektsii.ru/1-34279.html**

во время полёта, при выводе самолёта из сложного положения, при выполнении фигур пилотажа.

Гироскоп, размещённый внутри авиагоризонта, «отслеживает» изменения положения самолёта относительно его продольной и поперечной осей и отображает эти изменения на экране авиагоризонта в виде схематически представленных неба (синяя полусфера вверху), земли (тёмная полусфера внизу) и самолёта — U-образная конструкция в центре, с жёлтыми горизонтальными «крылышками» (см. рис. 4).

Рис. 4. Авиагоризонт

Граница между небом и землёй на приборе и есть линия *искусственного горизонта*. Если в обычном полёте (при хорошей видимости) обратить внимание на взаимное расположение *естественного горизонта* (впереди за лобовым стеклом) и искусственного (на приборе), то окажется, что они всегда параллельны.

Если «приподнять» нос самолёта подачей штурвала на себя, то искусственный горизонт так же, как и естественный, переместится вниз. Опускаем «нос» — и линии обоих горизонтов синхронно поползут вверх.

Вооружённые таким прибором пилоты всегда могут выбрать безопасное положение самолёта в пространстве, ориентируясь на показания авиагоризонта.

Авиагоризонтами оснащены все современные самолёты как гражданской, так и военной авиации.

В нашем проекте мы будем собирать и управлять моделью штурвала пассажирского самолёта Airbus-A320. При управлении самолётом с целью обеспечения безопасности необходимо учитывать следующие параметры:

- скорость полёта: 800–840 км/ч, или 222–233 м/с;
- задаваемые значения углов: тангажа — от –20 градусов до 22 градусов, крена — от –15 градусов до 15 градусов.

Оборудование:

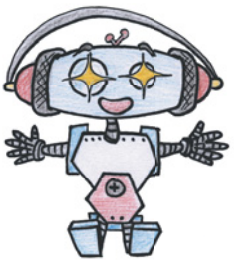
- Базовый набор LEGO® MINDSTORMS® Education EV3.
- Компьютер (минимальные системные требования): Windows XP, Vista, Windows 7, Windows 8 (за исключением METRO), Windows 10 (32/64 бит), оперативная память не менее 1 Гб, процессор — 1,6 ГГц (или быстрее), разрешение экрана — 1024 × 600, свободное место на диске — 5 Гб.
- Программное обеспечение LEGO® MINDSTORMS® Education EV3 (LME-EV3).

Обозначения:

В тексте тебе встретятся обозначения, которые мы сейчас поясним на примерах.

1. Балка № 7 — это балка с семью модулями (отверстиями).
2. 3-модульный штифт — штифт, длина которого равна длине балки № 3.
3. Ось № 5 — ось, длина которой равна длине балки № 5.

Этап 1. Устройство Авиасимулятора

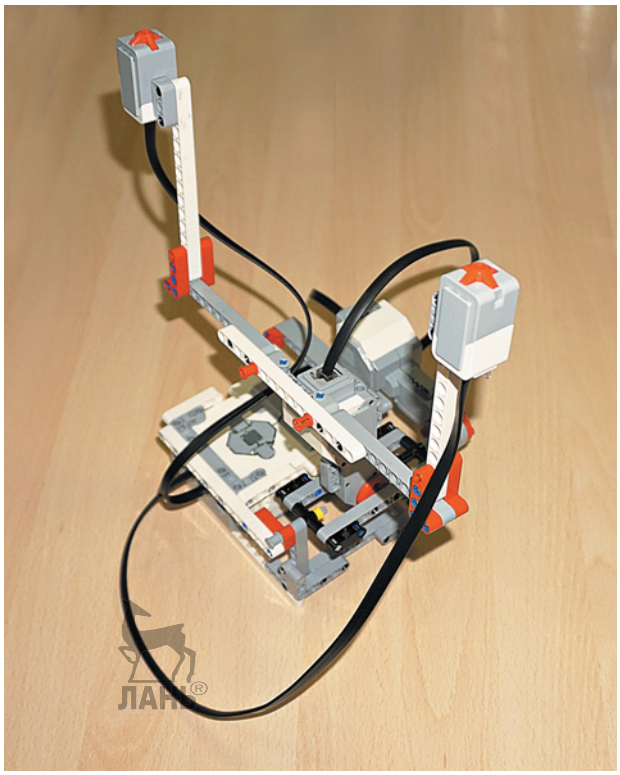


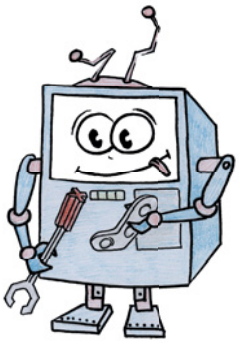
Рассмотри модель робота-авиасимулятора, собранную на основе набора LEGO® MINDSTORMS® Education EV3.

Выдели на ней рабочие детали: модель штурвала; два мотора, отвечающие за углы крена и тангажа; два датчика касания — газ и тормоз.

Обрати внимание, в конце книги, в таблице даны все детали, которые потребуются тебе для сборки. Эта таблица поможет быстро найти то, что необходимо, и не ошибиться при конструировании.

А теперь давай соберём эту модель!



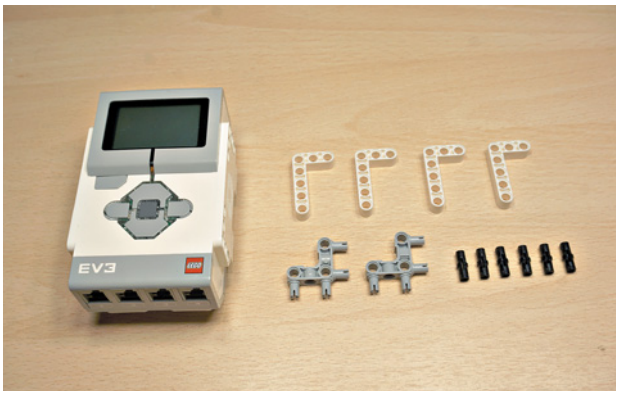


Этап 2. Сборка Авиасимулятора

ШАГ 1. СБОРКА КОНТУРА ЖЁСТКОСТИ

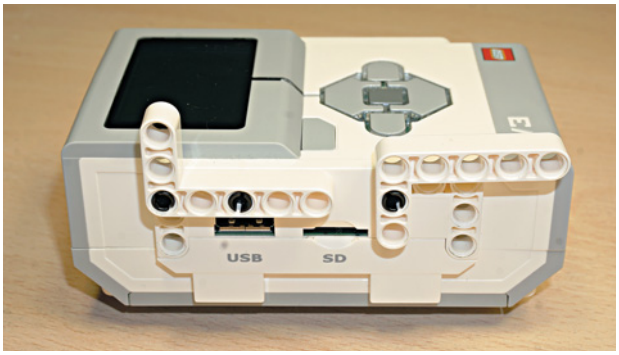
Детали для сборки:

- программируемый модуль EV3, 1x;
- балка прямоугольная 3 × 5, белая 4x;
- угловой штифт 3 × 3, 2x;
- соединительный штифт, 2-модульный, чёрный, 6x.



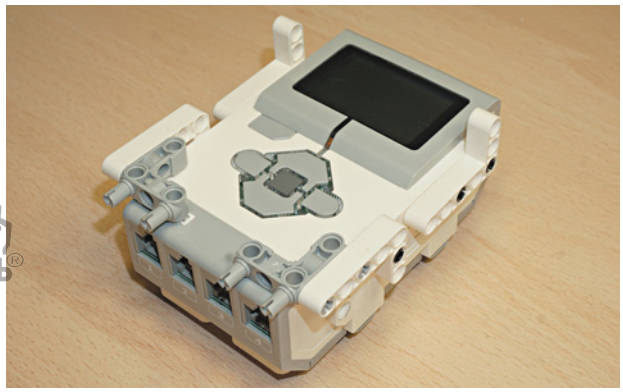
1. На левой боковой панели программируемого модуля в первый, третий и четвёртый верхние модули вставь чёрные штифты.

2. На двух левых штифтах закрепи прямоугольную балку, на правом штифте установи вторую прямоугольную балку, как показано на рисунке.



3. С обратной стороны этой балки установи угловой штифт в её третьем и пятом модулях.

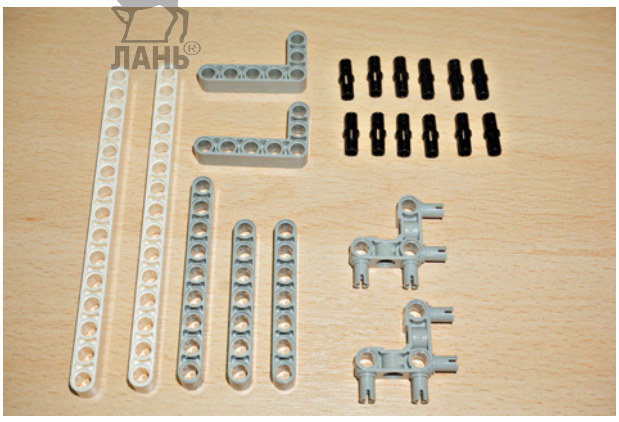
4. Перейди на правую боковую панель EV3 и зеркально собери то же самое.



ШАГ 2. СБОРКА ОСНОВАНИЯ РУЛЕВОГО МЕХАНИЗМА

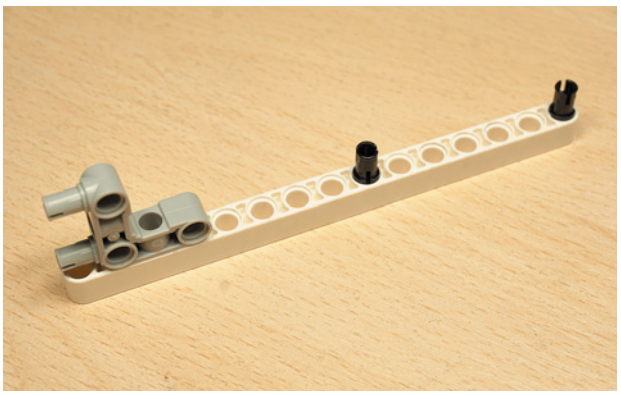
Детали для сборки:

- балка № 15, белая, 2x;
- балка № 9, серая, 1x;
- балка № 7, серая, 2x;
- прямоугольная балка 3 × 5, серая, 2x;
- угловой штифт 3 × 3, 2x;
- соединительный штифт, 2-модульный, чёрный, 12x.

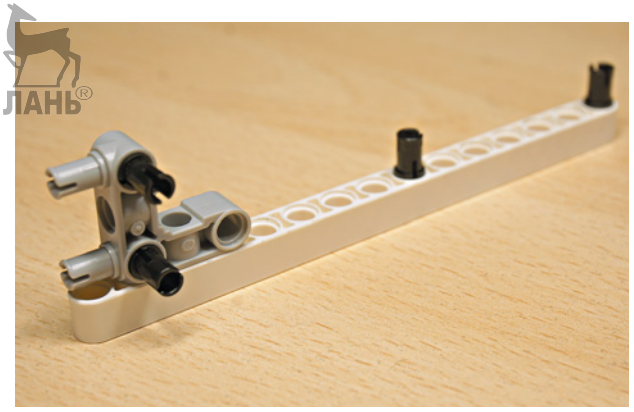


Основание состоит из двух одинаковых половинок. Сначала собери левую.

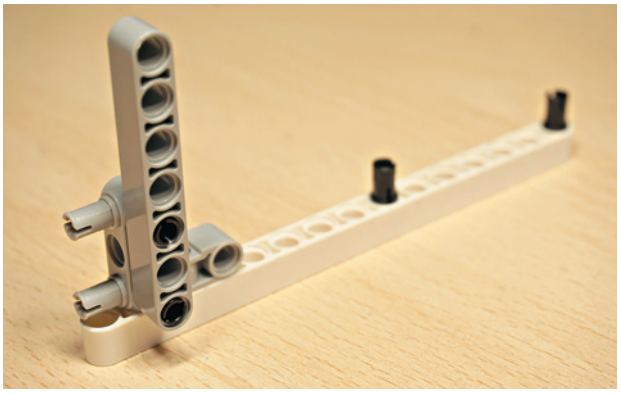
1. В девятый и пятнадцатый модули балки № 15 вставь чёрные штифты. Во втором и четвёртом модуле установи угловой штифт, как показано на рисунке.



2. В его вертикальные модули вставь чёрные штифты.

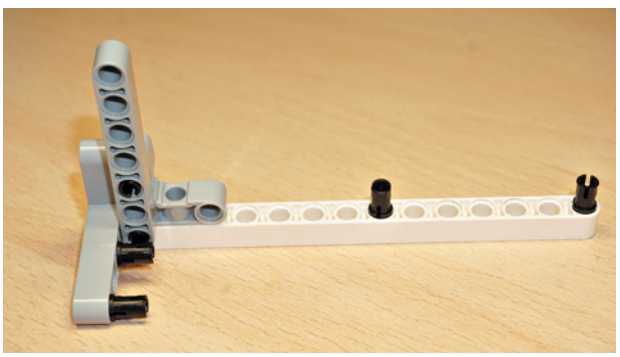


3. На этих штифтах закрепи балку № 7.

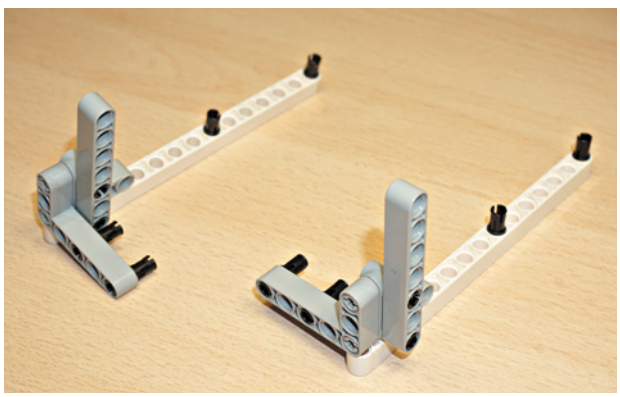


4. На угловом штифте установи с торца прямоугольную балку.

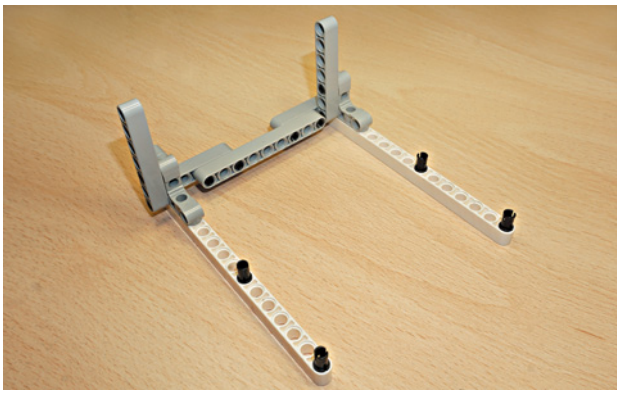
5. В первый и третий модули балки с внутренней стороны вставь чёрные штифты.



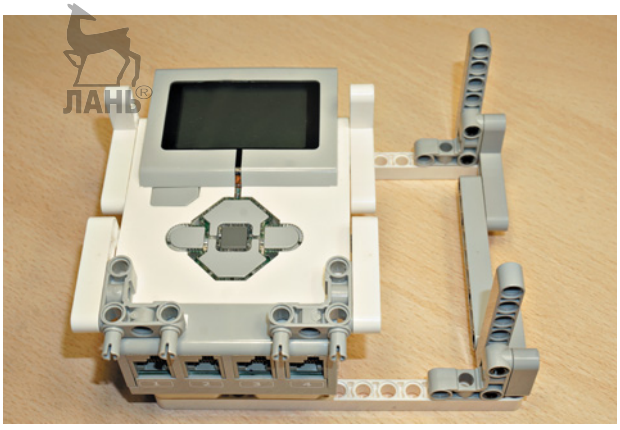
6. Правую половинку основания рулевого механизма собери зеркально, но балку № 7 установи с внешней стороны.



7. Соедини получившиеся половинки балкой № 9, как показано на рисунке.



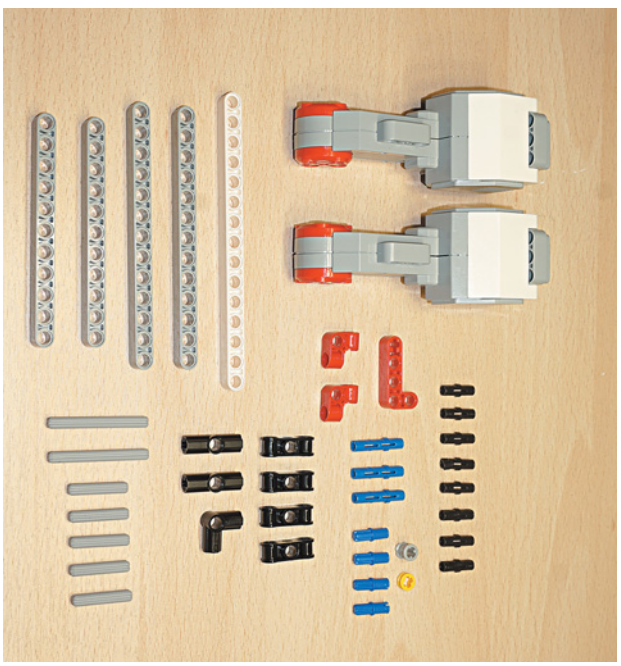
8. На четырёх свободных чёрных штифтах установи программируемый модуль.



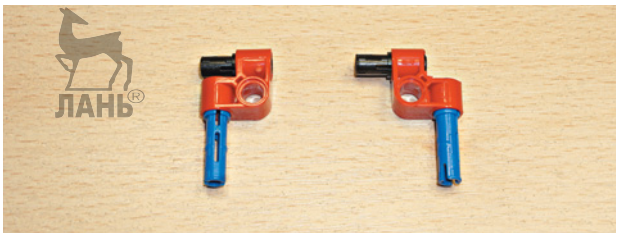
ШАГ 3. СБОРКА РУЛЕВОГО МЕХАНИЗМА

Детали для сборки:

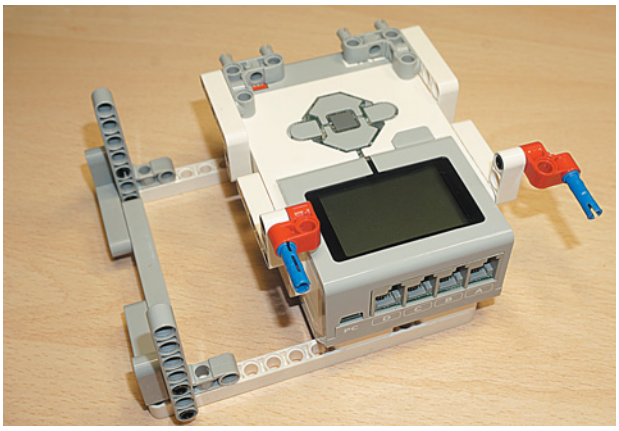
- балка № 15, белая, 1x;
- балка № 13, серая, 2x;
- балка № 11, серая, 2x;
- балка прямоугольная 2 × 4, красная, 1x;
- балка поперечная 2 × 1, красная, 2x;
- втулка жёлтая, 1x;
- втулка серая, 1x;
- ось № 5, серая, 2x;
- ось № 3, серая, 5x;
- втулка 3-модульная, чёрная, 2x;
- втулка угловая 3-модульная, чёрная, 1x;
- фиксатор 3-модульный, чёрный, 4x;
- штифт 3-модульный, синий, 3x;
- штифт 2-модульный, синий, 4x;
- штифт 2-модульный, чёрный, 8x;
- мотор большой, 2x.



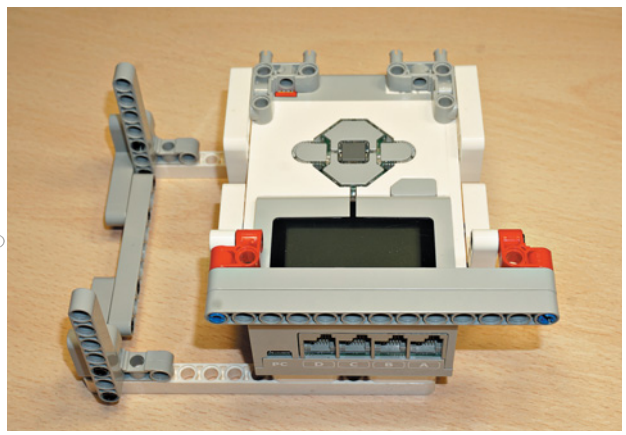
1. В красные поперечные балки 2 × 1 вставь по одному чёрному 2-модульному и по одному синему 3-модульному штифту.



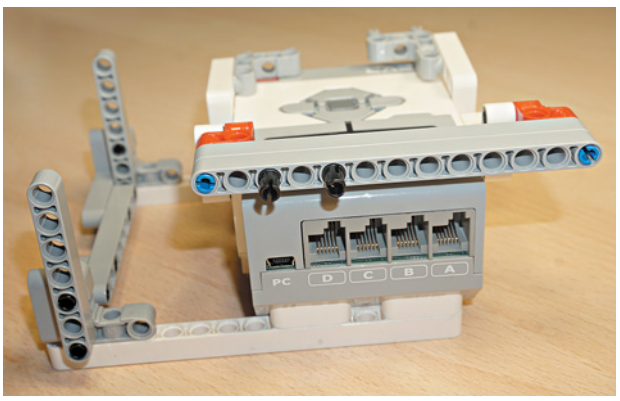
2. Закрепи получившиеся элементы на балках контура жёсткости, как показано на рисунке. Обрати внимание на то, как они должны быть расположены!



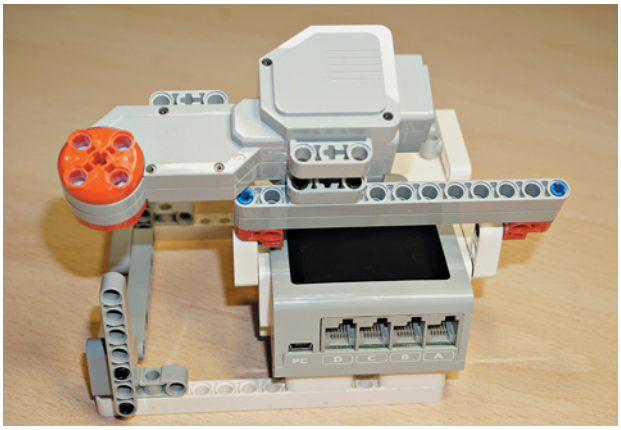
3. На синих штифтах установи друг за другом две балки № 13.

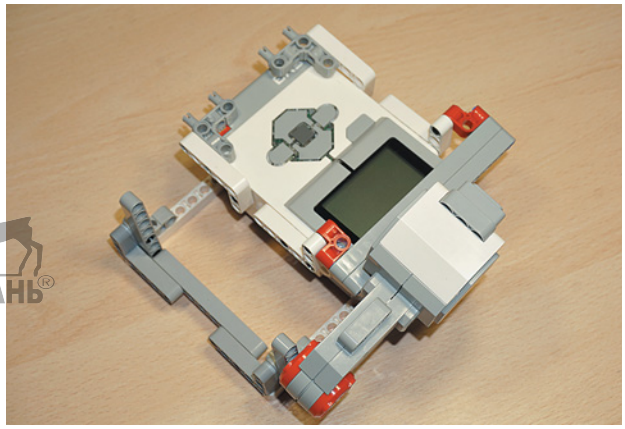


4. В третий и пятый модули крайней балки № 13 вставь чёрные штифты.

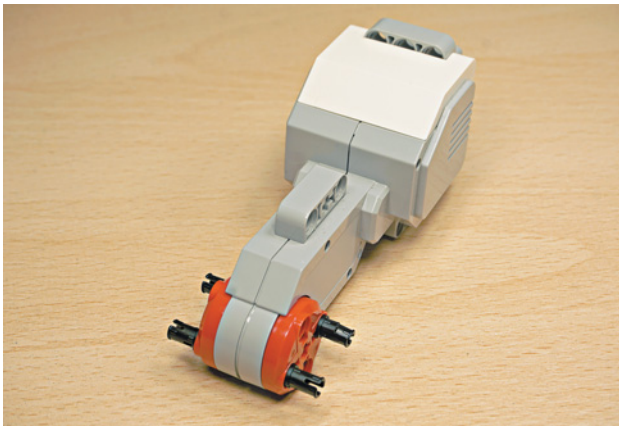


5. На этих штифтах установи большой мотор. Этот мотор управляет углом крена самолёта. Мы будем называть его *горизонтальным*.

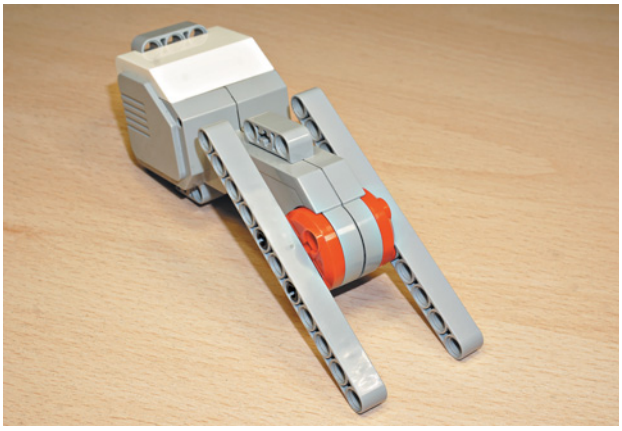




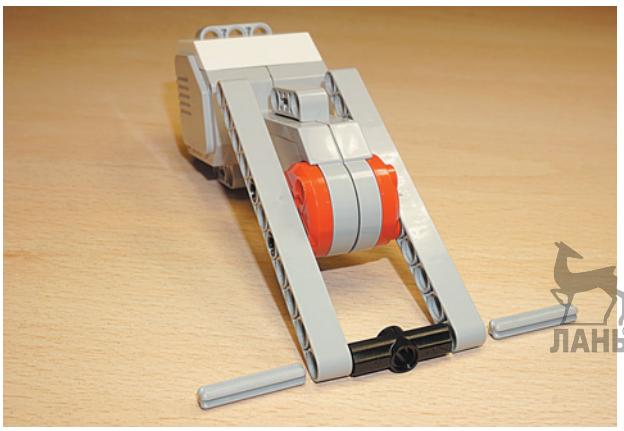
6. Возьми второй мотор. С обеих сторон привода в круглые отверстия вставь симметрично по два чёрных штифта. Этот мотор управляет углом тангажа самолёта. Мы будем называть его *вертикальным*.

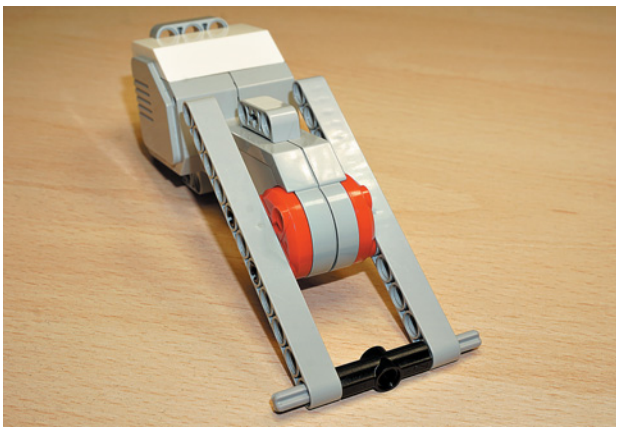


7. Закрепи на них балки № 11 пятым и седьмым модулями.

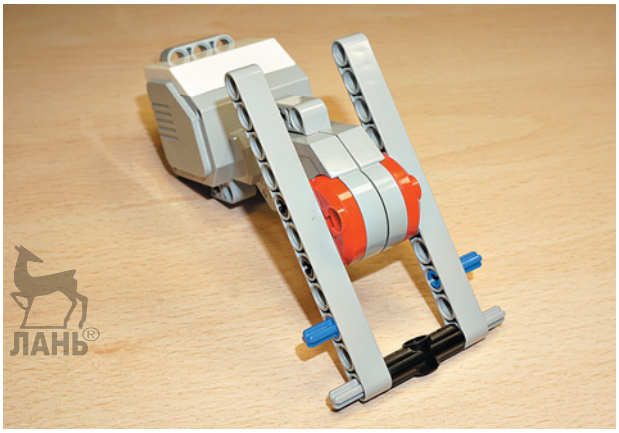


8. Вложи чёрную втулку между концами балок так, чтобы её центральный модуль «смотрел» на мотор. С обеих сторон закрепи её осями № 3.

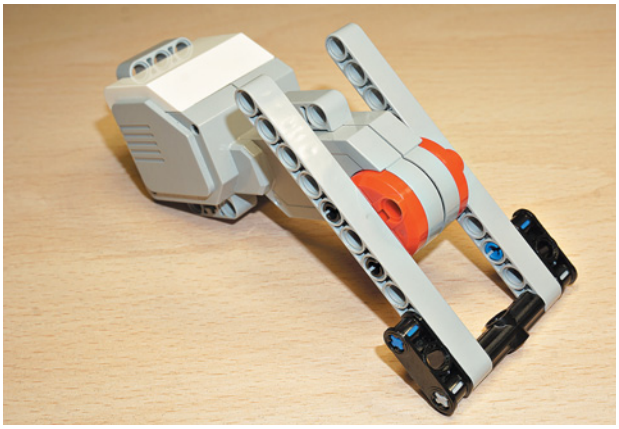




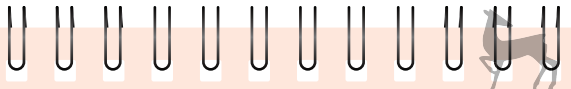
9. В третий модуль балок № 11 вставь по одному синему 2-модульному штифту.



10. С обеих сторон на штифте и на выступающем конце оси закрепи по чёрному фиксатору.

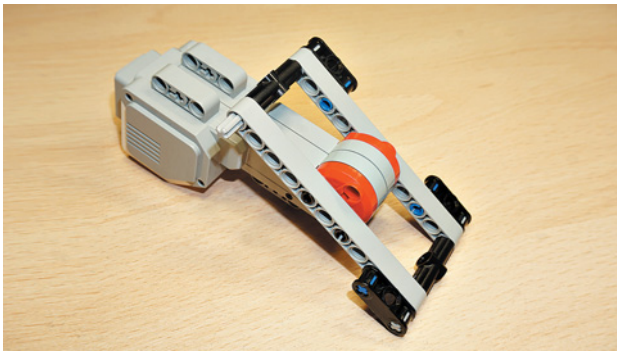


11. Переверни мотор «вверх дном» и прокрути рулевой механизм так, чтобы чёрная втулка опустилась на стол.

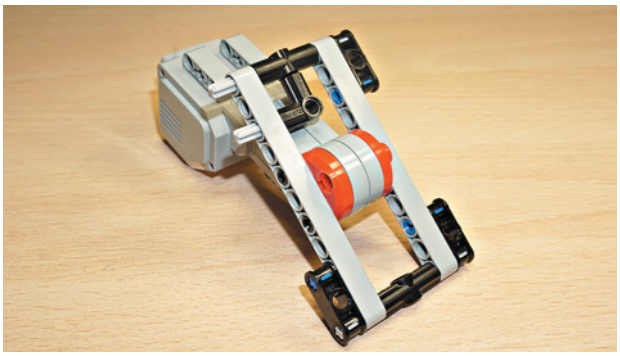


Внимание!
Для продолжения сборки мотор должен находиться в таком положении!

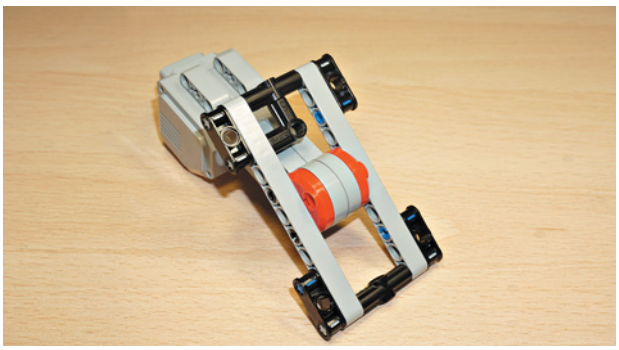
12. Собери на свободных концах балок № 11 такую же конструкцию, как внизу, только с левой стороны не ставь синий штифт и фиксатор.



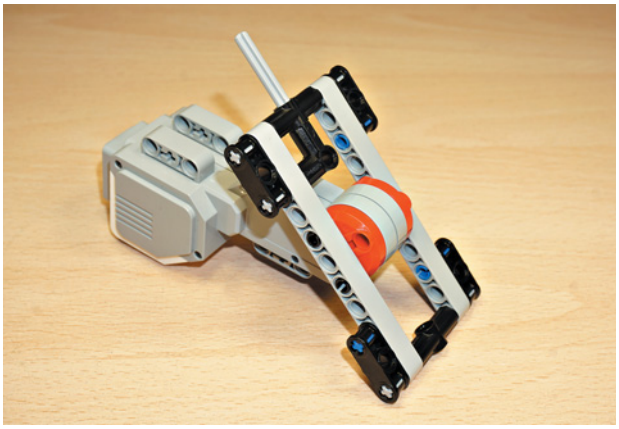
13. В девятый модуль левой балки вставь ось № 3 и придержи её рукой. На внутренний конец оси надень угловую втулку, как показано на рисунке.



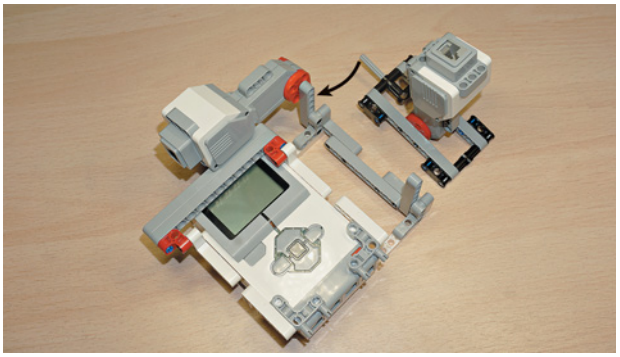
14. На выступающие концы осей надень фиксатор.

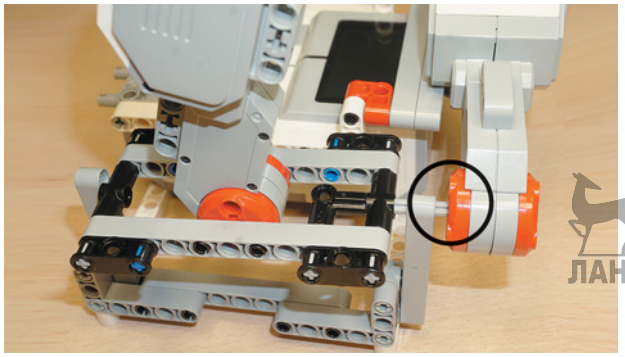


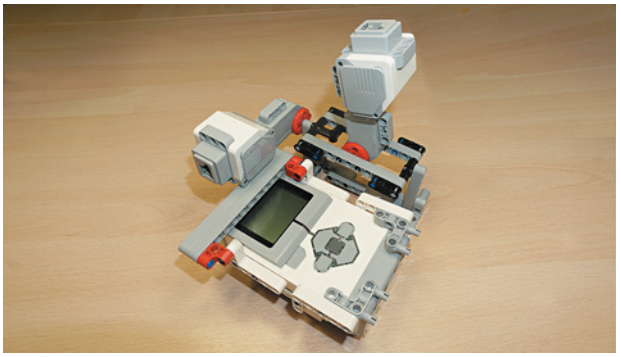
15. Продень ось № 5 сверху так, чтобы она прошла в обе втулки.



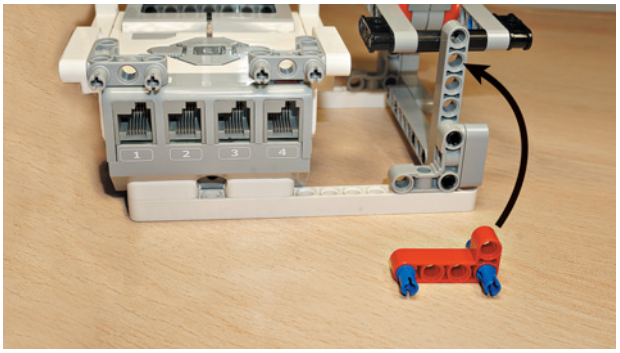
16. Получившуюся конструкцию присоедини к приводу горизонтального мотора. Для этого выступающий конец оси № 5 продень через верхний модуль задней балки основания рулевого механизма так, чтобы он вошёл в центральный крестообразный модуль привода.



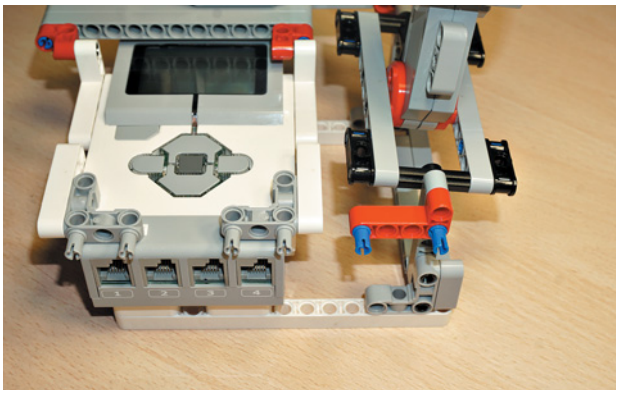




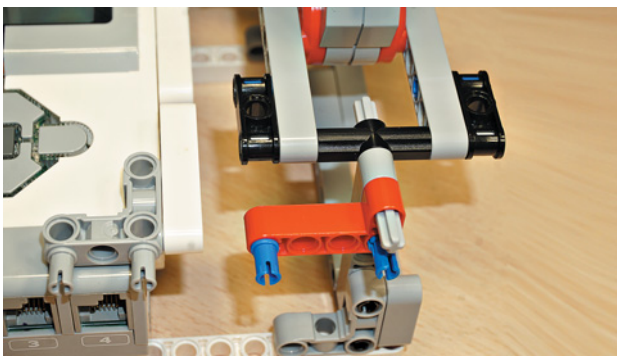
17. Возьми красную прямоугольную балку и в её угловой модуль вставь синий 3-модульный штифт длинным концом вперёд, а в крестообразный модуль — синий 2-модульный.



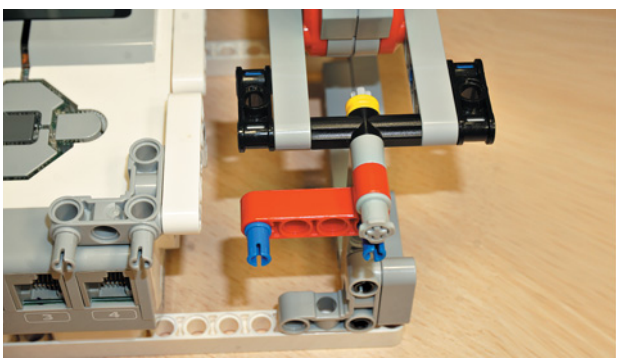
18. Установи получившийся элемент на передней балке основания во втором модуле сверху.



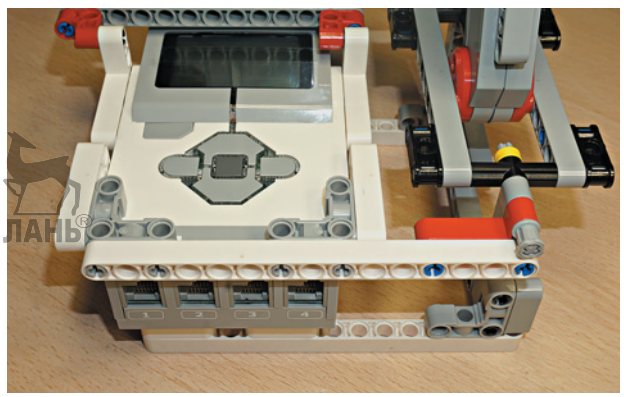
19. Продень ось № 5 через верхний модуль красной балки так, чтобы она прошла в центральный модуль чёрной втулки рулевого механизма.



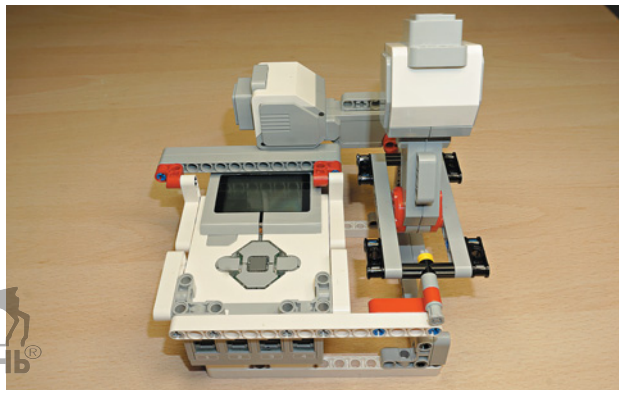
20. Зафиксируй эту ось жёлтой и серой втулками.



21. На свободных штифтах установи балку № 15.



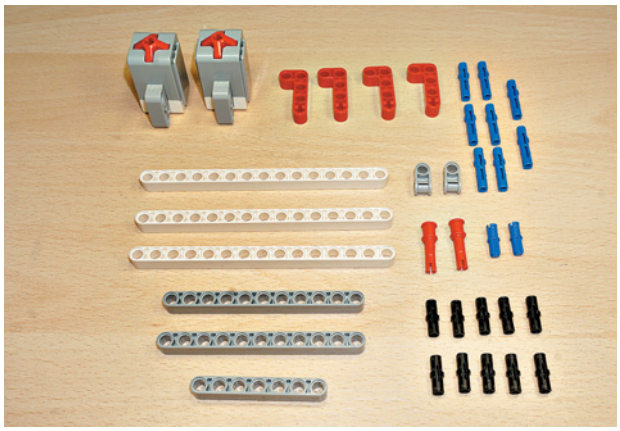
Рулевой механизм готов!



ШАГ 4. СБОРКА ШТУРВАЛА

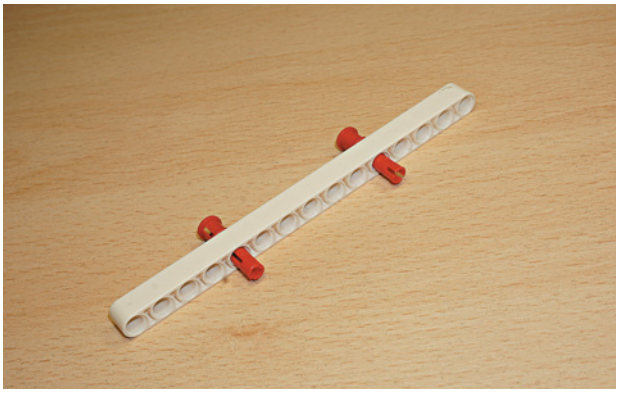
Детали для сборки:

- балка № 15, белая, 3x;
- балка № 11, серая, 2x;
- балка № 7, серая, 1x;
- балка прямоугольная 2 × 4, красная, 4x;
- соединительный штифт с втулкой, 3-модульный, красный, 2x;
- поперечный блок, 2-модульный, серый, 2x;
- соединительный штифт, 2-модульный, синий, 2x;
- соединительный штифт, 2-модульный, чёрный, 10x;
- соединительный штифт, 3-модульный, синий, 8x;
- датчик касания, 2x.

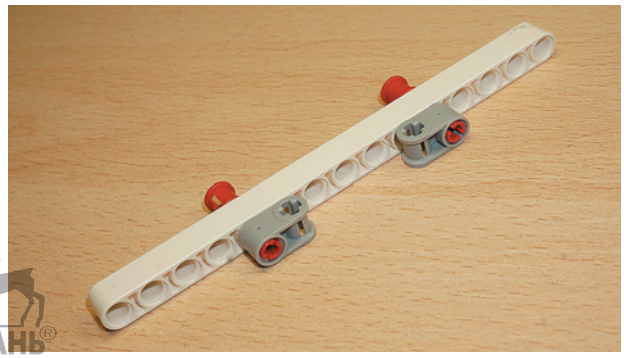


Штурвал состоит из трёх частей: центральной и двух одинаковых боковых. Начни с центральной части.

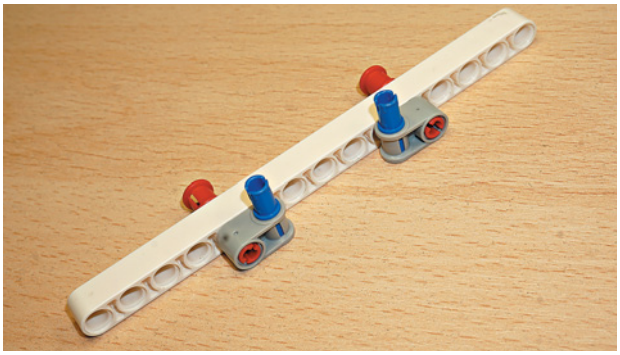
1. В пятый и одиннадцатый модули балки № 15 с обратной стороны вставь по одному красному штифту с втулкой.



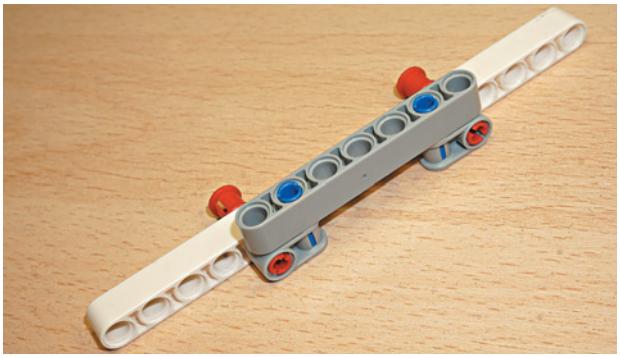
2. На их концы надень серые поперечные блоки.



3. В крестообразные модули блоков вставь синие 2-модульные штифты.

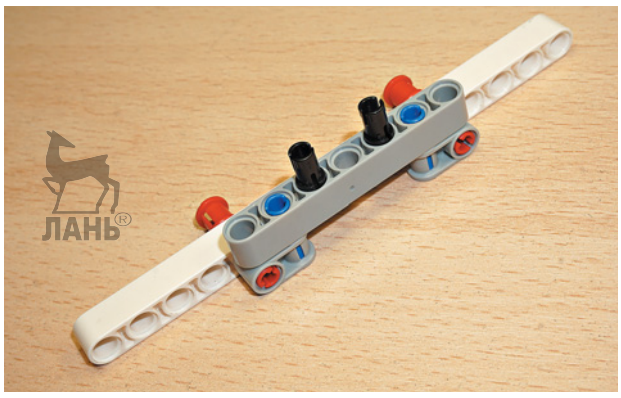


4. Закрепи балку № 7 на этих штифтах вторым и шестым модулями.



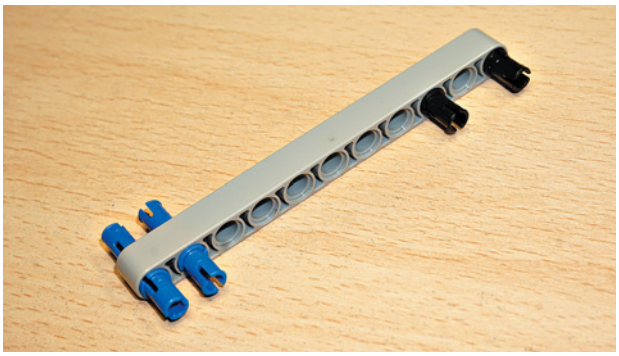
5. В третий и пятый модули этой балки штурвала вставь по одному чёрному штифту.

Центральная часть штурвала готова!

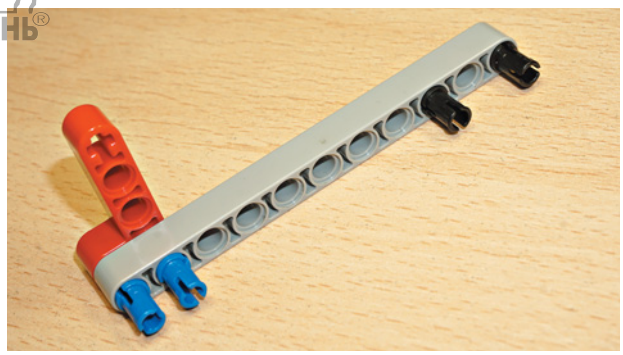


Теперь собери левую часть штурвала.

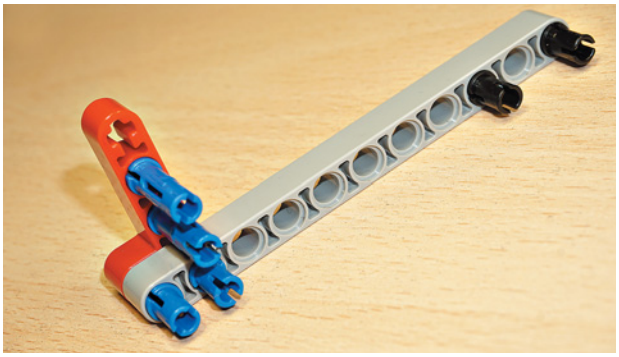
1. В первый и второй модули балки № 11 вставь синие 3-модульные штифты длинными концами вперёд, а в девятый и одиннадцатый модули — чёрные штифты.



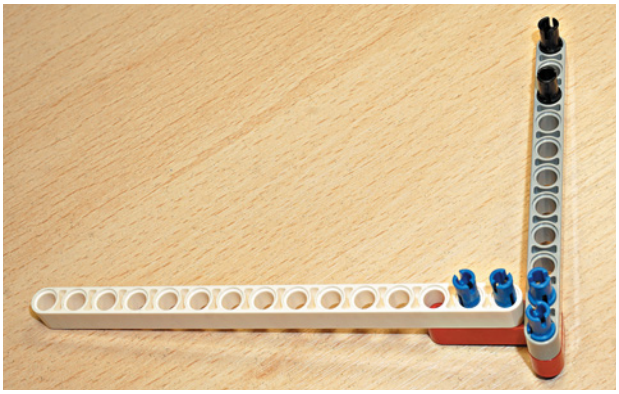
2. На синих штифтах с обратной стороны установи красную балку, как показано на рисунке.



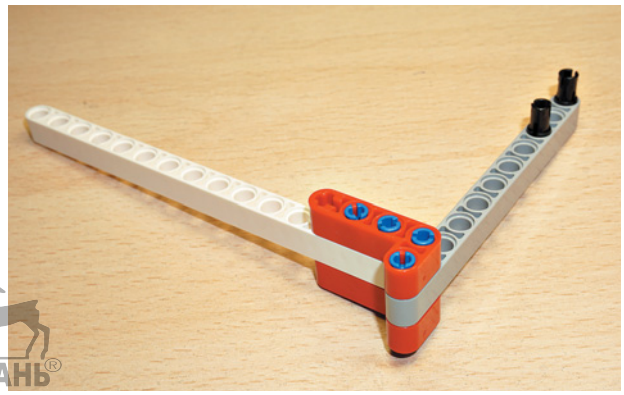
3. В её свободные модули вставь два синих 3-модульных штифта короткими концами вперёд.



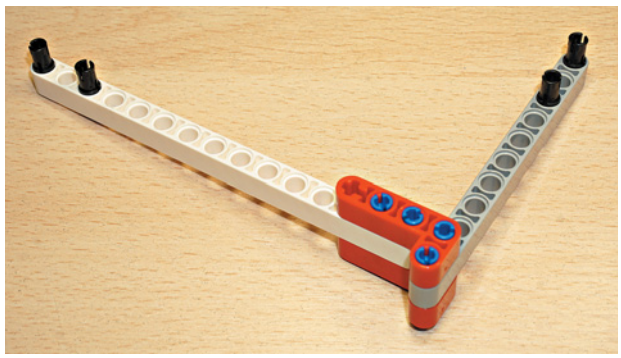
4. На этих двух штифтах закрепи балку № 15.



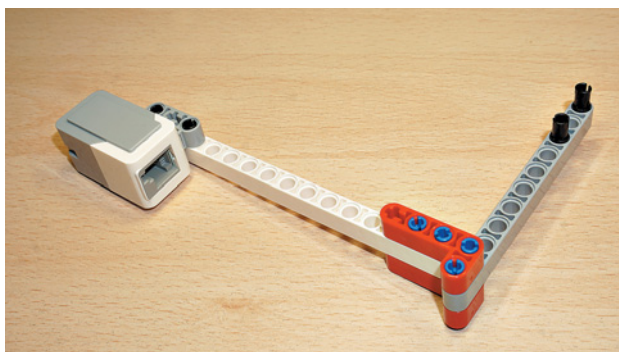
5. Вторую красную балку надень на свободные концы синих штифтов.



6. В первый и третий модули балки № 15 вставь чёрные штифты.

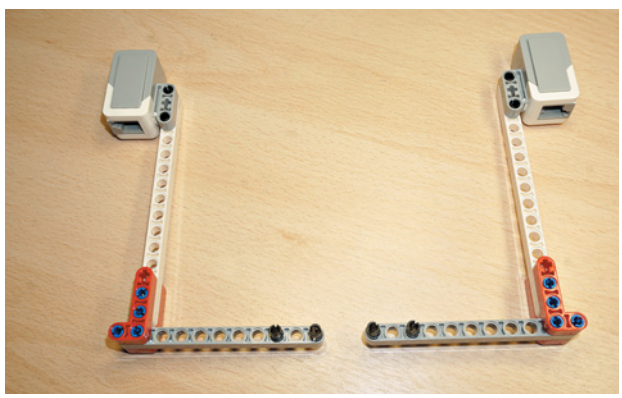


7. Закрепи на этих штифтах датчик касания.

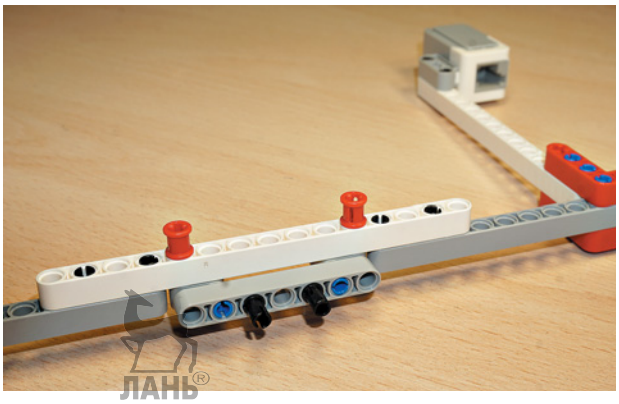


Левая часть штурвала готова!

Правую часть штурвала собери зеркально.



Получившиеся три части штурвала соедини вместе, как показано на рисунке.

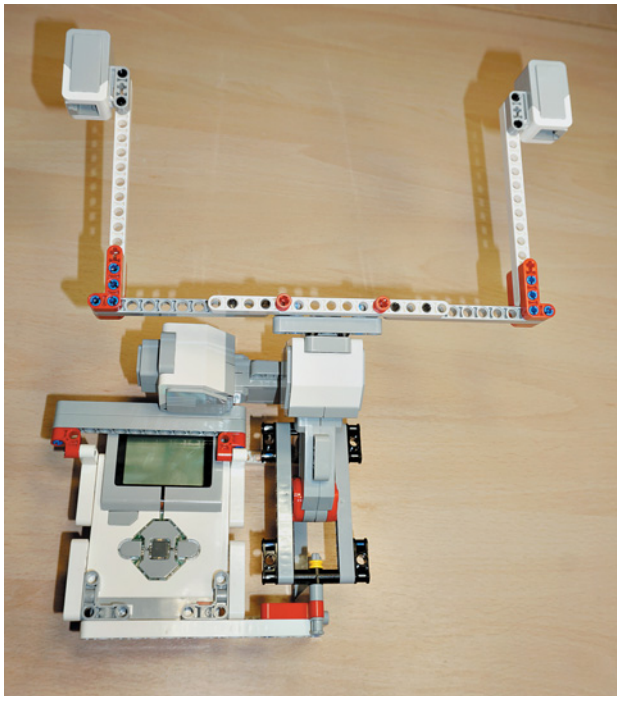


Сборка штурвала завершена!

ШАГ 5. СОЕДИНЕНИЕ РУЛЕВОГО МЕХАНИЗМА СО ШТУРВАЛОМ

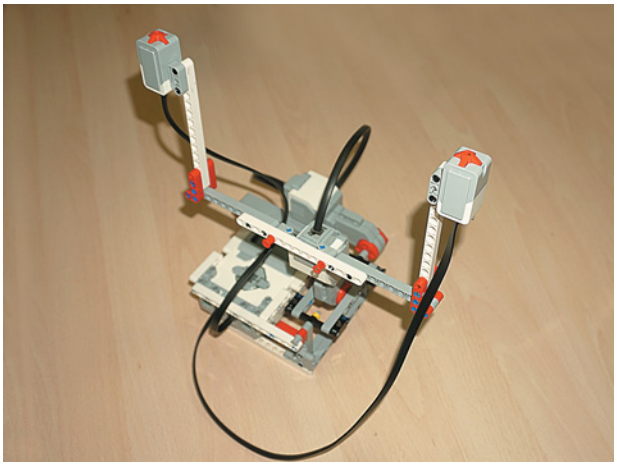
Детали для сборки:

- кабель, 50 см, 1x;
- кабель, 35 см, 2x;
- кабель, 25 см, 1x.

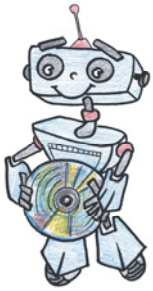


1. Закрепи штурвал на рулевом механизме, вставив выступающие чёрные штифты в модули на вертикальном (см. с. 16, пункты 5 и 6) моторе.

2. Подключи кабели:
   - горизонтальный мотор кабелем длиной 25 см к порту B;
   - вертикальный мотор кабелем длиной 35 см к порту C;
   - левый датчик касания кабелем длиной 35 см к порту 3;
   - правый датчик касания кабелем длиной 50 см к порту 2.



Молодец! Ты справился со сборкой авиасимулятора.



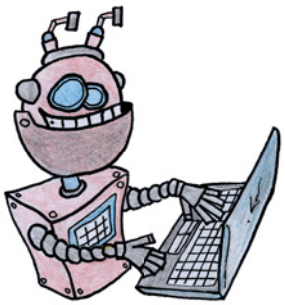
Этап 3. Установка программного обеспечения на компьютере

1. Если ты приобрёл базовый набор LEGO® MINDSTORMS® Education EV3 (LME-EV3) с лицензией на программное обеспечение LME-EV3, то действуй так, как написано в информационном листке, вложенном в набор.

2. Если такой лицензии у тебя нет, зайди на сайт **http://education.LEGO.com** и перейди в раздел «Техническая поддержка», где ты сможешь скачать установочный файл LME-EV3.

Внимание!
При любых затруднениях с установкой программного обеспечения обращайся в службу технической поддержки компании LEGO® Education на сайте **http://education.LEGO.com**

Этап 4. Создание программы для робота-авиасимулятора



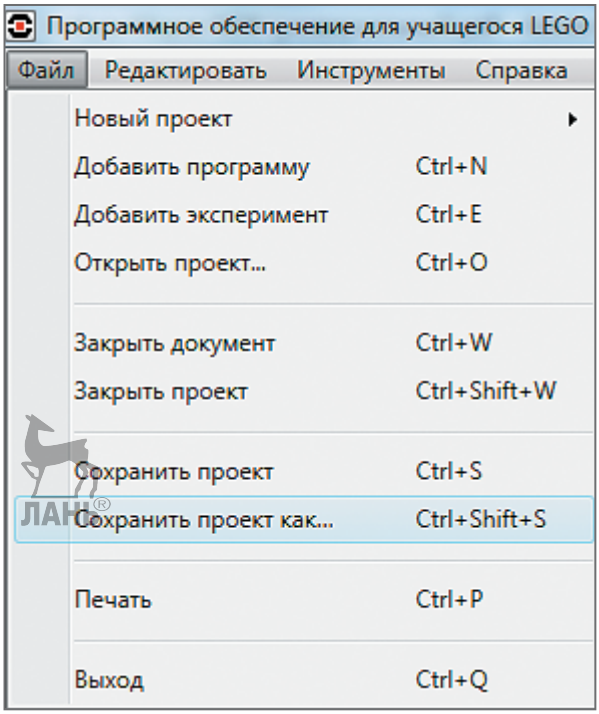
ЗАПУСК ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ LME-EV3

1. Запусти программное обеспечение LME-EV3.
2. В открывшемся окне слева выбери пункт меню **Файл**.
3. Выбери пункт **Новый проект** → **Программа** → **Открыть**.

СОЗДАНИЕ НОВОГО ПРОЕКТА В ПАМЯТИ EV3

Программируемый модуль EV3 позволяет хранить в своей памяти десятки проектов. Это очень удобно: если захотел вернуться к какому-то проекту, не нужно искать, где он записан.

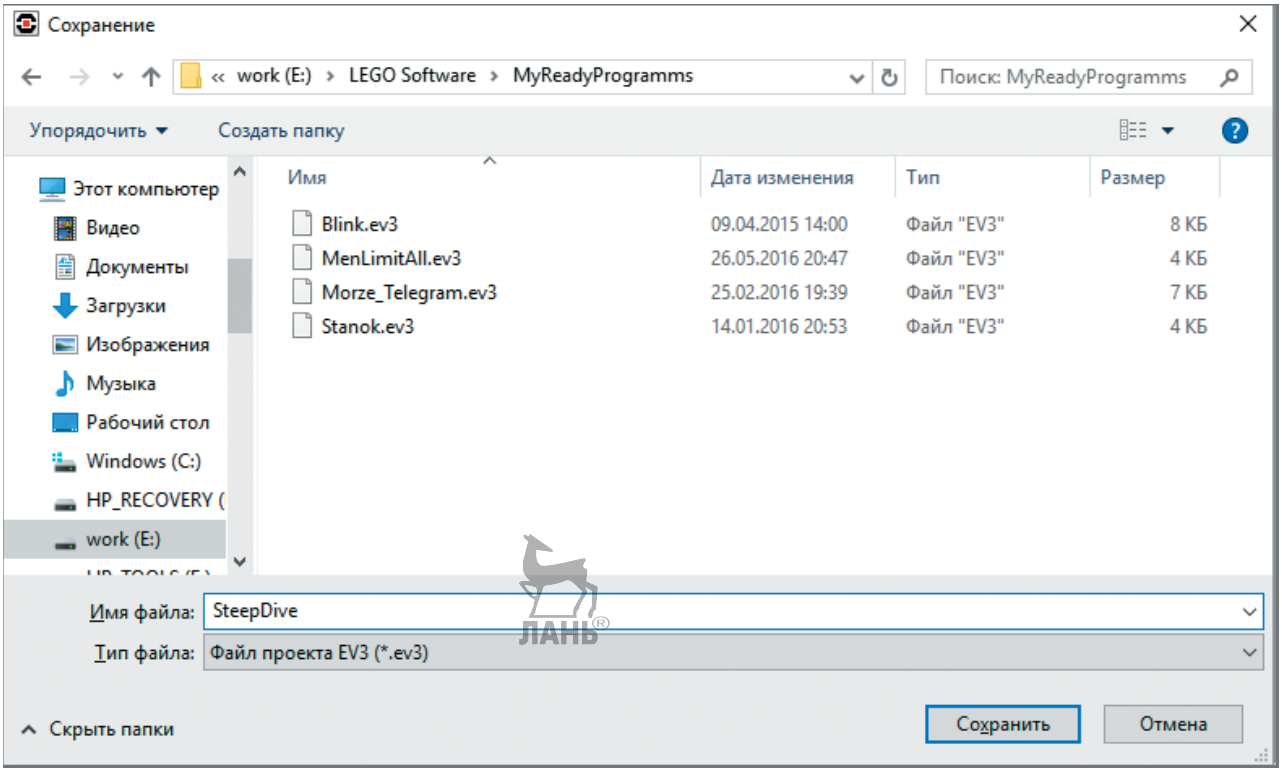
1. Для начала сохрани твой проект в памяти компьютера, чтобы избежать его потери. Для этого выбери меню **Файл** в левом верхнем углу окна программы и в контекстном меню выбери пункт **Сохранить проект как…**



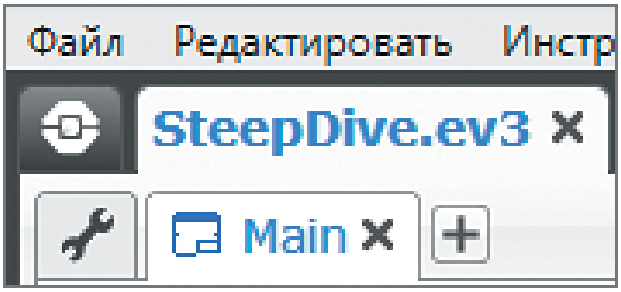
2. В открывшемся окне выбери удобное расположение для всех файлов твоей программы, затем задай **Имя файла**: *SteepDive* (от английского «Крутое пике») в нижней части окна и нажми **Сохранить**. Именно так будет называться в памяти программируемого модуля EV3 папка для хранения программ, которые ты напишешь в рамках этого проекта.

Внимание!

Настоящие программисты всегда дают имена своим программам на английском языке и без использования пробелов. Вместо них они пишут, например, каждое слово с большой буквы (как это сделал и ты) или добавляют нижнее подчёркивание.



3. Осталось дать имя самой программе (в нашем проекте их будет несколько). Для этого дважды кликни по имени *Program* в левом верхнем углу поля программы и задай новое имя: *Main* (от английского «Главный», именно так программисты называют основную программу, если она в проекте не одна) и нажми клавишу **Enter**.



ЛОГИКА ПРОГРАММЫ

Так как мы создаём настоящий авиасимулятор, то программа получится довольно сложной и объёмной. Причём она будет содержать внутри себя другие, совершенно самостоятельные подпрограммы. Поэтому программирование всего устройства мы разобьём на семь частей и опишем логику для каждой из них в отдельности:

Кстати!
В этой программе будет множество интересных алгоритмических конструкций. Самой первой из них будет **подпрограмма**.

Часть 1. Исходные положения. Переменные и начальные параметры.
Часть 2. Двигатели — на старт! Увеличение и уменьшение мощности турбин.
Часть 3. Скорость, крен, тангаж — завернём крутой вираж! Показания спидометра и авиагоризонта.
Часть 4. Тревога! Тревога! Система сигнализации об опасном уровне тангажа.
Часть 5. Бесконечность — не предел! Работа одометра, расчёт пройденного расстояния.
Часть 6. Всё выше, и выше, и выше! Работа альтиметра, расчёт набранной высоты.
Часть 7. Последний рывок. Вывод на экран показаний всех приборов, завершение.

СОСТАВЛЕНИЕ ПРОГРАММЫ ДЛЯ РОБОТА-АВИАСИМУЛЯТОРА

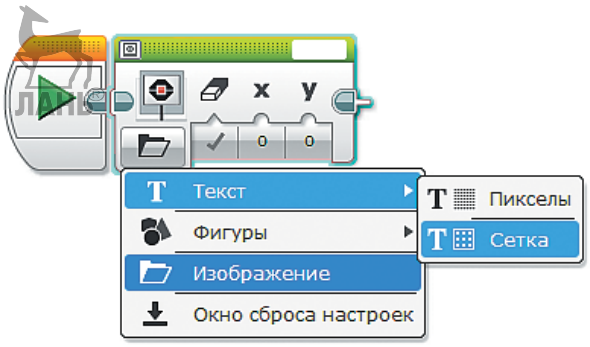
В открывшемся окне проекта начинай составлять программу для робота-авиасимулятора.

Часть 1. Исходные положения. Переменные и начальные параметры

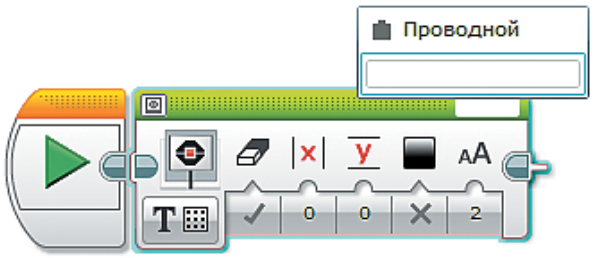
Логика. В этой части ты инициализируешь в программе несколько переменных, которые понадобятся для ведения подсчётов. Кроме того, ты очистишь экран (ещё его называют «дисплей») программируемого модуля и подготовишь к работе параметр времени.

1. Для начала нужно очистить экран от всех надписей. Для этого добавь команду **Экран** (*зелёный блок*).

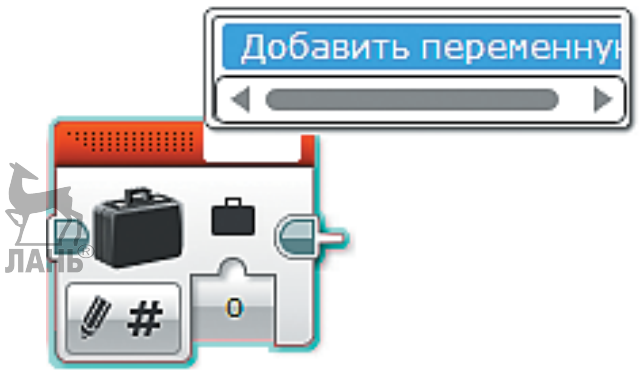
2. Выбери опцию **Текст** → **Сетка**.



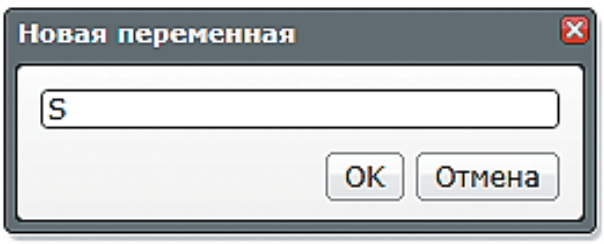
3. В правом верхнем углу, где написано слово MINDSTORMS, кликни мышью, сотри эту надпись и нажми клавишу **Enter**. Именно так ты очистишь экран, по сути выведя на него «пустой» текст.



4. Теперь нужно объявить (то есть создать в программе) сразу четыре переменные. Добавь команду **Переменная** (*красный блок*) и в правом верхнем окошке выбери пункт **Добавить переменную**.



5. В открывшемся окне введи с клавиатуры название первой переменной:
   - **S** (от латинского *Spatium* — расстояние) — в этой переменной будет рассчитываться и храниться значение пройденного расстояния, то есть показания прибора **Одометр**. Нажми **ОК**.

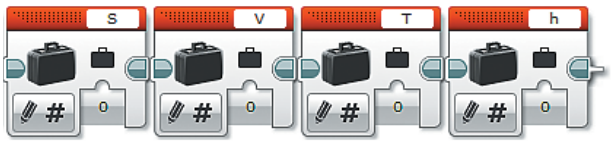


   - Опцию **Записать числовое значение** и нуль на входе данной команды оставь без изменений. Это означает, что ты создал переменную и **инициализировал** её, то есть записал начальное значение, равное нулю.

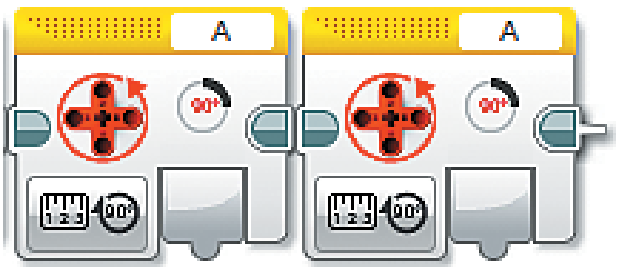
6. Остальные переменные добавь аналогичным способом, задавая им имена:
   - **V** (от латинского *Velocitas* — скорость) — здесь будет рассчитываться и храниться значение текущей скорости самолёта, то есть показания прибора **Спидометр**.

- **T** (от латинского *Tempus* — время) — в этой переменной будет храниться значение времени, которое тратится на обработку одного измерения приборов (в нашем случае мы возьмём одну секунду).
- **h** (от английского *Height* — высота) — здесь будет храниться временное значение угла тангажа для расчёта высоты.
  Обрати внимание! Эта переменная именована строчной (маленькой) буквой.

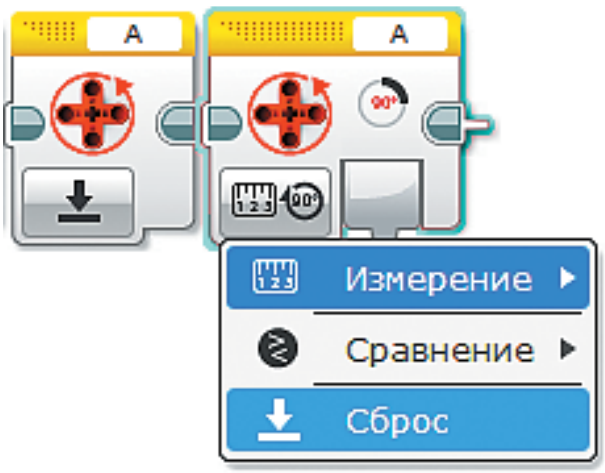
7. В итоге должна получиться такая цепочка команд.



8. Далее тебе нужно сбросить показания моторов, то есть обнулить значения энкодеров[1]. Для этого добавь две команды **Вращение мотора** (*жёлтый блок*).

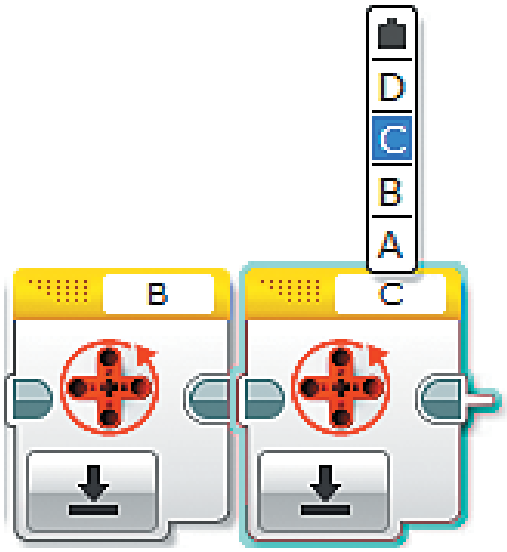


9. Выбери в обеих командах опцию **Сброс**.

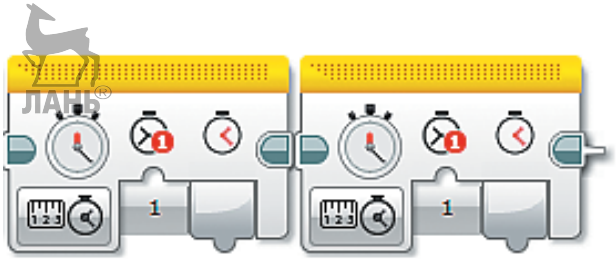


[1] Энкодер (от английского *Encoder* — кодировщик) — устройство внутри мотора, позволяющее снимать различные показания: число совершённых оборотов или угол поворота привода.

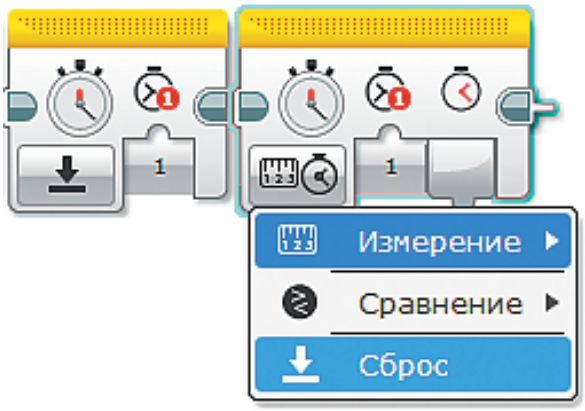
10. Теперь осталось проверить и при необходимости задать порты моторов. При сборке они были подключены к портам **B** и **C**, поэтому в правых верхних окошках команд **Вращение мотора** должны быть заданы именно эти порты.



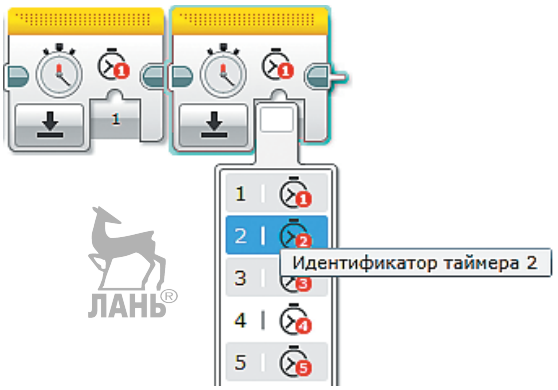
11. Последними командами перед началом основной программы будут команды таймера. Нам необходимо инициализировать два таймера, каждый из которых будет отвечать за частоту обновления показаний приборов на экране EV3, выполняющего роль приборной панели самолёта.
Добавь две команды **Таймер** (*жёлтый блок*).



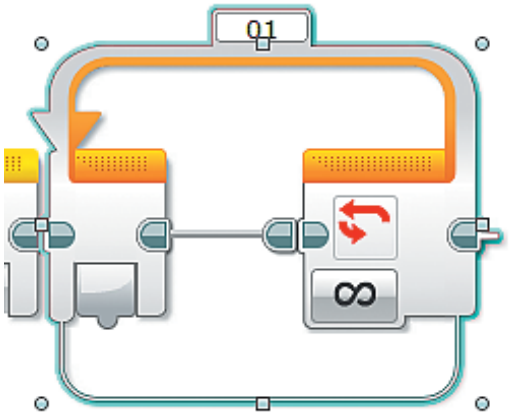
12. В каждой из них выбери опцию **Сброс**.



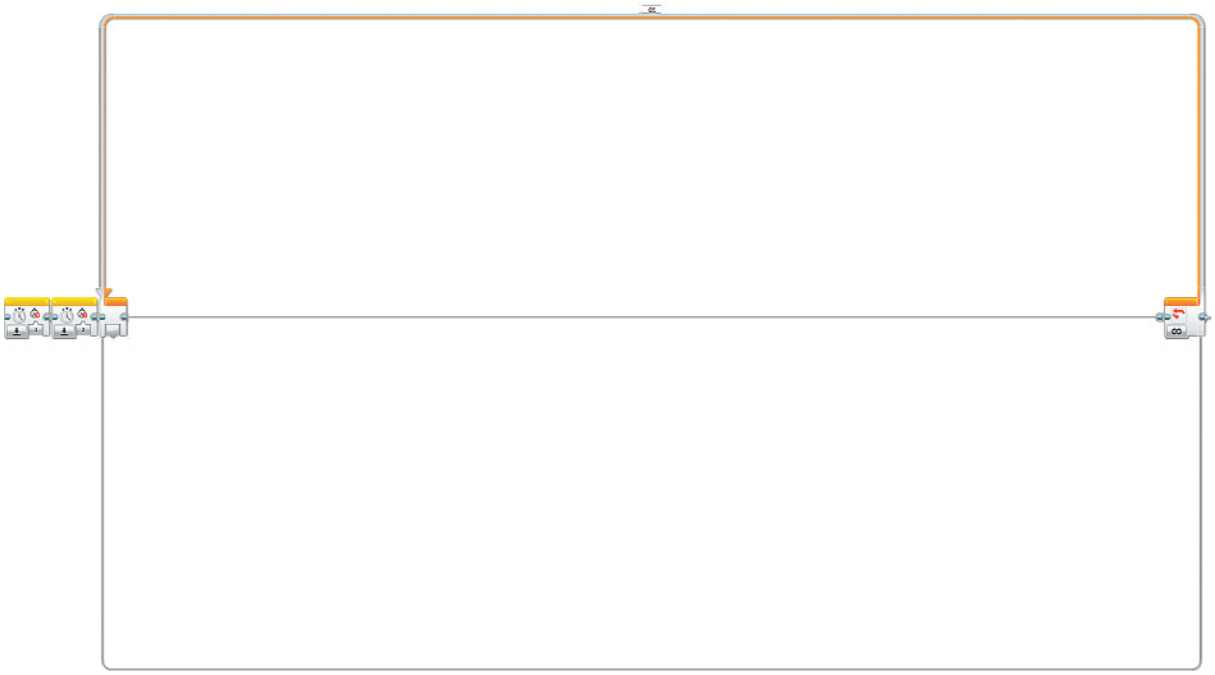
13. В первой команде **Индентификатор таймера** оставь **1**, а во второй выбери **2**. Первый таймер будет отвечать за показания одометра, а второй — за другие приборы.



14. Теперь время приступить к основной программе. Она будет заключена в большом бесконечном **Цикле** (*оранжевый блок*), который тебе и надо добавить.

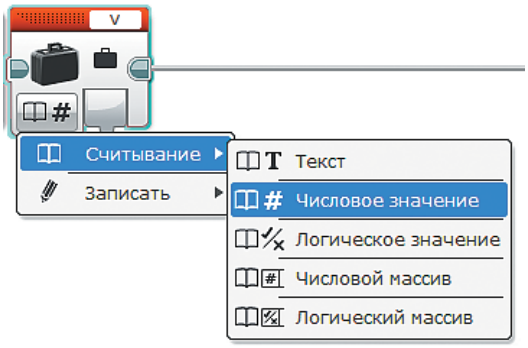


15. Программа внутри цикла будет очень большой, поэтому целесообразно его сразу расширить. Это можно сделать, потянув курсором мыши за кружочки и квадратики, окружающие цикл. В результате должно получиться примерно так:

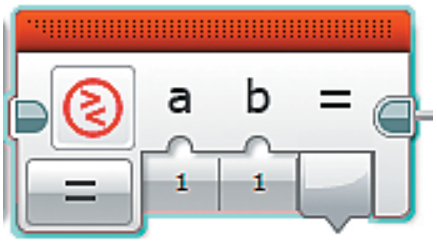


16. Далее все команды добавляются внутрь этого цикла, и первой из них будет команда **Переменная** (*красный блок*). В правом верхнем углу выбери для обработки переменную **T**, опцию и входное значение оставь без изменений — каждый повтор цикла эта переменная будет обнуляться.

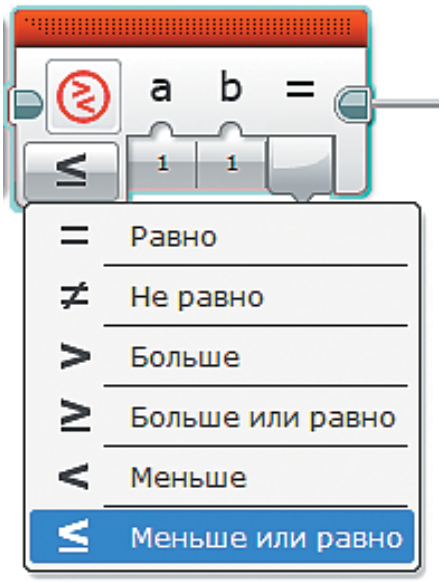
17. Следующей командой тоже будет **Переменная** (*красный блок*), в которой будет обрабатываться переменная **V**. Выбери в ней опцию **Считывание → Числовое значение.**



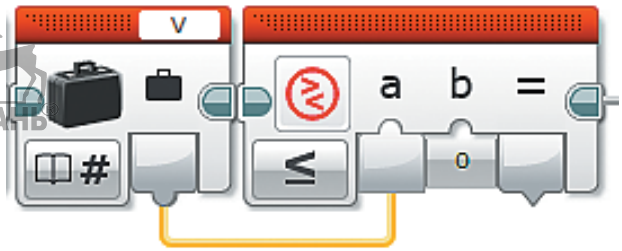
18. Теперь добавь команду **Сравнение** (*красный блок*).



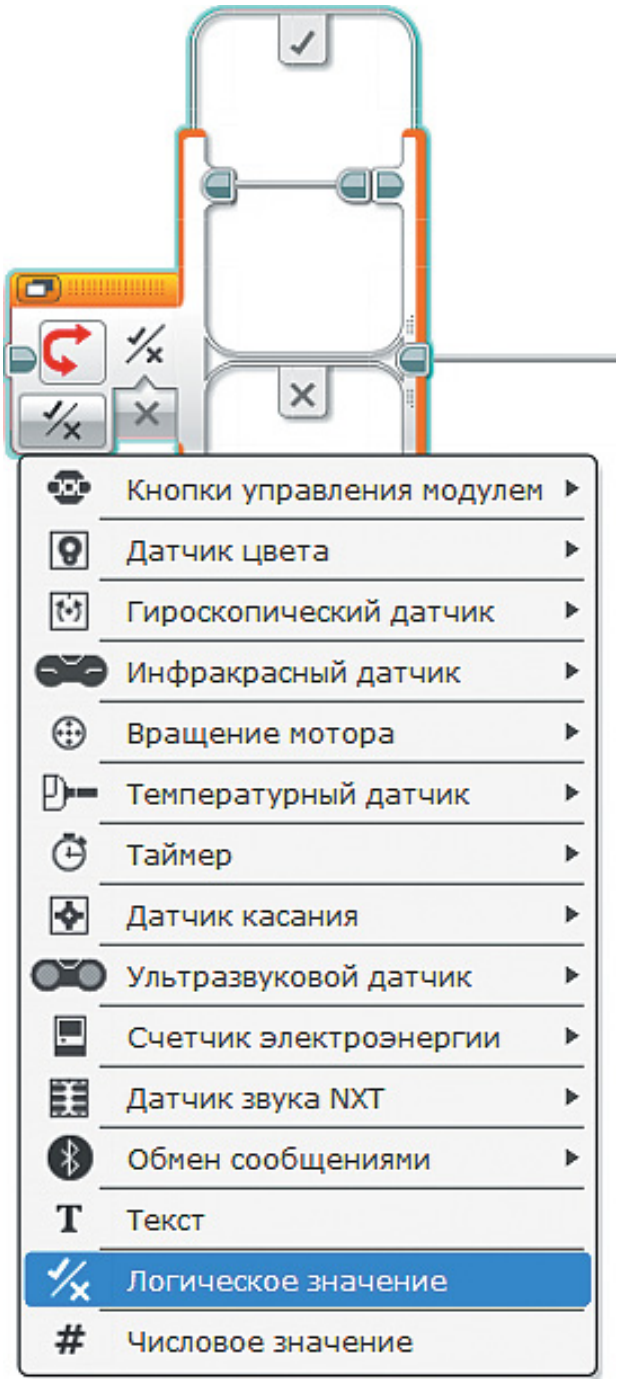
19. Выбери опцию **Меньше или равно**.



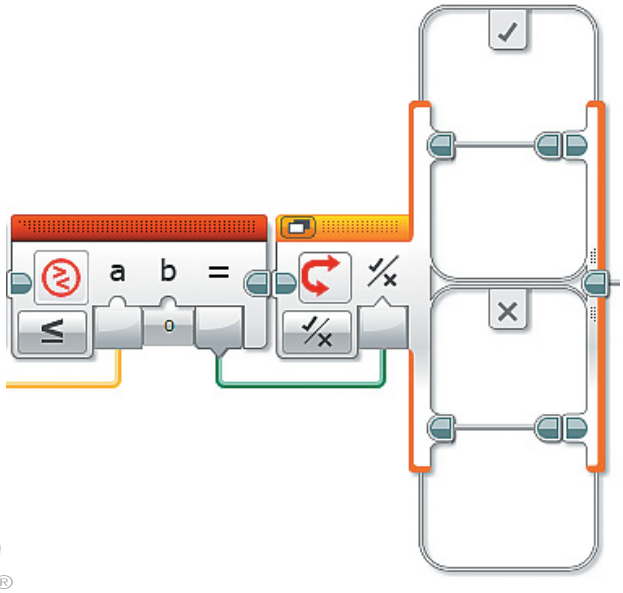
20. Соедини выход переменной **V** со входом **a** команды **Сравнение**. Во вход **b** с клавиатуры введи **0**.



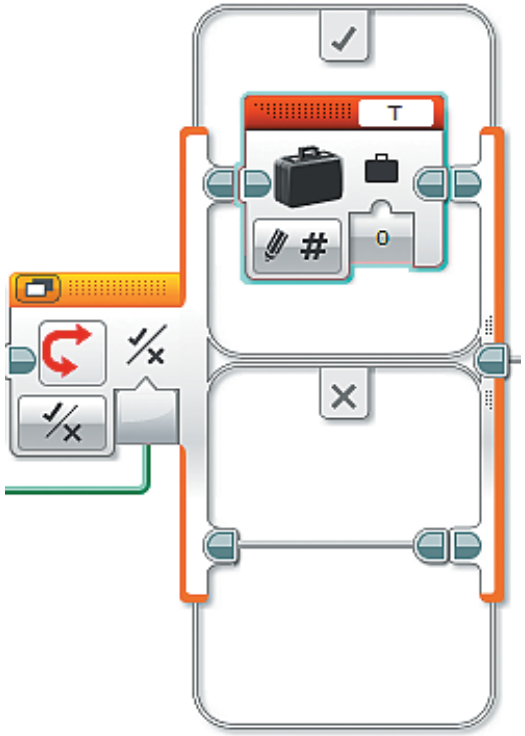
21. Добавь команду **Переключатель** (*оранжевый блок*) и выбери опцию **Логическое значение**.



22. Соедини выход «=» команды **Сравнение** со входом **Переключателя**.



23. Последние три команды позволяют тебе проверить следующее условие: «*Если значение скорости меньше или равно нулю, то...*»; в случае истинности (выполнения) этого условия, необходимо **обнулять** переменную **T**, а в случае ложности (невыполнения) устанавливать её равной **0,00027**. Разберёмся.



24. Добавь в верхнюю часть (условие = истина) **Переключателя** команду **Переменная** (*красный блок*) и выбери для обработки переменную **T**. Опцию и входное значение не изменяй.

25. В нижнюю часть **Переключателя** (условие = ложь) добавь такую же команду **Переменная** (*красный блок*) и выбери ту же переменную **T**, но входное значение будет другое: тебе нужно рассчитать пройденное расстояние. Оно вычисляется по формуле $S = V \cdot T$, где S — расстояние в километрах; V — скорость в км/ч; T — время в секундах. Так как значения S и V связаны с километрами и часами, значит, и значение T должно быть в часах. Поэтому переведём секунды в часы

и заодно договоримся, что **частота обновления** показаний приборов на экране будет **1 раз в секунду**, или **1 Герц**[1]. Тебе надо выполнить следующую цепочку преобразований:

Секунды → Минуты → Часы.

Ты знаешь, что секунды — это «маленькие» единицы измерения времени, в отличие от минут и часов. Чтобы узнать, какую часть составляет 1 секунда от 1 часа, посчитай, сколько же всего в нём секунд:

1 час = 60 мин
1 мин = 60 сек
1 час = 1 · 60 мин = 1 · 60 · 60 = 3600 сек.

В итоге: 1 час = 3600 сек.

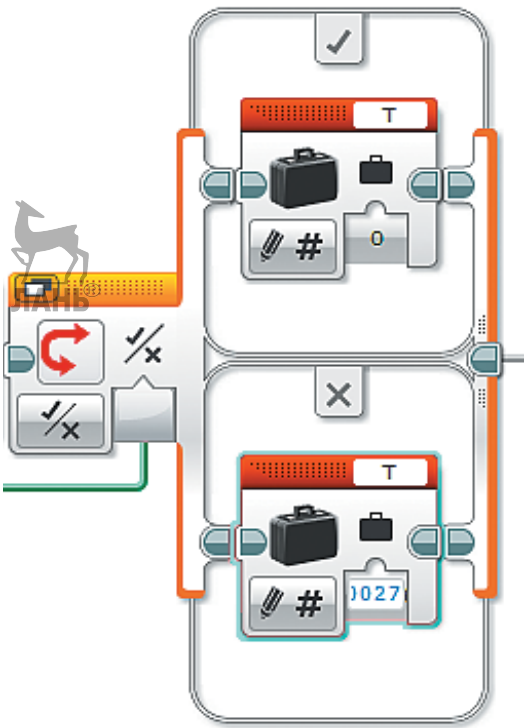
Теперь ты можешь сказать, что 1 секунда — это одна часть из 3600 частей, содержащихся в часе, то есть это обыкновенная дробь вида $\frac{1}{3600}$.

Черта дроби обозначает операцию деления. Разделим единицу на 3600:

1 : 3600 ≈ 0,00027 часа.

Вот так ты получил число для входного значения переменной **T**.

На этом первая часть программы окончена. В ней ты задал все необходимые первоначальные параметры и теперь готов «оживлять» свой авиасимулятор.

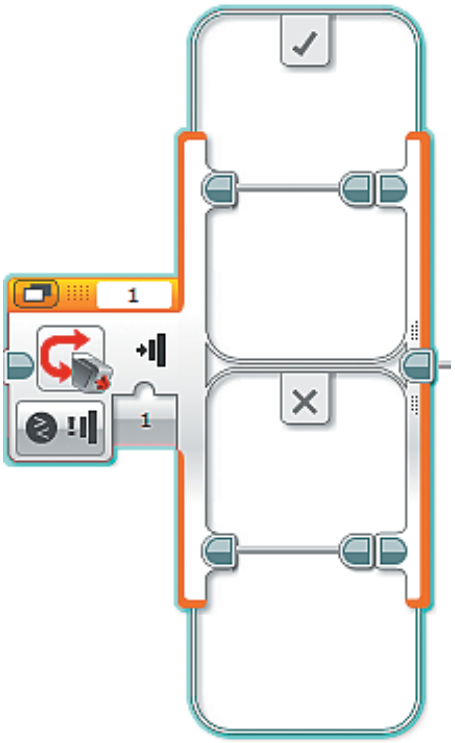


[1] Герц — единица частоты периодических процессов (например, колебаний) в Международной системе единиц (СИ).

Часть 2. Двигатели — на старт! Увеличение и уменьшение мощности турбин

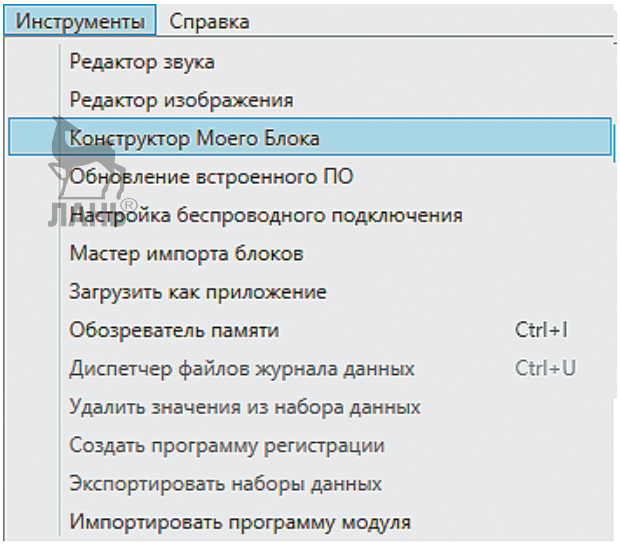
Логика. В этой части ты создашь подпрограмму, которая будет отвечать за работу турбин самолёта и управление ими. Так как самолёт не может равномерно разгоняться на протяжении всего движения, то в зависимости от набранной скорости самолёт будет ускоряться или замедляться по-разному. Для увеличения мощности ты запрограммируешь правый датчик касания на штурвале, а для уменьшения — левый.

Подпрограмма необходима для компактности вида всей программы и удобной работы с разными её частями. Опытные программисты всегда выделяют из основной программы некоторые её части, которые занимаются отдельной обработкой однородных событий. В нашем случае — это обеспечение работы турбин.

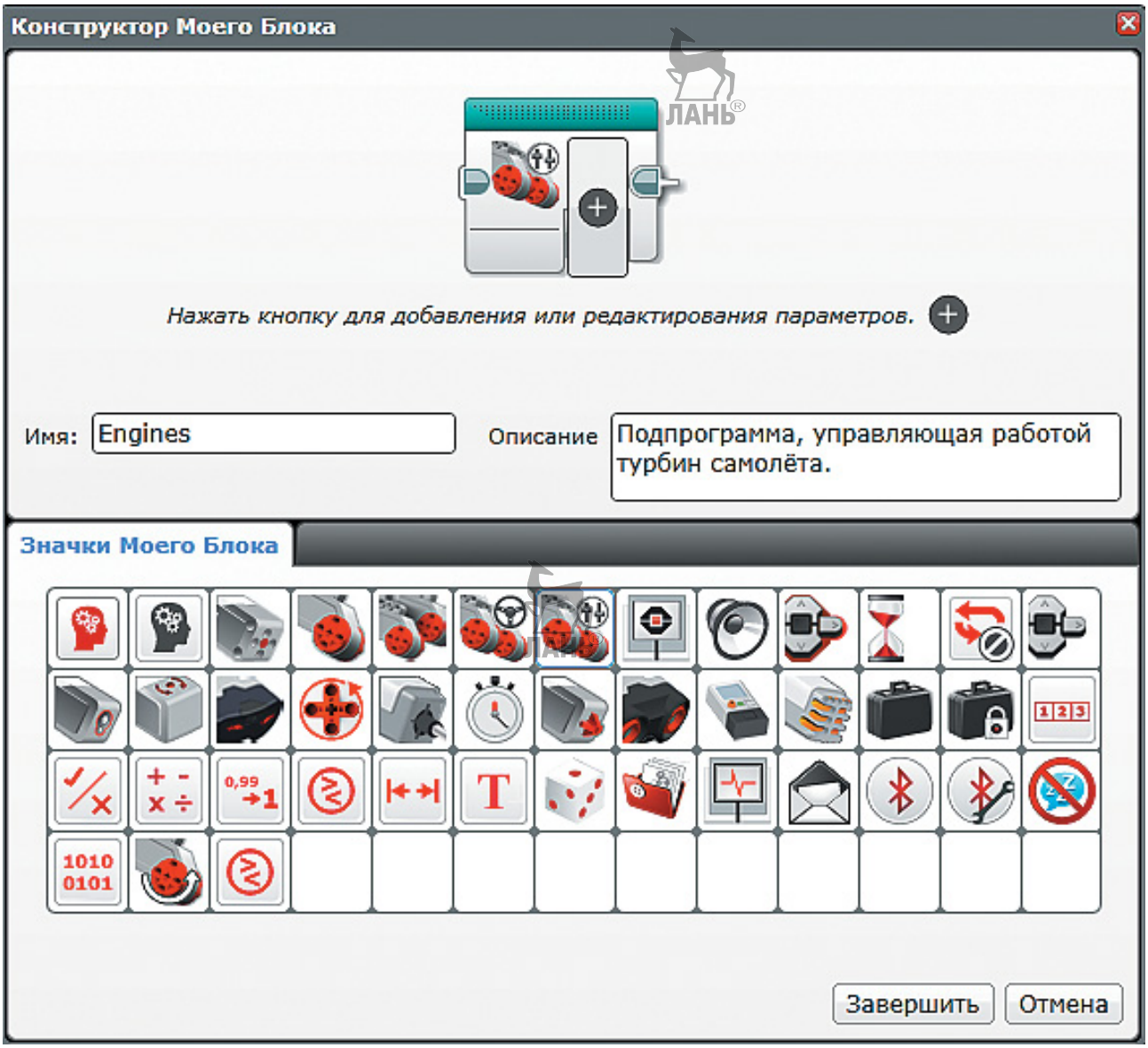


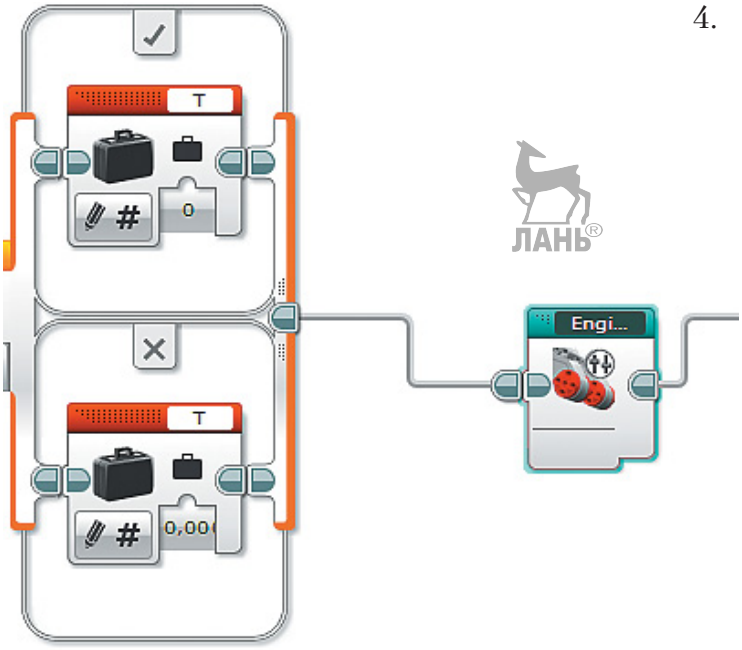
1. Перед тем как приступать к созданию подпрограммы, тебе необходимо сначала добавить хотя бы одну команду из неё. И у тебя это будет команда **Переключатель** (*оранжевый блок*).

2. Когда ты добавляешь новую команду, она выделена голубым контуром, то есть активна. Если этого не произошло, кликни по выбранной команде мышью, и она активируется. Теперь помести эту команду в подпрограмму, то есть создай **свой блок**. Для этого в верхней части окна программы выбери меню **Инструменты** → **Конструктор Моего Блока**.

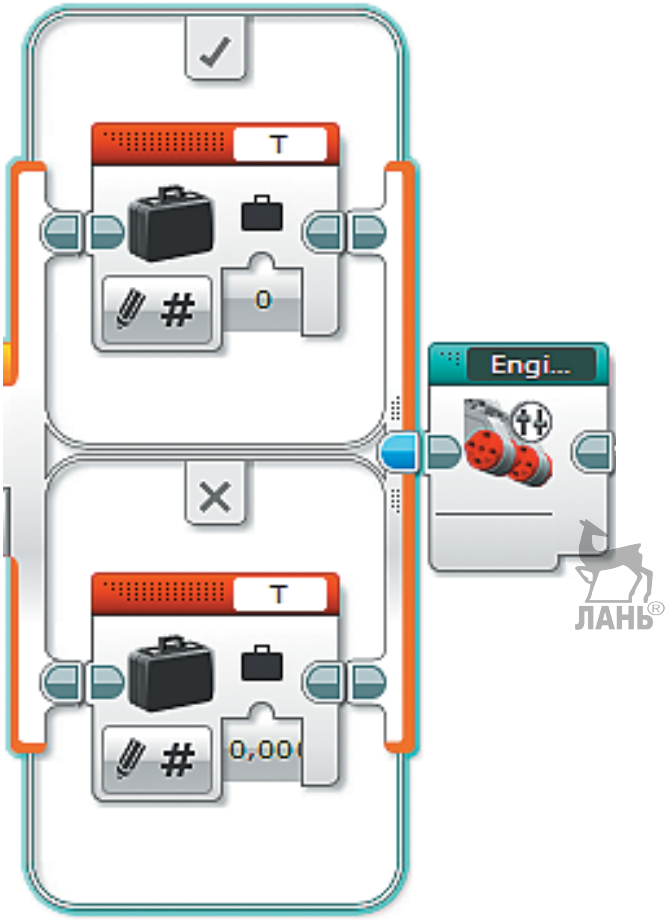


3. В открывшемся окне задай имя подпрограммы, введи её краткое описание и выбери подходящую пиктограмму (маленькое изображение). Для турбин логично задать имя, например, **Engines** (от английского — двигатели).
Помни, чтобы легко ориентироваться в программах, всегда следует называть их сообразно тому, какую функцию они выполняют. Затем нажми кнопку **Завершить**. Краткое описание и пиктограмму можешь задать, как на рисунке или самостоятельно.



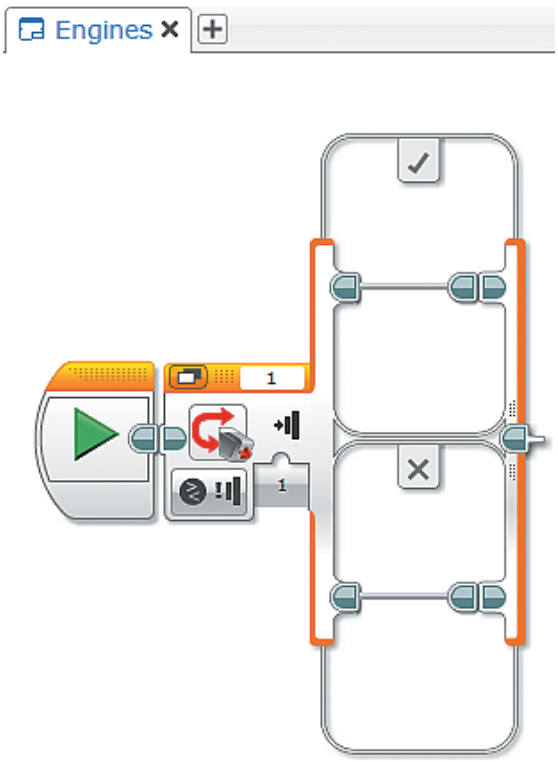


4. В основной программе моментально появится новая команда.

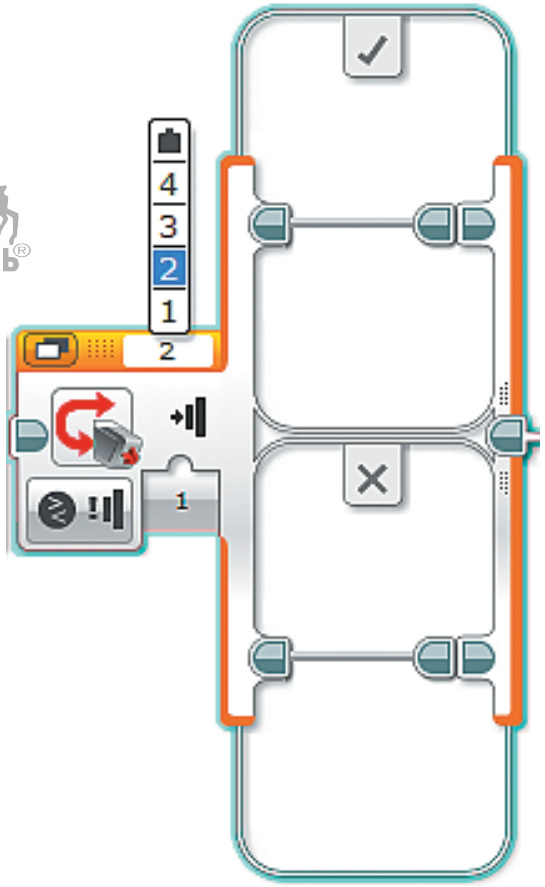


5. Для удобства работы можно убрать расстояние между этими командами, кликнув по выходной связке команды **Переключатель**.

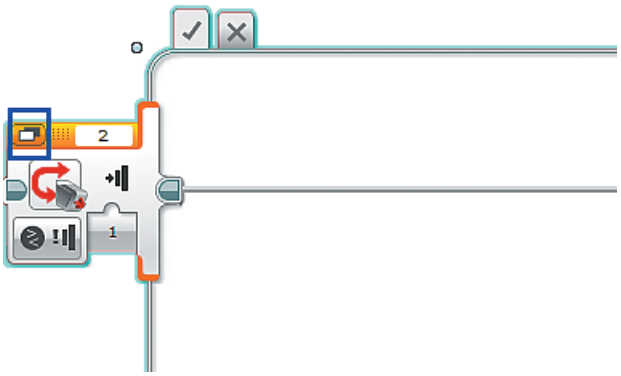
6. Дважды кликни по созданной команде **Engines**. На экране откроется новое рабочее поле для редактирования подпрограммы. Более того, сверху появится специальная вкладка с именем *Engines*.



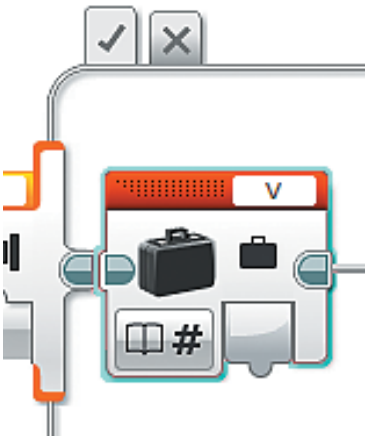
7. Теперь приступай непосредственно к самой подпрограмме. В команде **Переключатель** по умолчанию установлены нужная тебе опция **Датчик касания → Сравнение → Состояние** и проверка состояния датчика (нажат **[1]**). Проверь, чтобы в верхнем правом окошке этой команды стоял номер порта, к которому подключён правый датчик касания. В нашем случае это порт № 2.



8. Для более удобной работы с **Переключателем** переведи его в **Вид с вкладками**, нажав на кнопку в левом верхнем углу этой команды. И сразу расширь эту команду, так как в ней будет содержаться всё остальное.



9. Сначала пропиши все команды для случая, когда условие выполнено, то есть датчик касания, подключённый к порту № 2, нажат. Добавь команду **Переменная** (*красный блок*) и выбери для обработки переменную **V**.
Установи опцию **Считывание** → **Числовое значение**.

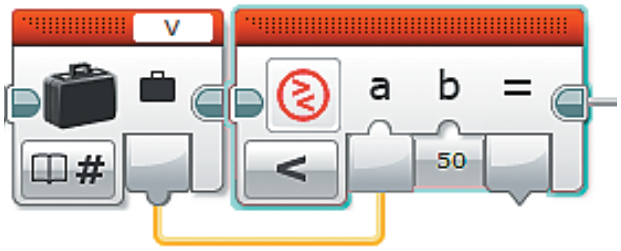


10. Разберёмся, как же мы будем управлять работой турбин. Датчик касания, подключённый к порту № 2, то есть расположенный на штурвале справа, будет отвечать за увеличение мощности. Однако на экран будет выводиться скорость самолёта. Обратимся к инженерной хитрости и будем наращивать скорость в соответствии со следующей таблицей:

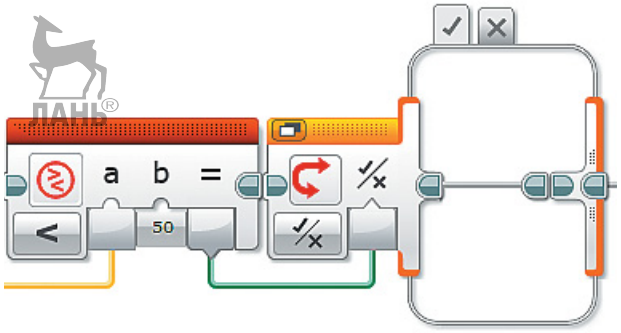
Скорость самолёта, км/ч	**Соответствующее неравенство**	**Коэффициент прироста скорости, км/ч**
Меньше 50	$V < 50$	0,05
От 50 до 100	$50 \le V \le 100$	0,1
От 100 до 200	$100 \le V \le 200$	0,3
От 200 до 300	$200 \le V \le 300$	0,5
От 300 до 600	$300 \le V \le 600$	0,7
От 600 до 850	$600 \le V \le 850$	1

Пояснение. Подпрограмма будет считывать значение переменной **V**, отвечающей за скорость, и проверять, в каком диапазоне сейчас находится её значение. Чем выше скорость, тем «проще» самолёту её набирать; поэтому для разных диапазонов будет разный коэффициент прироста скорости (то есть на сколько будет увеличиваться значение этой переменной в каждый момент времени).

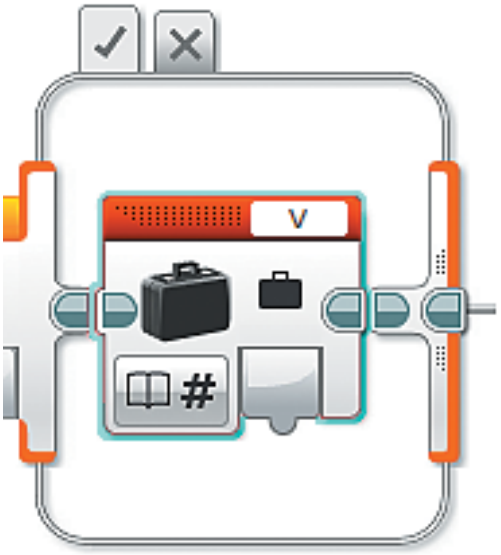
11. Добавь команду **Сравнение** (*красный блок*) и выбери опцию **Меньше**. Соедини выход команды **Переменная** со входом **a** команды **Сравнение**; во вход **b** с клавиатуры введи значение **50**.
Таким образом ты обеспечил проверку самого первого диапазона скорости самолёта с помощью соответствующего неравенства.



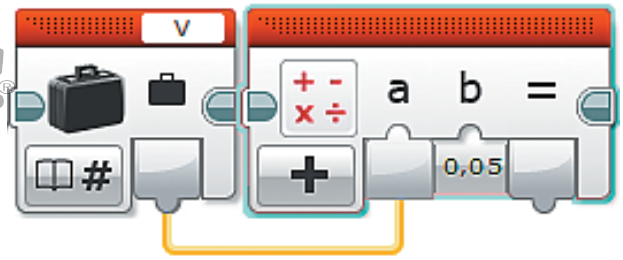
12. Результат этой проверки нужно обработать. Добавь команду **Переключатель** (*оранжевый блок*) и выбери опцию **Логическое значение**. Для удобства работы перейди к **Виду с вкладками**. Соедини выход «=» команды **Сравнение** с входом **Переключателя**.



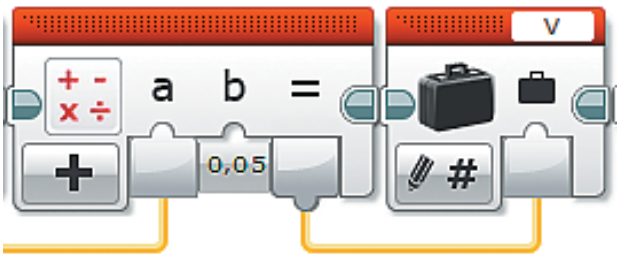
13. Ты пропишешь команды только для **Истины**, так как сейчас нас не интересует, что будет происходить в случае ложности, то есть если $V \geq 50$.
Здесь ты и обеспечишь прирост скорости. Для этого добавь команду **Переменная** (*красный блок*) и выбери для обработки переменную **V**, затем установи опцию **Считывание → Числовое значение**.



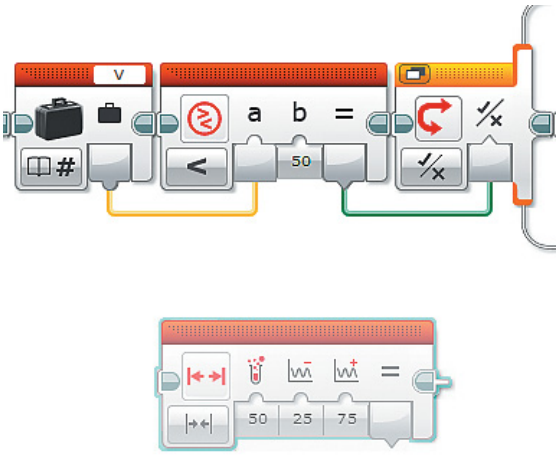
14. Добавь команду **Математика** (*красный блок*), опцию **Сложение** оставь без изменений. Соедини выход команды **Переменная** с входом **a** команды **Математика**, во вход **b** введи с клавиатуры коэффициент прироста скорости **0,05**.



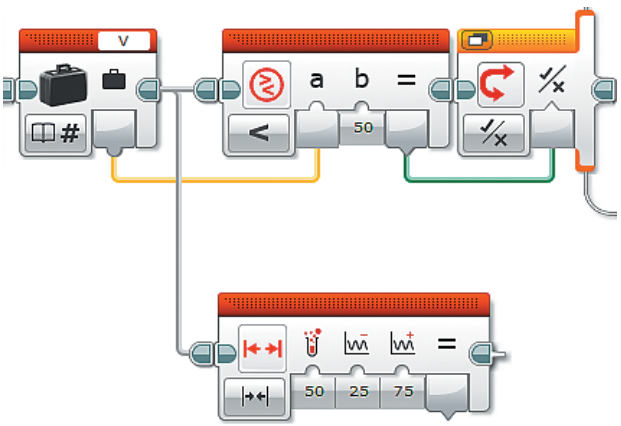
15. Результат сложения перезапиши в переменную **V**. Для этого добавь команду **Переменная**, для обработки выбери переменную **V** и соедини выход «=» команды **Математика** с входом переменной **V**.



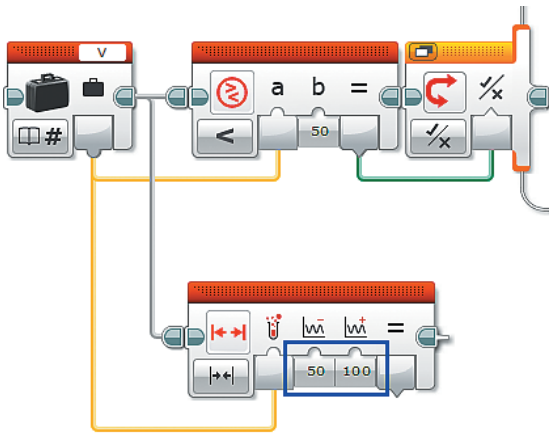
16. Теперь реализуем набор скорости в другом скоростном диапазоне. Для этого размести команду **Интервал** (*красный блок*) чуть ниже написанного фрагмента программы.



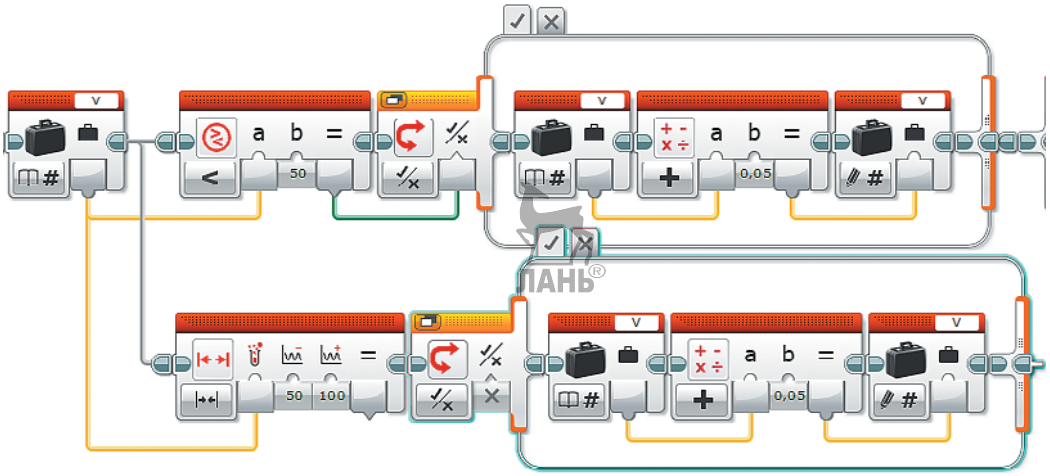
17. Заметь, что эта команда пока неактивна, так как она ни к чему не привязана. Нужно задать с клавиатуры момент времени, с которого она должна начать выполняться в программе. Мы проверяем величину переменной **V** сразу после считывания её значения. Поэтому соедини эту команду с командой **Переменная**. Для этого курсором мыши перетащи соединительный шлейф этой команды к соединительному порту команды **Интервал**.



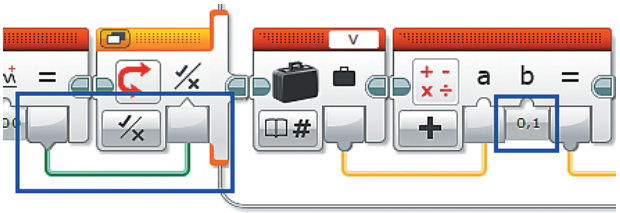
18. Выход команды **Переменная** соедини с входом **Тестовое значение** команды **Интервал**, а во входы **Нижняя граница** и **Верхняя граница** с клавиатуры запиши, соответственно, числа **50** и **100**.



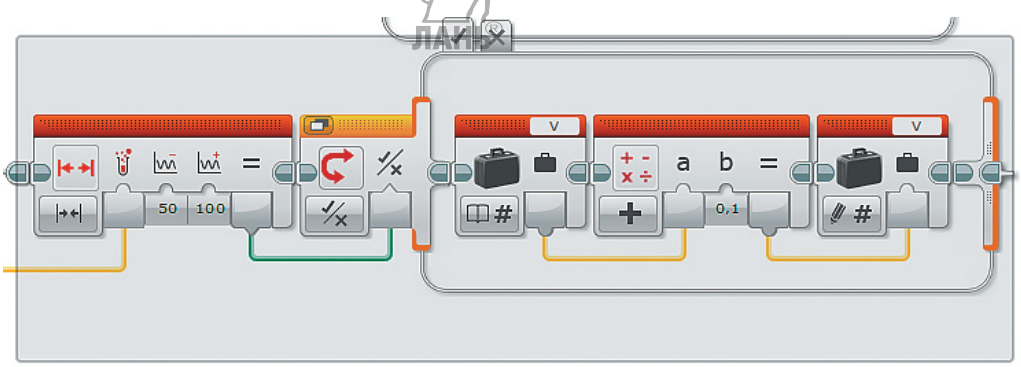
19. Следующий фрагмент программы, регулирующий изменение скорости, будет точно таким же, как и в пунктах 12–15. Для ускорения процесса написания программы можно копировать одинаковые её части. Сейчас тебе нужно скопировать команду **Переключатель** с её внутренним наполнением и настройками. Для этого зажми клавишу **Ctrl** на клавиатуре и курсором мыши перетащи команду **Переключатель** вниз, подсоединив к команде **Интервал**.



20. Соедини выход «=» команды **Интервал** с входом **Переключателя**. Единственное изменение, которое нужно внести в скопированный фрагмент, — это коэффициент прироста скорости: во вход **b** команды **Математика** запиши новое значение: **0,1**.



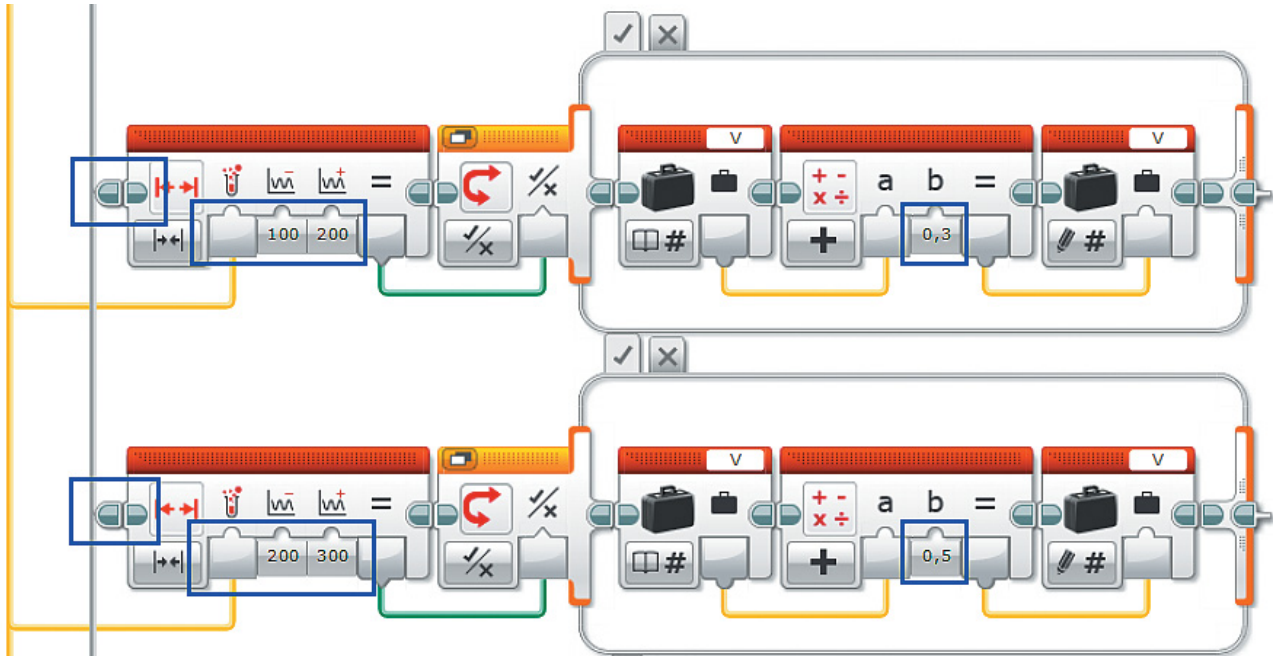
21. Осталось ещё четыре скоростных интервала. По сравнению с предыдущим фрагментом в них будут уже два отличия — скоростной интервал и коэффициент прироста скорости. Поэтому тебе нужно копировать только две команды, **Интервал** и **Переключатель**, и менять соответствующие значения.
Чтобы скопировать сразу две команды, **выдели** их. Это можно сделать с помощью мыши, растянув область выделения... :

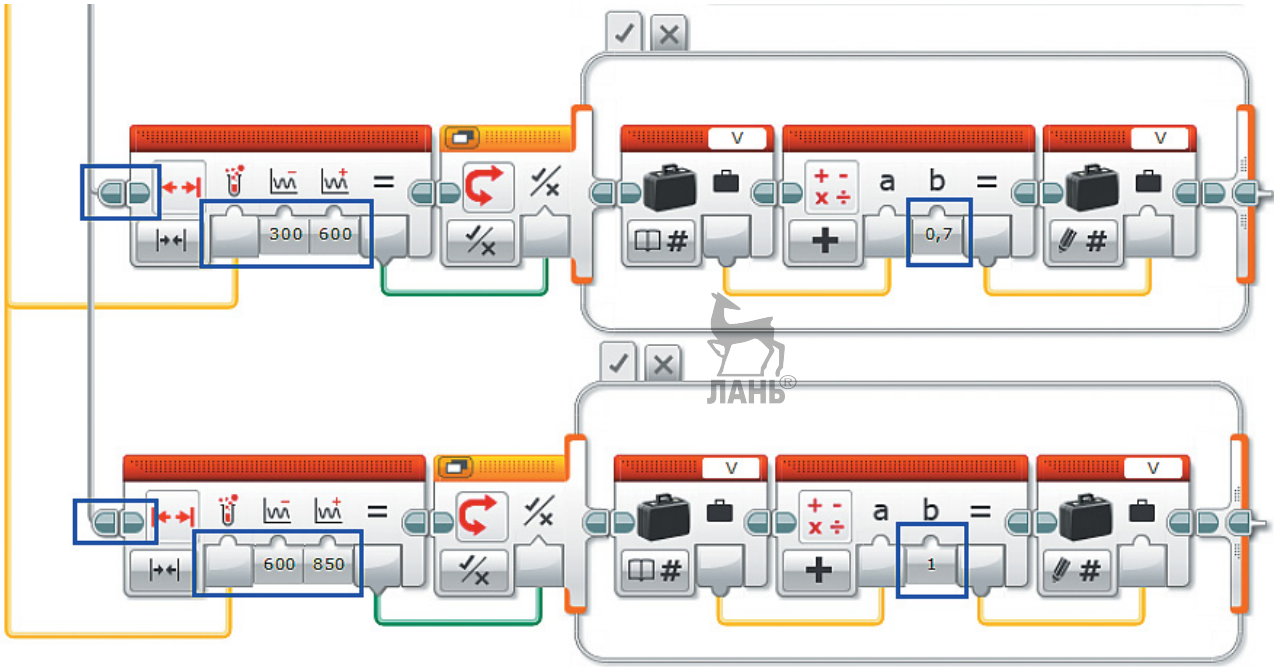


или, зажав клавишу **Ctrl**, кликать по тем командам, которые ты будешь копировать, — они сразу же будут подсвечиваться голубой рамкой.

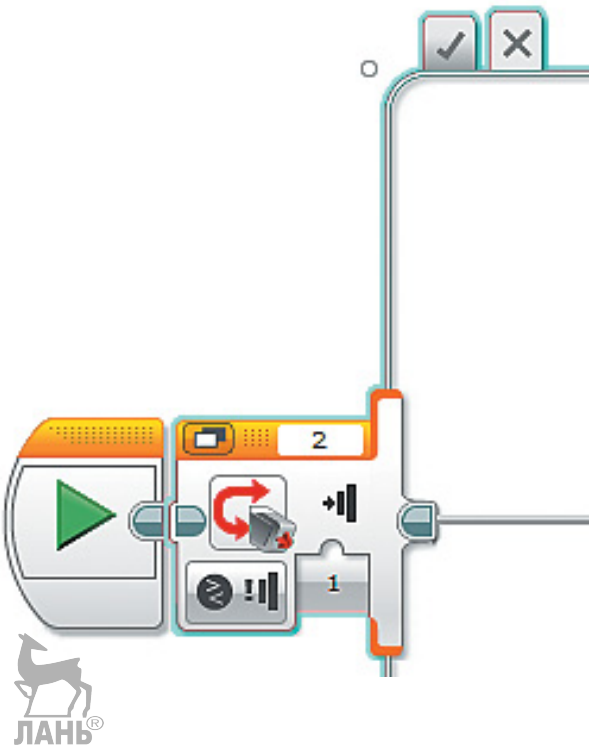
Внимание! Не забывай соединять команду **Переменная** с командой **Интервал** шлейфом, а также их выход и вход.

22. Проверь себя, должны получиться следующие фрагменты:

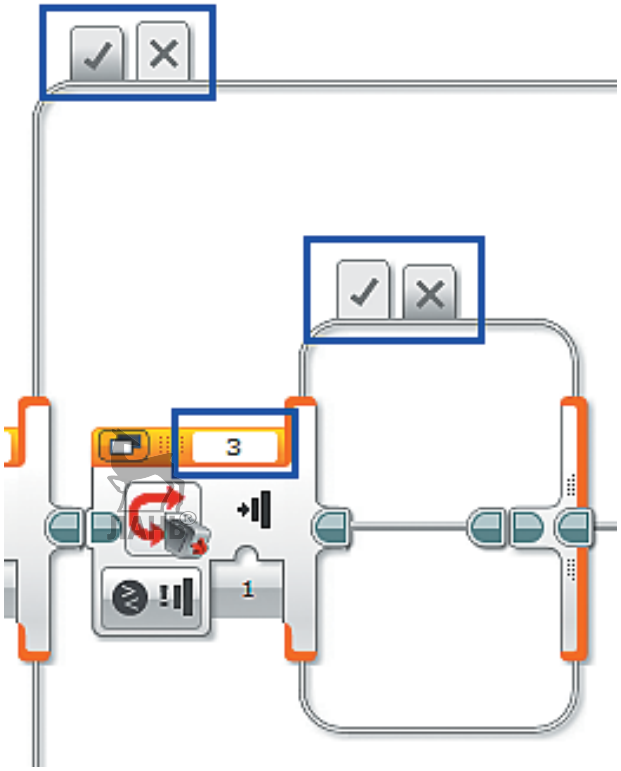




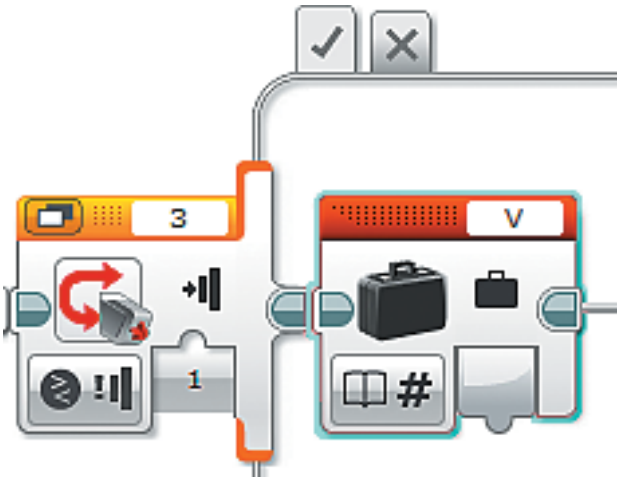
23. Вернись в начало подпрограммы. Ты прописал ту часть, в которой условие самого первого внешнего переключателя выполнено, то есть **правый датчик касания** был действительно нажат и мощность турбин необходимо увеличивать.
Теперь тебе надо рассмотреть вариант, когда проверочное условие оказалось ложным, то есть правый датчик касания нажат не был.
В этом случае появляется вторая возможность: а был ли при этом нажат **левый датчик касания**? Если да, то мощность турбин нужно *уменьшать*, то есть обеспечить замедление и торможение воздушного судна.
Перейди во внешнем **Переключателе** на **вкладку действий при ложном условии**.



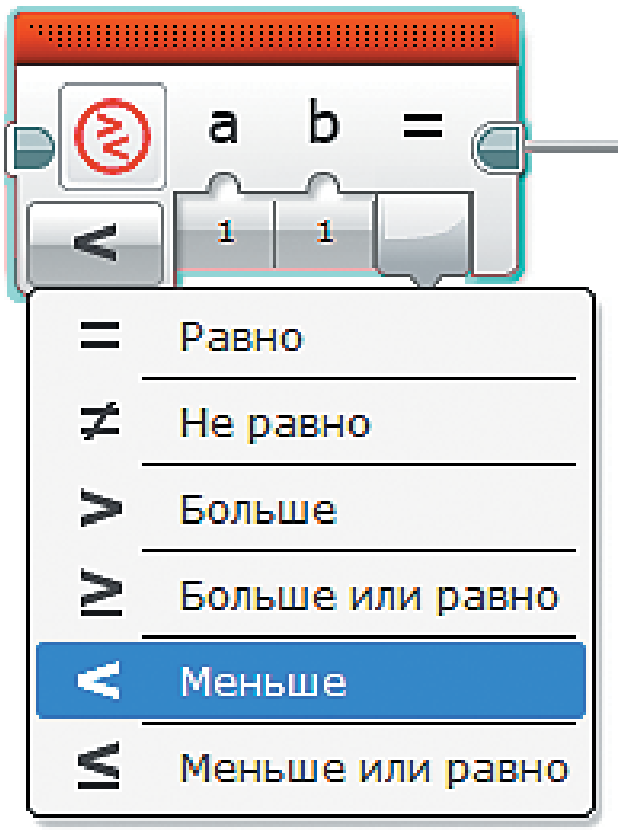
24. Добавь внутрь этого переключателя ещё одну команду **Переключатель** (*оранжевый блок*) и перейди к **Виду с вкладками**. Опции оставь без изменения. В правом верхнем окошке проверь номер порта, к которому подключён левый датчик касания (в нашем проекте это порт **№ 3**).



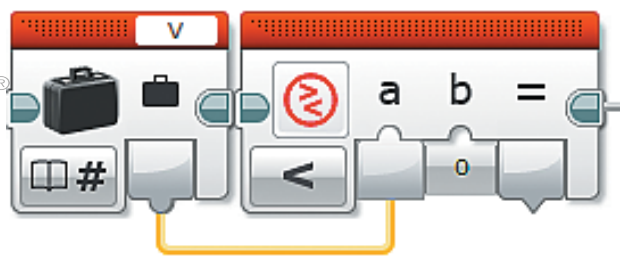
25. Рабочее поле внутреннего **Переключателя** сразу расширь, а затем напиши практически такой же фрагмент программы, как при увеличении мощности, только коэффициент прироста скорости станет **коэффициентом убывания** скорости, который нужно будет вычитать.
Первой командой будет **Переменная** (*красный блок*). Для обработки выбери переменную **V** и опцию **Считывание → Числовое значение**.



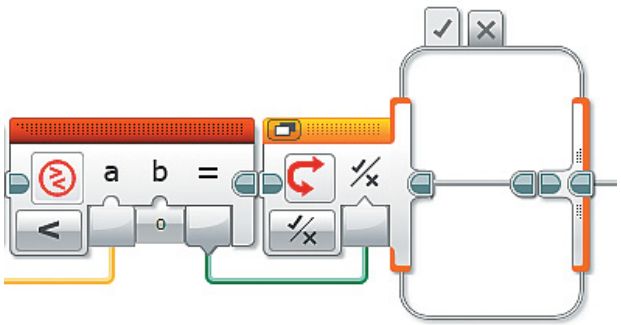
26. Перед началом работы по уменьшению мощности турбин нужно исключить ошибку ухода показаний спидометра **ниже нуля** (при увеличении мощности за это автоматически отвечала команда **Интервал**).
Для этого добавь команду **Сравнение** (*красный блок*) и выбери опцию **Меньше**.



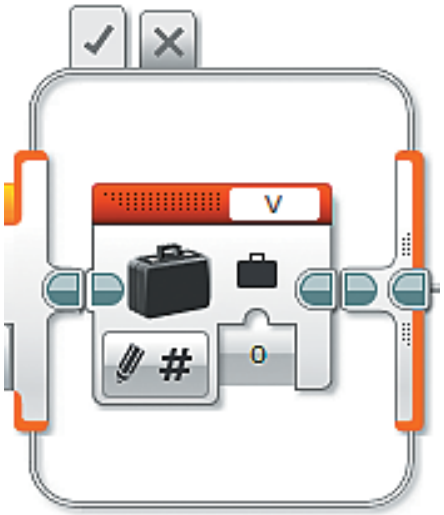
27. Соедини выход команды **Переменная** с входом **a** команды **Сравнение**, во вход **b** введи с клавиатуры число **0**.



28. Теперь добавь команду **Переключатель** (*оранжевый блок*), перейди к **Виду с вкладками** и выбери опцию **Логическое значение**. Соедини выход «=» команды **Сравнение** с входом **Переключателя**.



29. Если скорость действительно попытается «убежать» ниже нуля (в отрицательные значения), «верни» её к нулю.
Для этого добавь внутрь этого **Переключателя** команду **Переменная** (*красный блок*) и выбери для обработки переменную **V**. По умолчанию в ней настроена запись нуля. Именно это тебе и нужно. Теперь ниже нуля скорость не опустится.

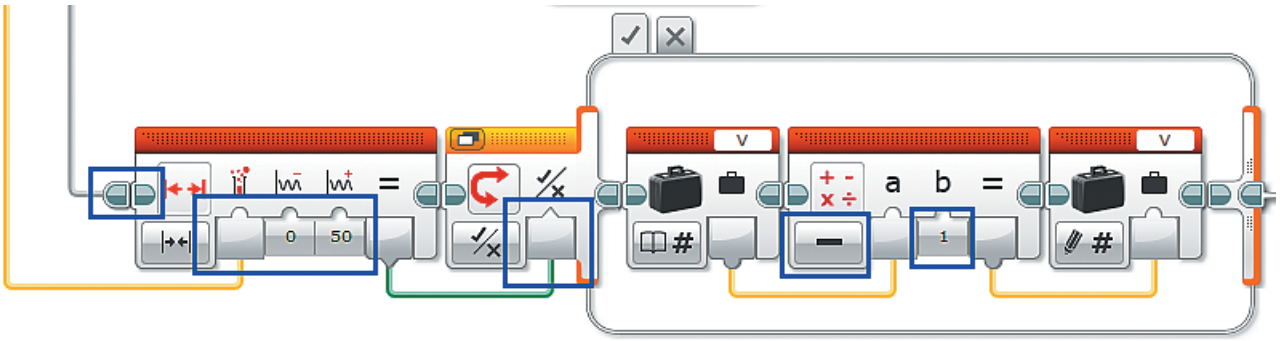


30. Далее пропиши фрагмент программы, регулирующий уменьшение скорости. Для удобства воспользуйся таблицей:

Скорость самолета, км/ч	**Соответствующее неравенство**	**Коэффициент убывания скорости, км/ч**
От 0 до 50	$0 \le V \le 50$	1
От 50 до 100	$50 \le V \le 100$	0,7
От 100 до 200	$100 \le V \le 200$	0,5
От 200 до 300	$200 \le V \le 300$	0,3
От 300 до 600	$300 \le V \le 600$	0,1
Свыше 600	$V > 600$	0,05

Пояснение. Чем ниже скорость, тем «проще» самолёту тормозить.

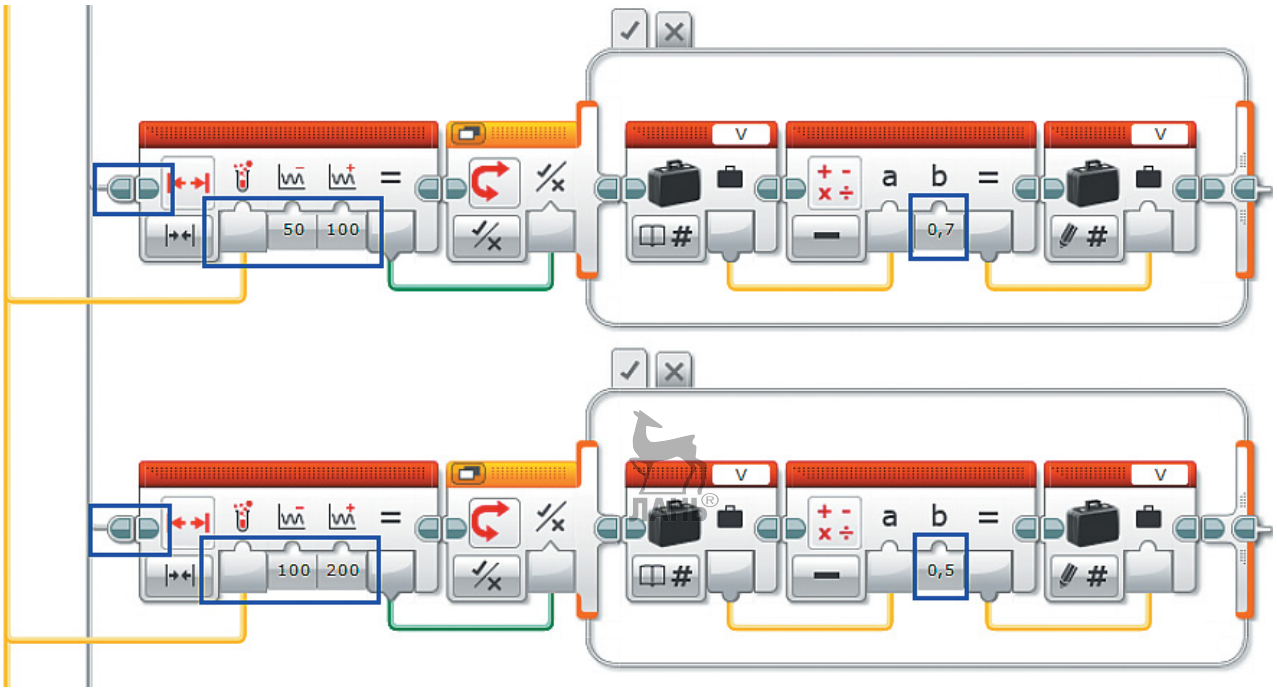
31. Реализация каждого из диапазонов, кроме последнего, практически ничем не отличается от наращивания скорости. Логика первого фрагмента такова: в диапазоне от 0 до 50 км/ч скорость должна уменьшаться на 1 км/ч в каждый момент времени. Тогда получится следующая цепочка команд:

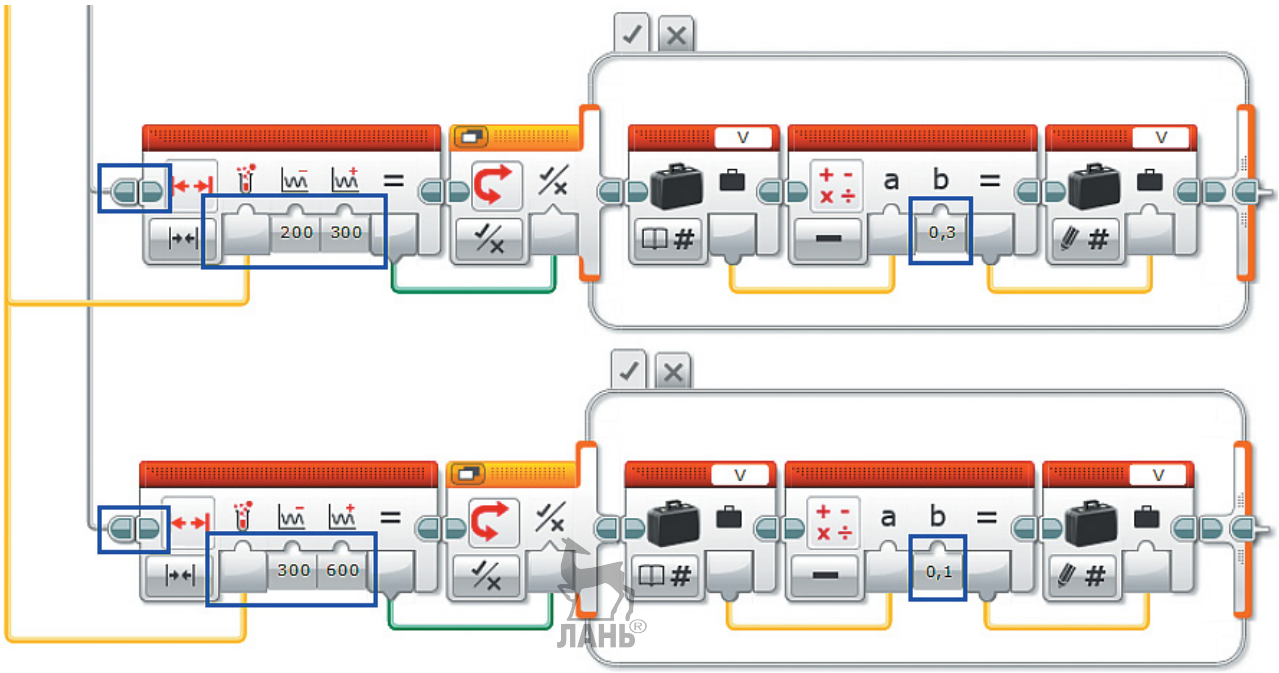


32. Для следующих четырёх диапазонов скорости фрагменты будут аналогичные, поэтому их удобно копировать, изменяя границы диапазона и значение коэффициента убывания скорости.

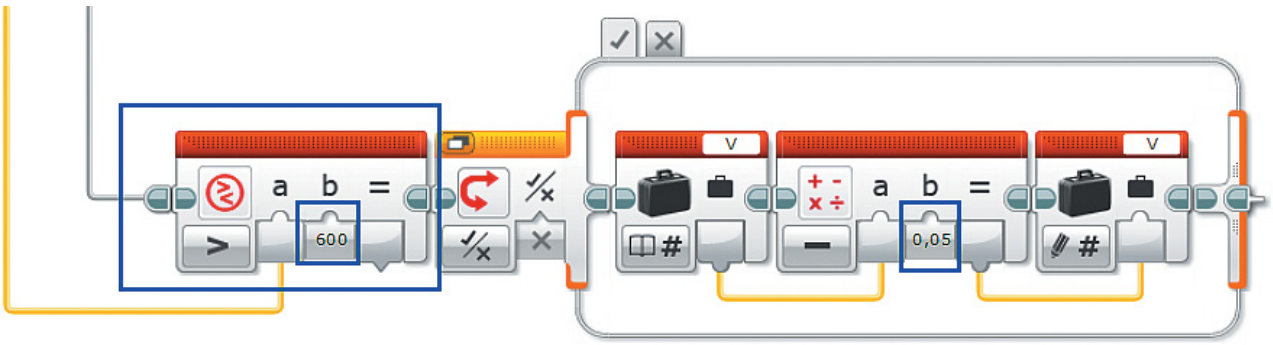
Внимание! Не забывай соединять команду **Переменная** с командой **Интервал** шлейфом, а также их выход и вход.

Проверь себя, должны получиться следующие фрагменты:





33. Последний фрагмент в этой подпрограмме отличается проверкой условия. Здесь значение скорости сравнивается только с одним числом, поэтому команда **Интервал** заменится на **Сравнение**:



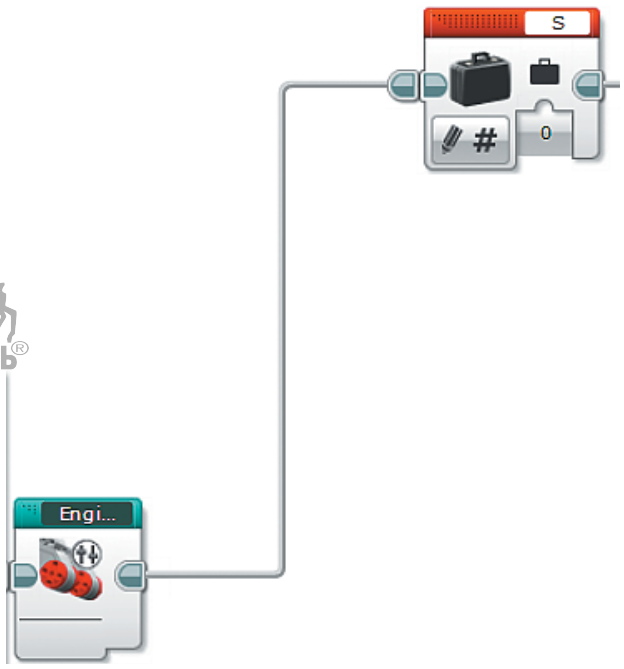
Поздравляем! Первая подпрограмма составлена, благодаря ей твой самолёт готов отправиться в путешествие!

Часть 3. Скорость, крен, тангаж — завернём крутой вираж! Показания спидометра и авиагоризонта

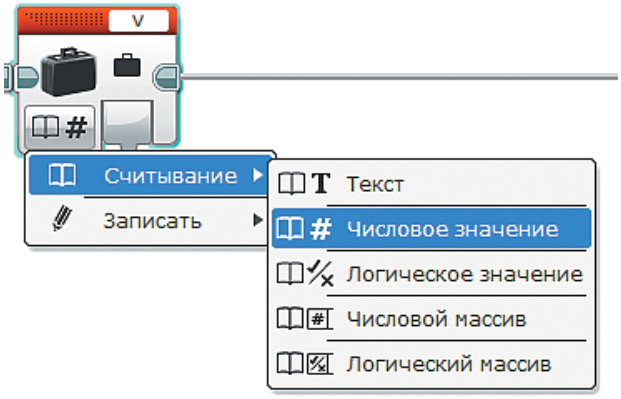
Логика. В третьей части программирования ты выведешь на приборную панель первые показания приборов, а именно: скорость и углы крена и тангажа.

Для удобства вывода на экран и компактности программы воспользуйся возможностью составления текстовых строк из нескольких различных объектов: чисел, значений переменных и символов.

1. Вернись в основную программу **Main**. После подпрограммы **Engines** у тебя будут ещё три ветви, и сейчас ты пропишешь первую из них. Для удобства их лучше расположить друг под другом, поэтому первую команду **Переменная** (*красный блок*) добавь чуть правее и выше от **Engines**.
Внимание! Не забудь соединить эти команды шлейфом.

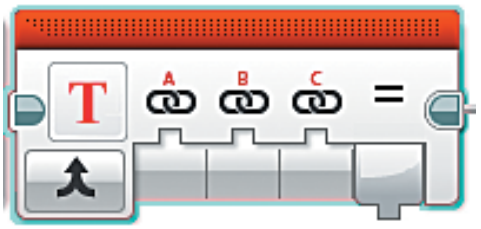


2. Выбери для обработки переменную **V** и опцию **Считывание → Числовое значение.**



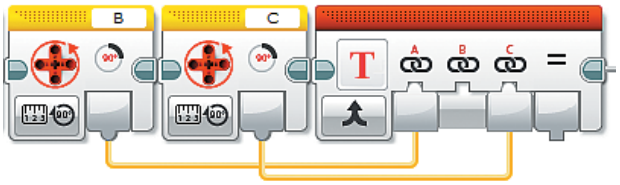
3. Далее добавь две команды **Вращение мотора** (*жёлтый блок*). Проверь, чтобы в правом верхнем окошке значились порты **B** и **C**.

4. Теперь добавь команду **Текст** (*красный блок*). С её помощью ты сможешь вывести на экран программируемого модуля сразу несколько различных значений.



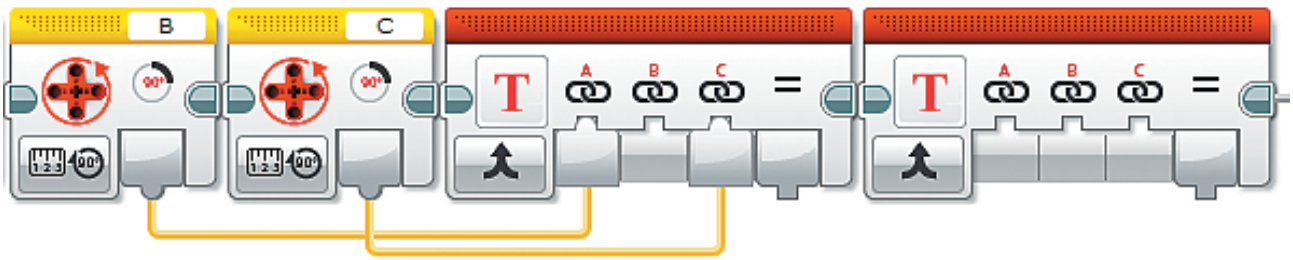
Пояснение. Эта команда позволяет сформировать для отображения на экранчике программируемого модуля текст, состоящий из трёх различных фрагментов или, как говорят программисты, из трёх строк.

5. Мы зададим следующий порядок в тексте: <Угол тангажа><Пробел><Угол крена><Пробел><Скорость> Получилось пять условных объектов для вывода, значит, одной командой **Текст** тебе не обойтись.

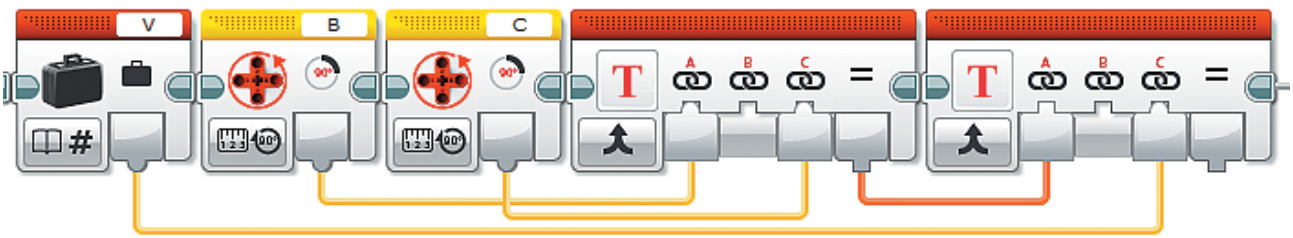


Для начала соедини выход первой команды **Вращение мотора** с входом **A**, во входе **B** с клавиатуры введи один пробел, выход второй команды **Вращение мотора** соедини со входом **C**.

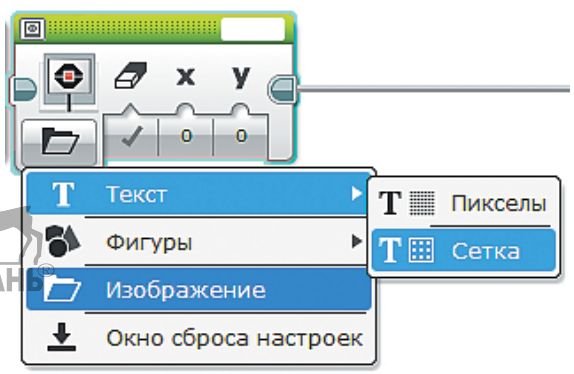
6. Добавь ещё одну команду **Текст** (*красный блок*).



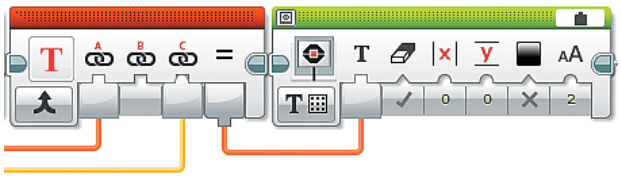
7. Соедини выход «=» первой команды **Текст** с входом **A**, во вход **B** с клавиатуры введи один пробел, выход переменной **V** соедини со входом **C** второй команды **Текст**.



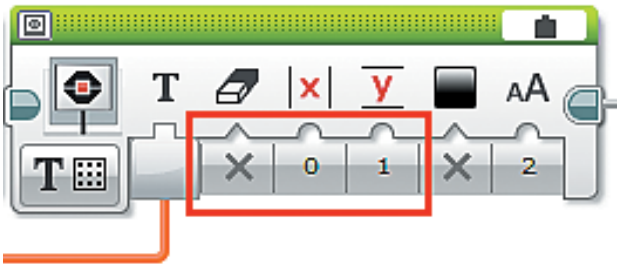
8. Осталось вывести результат этих слияний на экран EV3. Добавь команду **Экран** (*зелёный блок*) и выбери опцию **Текст** → **Сетка**.



9. В правом верхнем окошке выбери пункт **Проводной**, а затем соедини выход «=» с входом **Текст**.



10. Осталось настроить параметры вывода на экран:
Очистить экран — ложность (тебе ведь не нужно, чтобы экран то и дело мерцал очищаясь).
Введи с клавиатуры во входы **x** и **y** значения: **X** = 0, **Y** = 1.
В результате текст будет выводиться от левого края экрана, но чуть ниже первой строки.
Остальные параметры оставь **без изменений**.

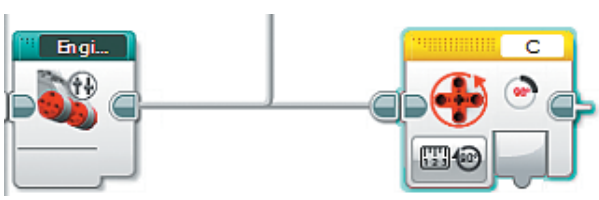


Часть 4. Тревога! Тревога! Система сигнализации об опасном уровне тангажа

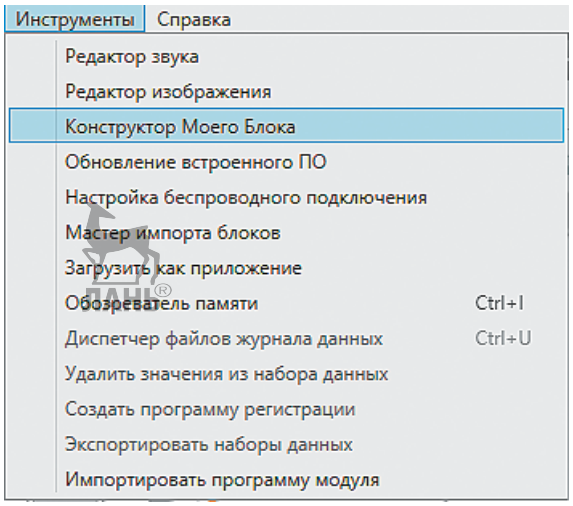
Логика. Ты подошёл к одному из очень важных этапов — созданию **системы сигнализации** опасного угла тангажа. Если бы в сложных технических устройствах, тем более таких, как система управления воздушным судном, не было подобных систем, людям было бы очень сложно замечать и ликвидировать опасные ситуации и неисправности.

Наша система сигнализации будет подавать сигнал **Down** (от английского — вниз), если нос самолёта поднят слишком высоко, и сигнал **Up** (от английского — вверх), если нос слишком низко опущен.

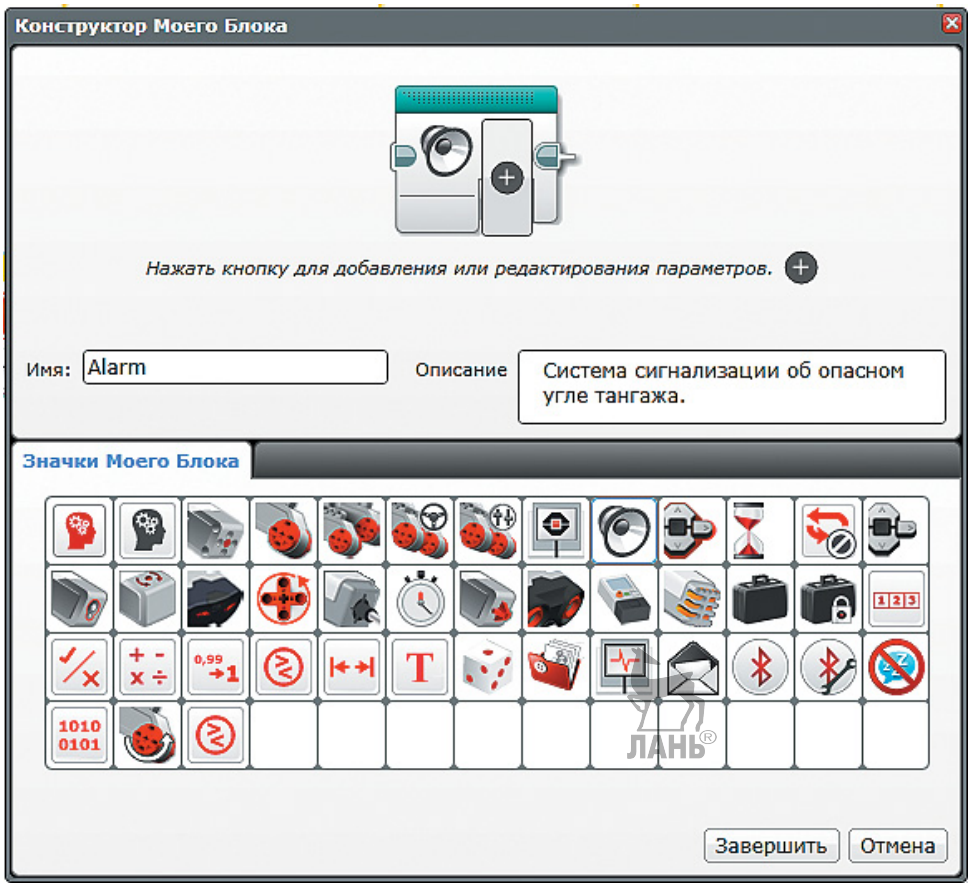
1. Чтобы узнать, действительно ли нос самолёта находится в опасной зоне, нужно считать показания мотора, который отвечает за тангаж. В нашем случае это мотор, подключенный к порту **C**.
Так как данная система представляет собой самостоятельную часть программы, то, естественно, ты создашь её как подпрограмму.
Для этого добавь команду **Вращение мотора** (*жёлтый блок*), расположи её напротив команды **Engines** и соедини их с помощью шлейфа. Опцию оставь без изменений, в данном случае нам нужно знать, на сколько градусов был повернут мотор, то есть как сильно отклонён штурвал вперёд или назад.
Проверь, чтобы в правом верхнем окошке значился порт **C**.



2. Теперь приступай к составлению подпрограммы. Команда **Вращение мотора** должна быть выделена (подсвечена голубым контуром). В верхней части программы выбери меню **Инструменты → Конструктор Моего Блока**.

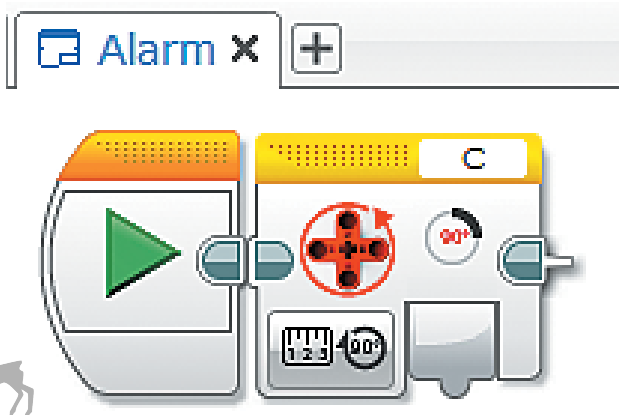


3. В открывшемся окне задай имя подпрограммы, введи её краткое описание и выбери подходящую пиктограмму. Логично задать имя **Alarm** (от английского — тревога). Помни, чтобы легко ориентироваться в программах, следует называть их сообразно тому, какую функцию они выполняют. Затем нажми кнопку **Завершить**. Краткое описание и пиктограмму можешь задать, как на рисунке или самостоятельно.



4. В основной программе **Main** появится новая команда.

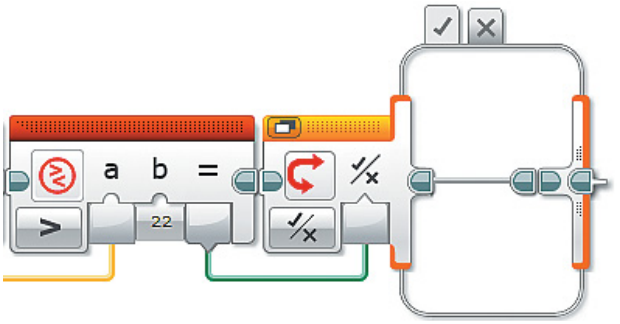
5. Дважды кликни по ней. На экране откроется новое рабочее поле для редактирования подпрограммы. Более того, сверху появится вкладка с именем *Alarm*.



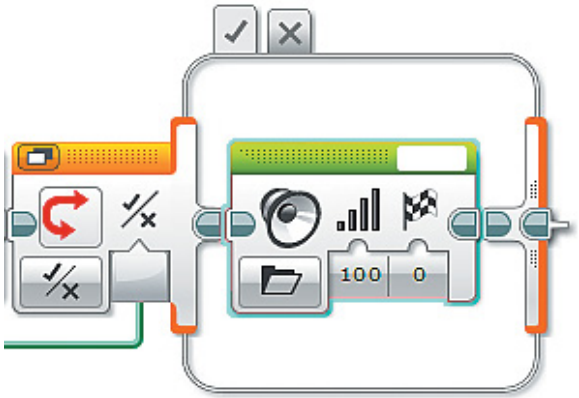
6. Шкала градусов, показывающая угол отклонения носа самолёта, может иметь как отрицательные, так и положительные значения. Для нашего расположения вертикального мотора подъём носа (штурвал на себя) соответствует перемещению по шкале в отрицательном направлении (отрицательный тангаж), а опускание носа (штурвал от себя) соответствует перемещению по шкале в положительном направлении — угол в градусах увеличивается (положительный тангаж).

Отработаем для начала ситуацию с критическим уровнем опущенного носа самолёта, составляющим угол в 22 градуса. При таком уклоне самолёт переходит в пикирование. Добавь команду **Сравнение** (*красный блок*), выбери опцию **Больше** и соедини выход команды **Вращение мотора** и вход **a** команды **Сравнение**. Во вход **b** запиши число **22** — критическое значение.

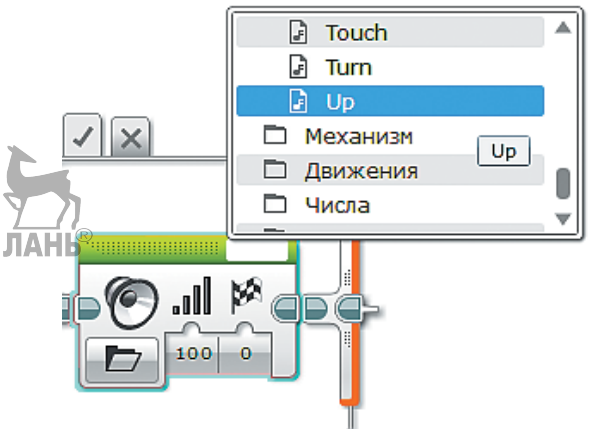
7. Далее добавь команду **Переключатель** (*оранжевый блок*), перейди к **Виду с вкладками** и выбери опцию **Логическое значение**. Соедини выход «=» команды **Сравнение** с входом **Переключателя**.



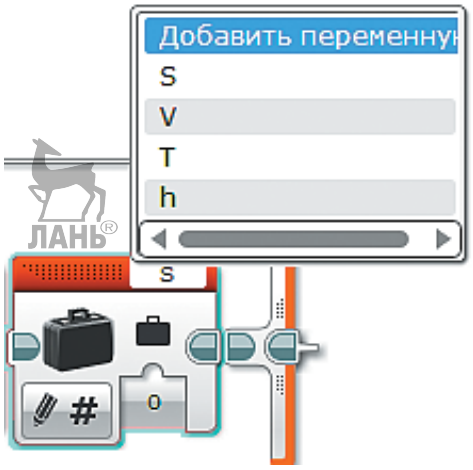
8. Чтобы просигнализировать об опасности и дать инструкцию **Up**, внутрь **Переключателя** добавь команду **Звук** (*зелёный блок*).



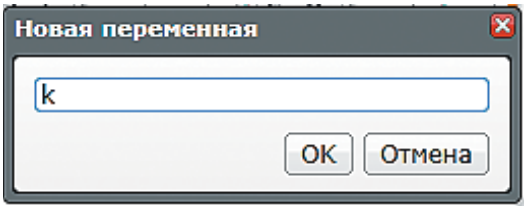
9. Сейчас эта команда настроена на воспроизведение звукового файла. Можно выбрать файл из уже имеющихся в библиотеке звуков, а можно добавить собственный (с этим ты познакомишься позже). В правом верхнем окошке этой команды выбери папку **Звуковые файлы LEGO**, в ней — папку **Информация** и затем файл **Up**. При нажатии на него он воспроизведётся.



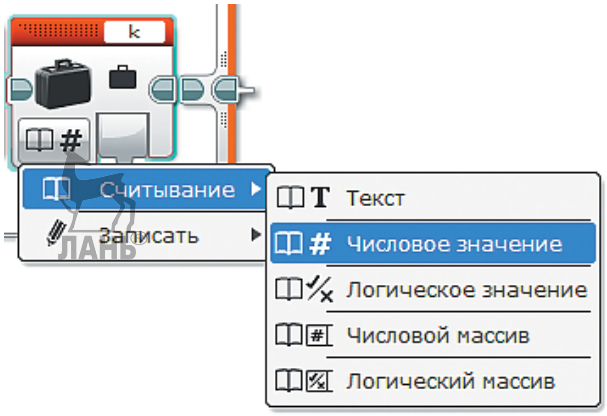
10. Ты понимаешь, что если долго держать самолёт в таком состоянии, произойдет авиакатастрофа. Это тебе тоже надо учесть.
Чтобы понять, что самолёт перешёл в сваливание и разбился, можно посчитать, сколько раз был подан сигнал предупреждения. Для этого добавь команду **Переменная** (*красный блок*) и в правом верхнем окошке выбери пункт **Добавить переменную**.



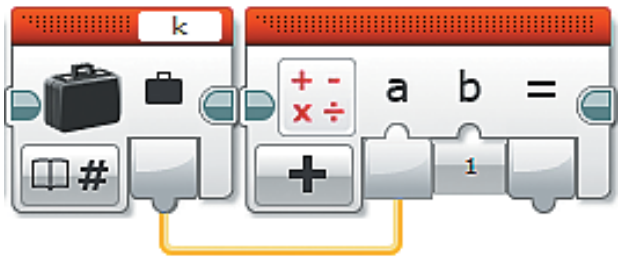
11. В открывшемся окне введи имя **k** (эта переменная понадобится тебе для подсчёта количества поданных сигналов) и нажми **ОК**.



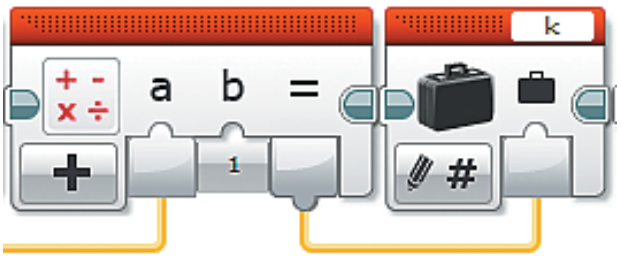
12. Переменная **k** будет увеличиваться на единицу после каждого поданного сигнала. Если сигнал не подаётся, то есть самолёт находится в нормальном положении, эта переменная будет заново обнуляться.
Для этого выбери опцию **Считывание** → **Числовое значение**.



13. Далее добавь команду **Математика** (*красный блок*). Соедини выход переменной **k** с входом **a** команды **Математика**. Остальные опции и параметры оставь без изменений.

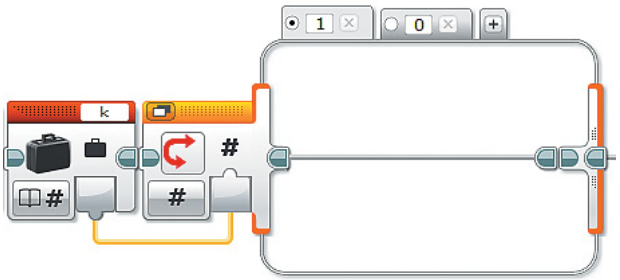


14. Добавь ещё одну команду **Переменная** (*красный блок*). Для обработки выбери переменную **k** и соедини выход «=» команды **Математика** с входом переменной **k**.

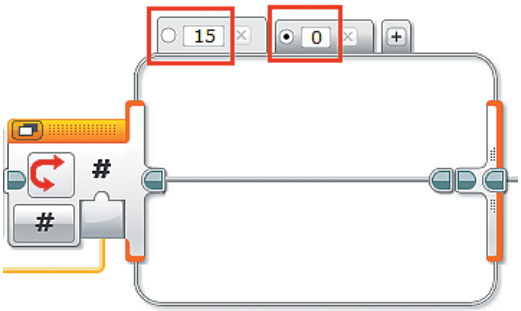


15. Давай договоримся, что если сигнал **Up** был подан **15** раз, то эта ситуация будет считаться крушением, после которого авиасимулятор прекращает свою работу. Для проверки добавь команду **Переменная** (*красный блок*), выбери для обработки переменную **k** и установи опцию **Считывание** → → **Числовое значение**.

16. Добавь команду **Переключатель** (*оранжевый блок*), перейди к **Виду с вкладками** и выбери опцию **Числовое значение**. Затем соедини выход переменной **k** с входом **Переключателя**.

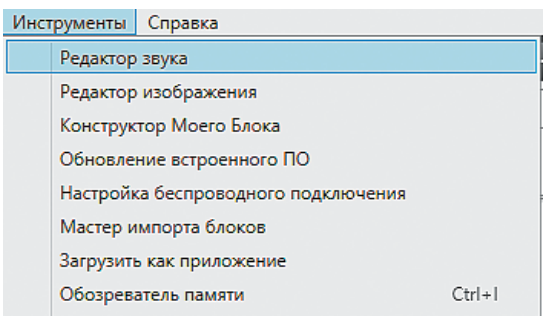


17. В верхней части **Переключателя** вместо единицы задай значение **15** — количество повторов сигнала, а второе значение оставь равным **нулю**, только переставь к нему точку выбора значения по умолчанию. Для этого кликни в кружок рядом с нулём.

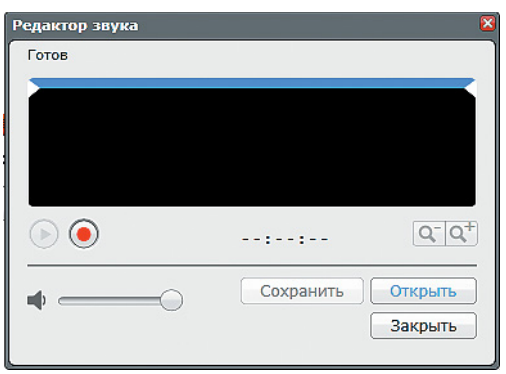


Пояснение. Данный **Переключатель** будет проверять, достигла ли переменная **k** значения, равного **15**. Для этого случая ты пропишешь определённые действия. Если же **k** будет иметь иное значение, то **Переключатель** будет считать его равным нулю и ничего не делать, так как для этой ситуации ты не будешь задавать ему действий на исполнение. **Важно**! Переключатель **НЕ** обнуляет переменную **k**.

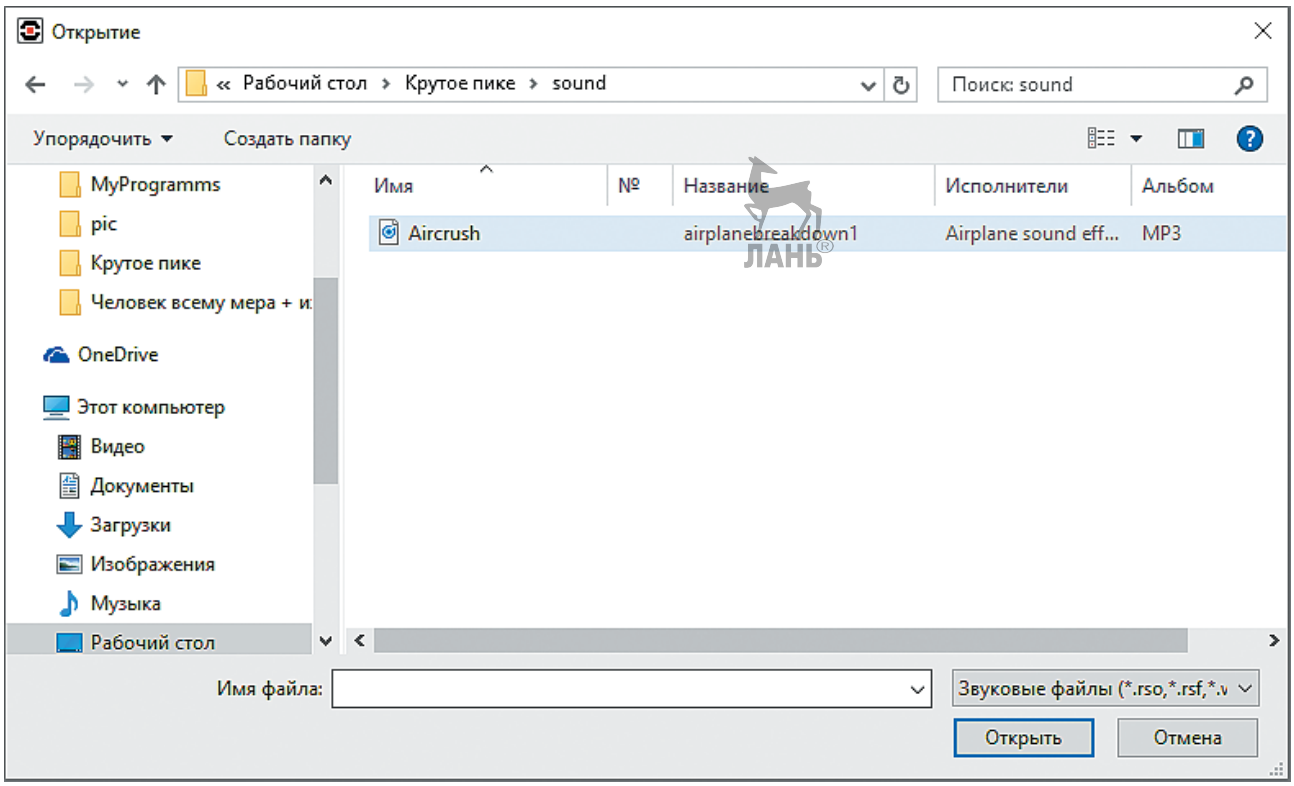
18. Теперь «научим» твой авиасимулятор воспроизводить звук крушения самолёта. Скачай его на сайте издательства по ссылке **http://pilotlz.ru/files/10045/**, выбери файл **Aircrush.mp3** и сохрани в памяти компьютера. Затем вверху экрана выбери меню **Инструменты → Редактор звука.**



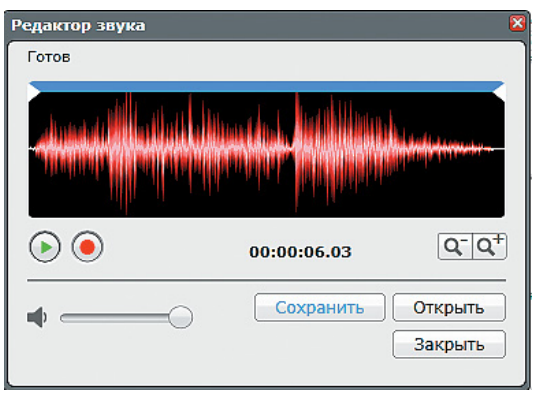
19. В открывшемся окне нажми кнопку **Открыть**.



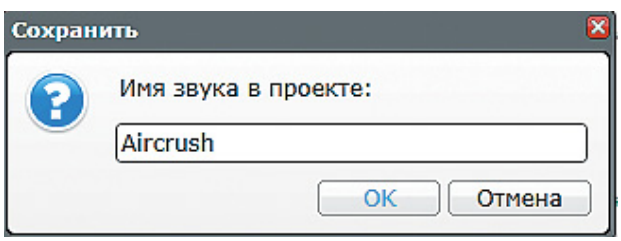
20. В новом окне найди файл звука **Aircrush** (от английского — авиакатастрофа) там, где ты его сохранил при скачивании, и кликни по нему дважды.



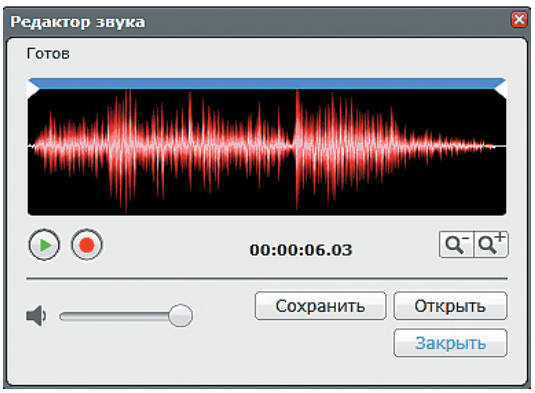
21. Файл сразу загрузится в редактор, где ты можешь его обработать. Данный файл не требует никаких дополнительных действий, поэтому ты можешь его прослушать, нажав клавишу **Play** (*зелёный треугольник*). Затем нажми кнопку **Сохранить**.



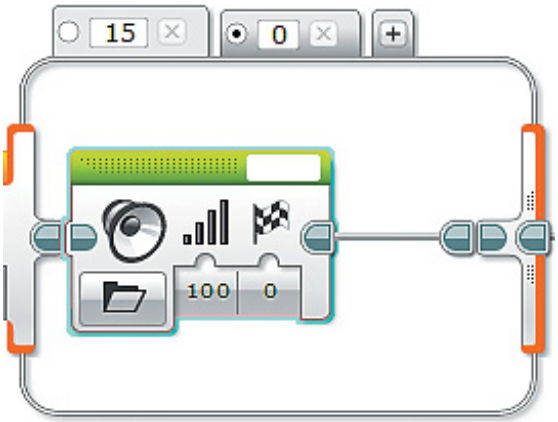
22. В открывшемся окне введи имя этого звука, например тоже **Aircrush,** и нажми **ОК**. По этому имени он будет идентифицироваться в твоём проекте.



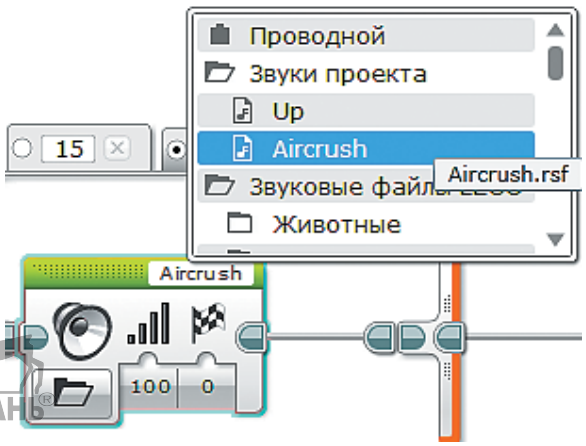
23. Теперь нажми кнопку **Закрыть**.



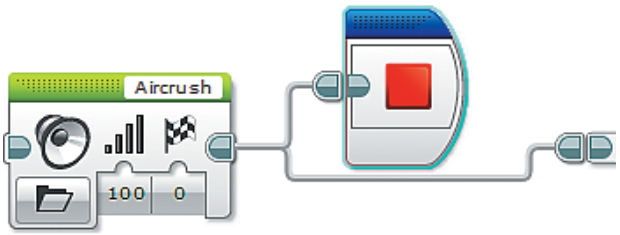
24. Перейди в **Переключателе** на вкладку с числом **15** и добавь туда команду **Звук** (*зелёный блок*).



25. В правом верхнем окошке этой команды выбери папку **Звуки проекта** и в ней файл **Aircrush**.



26. После воспроизведения этого звука авиасимулятор должен прекратить свою работу, то есть программа должна полностью завершиться.
Для этого добавь команду **Остановить программу** (*синий блок*), только

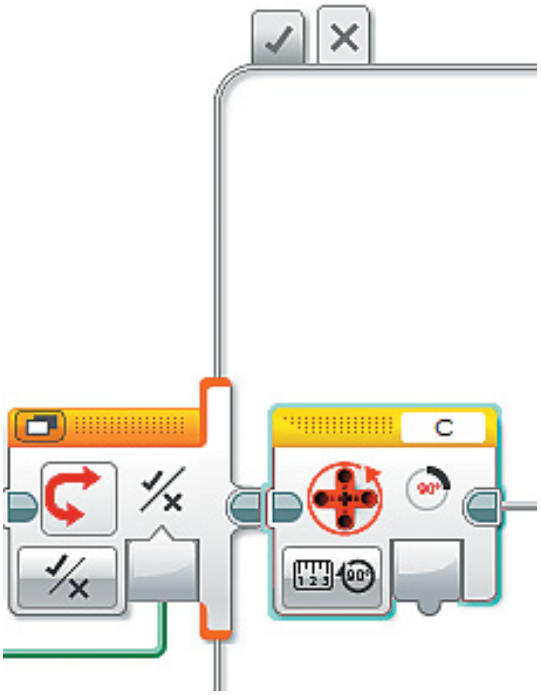


расположи её так, чтобы она **не присоединилась** к команде **Звук**. Её надо соединить с командой **Звук** при помощи шлейфа — это нужно, чтобы обеспечить возможность повтора тревоги в случае, когда самолёт выведен в нормальное положение после неё.

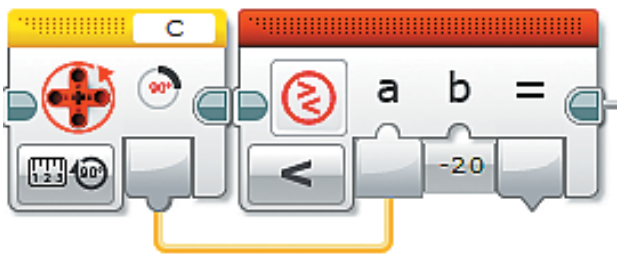
27. Итак, ты создал предупреждающую сигнализацию для случая, когда нос самолёта слишком опущен, то есть первый **Переключатель** в этой подпрограмме принял значение **Истина**.

    Теперь рассмотрим случай, когда нос слишком сильно поднят. Как мы договаривались ранее, критическим значением будем считать **20** градусов.

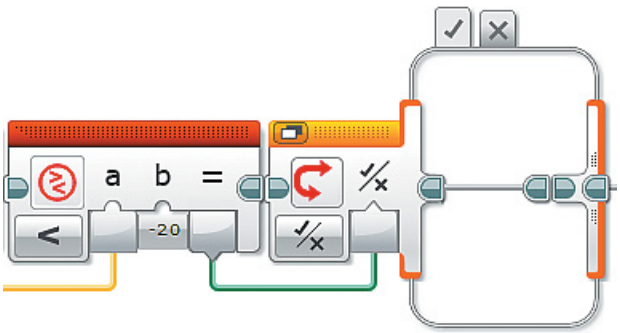
    Перейди на вкладку **Ложь** внешнего **Переключателя** и добавь команду **Вращение мотора** (*жёлтый блок*). Проверь, что в верхнем правом окошке выбран порт **C**.



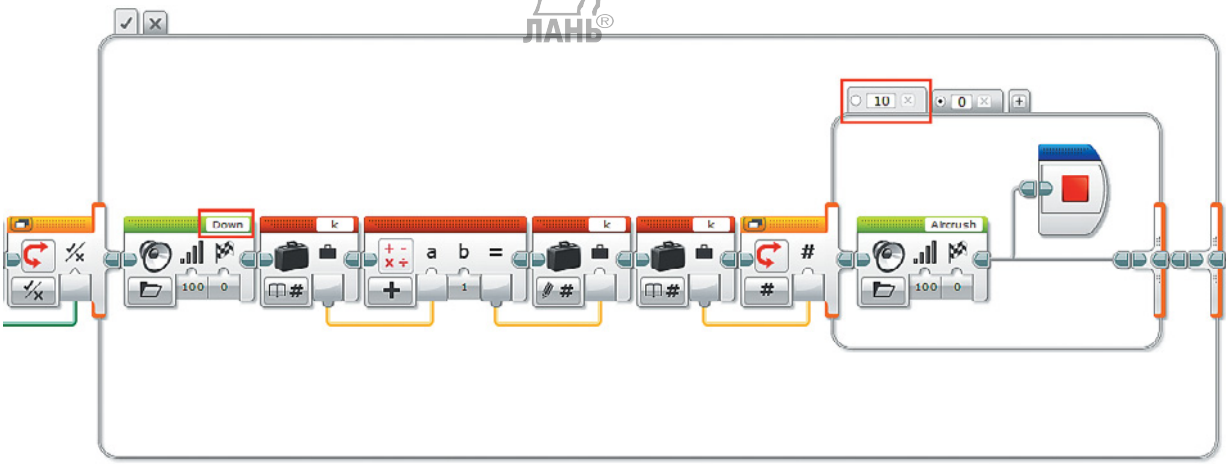
28. Добавь команду **Сравнение** (*красный блок*), выбери опцию **Меньше** и соедини выход команды **Вращение мотора** с входом **a** команды **Сравнение**. Во вход **b** с клавиатуры введи **–20**.



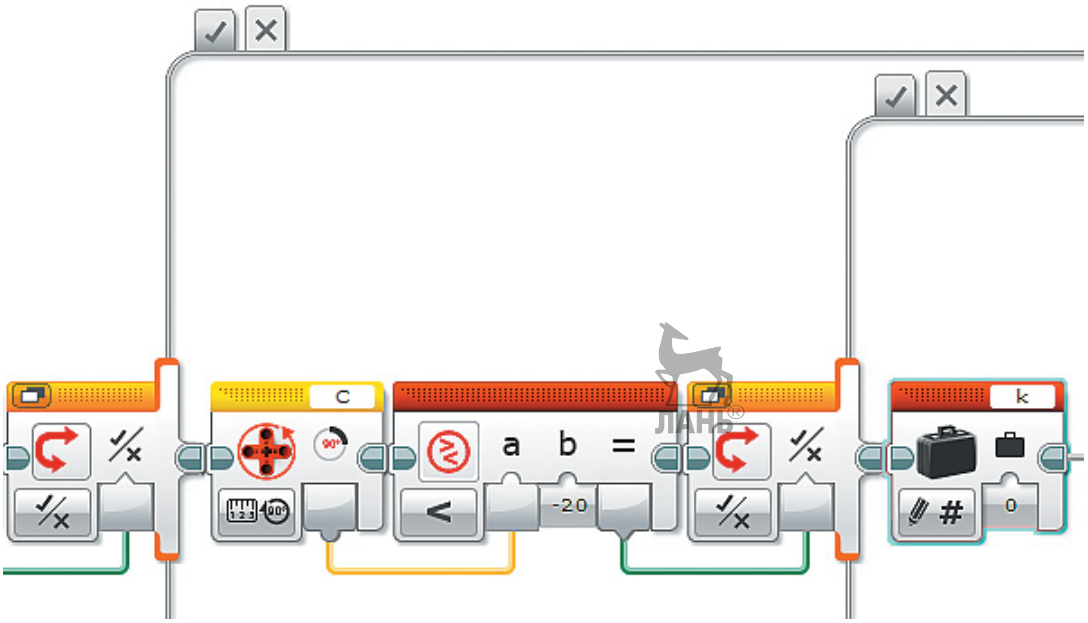
29. Далее добавь команду **Переключатель** (*оранжевый блок*), перейди к **Виду с вкладками**, выбери опцию **Логическое значение** и соедини выход «=» команды **Сравнение** с входом **Переключателя**.



30. Напиши такой же фрагмент программы, как и для предыдущего случая, но с небольшими изменениями:
    - звуком сигнализации будет **Down**; он находится в папке **Звуковые файлы LEGO** → → **Информация** → **Down;**
    - числовое значение для второго переключателя уже не 15, а **10**, так как слово **Down** звучит дольше, чем **Up**, значит, количество его повторов должно быть меньше.



31. Осталось рассмотреть третий случай, когда самолёт находится в нормальном состоянии или пилот выровнял его. В подпрограмме во втором по счёту **Переключателе** перейди на вкладку **Ложь** и добавь команду **Переменная** (*красный блок*). Для обработки выбери переменную **k**, остальные параметры оставь без изменений.



Пояснение. Если самолёт находится в нормальном состоянии или выведён в него, то счётчик сигналов бедствия должен быть обнулён. Именно это ты и реализовал.

Часть 5. Бесконечность — не предел! Работа одометра, расчёт пройденного расстояния

Логика. В этой части ты обеспечишь корректную работу прибора под названием **одометр** (от греческого οδος — дорога + μετρον — мера), который будет показывать пройденный самолётом путь. Без такого прибора невозможно определить, сколько километров пролетел самолёт или пробежал по взлётно-посадочной полосе, а также оценить запасы топлива.

Ты уже, конечно, знаешь, что пройденное расстояние — *это скорость движения, умноженная на затраченное в пути время*. Однако эта формула (которую мы рассматривали в пункте 24 первой части) действительна только для какого-то конечного и конкретного значения времени. Одометр же должен рассчитывать путь, пройденный за какое-то время, и наращивать (то есть прибавлять) это значение к предыдущему, начиная с нуля.

Давай выберем таким промежутком времени 1 секунду, то есть показания одометра на экране EV3 будут обновляться раз в секунду и расти в зависимости от скорости.

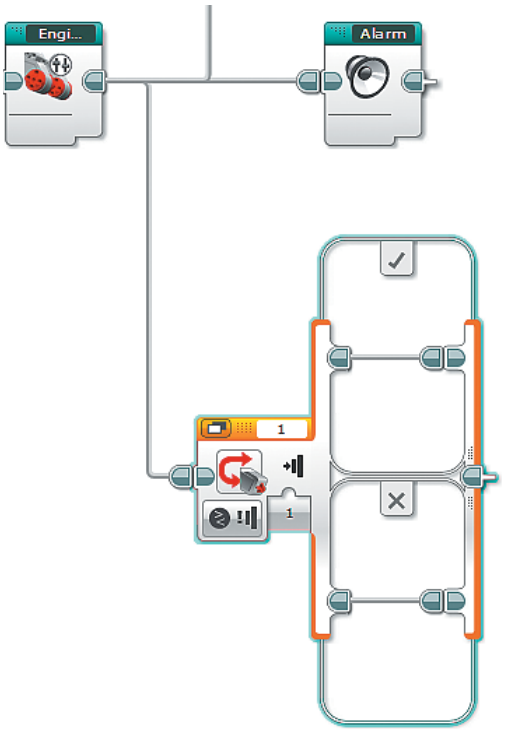
Пройденное расстояние S_2 на момент отсчёта определяется по формуле:

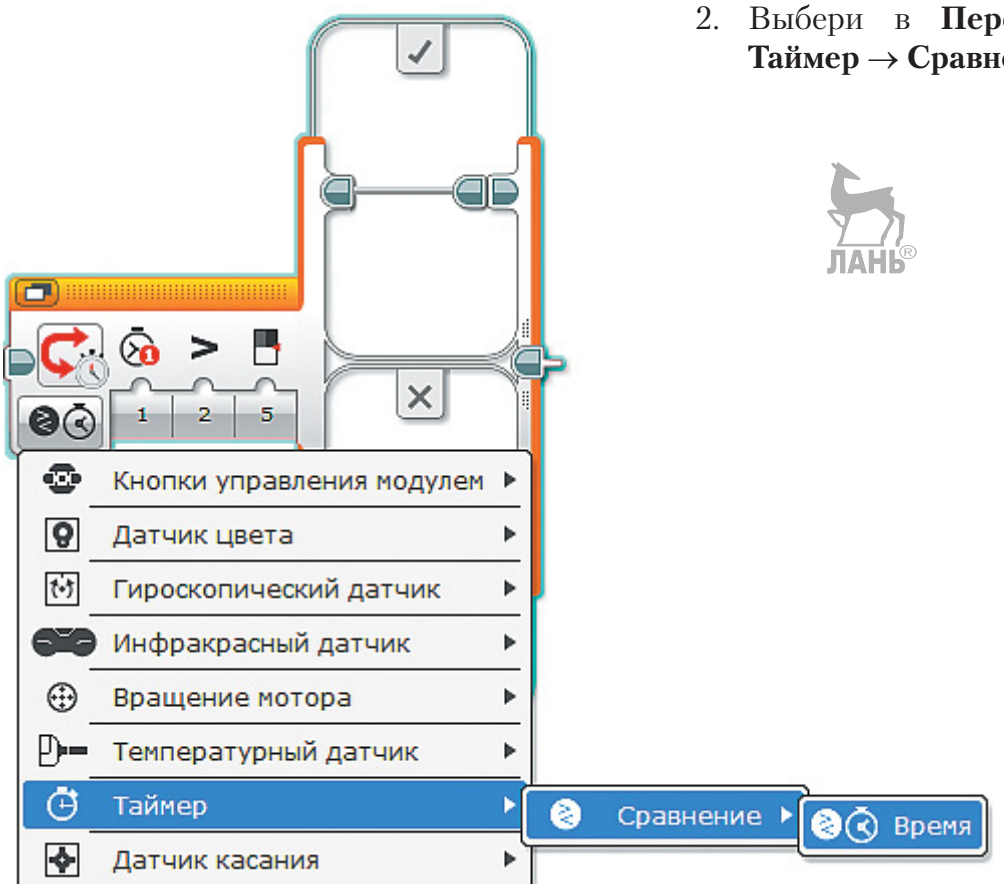
$$S_2 = S_1 + (V \cdot T),$$

где S_1 — значение расстояния, пройденного за предыдущее время; V — скорость в определенный момент времени; T — время обновления информации на экране, равное 0,00027 часа при скорости, не равной нулю (это значение ты рассчитывал тоже в пункте 24 первой части).

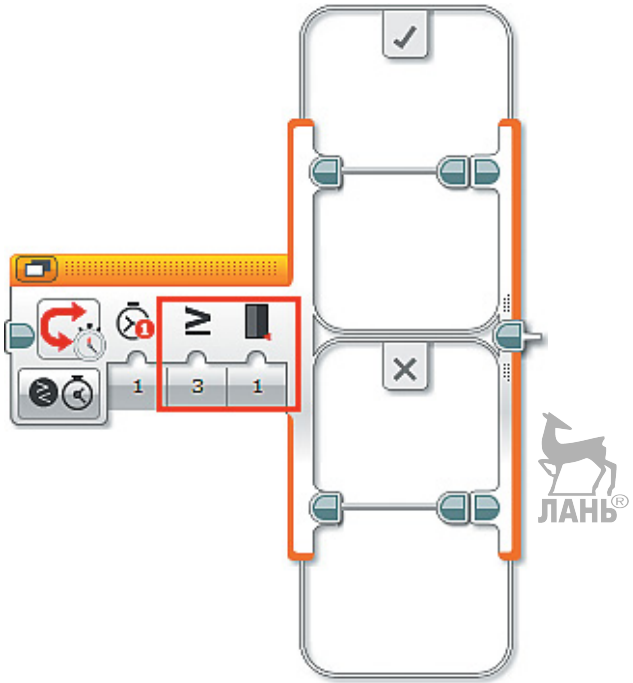
Приступай!

1. Вернись в основную программу **Main**. Тебе осталось построить последнюю, третью ветку в этой программе. Добавь команду **Переключатель** (*оранжевый блок*) и расположи её под командой **Alarm**, а затем соедини её с помощью шлейфа от команды **Engines**.





2. Выбери в **Переключателе** опцию **Таймер → Сравнение → Время**.

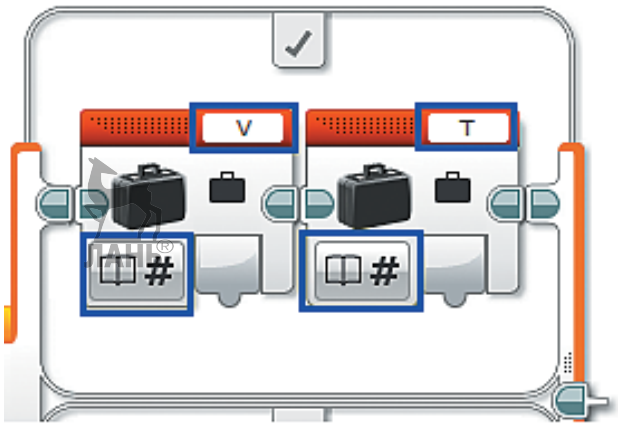


3. Этот **Переключатель** заработает сразу, как только в одном из **таймеров** (например, в первом) будет превышаться значение 1 секунда. Для этого настрой параметры **Переключателя**:
   - **Тип сравнения** — больше или равно (3);
   - **Пороговое значение** — 1 (введи с клавиатуры).

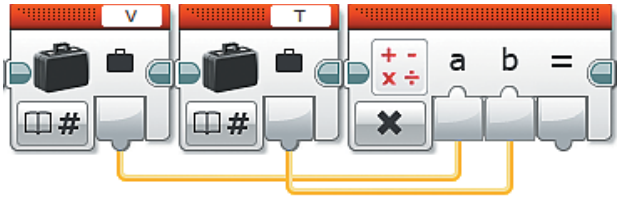
4. Нас **не интересует** интервал времени от 0 до 1 секунды, поэтому весь фрагмент программы для одометра будет внутри вкладки **Истина**. Реализуем формулу подсчёта пройденного расстояния:

$$S_2 = S_1 + (V \cdot T).$$

Добавь две команды **Переменная** (*красный блок*), выбери в них опцию **Считывание → Числовое значение**. В первой команде для обработки установи переменную **V**, а во второй — **T**.

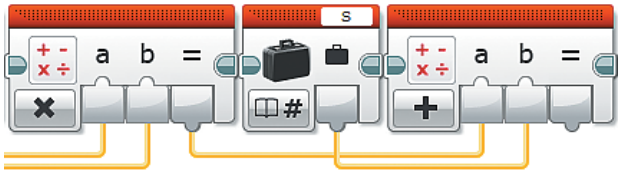


5. Добавь команду **Математика** (*красный блок*) и выбери опцию **Умножить**. Соедини выход переменной **V** с входом **a**, выход переменной **T** с входом **b**.

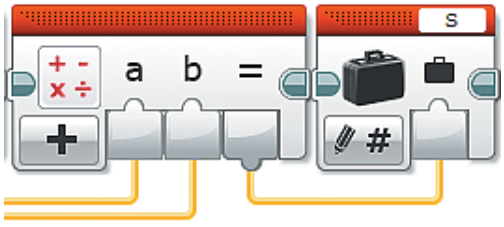


6. Теперь рассчитанное значение расстояния, пройденного за 1 секунду (0,00027 часа) нужно прибавить к предыдущему значению **S**.
Добавь команду **Переменная** (*красный блок*). Для обработки установи переменную **S** и выбери опцию **Считывание → Числовое значение**.

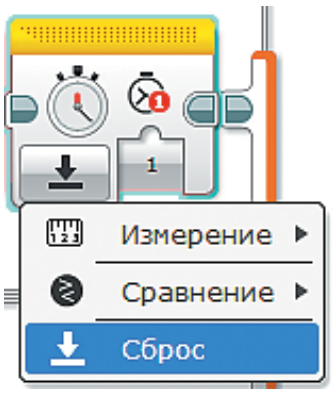
7. Добавь команду **Математика** (*красный блок*), опцию оставь без изменений, соедини выход «=» предыдущей команды **Математика** с входом **a**, выход переменной **S** с входом **b**.



8. Осталось записать результаты подсчёта в переменную **S**. Добавь команду **Переменная** (*красный блок*) и выбери для обработки переменную **S**, затем соедини выход «=» команды **Математика** с входом переменной **S**.



9. После этого остаётся только сбросить значение таймера № 1, используемого для одометра. Добавь команду **Таймер** (*жёлтый блок*), по умолчанию в ней выбран таймер № 1. Установи опцию **Сброс**.



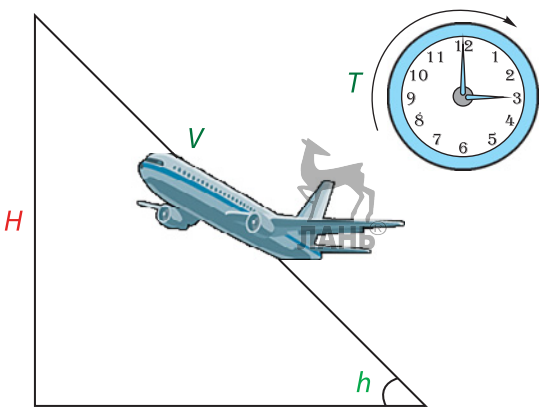
Так держать! Теперь ты всегда будешь знать, какое расстояние преодолел твой самолёт!

Часть 6. Всё выше, и выше, и выше! Работа альтиметра, расчёт набранной высоты

Логика. В этой части ты обеспечишь работу одного из наиболее сложных приборов — **альтиметра** (от латинского altus — высоко). Вообще, для расчёта высоты нужны знания такого раздела математики, как тригонометрия, который изучают старшеклассники. Но учёные придумали методы преобразования различных сложных формул в более лёгкие и понятные. Воспользуемся ими.

Ты уже с уверенностью можешь сказать, какие параметры полёта тебе известны. Для расчёта высоты тебе понадобятся:

- скорость V в расчётный момент времени;
- конкретный момент времени $T = 0{,}00027$ часа;
- угол тангажа (показания мотора, подключенного к порту **C**).



Как и для других приборов, ты будешь рассчитывать приращение высоты за определённый промежуток времени (1 секунда = 0,00027 часа) и прибавлять это значение к общему значению высоты.

Набранная высота H_2 на момент отсчёта определяется по формуле:

$$H_2 = H_1 + \left(\frac{(h^2 + 3) \cdot h \cdot T \cdot V}{3} \cdot 1000 \right),$$

где H_1 — высота, набранная за предыдущее время;

h — угол тангажа, переведённый из градусной меры в радианную (дело в том, что градусы плохо пригодны для подобных «плоских» подсчётов, поэтому существует другая мера — радианы);

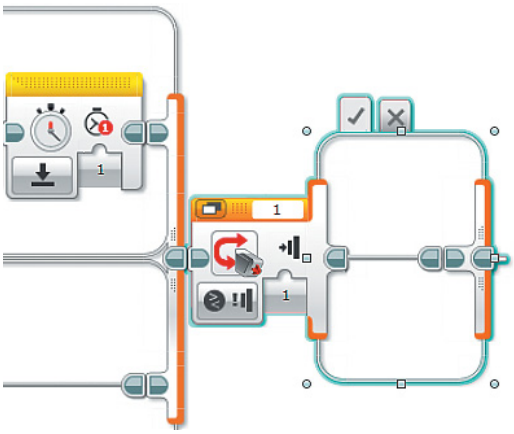
T — фиксированное время; V — скорость в определённый момент времени.

Результат расчётов по этой формуле получится в километрах, а высоту удобнее и правильнее представлять в метрах, поэтому в конце формулы стоит множитель 1000.

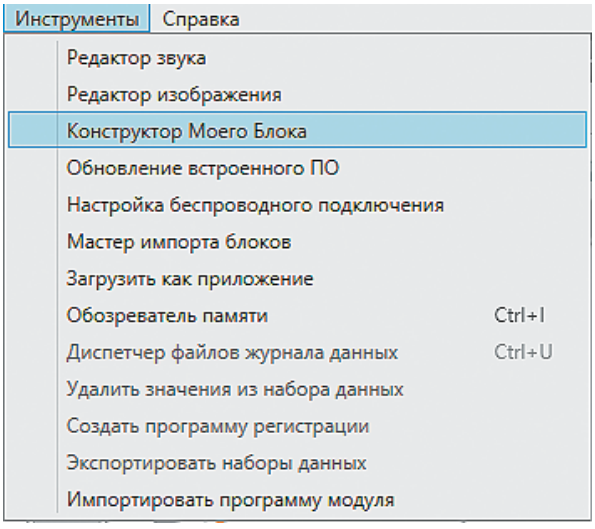
Часть формулы в больших скобках и есть приращение высоты.

Такой сложный процесс целесообразно вынести в отдельную **подпрограмму**.

1. В основной программе **Main** продолжи третью, нижнюю, ветку. После **Переключателя** добавь ещё один **Переключатель** (*оранжевый блок*) и перейди к **Виду с вкладками**.

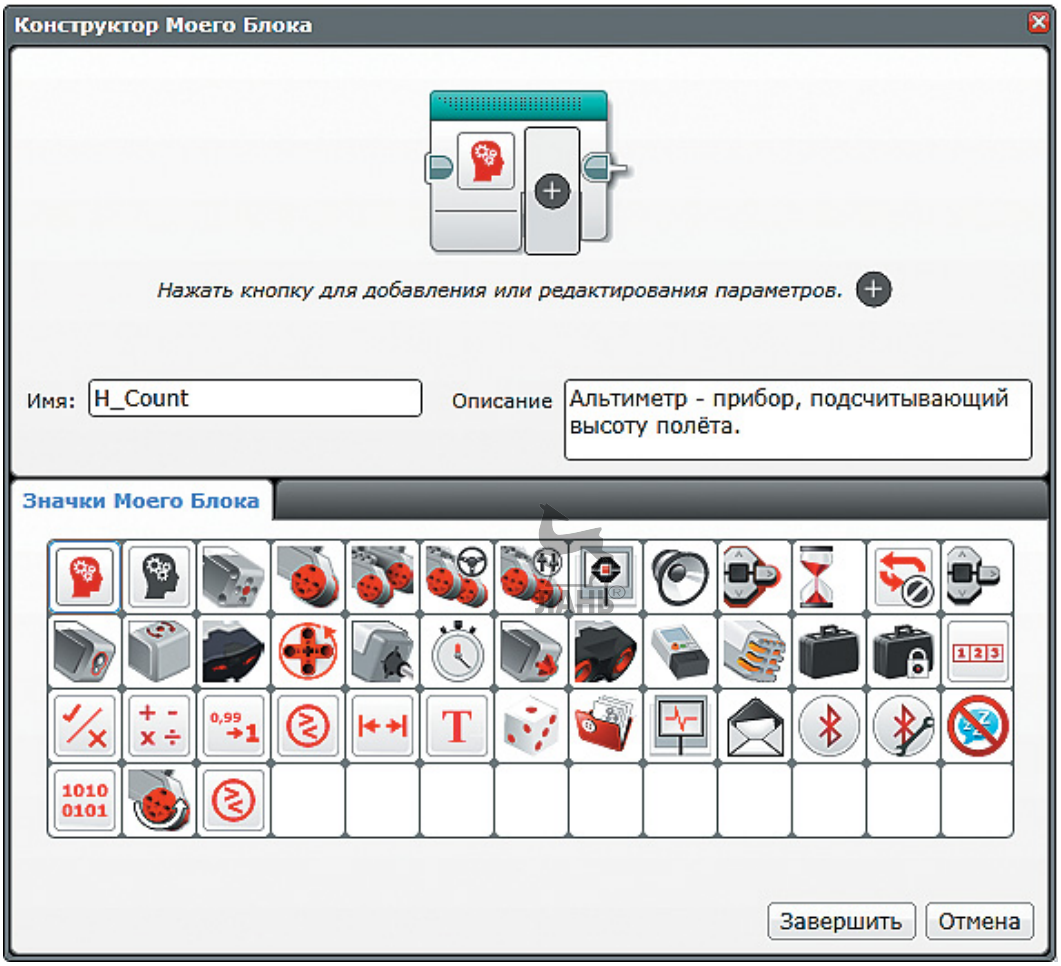


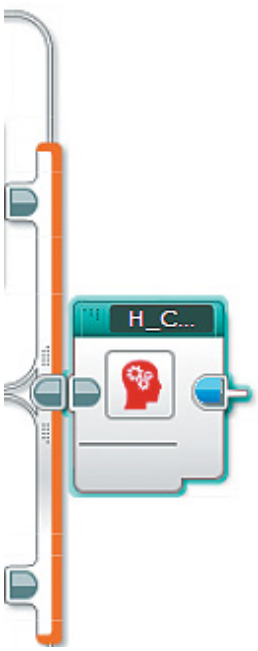
2. Теперь начинай создавать подпрограмму. Команда **Переключатель** должна быть выделена (подсвечена голубым контуром). В верхней части программы выбери меню **Инструменты → Конструктор Моего Блока.**



3. В открывшемся окне задай имя подпрограммы, введи её краткое описание и выбери подходящую пиктограмму.

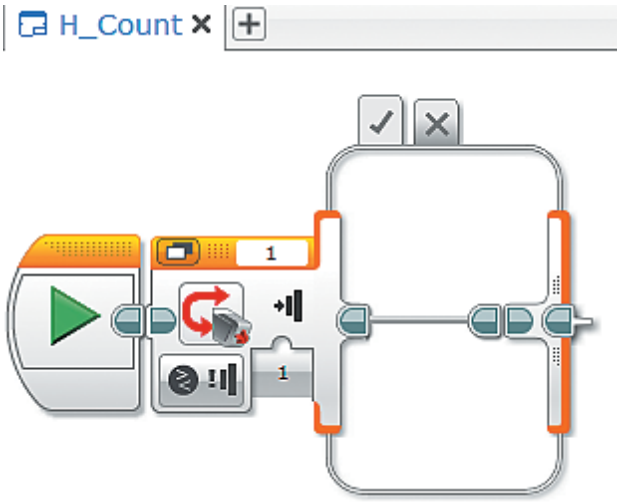
Имя логично задать, например, **H_Count** (от английского height — высота; count — подсчёт). Затем нажми кнопку **Завершить**. Краткое описание и пиктограмму можешь задать, как на рисунке или самостоятельно.



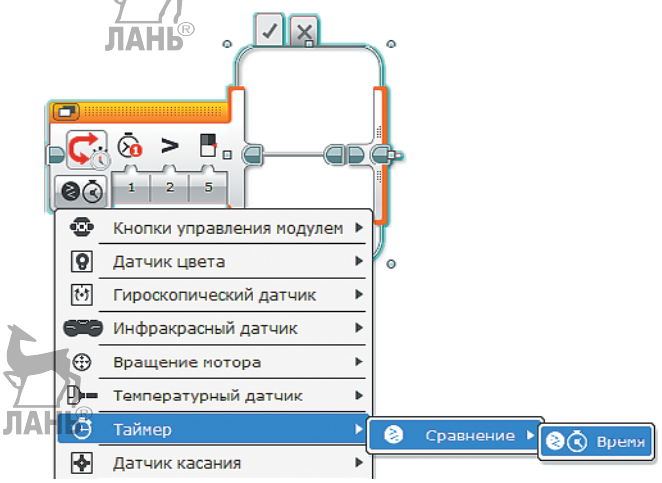


4. В основной программе появится новая команда **H_Count**. Именно в ней будет происходить подсчёт показаний альтиметра.

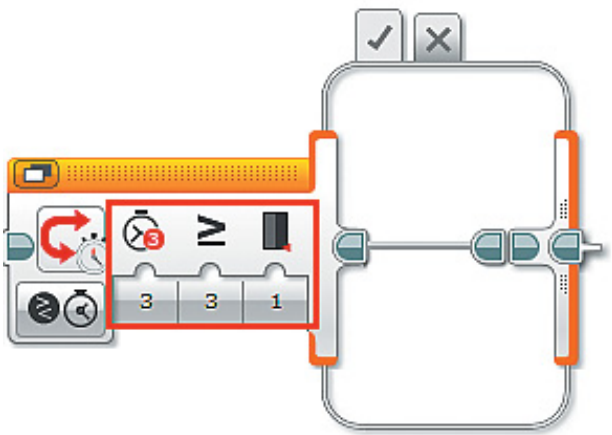
5. Дважды кликни по ней. На экране откроется новое рабочее поле для редактирования подпрограммы. Сверху появится вкладка с именем **H_Count**.



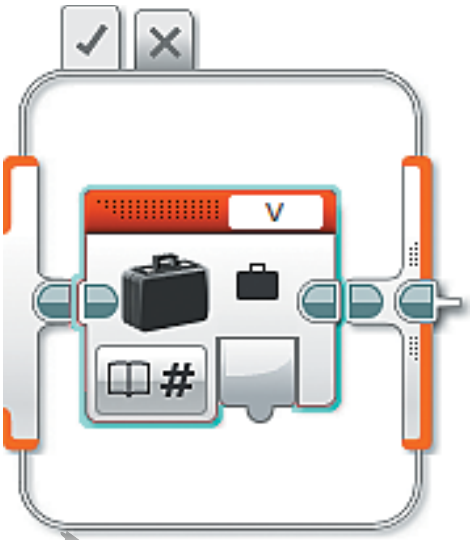
6. Для **Переключателя** выбери опцию **Таймер → Сравнение → Время**.



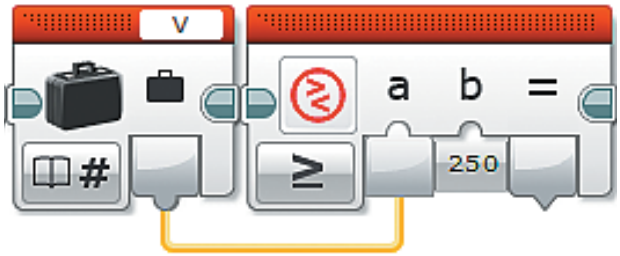
7. Настрой параметры:
   - **Идентификатор таймера** — 3;
   - **Тип сравнения** — больше или равно (3);
   - **Пороговое значение** — 1 (введи с клавиатуры).



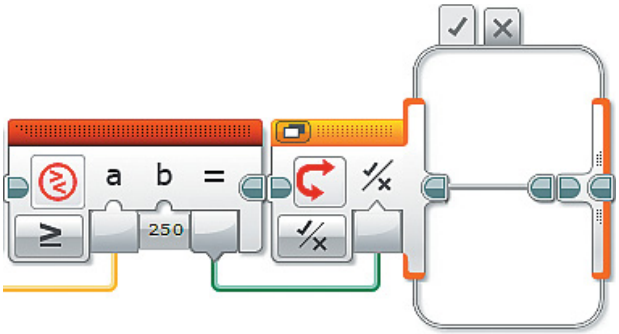
8. Такую конструкцию ты уже реализовывал в предыдущей части — она обеспечивает частоту обновления информации 1 раз в секунду. Поэтому в **Переключателе** нас будет интересовать только вкладка **Истина**.
Будет правильно обеспечить минимальные скорости взлёта и посадки. В среднем, в зависимости от модели самолёта, она составляет 250 км/ч, поэтому введём ограничение на возможность изменения высоты при скорости меньше 250 км/ч.
Добавь команду **Переменная** (*красный блок*), для обработки установи переменную **V**, выбери опцию **Считывание → Числовое значение**.



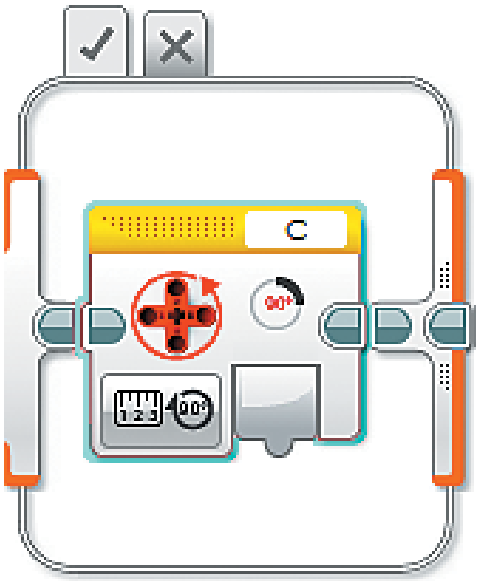
9. Далее добавь команду **Сравнение** (*красный блок*), выбери опцию **Больше или равно**, затем соедини выход переменной **V** с входом **a** команды **Сравнение**, а во вход **b** введи с клавиатуры число **250**.



10. Теперь добавь команду **Переключатель** (*оранжевый блок*), перейди к **Виду с вкладками**, выбери опцию **Логическое значение** и соедини выход «=» команды **Сравнение** с входом **Переключателя**.

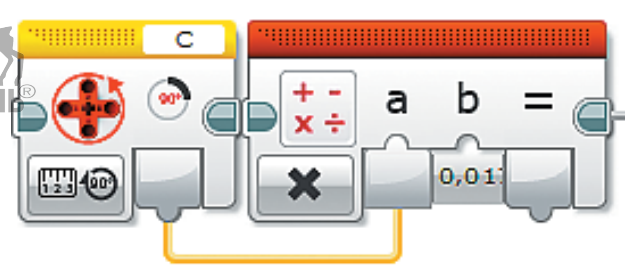


11. Для начала поработай с вкладкой **Истина**. В ней нужно реализовать формулу для расчёта высоты, поэтому приготовься к большому объёму математических действий. Добавь команду **Вращение мотора** (*жёлтый блок*). Проверь, чтобы в верхнем правом окошке значился порт **C**.

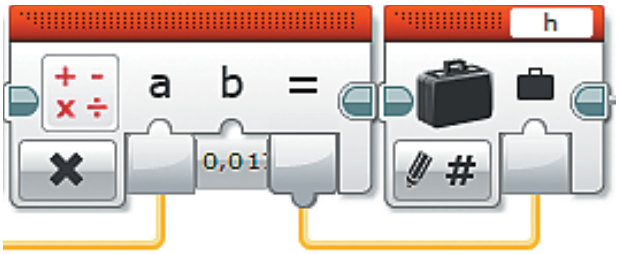


12. Далее добавь команду **Математика** (*красный блок*) и выбери опцию **Умножить**. Затем соедини выход команды **Вращение мотора** с входом **a**, во вход **b** с клавиатуры введи число **0,0175**.

Пояснение. Именно это действие позволяет перевести градусную меру угла в радианную.

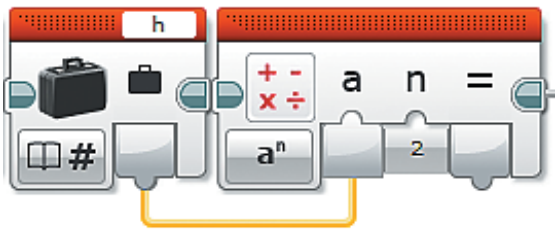


13. Добавь команду **Переменная** (*красный блок*). Для обработки выбери переменную **h** и соедини выход «=» команды **Математика** с входом переменной **h**.

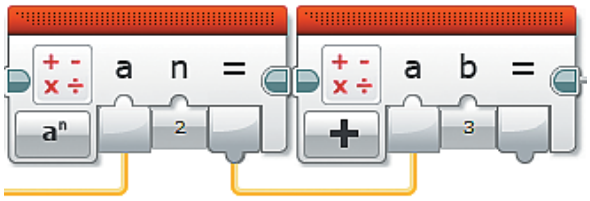


14. Согласно формуле, тебе нужно возвести полученное значение в квадрат. Добавь команду **Переменная** (*красный блок*), установи для обработки переменную **h**, выбери опцию **Считывание → Числовое значение**.

15. Добавь команду **Математика** (*красный блок*), выбери опцию **Показатель степени** и соедини выход переменной **h** с входом **a**, во вход **n** с клавиатуры введи **2**.

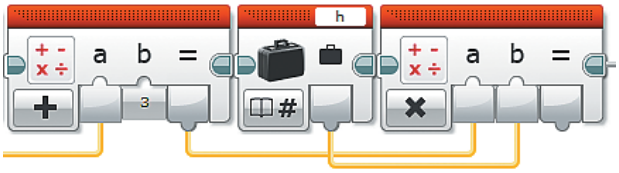


16. К этому значению нужно прибавить 3. Добавь команду **Математика** (*красный блок*) и соедини выход «=» предыдущей команды с входом **a** новой, во вход **b** введи с клавиатуры число **3**.

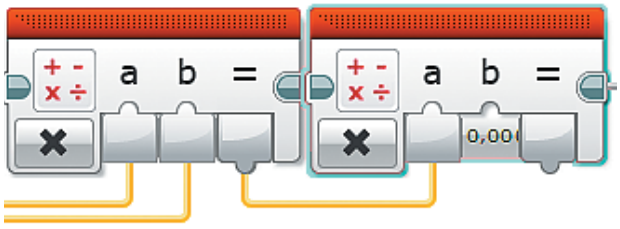


17. Полученный результат нужно умножить на **h**. Добавь команду **Переменная** (*красный блок*). Установи для обработки переменную **h** и выбери опцию **Считывание → Числовое значение**.

18. Далее добавь команду **Математика** (*красный блок*) и выбери опцию **Умножить**. Соедини выход «=» предыдущей команды **Математика** с входом **a**, а выход переменной **h** с входом **b**.

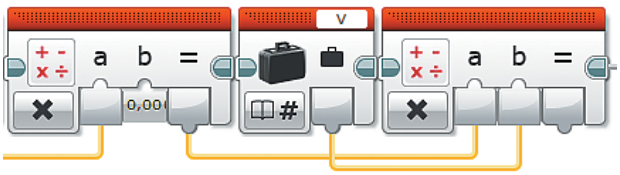


19. Теперь полученное значение умножим на фиксированное время. Добавь команду **Математика**, выбери опцию **Умножить** и соедини выход «=» предыдущей команды **Математика** с входом **a**, во вход **b** с клавиатуры введи число **0,00027**.

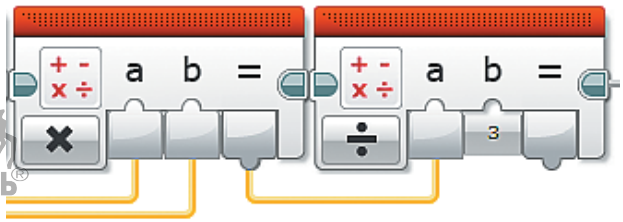


20. Согласно формуле, в числителе остаётся умножить результат на скорость *V*. Добавь команду **Переменная** (*красный блок*), установи для обработки переменную **V** и выбери опцию **Считывание → Числовое значение**.

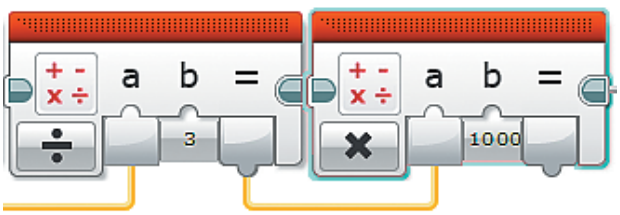
21. Добавь команду **Математика** (*красный блок*). Выбери опцию **Умножить**, а затем соедини выход «=» предыдущей команды **Математика** с входом **a**, выход переменной **V** с входом **b**.



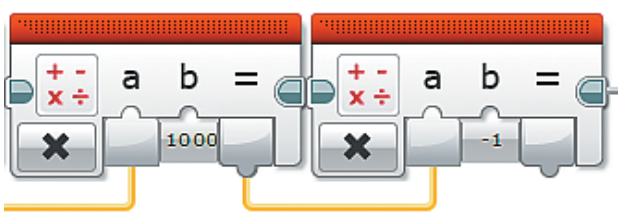
22. Теперь результат всех вычислений нужно разделить на 3. Добавь команду **Математика** (*красный блок*). Выбери опцию **Разделить** и соедини выход «=» предыдущей команды **Математика** с входом **a**, во вход **b** с клавиатуры введи **3**.



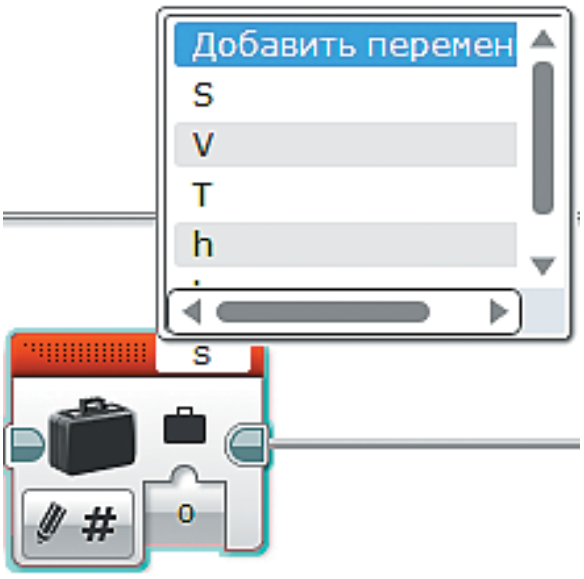
23. Для перевода километров в метры результат нужно умножить на 1000. Добавь команду **Математика** (*красный блок*). Выбери опцию **Умножить** и соедини выход «=» предыдущей команды **Математика** с входом **a**, во вход **b** с клавиатуры введи число **1000**.



24. Подсчёты практически завершены. Последний шаг, который тебе надо сделать, — скорректировать вычисление, умножив результат на –1. Это связано, как ты помнишь, с расположением вертикального мотора. Таким образом ты произведёшь ***инвертирование*** (одна из математических хитростей) результатов подсчёта. Тогда значение высоты будет положительным.
Добавь команду **Математика** (*красный блок*). Выбери опцию **Умножить**, затем соедини выход «=» предыдущей команды **Математика** с входом **a**, а во вход **b** с клавиатуры введи **–1**.

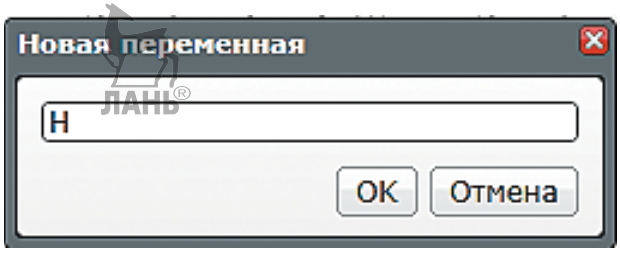


25. Рассчитанное мгновенное значение высоты нужно прибавить (то есть прирастить) к основной высоте. Для неё ещё не объявлена переменная, поэтому добавь команду **Переменная** (*красный блок*) и в правом верхнем окошке выбери пункт **Добавить переменную**.

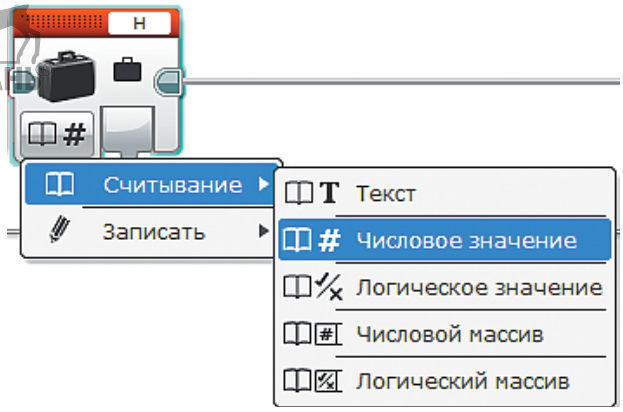


26. В открывшемся окне введи с клавиатуры **H** и нажми **ОК**.

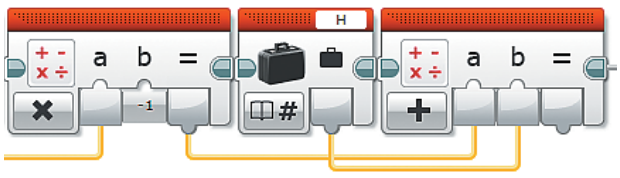
Пояснение. Прописные и строчные буквы для обозначения переменных – это разные буквы, и их можно и нужно использовать как отдельные объекты.



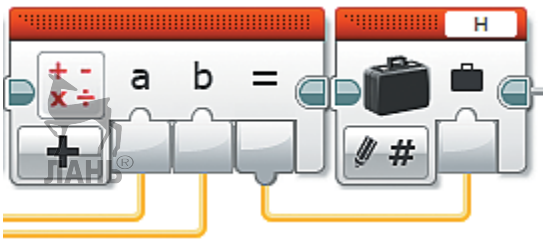
27. В команде **Переменная** выбери опцию **Считывание → Числовое значение**.



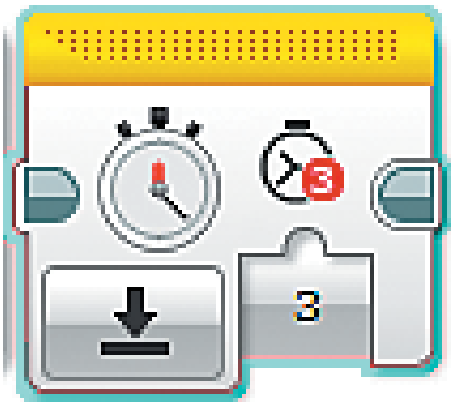
28. Далее добавь команду **Математика** (*красный блок*) и соедини выход «=» предыдущей команды **Математика** с входом **a**, выход переменной **H** с входом **b**.



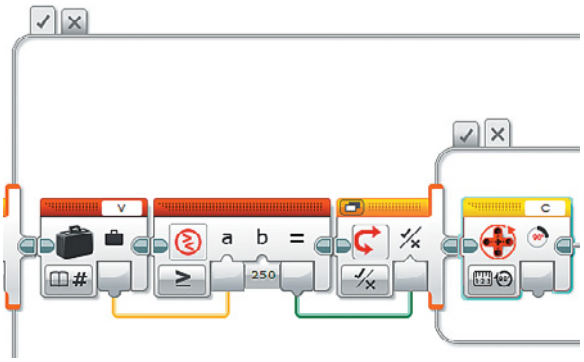
29. Запиши результат в блок **H**. Для этого добавь команду **Переменная** (*красный блок*), установи для обработки переменную **H** и соедини выход «=» команды **Математика** с входом переменной **H**.



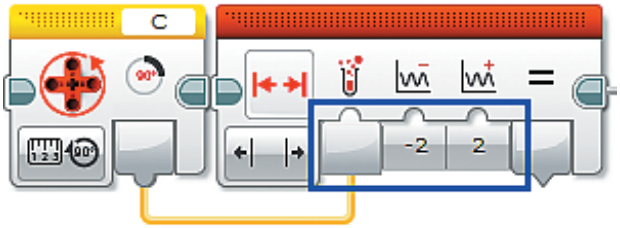
30. Осталось сбросить значение таймера. Добавь команду **Таймер** (*жёлтый блок*), выбери опцию **Сброс** и установи **Идентификатор таймера**: 3.



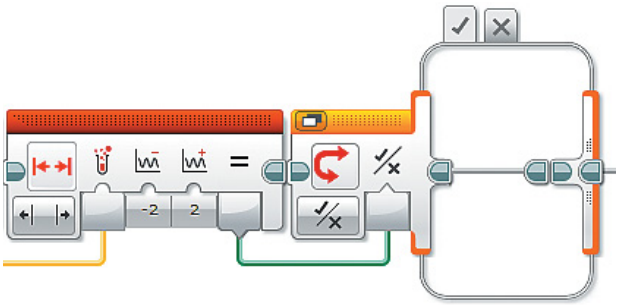
31. Вернёмся к случаю, когда условие «скорость превысила 250 км/ч» **не выполнено**. Перейди к вкладке **Ложь** (крестик наверху команды) внутреннего **Переключателя** и добавь туда команду **Вращение мотора** (*жёлтый блок*). Проверь, чтобы в верхнем правом окошке значился порт **C**.



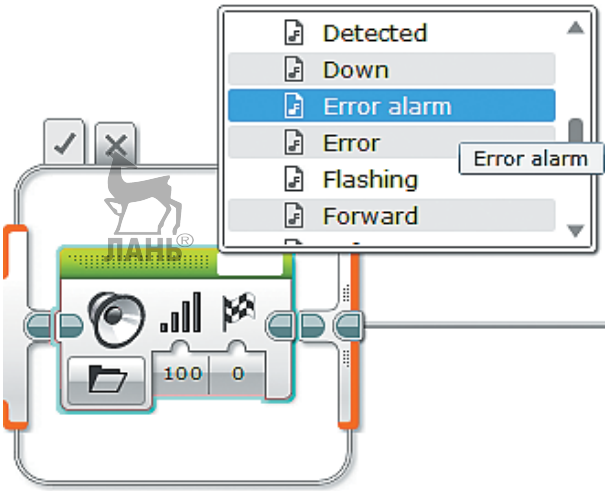
32. При разбеге самолёта штурвал может незначительно отклоняться, такой случай нужно обязательно предусмотреть. Пусть допустимым отклонением штурвала будет интервал от −2 до 2 градусов.
Добавь команду **Интервал** (*красный блок*) и выбери опцию **За пределами**. Далее соедини выход команды **Вращение мотора** с входом **Тестовое значение**. Во входы **Нижняя граница** и **Верхняя граница** введи с клавиатуры соответственно **−2** и **2**.



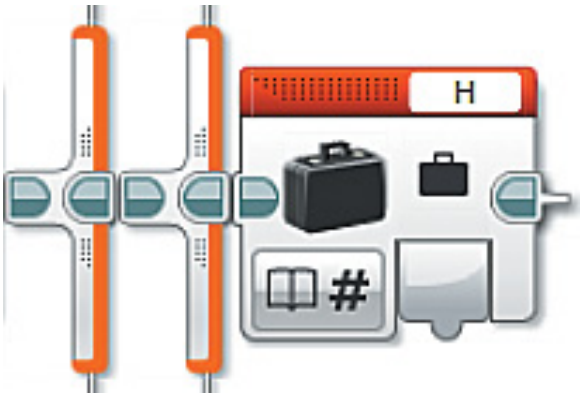
33. Теперь, если допустимое отклонение действительно превышено, необходимо подать сигнал тревоги. Добавь команду **Переключатель** (*оранжевый блок*). Перейди к **Виду с вкладками**, выбери опцию **Логическое значение** и соедини выход «=» команды **Интервал** с входом **Переключателя**.



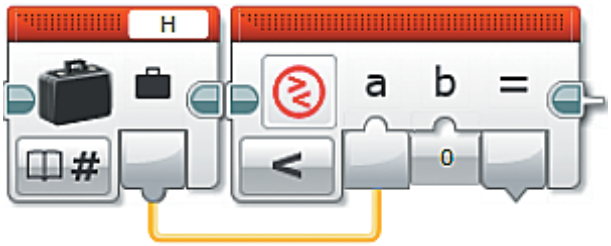
34. Тебя будет интересовать только вкладка **Истина**. Добавь в неё команду **Звук** (*зелёный блок*) и в правом верхнем окошке выбери в меню папку **Звуковые файлы LEGO**, в ней папку **Информация** и наконец файл **Error alarm**.



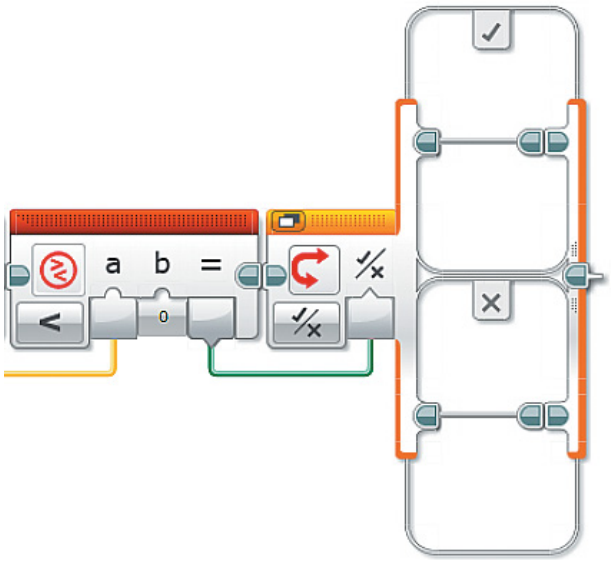
35. Осталось предусмотреть тот случай, когда значение высоты попытается «убежать» в отрицательную сторону (самолёт же не может улететь под землю!).
В конец этой подпрограммы после внешнего **Переключателя** добавь команду **Переменная** (*красный блок*), установи для обработки переменную **H**, выбери опцию **Считывание** → → **Числовое значение**.



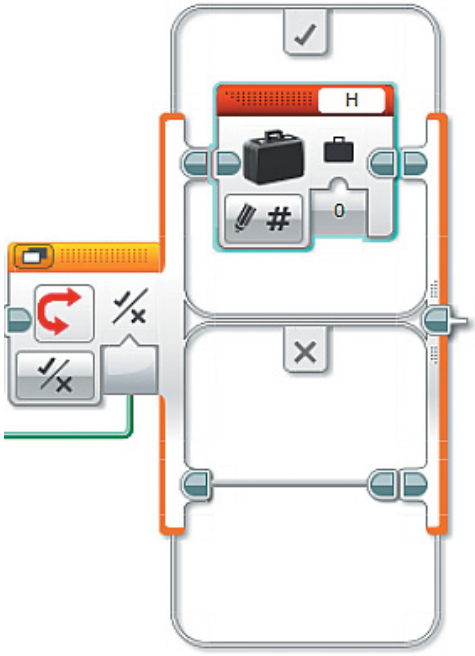
36. Далее добавь команду **Сравнение** (*красный блок*), выбери опцию **Меньше** и соедини выход переменной **H** с входом **a**, во вход **b** с клавиатуры введи число **0**.



37. Теперь добавь команду **Переключатель** (*оранжевый блок*), выбери опцию **Логическое значение** и соедини выход «=» команды **Сравнение** с входом **Переключателя**.



38. Во вкладку **Истина** добавь команду **Переменная** (*красный блок*) и установи для обработки переменную **H**. Опцию и входное значение менять не нужно.



Пояснение. К такой хитрости ты уже прибегал, когда настраивал подобный запрет для скорости — если она попытается «убежать» ниже нуля, то программа «вернёт» её значение к нулю.

Поздравляем! Самое сложное позади!

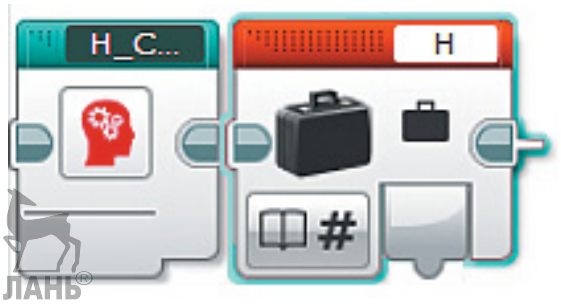
Часть 7. Последний рывок. Вывод на экран показаний всех приборов, завершение

Ты большой молодец! Ты проделал огромный путь к тому, чтобы твой авиасимулятор заработал в полную силу! Самое сложное позади, осталось сделать последние штрихи и вывести на экран показания всех приборов.

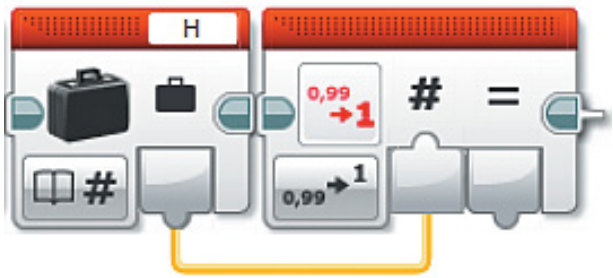
Некоторые показания приборов нужно будет округлить до целых значений, обеспечить корректный вывод показаний на экран EV3 с правильным расположением и задать частоту обновления информации на экране.

Вперёд!

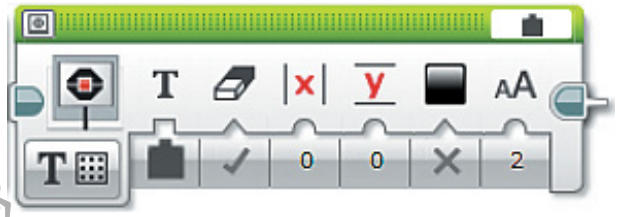
1. Вернись в основную программу **Main** и продолжи составление программы после команды **H_Count**. Добавь команду **Переменная** (*красный блок*), установи для обработки переменную **H** и выбери опцию **Считывание** → → **Числовое значение**.



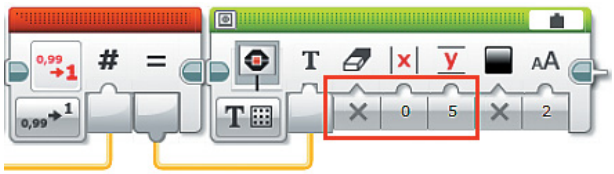
2. Так как формула для расчёта высоты довольно сложная, то и результат ты получишь далеко не целый. Удобнее будет его округлить. Добавь команду **Округление** (*красный блок*) и соедини выход переменной **H** с входом **Ввод** этой команды.



3. Выведи значение высоты на экран EV3. Для этого добавь команду **Экран** (*зелёный блок*), выбери опцию **Текст** → **Сетка**, в правом верхнем окошке выбери пункт **Проводной**.

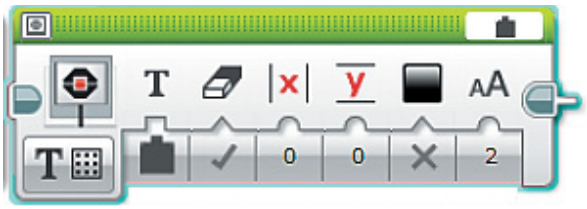


4. Теперь соедини выход «=» команды **Округление** с входом **Текст**. Настрой параметры команды **Текст**:
   - **Очистить экран** — ложь;
   - **X** = 0, **Y** = 5 (то есть значение будет выводиться у левого края экрана, но ниже первой строки, к середине).

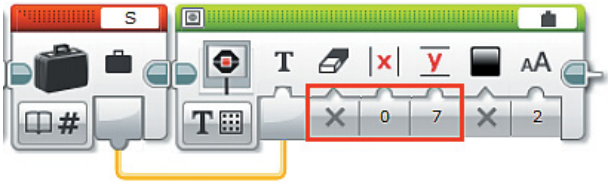


5. Осталось вывести на экран показания одометра. Добавь команду **Переменная** (*красный блок*), установи для обработки переменную **S**, выбери опцию **Считывание** → **Числовое значение**.

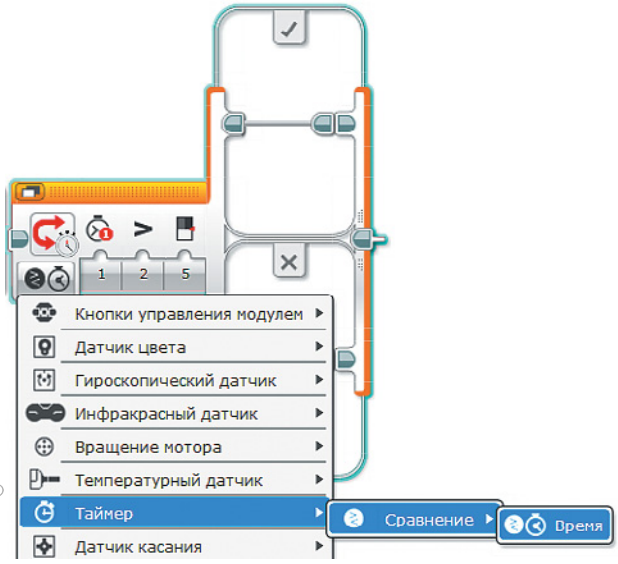
6. Добавь команду **Текст** (*зелёный блок*), выбери опцию **Текст** → **Сетка**.



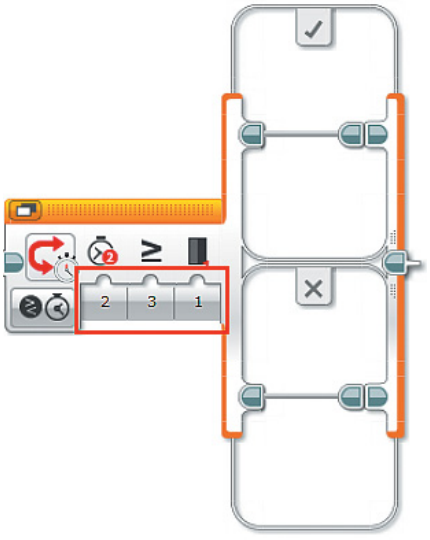
7. Соединии выход переменной **S** с входом **Текст**. Настрой параметры команды **Текст**:
   - **Очистить экран** — ложь;
   - **X** = 0, **Y** = 7 (то есть значение будет выводиться у левого края экрана, но ниже первой строки, к середине).



8. Последнее, что осталось сделать, — это то, о чём мы с тобой много говорили. Нужно обеспечить частоту обновления информации на экране EV3 1 раз в секунду. Сделать это очень просто: с помощью таймера очищать экран 1 раз в секунду. Добавь команду **Переключатель** (*оранжевый блок*) и выбери опцию **Таймер** → **Сравнение** → **Время**.

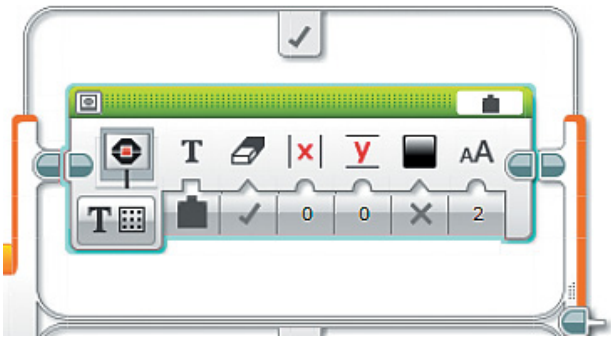


9. Настрой параметры команды:
   - **Идентификатор таймера** — 2;
   - **Тип сравнения** — больше или равно (3);
   - **Пороговое значение** — 1 (введи с клавиатуры).



10. Нас интересует только выполненное условие, поэтому во вкладку **Истина** добавь команду **Экран** (*зелёный блок*), выбери опцию **Текст** → **Сетка**, а в правом верхнем окошке — пункт **Проводной**.

    В этой команде никакие параметры настраивать не требуется. Здесь важно, что она ничего не выводит на экран, а параметр **Очистка экрана** — **Истина**.



11. Теперь нужно сбросить показания таймера № 2. Добавь команду **Таймер** (*жёлтый блок*), выбери опцию **Сброс** и **Идентификатор таймера**: **2**.

Внимание! Так как в твоём проекте содержится несколько программ, то при запуске авиасимулятора ты должен загрузить в память EV3 и запустить главную из них — **Main**.

Этап 5. Загрузка программы и её тестирование

ШАГ 1. ЗАГРУЗКА ПРОГРАММЫ В ПРОГРАММИРУЕМЫЙ МОДУЛЬ

Загрузи свою программу.

1. Подключи программируемый модуль с помощью USB-кабеля к компьютеру, соединив порты PC на модуле и USB на компьютере. В окне программного обеспечения станет активен блок EV3.

2. Нажми кнопку **Загрузить и запустить программу**. Программа запишется в память программируемого модуля и сразу запустится.

ШАГ 2. ТЕСТИРОВАНИЕ

1. Перед запуском программы штурвал должен находиться строго в вертикальном положении, то есть, если твой авиасимулятор стоит на столе, вертикальный мотор (управляет углом тангажа) должен быть перпендикулярен столу.

2. Нажми кнопку правого датчика касания — на экране EV3 в правом верхнем углу должно появиться нарастающее значение скорости. То же самое проделай с левым датчиком. Для него, наоборот, значение скорости начнёт уменьшаться. Убедись, что скорость не выходит из диапазона 0–850 км/ч.

3. Наклони штурвал на себя — в верхней части экрана показания угла *тангажа* должны измениться в отрицательную сторону. При наклоне штурвала от себя показания изменятся в положительную сторону.
   Проделай эти же действия для крена, наклонив штурвал вправо и затем влево.

4. При скорости менее 250 км/ч и отклонении угла тангажа в ту или другую сторону более, чем на 2 градуса, должен подаваться запрещающий сигнал.

5. При скорости более 250 км/ч и превышении критических углов тангажа должны подаваться предупреждающие сигналы **Up** или **Down.**

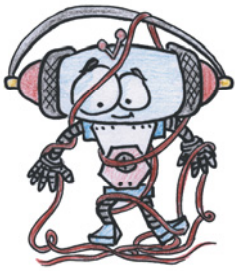
6. После 15 сигналов **Up** или 10 сигналов **Down** авиасимулятор должен воспроизвести звук крушения **Aircrush** и завершить работу.

7. При скорости более 250 км/ч в зависимости от угла тангажа должны увеличиваться (штурвал на себя) или уменьшаться (штурвал от себя) показания альтиметра — значения высоты полёта.

8. При скорости более 0 км/ч в нижней части экрана EV3 должны изменяться показания одометра: чем выше скорость, тем быстрее растёт значение.

Если ты хочешь повторно запустить программу или продемонстрировать работу своего робота другу, то для его запуска компьютер тебе уже не нужен: включи программируемый блок, затем выбери папку **SteepDive** и в ней программу **Main**.

Внимание!
Помни: наша программа — это бесконечный цикл. Поэтому остановить её можно только вручную с помощью кнопки **Отмена** на программируемом модуле.



Этап 6. От винта!

Теперь ты можешь позвать друзей, и вы вместе поиграете в «Экипаж самолёта»!

Роли можно распределить так:

Командир корабля, первый пилот — управляет самолётом при взлёте и дальнейшем полёте, ведёт переговоры с авиадиспетчером.

Второй пилот — управляет самолётом, производит посадку самолёта, ведёт переговоры с авиадиспетчером.

Авиадиспетчер — ведёт переговоры с пилотами, наблюдает за ходом полёта.

Сценарий, по которому вы можете сыграть:

- командир корабля запрашивает у авиадиспетчера подробную сводку о погодных условиях на заданном маршруте в пунктах взлёта и посадки;
- авиадиспетчер сообщает сводку о погодных условиях (температуре воздуха, скорости ветра, солнечно/пасмурно/дождь/снег);
- командир корабля запрашивает у авиадиспетчера разрешение на взлёт;
- авиадиспетчер даёт разрешение на взлёт;
- командир информирует экипаж о готовности к взлёту, набирает скорость, плавно тянет штурвал на себя, взлетает, набирая высоту.

Второй пилот в это время следит за показаниями на приборной панели и информирует командира.

Затем:

- командир корабля и второй пилот меняются ролями;
- второй пилот запрашивает у авиадиспетчера разрешение на посадку;
- авиадиспетчер даёт разрешение на посадку, сообщает об атмосферном давлении, скорости и направлении ветра, температуре воздуха в пункте посадки;
- второй пилот сообщает о готовности к посадке, уменьшает скорость, отклоняет штурвал от себя, плавно снижаясь, а затем производит посадку при **безопасной** скорости 300–350 км/ч и угле тангажа не более 2 градусов;
- командир корабля следит за скоростью самолёта и информирует второго пилота о показаниях приборов;
- авиадиспетчер наблюдает за посадкой самолёта, даёт коррективы.

В завершение командир корабля сообщает авиадиспетчеру об успешном приземлении самолёта.

Это ещё не всё! Твой авиасимулятор может ещё больше приблизить тебя к романтике настоящего полёта!

Горизонт НЕ завален!

Найди в сети Интернет характеристики одного из популярных пассажирских самолётов (например, Sukhoi Superjet100 или Ту-204) и определи средний показатель допустимого угла крена самолёта — усовершенствуй свою программу так, чтобы авиасимулятор сигнализировал об опасности.

Захожу на второй круг!

Часто случается так, что самолёт не может зайти на посадку с первого раза, и экипажу приходится с помощью авиадиспетчера заходить на повторные круги. Для этого, несомненно, нужно знать, на какой угол выполнил поворот самолёт и каково его отклонение от первоначальной траектории.

Дополни авиасимулятор новым прибором, который будет показывать угол, на который отклонился самолёт. Для расчётов можешь воспользоваться следующей формулой:

$$Q = \frac{r \cdot 0{,}000046}{V},$$

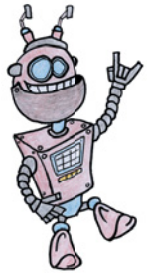
где Q — угол, на который повернулся самолёт; r — угол крена (в градусах); V — текущая скорость воздушного судна (в км/ч).

Помни! Расчёт этого значения необходимо делать с периодичностью 1 раз в секунду и каждый раз прибавлять его к предыдущему (суммировать общее значение).

Когда самолёт поворачивает, у него появляется крен в ту или иную сторону. Прибор рассчитывает по формуле, на какой угол самолёт отклонился от прежней траектории, и этот угол нужно «запомнить». Но когда самолёт закончит поворот, значение крена станет равным нулю, и показания прибора тоже станут равными нулю. Вот здесь и нужен авиадиспетчер, который с помощью линейки и транспортира фиксирует на бумаге все значения угла поворота самолёта, то есть его новой траектории.

Дозаправка требуется?

Придумай, как реализовать в программе учёт расхода топлива и упреждающую сигнализацию об его уровнях, особенно о критическом значении. В нижней части дисплея EV3 ещё осталось место, поэтому можно написать подпрограмму **Fuel**, указывающую заполненность топливных баков (например, в процентах). В зависимости от развиваемой мощности турбин и этапа полёта запасы топлива будут уменьшаться быстрее или медленнее.



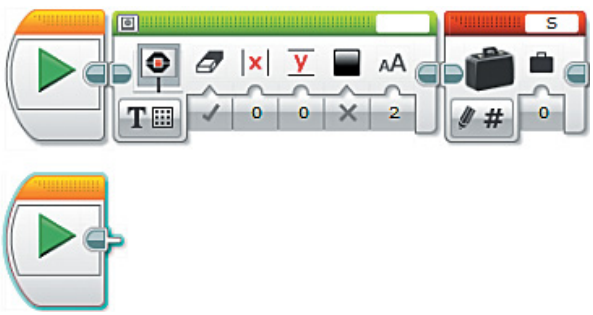
И ГУЛ ТУРБИН МЫ СЛЫШИМ ВНОВЬ!

Сделай авиасимулятор более реалистичным — добавь в него звук работающих турбин! Для этого скачай с сайта издательства по ссылке **http://pilotlz.ru/files/10045/** три файла: **Low** (низкий), **Medium** (средний) и **High** (высокий), каждый из которых соответствует разным уровням мощности работы турбин.

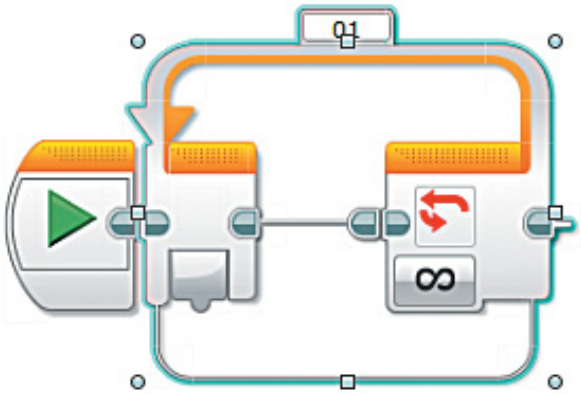
Для примера зададим звучание турбин на низкой мощности.

Снова открой свой проект **SteepDive** в LME-EV3 (**Файл → Открыть проект → Открыть**) и перейди к основной программе **Main**. Найди её начало, оно будет слева (это команда **Начало** с пиктограммой зелёного треугольника). На самом деле вся твоя программа и подпрограммы начинались именно с этой команды, и это неслучайно! В программировании есть такой приём, как **параллельное исполнение** двух различных процессов в одной программе. И вот теперь параллельно с работой основной программы ты запустишь воспроизведение аудиоэффектов, имитирующих звук турбин.

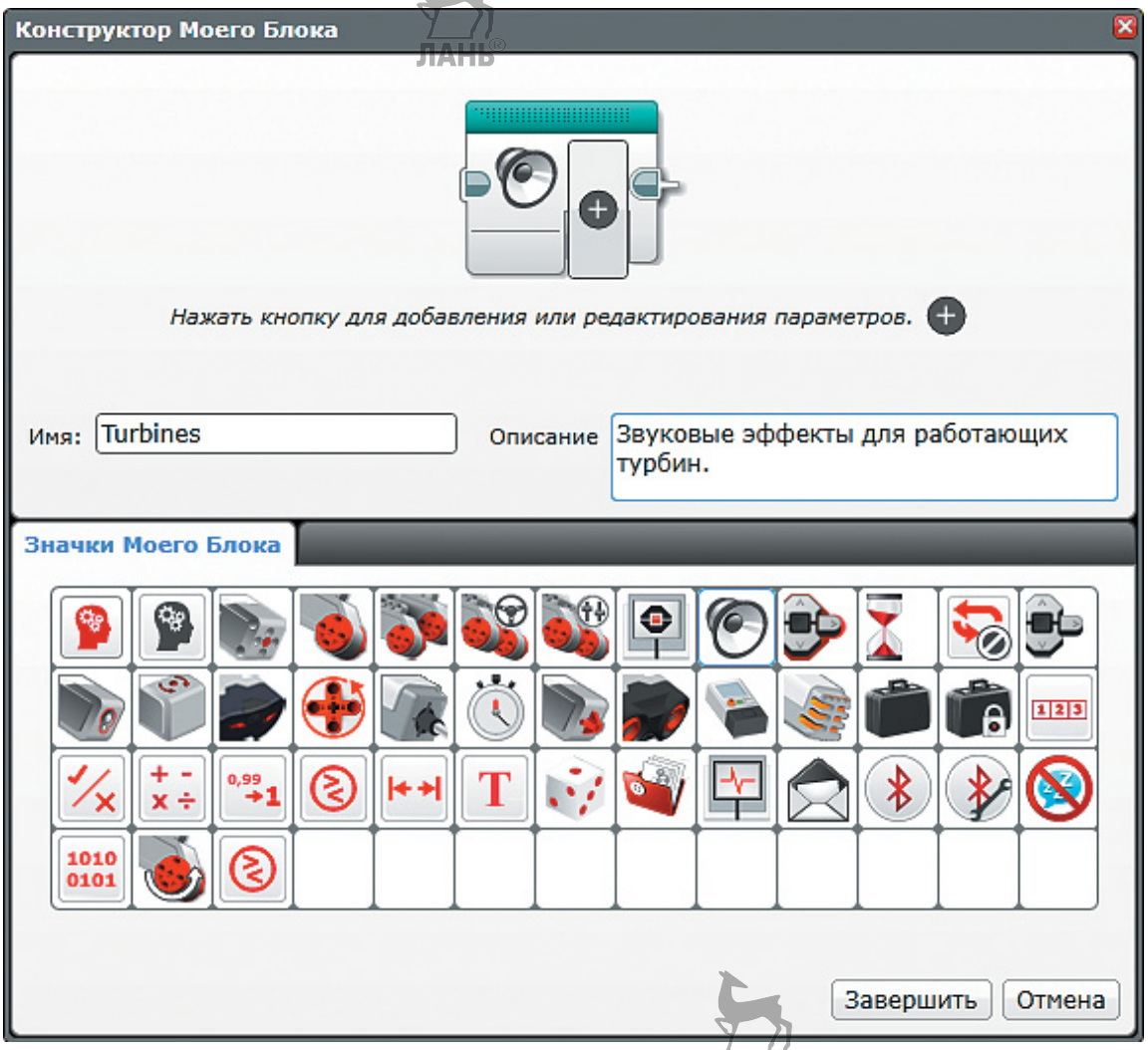
1. Добавь команду **Начало** (*зелёный блок*), как показано на рисунке.



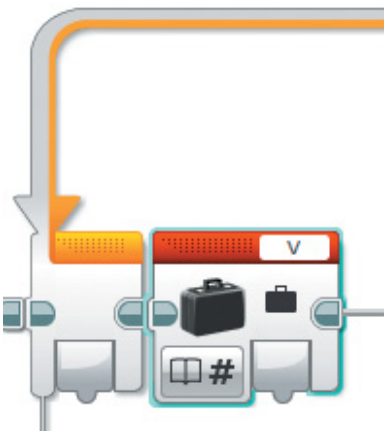
2. Далее добавь команду **Цикл** (*оранжевый блок*).



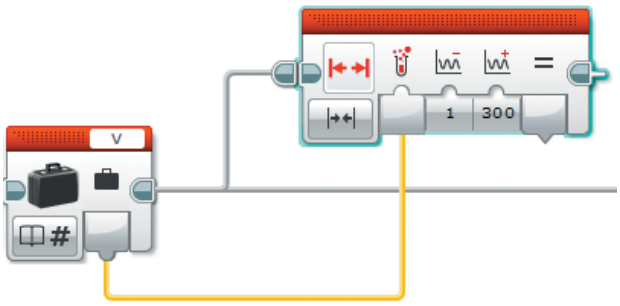
3. Теперь создай подпрограмму, которая будет отвечать за аудиоэффект турбин. Ты уже знаешь, как это делать. Назвать её можно, например, **Turbines** (от английского — турбины).



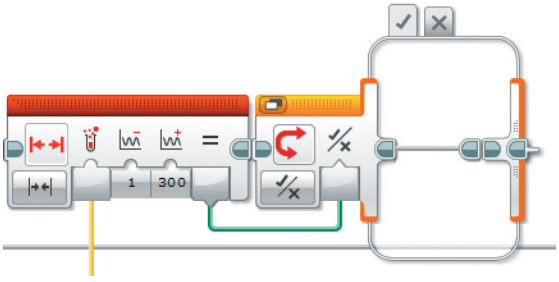
4. Кликни дважды по созданной команде **Turbines**. Учти, что звуковые эффекты должны соответствовать каждый своей скорости. Поэтому добавь внутрь цикла команду **Переменная** (*красный блок*), для обработки установи переменную **V** и выбери опцию **Считывание → Числовое значение**.



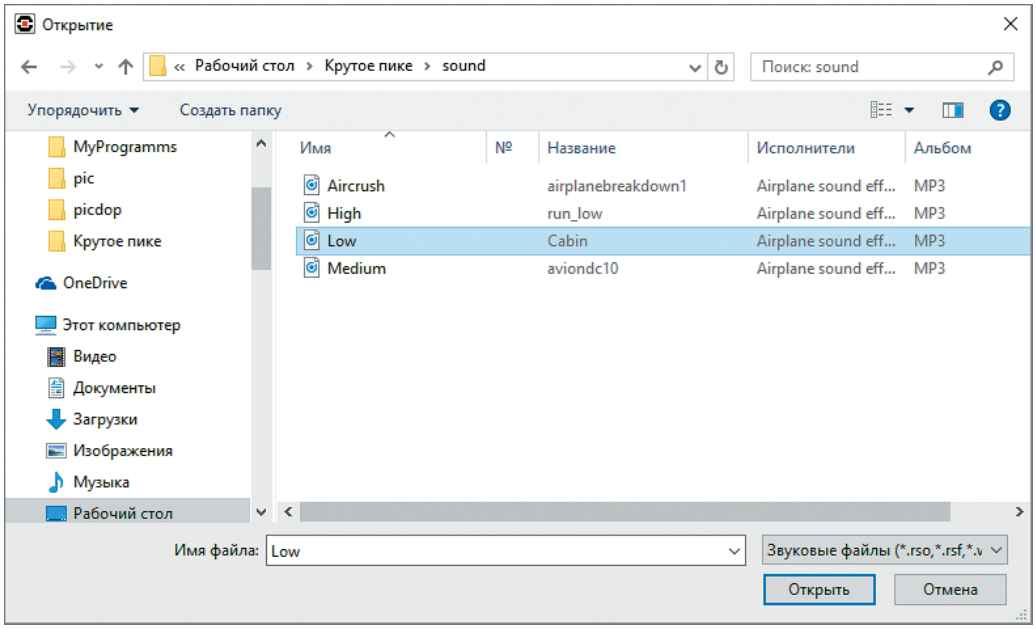
5. Теперь нужно создать похожую конструкцию, как ты делал для основной работы двигателей в подпрограмме **Engines**. Добавь команду **Интервал** (*красный блок*), соедини её с помощью шлейфа с командой **Переменная**, а выход переменной **V** соедини с входом **Тестовое значение**.
Во входы **Нижняя граница** и **Верхняя граница** с клавиатуры введи, соответственно, **1** и **300** — это будет диапазон скорости, при котором турбины будут издавать звук малой мощности.



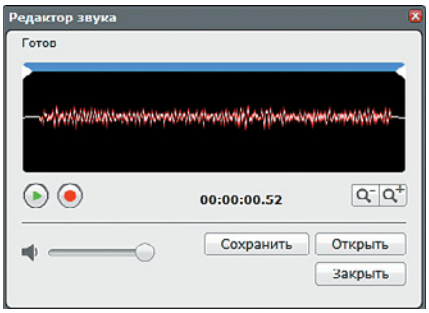
6. Добавь команду **Переключатель** (*оранжевый блок*), перейди к **Виду с вкладками**, выбери опцию **Логическое значение**, затем соедини выход «=» команды **Интервал** с входом **Переключателя**.



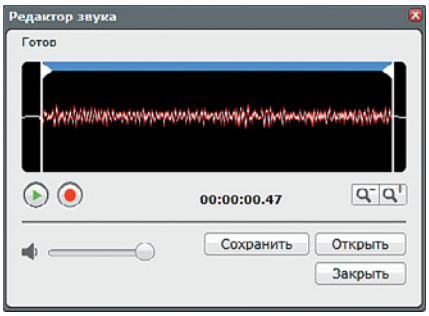
7. Внутри **Переключателя** понадобится всего лишь одна команда для воспроизведения звука. Однако сам звук ещё надо добавить в проект. Открой сверху меню **Инструменты → Редактор звука**. В открывшемся окне нажми кнопку **Открыть**, найди в памяти компьютера сохранённый звуковой файл **Low** и кликни по нему дважды.



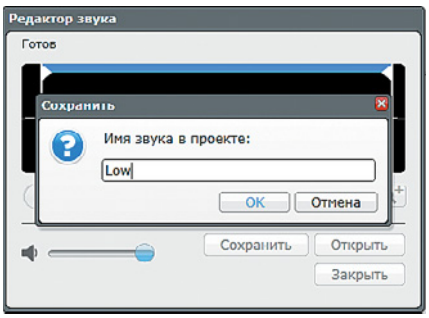
8. Звук откроется для предварительного редактирования.



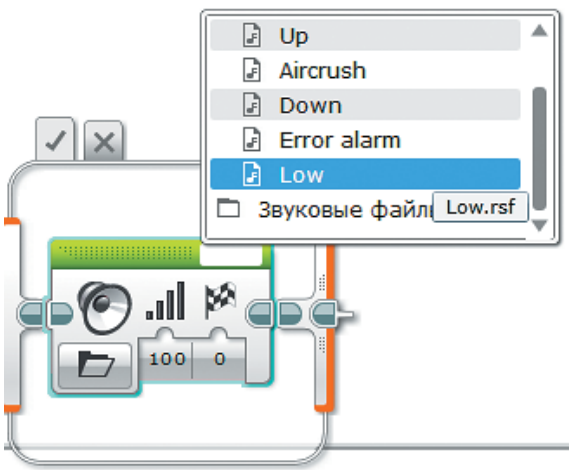
9. Ты можешь заметить, что у этого звука в начале и в конце есть тонкие линии — это тишина. Чтобы эффект звучал правильнее и непрерывно, тебе нужно **обрезать** тишину. Сделать это можно с помощью двух **бегунков сверху**: подвинь *левый бегунок вправо* до начала колебаний, а *правый влево*.



10. Прослушай этот аудиоэффект, нажав кнопку с зеленым треугольничком, а затем нажми кнопку **Сохранить**. В открывшемся окне задай имя, по которому данный звук будет идентифицироваться в твоём проекте, например, как и его оригинальное название, **Low**. И нажми **ОК**, затем кнопку **Закрыть**.



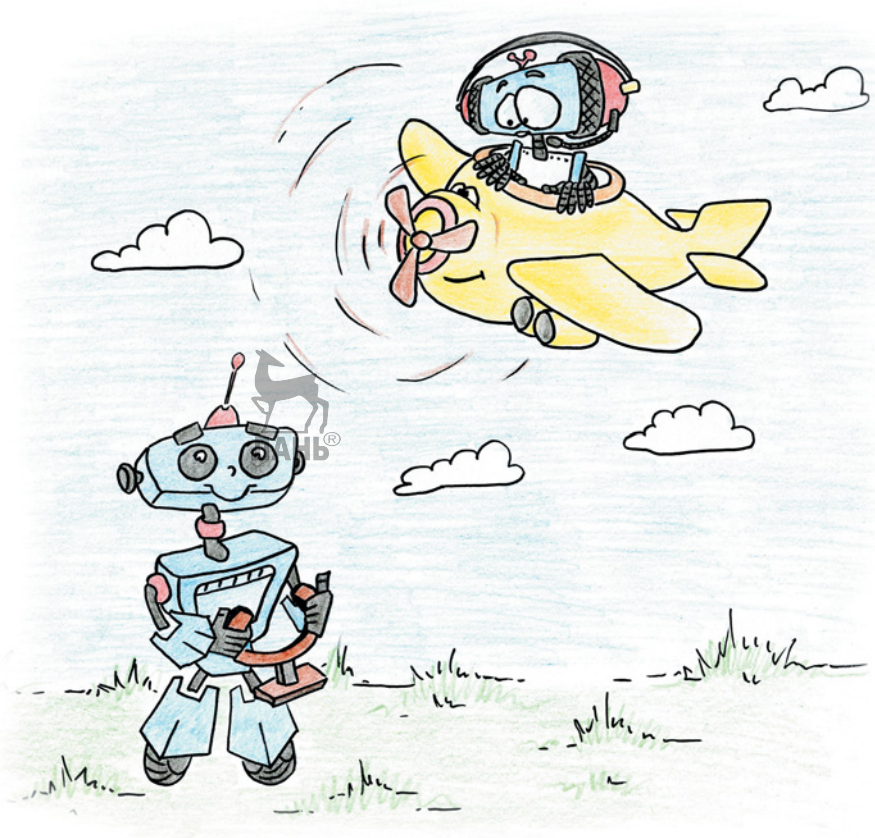
11. Добавь внутрь **Переключателя** команду **Звук** (*зелёный блок*), опции оставь без изменений. В правом верхнем окошке в папке **Звуки проекта** выбери файл **Low**.
Аналогично, копируя и изменяя команды **Интервал**, **Переключатель**, **Звук** и добавляя звуки с помощью **Редактора**, задай и остальные аудиоэффекты!



Как видишь, возможности и функции твоего авиасимулятора могут расширяться и дальше! Попробуй реализовать новые горизонты твоей инженерной мысли!

До новых встреч!

Ты создал робота своими руками и познакомился с работой пилота. Но впереди ещё так много интересного! В серии книг «РОБОФИШКИ» ты можешь найти другие замечательные проекты и стать настоящим изобретателем!

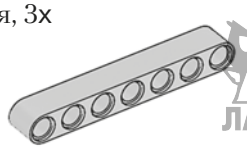


Содержание

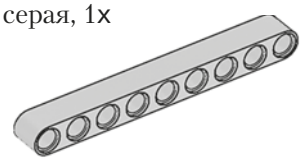
ЛАНЬ®

Балки прямые

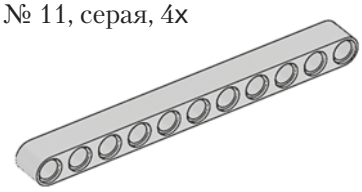
Балка № 7, серая, 3x



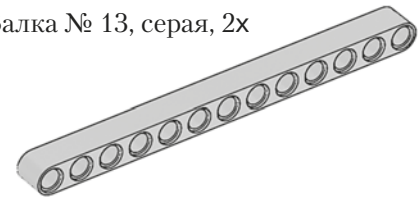
Балка № 9, серая, 1x



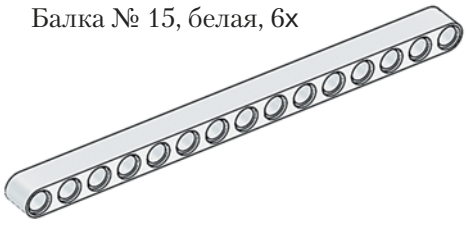
Балка № 11, серая, 4x



Балка № 13, серая, 2x

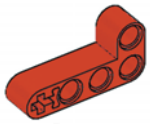


Балка № 15, белая, 6x

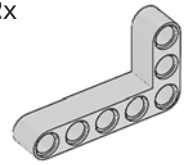


Балки угловые

Прямоугольная балка 2 × 4, красная, 5x



Прямоугольная балка 3 × 5, серая, 2x



Прямоугольная балка 3 × 5, белая, 4x

Поперечная балка 2 × 1, красная, 2x

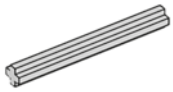
Поперечные блоки

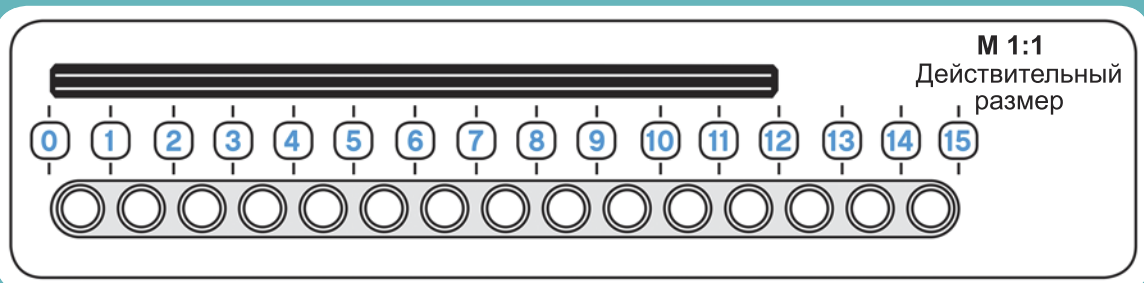
Поперечный блок, 2-модульный, серый, 2x

Оси

Ось № 3, серая, 5x

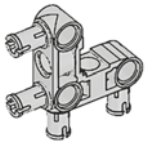
Ось № 5, серая, 2x



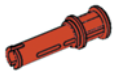


Штифты

Угловой соединительный штифт, 3 × 3-модульный, 4x



Соединительный штифт с втулкой, 3-модульный, красный, 2x



Соединительный штифт, 3-модульный, синий, 11x

Соединительный штифт, 2-модульный, синий, 6x

Соединительный штифт, 2-модульный, чёрный, 36x

Втулки

Втулка, жёлтая, 1x

Втулка, серая, 1x

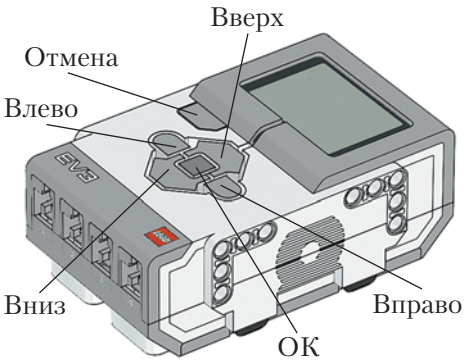
Втулка, 3-модульная, чёрная, 2x

Втулка, 3-модульная угловая, чёрная, 1x

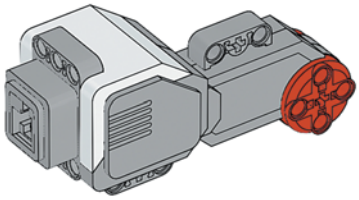
Фиксаторы

Фиксатор, 3-модульный, чёрный, 4x

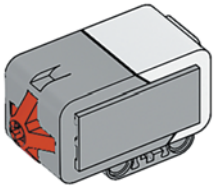
Программируемый модуль EV3, 1x



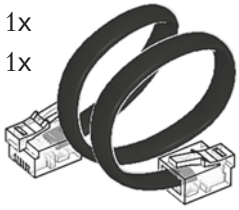
Большой мотор, 2x



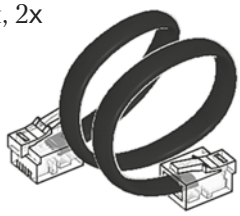
Датчик касания, 2x



Кабель, 25 см, 1x
Кабель, 50 см, 1x



Кабель, 35 см, 2x



USB-кабель, 1x

LEGO education

Минимальные системные требования определяются соответствующими требованиями программ Adobe Reader версии не ниже 11-й либо Adobe Digital Editions версии не ниже 4.5 для платформ Windows, Mac OS, Android и iOS; экран 10"

Электронное издание для досуга

Серия: «РОБОФИШКИ»

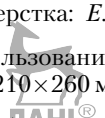
Рыжая Елена Ивановна
Удалов Виталий Владиславович
Тарапата Виктор Викторович

КОНСТРУИРУЕМ РОБОТОВ НА LEGO® MINDSTORMS® EDUCATION EV3. КРУТОЕ ПИКЕ

Для детей среднего и старшего школьного возраста

Ведущий редактор *Т. Г. Хохлова*
Руководители проекта от издательства *А. А. Елизаров, С. В. Гончаренко*
Научный консультант канд. пед. наук *Н. Н. Самылкина*
Ведущий методист *В. В. Тарапата*
Художники *В. Е. Шкерин, Я. В. Соловцова, И. Е. Марев, Ю. Н. Елисеев*
Корректор *М. Н. Угальская*
Компьютерная верстка: *Е. Г. Ивлева*

Подписано к использованию 05.04.21.
Формат 210×260 мм



Издательство «Лаборатория знаний»
125167, Москва, проезд Аэропорта, д. 3
Телефон: (499) 157-5272
e-mail: info@pilotLZ.ru, http://www.pilotLZ.ru

ЛОВИ НОВЫЕ «РОБОФИШКИ»

на **LEGO®** MINDSTORMS® Education EV3, Arduino® и ScratchDuino®:

ЛАНЬ®

- «Крутое пике»
- «Волшебная палочка»
- «Секрет ткацкого станка»
- «Тайный код Сэмюэла Морзе»
- «Посторонним вход воспрещён!»
- «В поисках сокровищ»
- «Умный замóк» и другие.

С серией **«РОБОФИШКИ»** самые удивительные и неожиданные идеи станут реальностью.

Создай своего робота, учись и играй вместе с ним!

Стань настоящим изобретателем!

ЛАНЬ®

info@pilotLZ.ru
www.pilotLZ.ru