

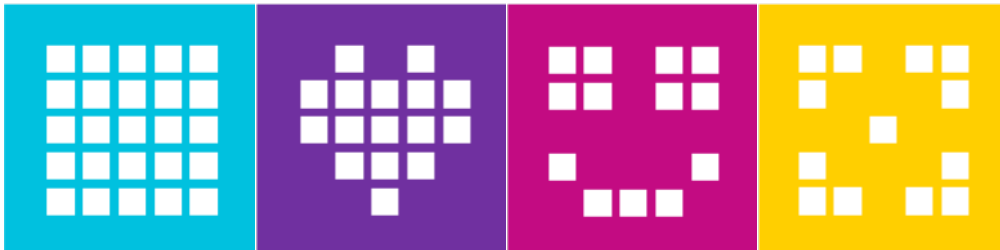
# УРОКИ ПО SPIKE PRIME

By the Makers of EV3Lessons



## РІД ДВИЖЕНИЕ ПО ЛИНИИ

BY SANJAY AND ARVIND SESHAN



# ЦЕЛИ УРОКА

- Узнаем ограничения пропорционального контроля.
- Узнаем, что означает PID.
- Узнаем, как программировать и настроить PID.

# КОГДА У ПРОПОРЦИОНАЛЬНОГО КОНТРОЛЯ ВОЗНИКАЮТ ТРУДНОСТИ?

## Что бы сделал человек?

На линии → еду прямо

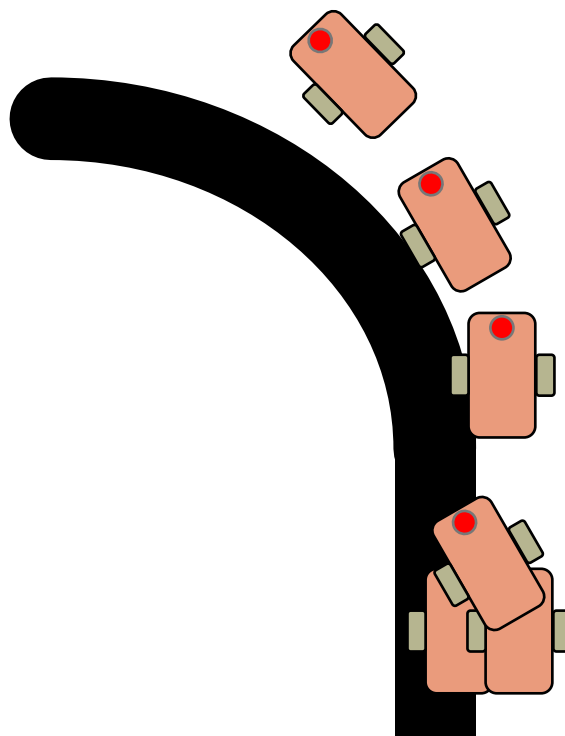
На белом → поворачиваю налево

Пересекли линию → поворачиваем направо

На белом → поворачиваю налево

Уехали далеко от линии → более резкий поворот!

Примечание: следующие несколько слайдов интерактивны. Используйте режим просмотра PowerPoint для этого.



## Что бы сделал пропорциональный контроль?

На линии → еду прямо

На белом → поворачиваю налево

**Пересекли линию → едем прямо!**

На белом → поворачиваю налево

**Уехали далеко от линии → поворот налево с тем же значением!**

ЗНАЧЕНИЕ ОСВЕЩЕННОСТИ= **500%**

# КАК МЫ МОЖЕМ ИСПРАВИТЬ ПРОПОРЦИОНАЛЬНЫЙ КОНТРОЛЬ?

Чтобы сделал человек

Поворот налево/на линии → поворачивают направо

Уехали далеко от линии → более резкий поворот!

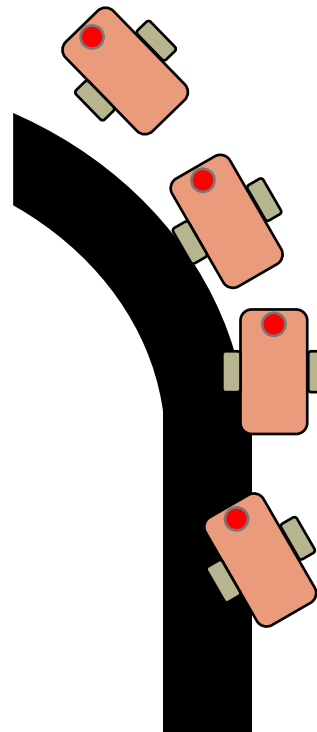
1. Предскажите, какое следующее значение считает датчик

Чтобы сделал пропорциональный контроль

Поворот налево / на линии → едем прямо!

Уехали далеко от линии → поворот налево с тем же значением!

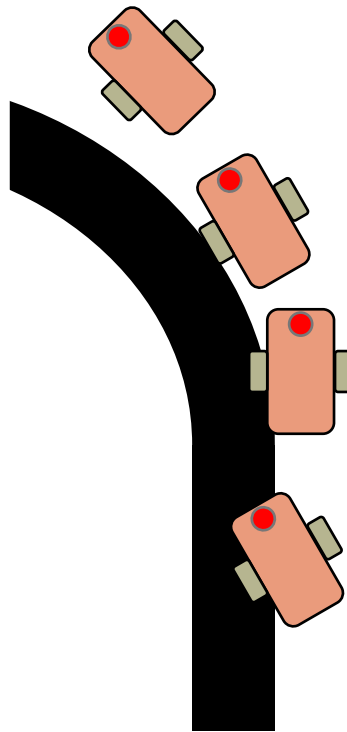
2. Это исправление поможет уменьшить ошибку



# ИНТЕГРАЛЫ И ПРОИЗВОДНЫЕ

## I. Предскажите, какое следующее значение считает датчик

- Считываем: 75, 65, 55 → какими будут следующие значения?
  - Может значения будут 57, 56, 55...
- Какую информацию вы используете для предположения?
- Производная → уровень, по которому изменяется скорость.



## 2. Это исправление поможет уменьшить ошибку?

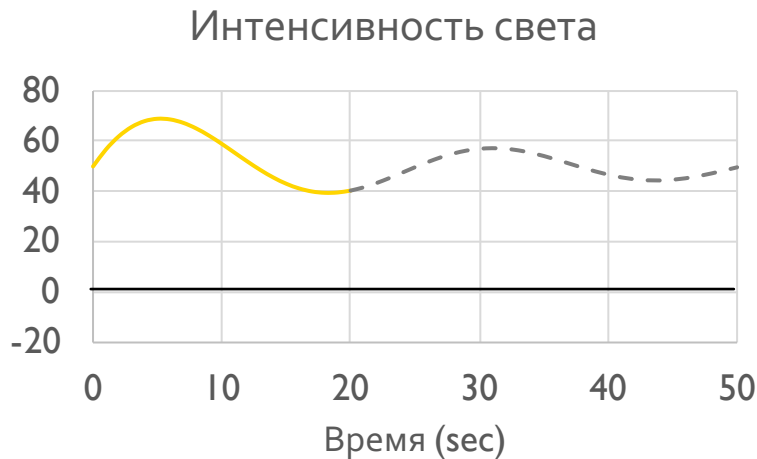
- Когда исправление работает правильно, какие ошибки мы считываем?
  - +5, -6, +4 -3... т.е. изменения около 0
- Когда исправление не работает, на что похожа ошибка?
  - +5, +5, +6, +5... т.е. на одной стороне от 0
- Как мы можем легко это обнаружить?
  - Подсказка: посмотрите на сумму всех прошлых ошибок.
- Что такое идеальное значение для этой суммы? Что означает, если сумма большая?
- Интеграл → «сумма» значений.

# ЧТО ТАКОЕ PID?

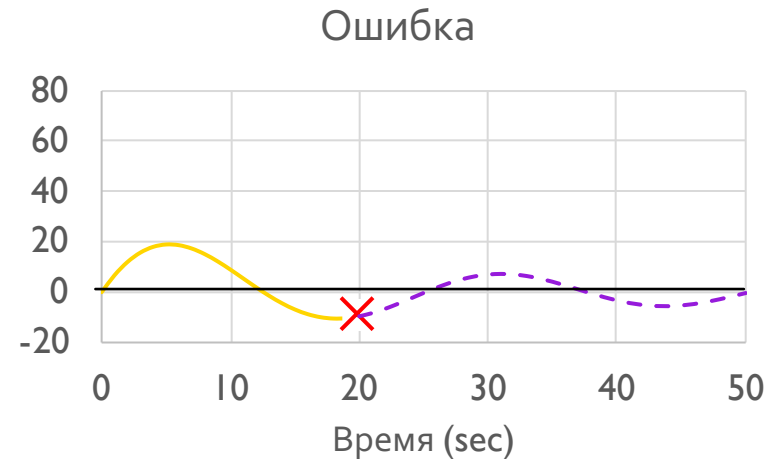
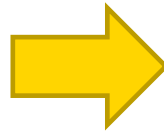
- Пропорциональный [Ошибка] → Насколько плохая ситуация теперь?
- Интеграл → Знает наше прошлое, фиксирует, помогли ли исправления?
- Производная → Как изменилась ситуация?
- PID контроль → объединение ошибки, интегралов и производных значений, для регулировки работа.

# ОШИБКА

- Сплошная линия представляет то, что произошло, пунктир – будущее
- Во время 20, Вы видите интенсивность света = 40 и ошибка = -10 (красный X)

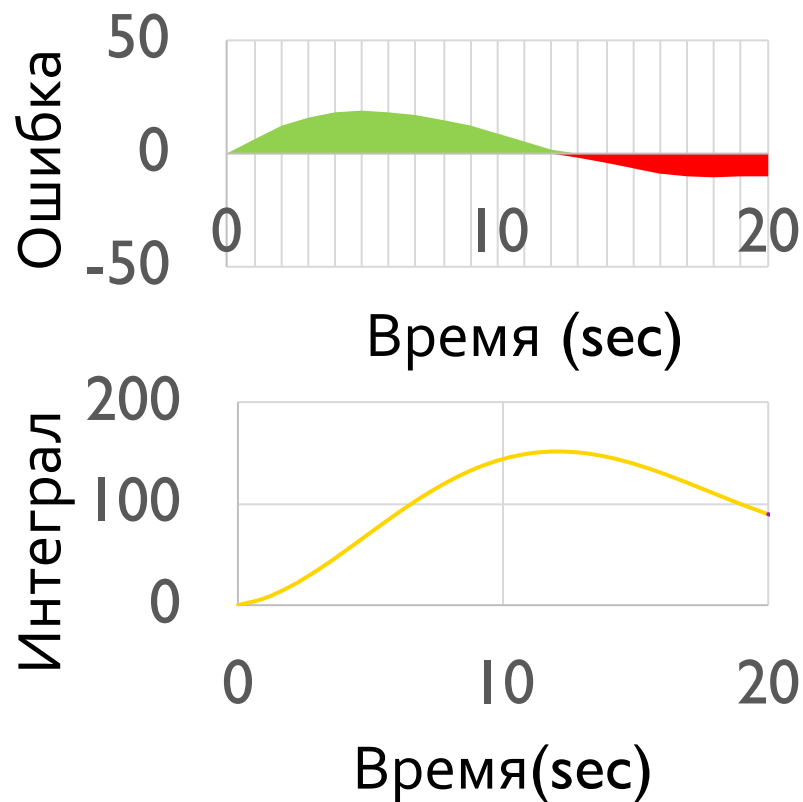


Вычитаем  
цель (50)



# ИНТЕГРАЛ

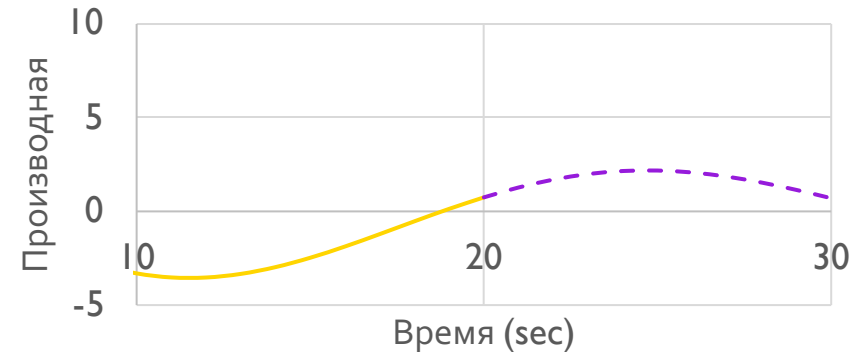
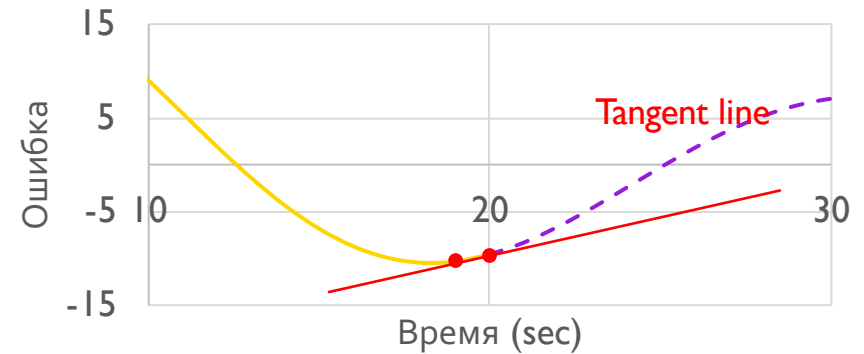
- Посмотрим на историю движения по линии.
- Суммируем прошлые ошибки.
- Какая область под кривой в графе (интеграл).
  - Зеленая = положительная область.
  - Красная = отрицательная область.





# ПРОИЗВОДНАЯ

- Как быстро изменяется положение?
  - Предсказываем, где робот будет находиться в ближайшем будущем.
  - То же самое, как быстро изменяется ошибка.
- Может быть измерено с помощью касательной линии к графику → производная
  - Аппроксимируем с использованием двух соседних точек на графике.



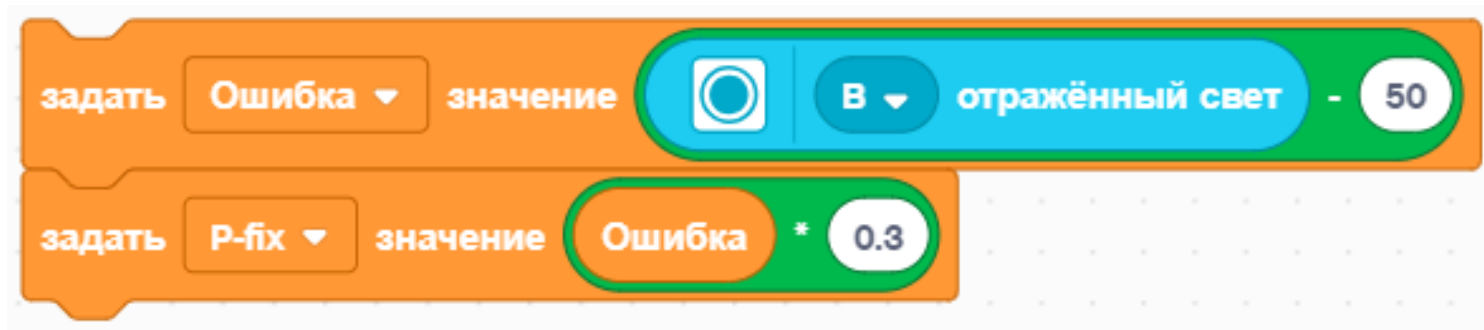
# ПСЕВДОКОД

1. Считываем новое значение датчика цвета.
2. Вычисляем «ошибку».
3. На основании величины ошибки определяем значение для исправления (пропорциональный контроль)
4. Используем ошибку для обновления интеграла (сумма всех прошлых ошибок).
5. Измеряем интеграл для определения значения для исправления (составной контроль).
6. Используем ошибку чтобы обновить производную (разность от последней ошибки).
7. Измеряем производную для определения значения для исправления (производный контроль)
8. Объединяем P, I, и D значения и управляем роботом.

# КОД - ПРОПОРЦИОНАЛЬНЫЙ

- Это совпадает с кодом пропорционального управления.

Ошибка = расстояние от линии = значение - цель

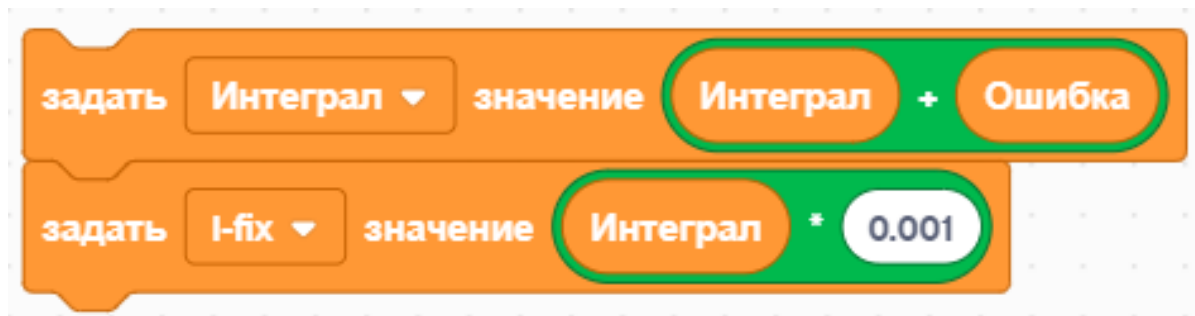


Исправление (P\_fix) = Ошибка измерения с пропорциональной константой ( $K_p$ ) = 0.3

# КОД - ИНТЕГРАЛЬНЫЙ

- Этот раздел вычисляет интеграл. Он добавляет текущую ошибку к переменной, у которой есть сумма всех предыдущих ошибок.
- Постоянная вычисления обычно маленькая, так как Интеграл может быть большим.

Интеграл = сумма всех прошлых ошибок = прошлый интеграл + следующий

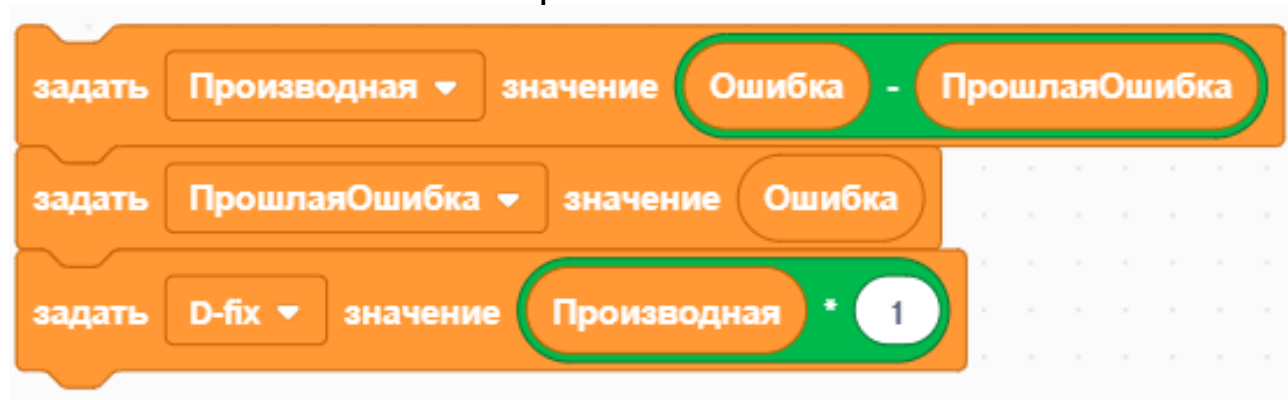


Исправление ( $I_{fix}$ ) = Интеграл вычисленный с пропорциональной константой ( $K_i$ ) = 0.001

# КОД - ПРОИЗВОДНЫЙ

- Этот раздел код вычисляет производную. Вычитает текущую ошибку из прошлой ошибки для определения изменения по ошибке.

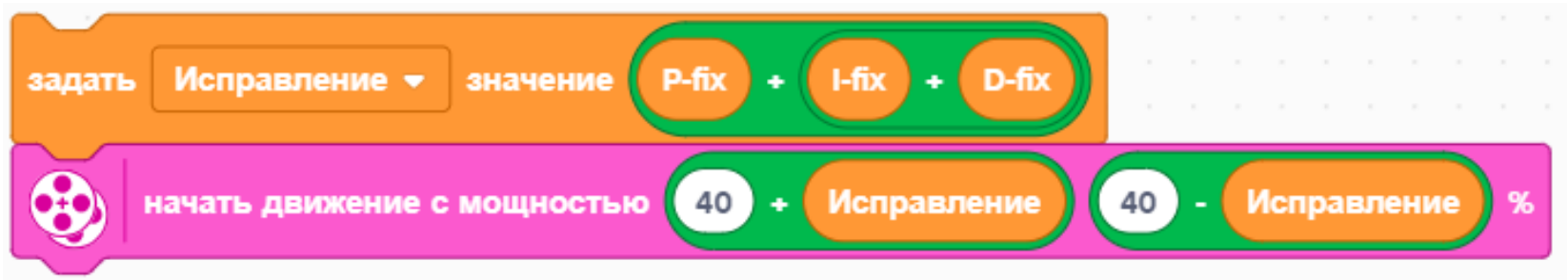
Производная = уровень изменения ошибки = текущая  
ошибка – прошлая ошибка



Исправление ( $D_{fix}$ ) = Производная с вычисленная с пропорциональной  
константой ( $K_d$ ) = 1.0

# ОБЪЕДИНЯЕМ ВСЕ ВЫЧИСЛЕНИЯ

- Каждый из компонентов был уже измерен. В этом пункте мы можем просто объединяем их вместе.
- Добавим все исправления для P, I и D. Это вычислит последнее исправление.
- В SPIKE Prime мы используем % мощности, т.к. моторы не регулируемые.



Применяем исправление для корректировки рулевого управления

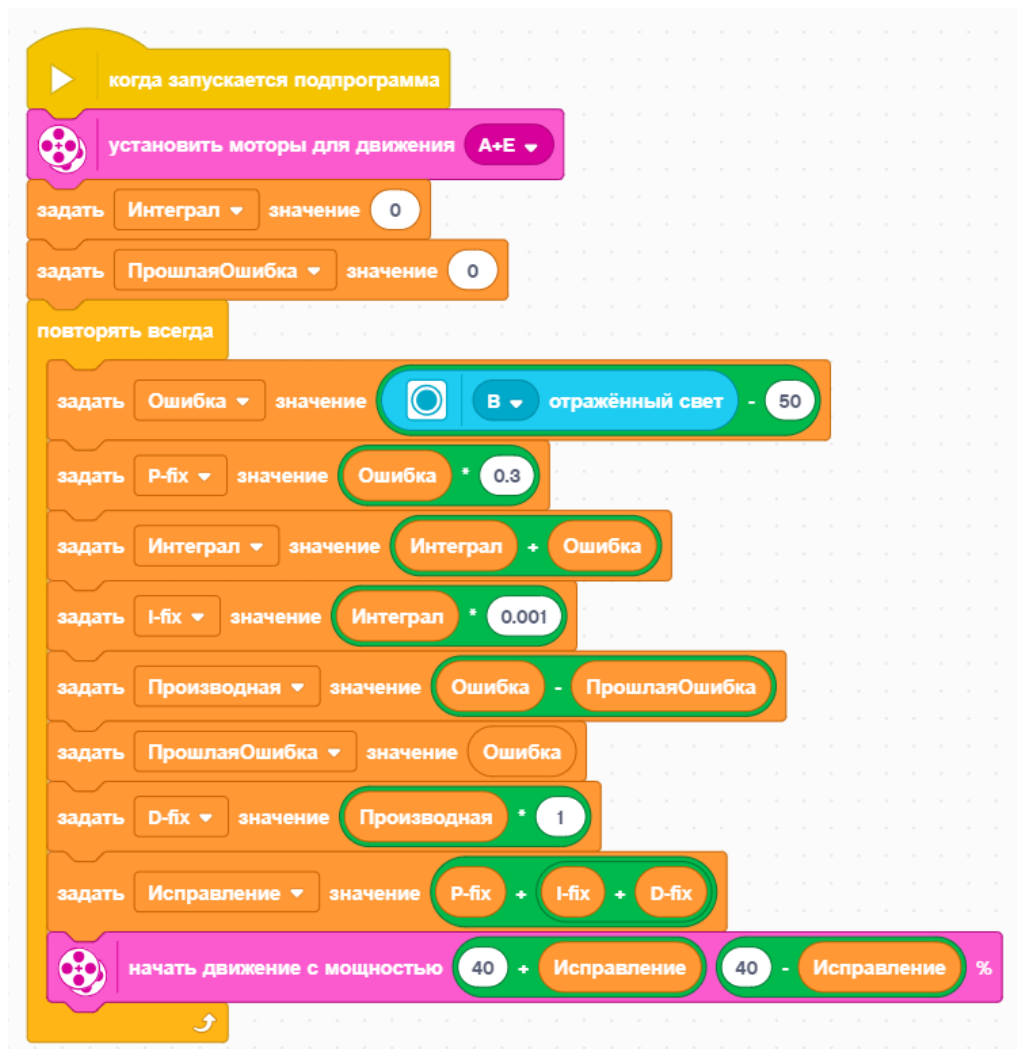
# ВСЕ КОД

- Это результат объединения всех частей.
- Мы надеемся, что Вы разобрались, как работает PID.



# ВСЕ КОД

Создайте переменные для последней ошибки и интеграла перед циклом и инициализируйте в 0. Кроме того, настройте моторы для движения.



```
когда запускается подпрограмма
  установить моторы для движения A+E
  задать Интеграл значение 0
  задать ПрошлаяОшибка значение 0
  повторять всегда
    задать Ошибка значение В отражённый свет - 50
    задать P-fix значение Ошибка * 0.3
    задать Интеграл значение Интеграл + Ошибка
    задать I-fix значение Интеграл * 0.001
    задать Производная значение Ошибка - ПрошлаяОшибка
    задать ПрошлаяОшибка значение Ошибка
    задать D-fix значение Производная * 1
    задать Исправление значение P-fix + I-fix + D-fix
  начать движение с мощностью 40 + Исправление 40 - Исправление %
```



# КЛЮЧЕВОЙ ШАГ: НАСТРОЙКА КОНСТАНТ ДЛЯ PID

- Наиболее распространенным способом вычислить Ваши константы для PID является метод проб и ошибок.
- Это может занять время. Вот некоторые подсказки:
  - Отключите всё кроме пропорциональной части (установите другие константы в ноль). Отрегулируйте только пропорциональную константу, пока робот не будет следовать по линии хорошо.
  - Затем используйте интеграл и скорректируйте, пока он не обеспечит хорошую работу в диапазоне.
  - Наконец, используйте производную и скорректируйте, пока Вы не будете удовлетворены движением по линии.
  - Настройте каждый сегмент, вот некоторые начальные значения для констант:
    - P: 1.0 изменяем на  $\pm 0.5$  от первоначального и  $\pm 0.1$  для точной настройки.
    - I: 0.05 изменяем на  $\pm 0.01$  от первоначального и  $\pm 0.005$  для точной настройки.
    - D: 1.0 изменяем на  $\pm 0.5$  от первоначального и  $\pm 0.1$  для точной настройки.

# ОЦЕНКА ДВИЖЕНИЯ ПО ЛИНИИ

## Пропорциональный

- Используем “P” в PID
- Делаем пропорциональные повороты.
- Хорошо работает и на прямой и на изогнутой линии.
- Хорошо для промежуточного звена продвинутым командам → необходимо знать математические блоки

## PID

- Это лучше, чем пропорциональный контроль при движении по очень кривой линии, поскольку робот адаптируется к её кривизне.
- Однако для FIRST LEGO League, в которой главным образом только прямые линии, пропорциональный контроль может быть достаточным.

# CREDITS

- This lesson was created by Sanjay Seshan and Arvind Seshan for SPIKE Prime Lessons
- More lessons are available at [www.primelessons.org](http://www.primelessons.org)



This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).