

Работа победителя заключительного этапа
командной инженерной олимпиады школьников
Олимпиада Национальной технологической инициативы

Профиль «Интеллектуальные робототехнические системы»

Курейкин Максим Сергеевич

Класс: 10

Город: Киров

Школа: КОГОАУ КФМЛ

Регион: Кировская область

Уникальный номер участника: 743

Команда на заключительном этапе: YoS

Параллель: 10-11

Результаты заключительного этапа:

Индивидуальная часть												Командная часть					Результат	
№	Математика			Информатика							Итого	Макс. балл	Решение задач этапов					Итого
	1	2	3	1.1	1.2	1.3	2.1	2.2	3.1	3.2			№1	№2	№3	№4		
743	8	2	0	0	0	0	0	0	0	0	10	200	62	12	31	18	123	133

Индивидуальная часть

Персональный лист участника с номером 743:



Олимпиада НТИ

ФИО Курейкин Максим Сергеевич

Город Курск

Школа № КГМЛ

Математика

Лист 1:

Командная инженерная олимпиада «Олимпиада НТИ»

Направление ИРС

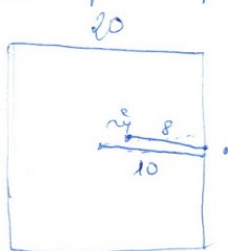
Предмет Математика

Номер участника 293

1	2	3	4	5	6	Σ
5	2	0	0	0	0	

№2

а) Действительно, робот не сможет добраться до границы не-за-
мечено при скорости меньшей 8 м/мин.



$$\text{Длина пути в данном случае} = \frac{20}{2} = 10$$

Рассмотрим старт (вздр) робота и самого робота, как точку. Чтобы робот успел пройти максимальное рас-
стояние за то время, пока его не обнаружит лазер, нужно, чтобы лазер был в крайнем правом положении.

За один оборот = 1 мин, робот успеет пройти $v \cdot t = 8 \cdot 1 = 8$ м, после чего он будет обнаружен лазером.

б) Если по условию v робота строго меньше 10 м/мин, то за 1 мин, как и в предыдущем пункте он не успеет добраться до центра, даже при условии $v = 10$ м/мин, робот имеет размер, которого рассматриваем его как точку. Следовательно, чтобы остальное перемещение лазер ролеком сдвинуть на минимальный угол перед стартом, а это значит, что роботу не хватает той секунды, чтобы достичь центра.

Робот может убежать от лазер.

№1.

Наименьшее кол-во прибавлений = 8.

	Сами	Устами
Корре	200	0
исход	0	200
I	100	100
II	150	50
III	75	125
IV	137.5	62.5
V	88.75	111.25

	Корре	исход
VI	134.37	65.63
VII	67.1	132.9
VIII	135	65

В погрешности может получиться не правильный ответ.
 $W_{\text{корре}} - W_{\text{исход}} = 100 \cdot 100\%$

± (50)

Командная инженерная олимпиада «Олимпиада НТИ»

Направление ЦРСПредмет МатематикаНомер участника 843

Дальнейшее перемещение имеет разницу между концентрациями кофе в кружках, но т.к. они постоянно перемешиваются и смесь становится более однородной, то после n -ого перемешивания количество кофе и молока в кружке будет меньше на $\frac{\text{кофе}}{2^n}$ либо $\frac{\text{молоко}}{2^n}$.

 x - кофе в кружке $(100-x)$ - молоко в кружке $(200-x)$ - кофе в кружке x - молоко в кружке

$$\frac{200-x}{200} = \frac{x}{100}$$

$$200x = 2000 - 100x$$

$$300x = 2000$$

$$x = 6\frac{2}{3} \text{ кофе в кружке}$$

$$W_{\text{кофе в кружке}} = 6\frac{2}{3}\%$$

$$W_{\text{кофе в кружке}} = 6\frac{2}{3}\%$$

$$\Delta W < 1\%$$

Поэтому равенство???

Оценка за задачу №1: 8

Комментарий к решению: Ответ дан верно, но есть ошибка в вычислениях.
3 балла было добавлено в ходе рассмотрения апелляции.

Оценка за задачу №2: 2

Комментарий к решению: Задача решена не полностью.

Оценка за задачу №3: 0

Комментарий к решению: Решение не предложено.

Информатика

Задача №1

Решение не было предложено (0 баллов).

Задача №2

Решение не было предложено (0 баллов).

Задача №3

Решение не было предложено (0 баллов).

Командная часть

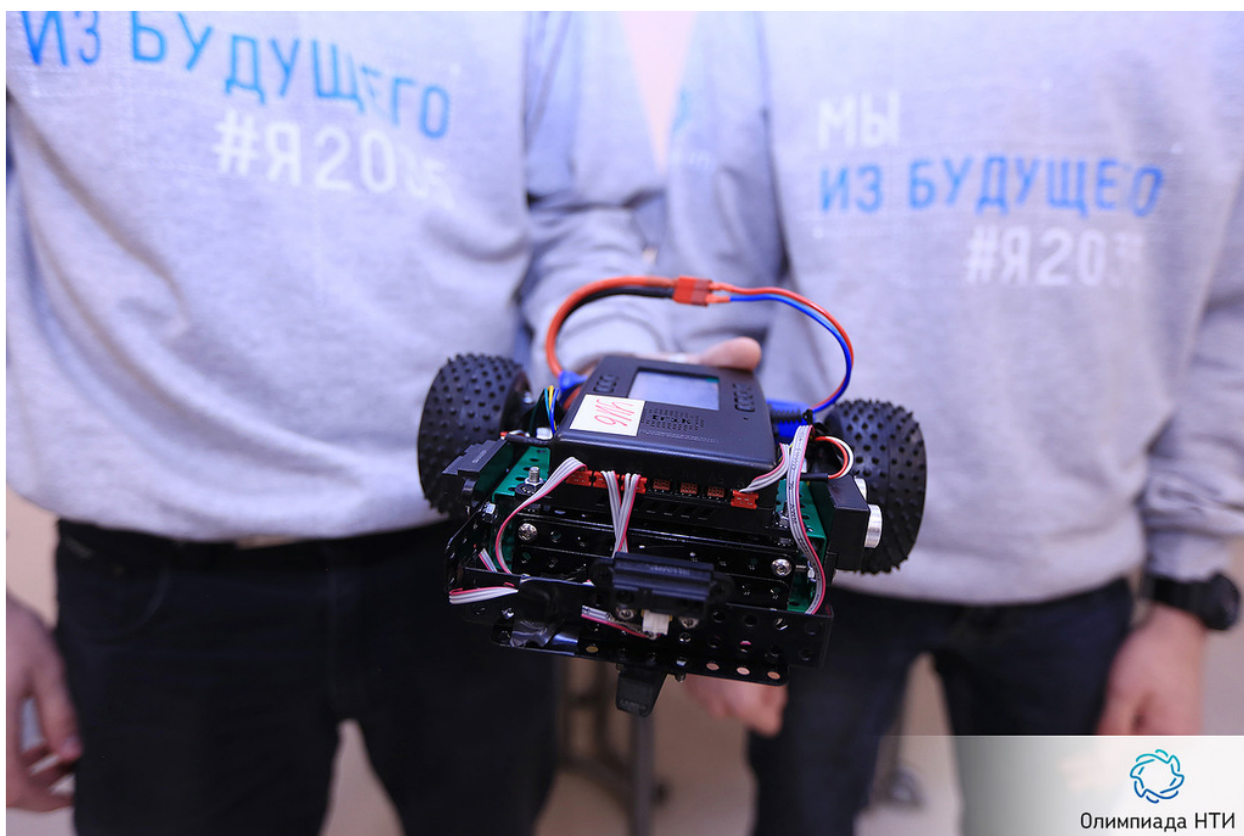
Результаты были получены в рамках выступления команды: YoS

Личный состав команды:

- Курейкин Максим Сергеевич
- Романов Вячеслав Сергеевич

Фотография команды и робототехнического устройства, оснащенного датчиками, перед проверочными испытаниями второго этапа:





Протокол выполнения заданий:

Первый этап:

№ задания	Критерий	Возможные баллы	Полученные баллы
1	Робот покинул сектор старта	6x2	12
2	Робот проехал половину от всего количества секторов от сектора старта до сектора финиша	13x2	26
3	Робот проехал от сектора старта до черной линии (в ходе движения робот пересек черную линию или остановился таким образом, что черная линия пересекает проекцию робота на поле)	6x2	12
4	Робот проехал от сектора старта, заехал в сектор финиша (проекция робота на поле внутри сектора) и остановился	6x2	12
5	Робот проехал от сектора старта, остановился в секции финиша (проекция робота внутри сектора) и вывел на экран верное количество пройденных секций	12x2	0
Дополнительные баллы		19	0
Всего за этап		105	62

Второй этап:

№ задания	Критерий	Возможные баллы	Полученные баллы
6	Робот проехал по периметру вокруг циклической структуры и остановился после определения цикла (допускается проезд по второму кругу до 3х секторов)	7х2	0
7	Робот после верного определения циклической структуры, остановился на 3 секунды, и продолжил движение по другим секторам полигона, не принадлежащим циклической структуре (движение не менее, чем по 5 не принадлежащим циклической структуре)	12х2	0
8	Робот проехал не меньше 7 секторов по полигону и остановился; на экране отображены координаты смещения относительно сектора старта: первое число – смещение по оси X, второе число – смещение по оси Y. При этом ось X совпадает с направлением робота в секторе старта, а ось Y направлена вправо. Данный критерий не будет оцениваться, если робот способен выполнять сразу следующий пункт списка критериев.	6х2	12
9	После остановки и отображения карты на экране устройства отображается 2 числа, определяющее позицию, где произошла остановка: первое число – сектор по оси X, второе число – сектор по оси Y (см. рис. 5)	31х2	0
Дополнительные баллы		19	0
<i>Всего за этап</i>		119	12

Третий этап:

№ задания	Критерий	Возможные баллы	Полученные баллы
10	Робот доехал до сектора сервисного обслуживания (проекция робота внутри сектора) и остановился	6х2	12
11	После остановки робота на экран выведено закодированное значение.	19х2	19
Дополнительные баллы		12	0
<i>Всего за этап</i>		62	31

Четвертый этап:

№ задания	Критерий	Возможные баллы	Полученные баллы
-----------	----------	-----------------	------------------

12	После запуска программы, роботу стоит на месте в течение на 10 секунд. На экран выведен путь от сектора старта до сектора финиша. При этом путь выведен в виде координат секций, через которые должен пройти робот. Данный критерий не будет оцениваться, если робот способен выполнять сразу следующий пункт списка критериев.	6x2	0
13	После запуска программы, роботу стоит на месте в течение на 10 секунд. На экран выведен путь от сектора старта до сектора финиша. При этом путь выведен в виде алгоритма перемещения (F - одна секция прямо, R- поворот направо, L - поворот налево) с учетом необходимых поворотов в секторе старта.	9x2	9
14	Путь, выведенный после запуска робота, является оптимальным по количеству секторов, необходимых для посещения.	3x2	3
15	Путь перемещения робота из сектора старта до сектора финиша является оптимальным по количеству секторов.	3x2	0
16	Робот доехал до финального сектора и остановился. И издал звуковой сигнал.	6x2	0
Дополнительные баллы		12	0
17	Робот выехал из сектора старта и остановился на 10 секунд в первом (по ходу движения робота) секторе сервисного обслуживания. Перед началом отсчета таймера робот подает звуковой сигнал.	6	6
18	После остановки робота в секторе сервисного обслуживания выведена одна компонента координаты финального сектора вместе с идентификатором оси (буква X или Y). Выведенное число совпадает со штрих-кодом нанесенным в данном секторе сервисного обслуживания.	5	0
19	Робот продолжил движение и остановился на 10 секунд во втором секторе сервисного обслуживания. Перед началом отсчета таймера робот подает звуковой сигнал.	11	0
20	После остановки робота в секторе сервисного обслуживания выведены обе компоненты координаты финального сектора вместе с идентификаторами осей (буква X или Y). Координата совпадает со штрих-кодом нанесенным в данном секторе сервисного обслуживания.	5	0
21	Путь перемещения робота из сектора, где робот локализовался до очередного сектора сервисного обслуживания или финального сектора является оптимальным по количеству секторов.	5	0
22	Робот доехал до финального сектора и остановился. И издал звуковой сигнал.	28	0
Всего за этап		114	18

Программа команды для решения задачи первого этапа:

```
var __interpretation_started_timestamp__;
var pi = 3.141592653589793;
var eLeft, eRight;
var mLeft, mRight;
var eEncL, eEncR;
var forward, right;
var kp = 3.5,
    err = 0,
    u = 0,
    m1 = 0,
    m2 = 0;
var cmtodeg = (pi * 37) / 3600;

var nakop = 0;

var rotate = function(deg) {
    var limit = 0;

    if ((deg <= 0) && (deg >= -180)) {
        var eR = eRight.readRawData();
        limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
        var target = Math.abs(eR) + limit;
        while (Math.abs(eR) < target) {
            mRight(25);
            mLeft(-25);
            eR = eRight.readRawData();
            script.wait(1);
        }
        mRight(-10);
        mLeft(10);
        mRight(0);
        mLeft(0);
    }
    if ((deg >= 0) && (deg <= 180)) {
        var eL = eLeft.readRawData();
        limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
        var target = Math.abs(eL) + limit;
        while (Math.abs(eL) < target) {
            mRight(-25);
            mLeft(25);
            eL = eLeft.readRawData();
        }
    }
}
```

```

        script.wait(1);
    }
    mRight(10);
    mLeft(-10);
    mRight(0);
    mLeft(0);
}
}

var main = function() {
    __interpretation_started_timestamp__ = Date.now();

    mLeft = brick.motor("M4").setPower;
    mRight = brick.motor("M3").setPower;

    eLeft = brick.encoder("E4");
    eRight = brick.encoder("E3");

    //sensor Light here
    while (brick.sensor("A6").read() < 70) {
        forward = brick.sensor("A2").read();
        right = brick.sensor("A1").read();
        if (right > 80) {
            right = 11;
        }

        if ((forward < 13) && (right > 15)) {
            nakop = nakop + Math.abs(eLeft.read());
            rotate(45);
            eLeft.reset();
        } else
        if ((forward < 13) && (right < 15)) {
            nakop = nakop + Math.abs(eLeft.read());
            rotate(-45);
            eLeft.reset();
        }

        err = 10 - right;
        u = err * kp;

        m1 = 23 - u;
        m2 = 23 + u;
    }
}

```

```

    if (m1 < 0) {
        m1 = -3;
    }
    if (m2 < 0) {
        m2 = -3;
    }

    if (m1 > 40) {
        m1 = 40;
    }
    if (m2 > 40) {
        m2 = 40;
    }

    mLeft(m1);
    mRight(m2);

    var data = eLeft.read();
    if (Math.abs(data) > 370) {
        eLeft.reset();
        mLeft(-5);
        mRight(-5);
        script.wait(100);
        mLeft(0);
        mRight(0);
    }

    script.wait(1);

}

//v zonu
eLeft.reset();
var range = Math.abs(eLeft.read());
while (range < 250) {
    range = Math.abs(eLeft.read());
    mLeft(40);
    mRight(40);
    script.wait(1);
}
mLeft(-10);
mRight(-10);

```

```

script.wait(100);

if (nakop < 200) nakop = nakop + 200;
print(nakop);
nakop = nakop / 200;
print(Math.floor(nakop + 2))

brick.display().addLabel(Math.floor(nakop + 2), 20, 20);
brick.display().redraw();
mLeft(0);
mRight(0);

script.wait(5000);
return;
}

```

Программа команды для решения задачи “определение циклической структуры” второго этапа:

```

var pi = 3.1415926535897931;
var d = 8.25; //8.7; //diameter of wheel, cm
var l = 17.5; // base of robot
var degInRad = 180 / pi;

var alpha = 0;
var gyro;
gyro = brick.gyroscope();
gyro.calibrate(10000);
print("initialized gyro");
script.wait(11000);
var fullTurn = 4 * 87; // from experimets. Not 360 degrees, but ....

//кнопки
var stopKey = KeyEnum.Esc;
var startKey = KeyEnum.Left;

//диод
var led = brick.led();

//моторы
var mLeft = brick.motor(M3).setPower;
var mRight = brick.motor(M4).setPower;

```



```

//сенсоры ИК
sA1 = brick.sensor(A1).read;
sA2 = brick.sensor(A2).read;

//энкодеры
var eLeft = brick.encoder(E3);
var eRight = brick.encoder(E4);
var cpr = 374; //560; //374; // количество сигналов на оборот

//длина клетки
var cellLength = 40 * cpr / (pi * d);

//переопределение метода нажатия клавиши
if (0) {
    var key = 0;

    function wasPressed(_key) {
        key = 0;
        if (brick.keys().wasPressed(_key)) {
            key = _key;
            return 1;
        } else {
            return 0;
        }
    }
}

var direction = 0; // absolute angle of direction movement
var directionOld = 0;

print("-----");

led.red();

eLeft.reset();
eRight.reset();

var el = eLeft.readRawData();
var er = eRight.readRawData();
mLeft(0);
mRight(0);
print("Start");

```

//инициализация и калибровка гироскопа

```
var v = 20; // velocity
```

```
var ex = 0;
```

```
var ey = 0;
```

```
var encLeftOld = 0;
```

```
var encRightOld = 0;
```

```
var encLeft = 0;
```

```
var encRight = 0;
```

```
var n = 0;
```

// вычисление абсолютного угла относительно начального положения

```
function angle() {
```

```
    var sgn = 0;
```

```
    var _direction = readYaw(); // mgrad
```

```
    var dtDirection = _direction - directionOld;
```

```
    sgn = directionOld == 0 ? 0 : directionOld /
```

```
Math.abs(directionOld);
```

```
    n += sgn * Math.floor(Math.abs(dtDirection / 320000));
```

```
    direction = _direction + n * 360000;
```

```
    directionOld = _direction;
```

```
}
```

```
var t = 0;
```

```
print("Run, robot, run!");
```

// делаем прерывание основной программы с частотой 200Гц, чтобы посчитать абсолютный угол с гироскопа

```
//var mtimer = script.timer(50);
```

```
//mtimer.timeout.connect(angle);
```

// вывод в консоль показаний гироскопа

```
function printGyro() {
```

```
    print("direction=" + direction + " sA1" + sA1() + " sA2" + sA2());
```

```
}
```

```
//var ptimer = script.timer(300);
```

```
//ptimer.timeout.connect(printGyro);
```

```
//поворот на угол по гироскопу
```

```
function turnDirection(_angle, _v) {  
  _angle = _angle * 1000; //toMGrad  
  var _vel = _v == undefined ? 10 : _v;  
  var angleOfRotate = _angle - direction;  
  print("ar=" + angleOfRotate);  
  var sgn = angleOfRotate == 0 ? 0 : angleOfRotate / Math.abs(  
    angleOfRotate);  
  mLeft(_vel * sgn);  
  mRight(-_vel * sgn);  
  while (Math.abs(angleOfRotate = _angle - direction) > 20000) {  
    script.wait(50);  
  }  
  brick.motor(M3).forceBreak(500);  
  brick.motor(M4).forceBreak(500);  
  script.wait(500);  
}
```

```
// проезд в перед на количество ячеек _kcell со скоростью _v выравниваясь  
по гироскопу на угол _alpha
```

```
function forward(_alpha, _v, _kcell) {  
  var _vel = _v == undefined ? 10 : _v;  
  var u = 0;  
  var e1 = Math.abs(eLeft.readRawData());  
  while (Math.abs(eLeft.readRawData()) < e1 + (_kcell * cellLength))  
  {  
    u = -0.7 * (_alpha - direction / 1000);  
    mLeft(_vel - u);  
    mRight(_vel + u);  
    script.wait(5);  
  }  
  brick.motor(M3).forceBreak();  
  brick.motor(M4).forceBreak();  
  script.wait(500);  
}
```

```
// проезд назад, чтобы упереться в стенку с целью выравнивания
```

```
function backward() {  
  mLeft(-50);  
  mRight(-50);  
}
```

```

var eLOld = eLeft.readRawData();
var eROld = eRight.readRawData();
var el = eLOld;
var er = eROld;
do {
    eLOld = el;
    eROld = er;
    script.wait(1000);
    el = eLeft.readRawData();
    er = eRight.readRawData();
}
while (Math.abs(eLOld - el) > 200 && Math.abs(eROld - er) > 200)
brick.motor(M3).forceBreak();
brick.motor(M4).forceBreak();
script.wait(300);

}

function get_odo() {
    tilt = (gyro.read()[6] / 1000) * pi / 180;
    var eL = brick.encoder("E3").read();
    var eR = brick.encoder("E4").read();

    block = ((eL - eoldL) * cmtodeg + (eR - eoldR) * cmtodeg);

    x = xold + (block / 2) * (Math.cos(told + (tilt - told) / 2));
    y = yold + (block / 2) * (Math.sin(told + (tilt - told) / 2));

    eoldL = eL;
    eoldR = eR;

    xold = x;
    yold = y;
    told = tilt;

    print("x= " + x + " y= " + y);
}

var dir = 0;
var isOnFinish = false;
var currentX = 0,
    currentY = 0;
var fwd = 0,

```

```

    right = 0;
    var cmtodeg = (pi * 37) / 3600;

    var forwardGyro = function(angle, speed, kcell) {
        if (angle == 180) {
            eLeft.reset();
            var e1 = Math.abs(eLeft.readRawData());
            while (Math.abs(eLeft.readRawData()) < e1 + (kcell * cellLength))
            {
                tilt = (gyro.read()[6] / 1000);
                var k = 0.7
                if (tilt > -180 && tilt < 0)
                    k = -0.7
                else
                    k = 0.7;
                var u = k * (180 - Math.abs(tilt))
                mLeft(30 - u);
                mRight(30 + u);
                script.wait(1);
            }
            mLeft(-10);
            mRight(-10);
            script.wait(250);
            mLeft(0);
            mRight(0);
        } else {
            eLeft.reset();
            var e1 = Math.abs(eLeft.readRawData());
            while (Math.abs(eLeft.readRawData()) < e1 + (kcell * cellLength))
            {
                tilt = (gyro.read()[6] / 1000);
                var u = 0.7 * (angle - tilt)
                mLeft(30 - u);
                mRight(30 + u);
                script.wait(1);
            }
            mLeft(-10);
            mRight(-10);
            script.wait(250);
            mLeft(0);
            mRight(0);
        }
    }

```

```

    }
}

var rotate = function(deg) {
    var limit = 0;

    if ((deg <= 0) && (deg >= -180)) {
        var eR = eRight.readRawData();
        limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
        var target = Math.abs(eR) + limit;
        while (Math.abs(eR) < target) {
            mRight(25);
            mLeft(-25);
            eR = eRight.readRawData();
            script.wait(1);
        }
        mRight(-10);
        mLeft(10);
        script.wait(200);
        mRight(0);
        mLeft(0);
    }
    if ((deg >= 0) && (deg <= 180)) {
        var eL = eLeft.readRawData();
        limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
        var target = Math.abs(eL) + limit;
        while (Math.abs(eL) < target) {
            mRight(-25);
            mLeft(25);
            eL = eLeft.readRawData();
            script.wait(1);
        }
        mRight(10);
        mLeft(-10);
        script.wait(200);
        mRight(0);
        mLeft(0);
    }
}

var main = function() {

```

```

mLeft(0);
mRight(0);

while (!isOnFinish) {
    right = brick.sensor("A1").read();
    fwd = brick.sensor("A2").read();
    if (right < 20 && fwd < 25) {
        //left

        dir = dir - 1;
        rotate(45);
        dir = dir + 4;
        dir = dir % 4;

    } else
    if (right > 30) {
        //right

        dir = dir + 1;
        dir = dir % 4;
        rotate(-45);
        if (dir != 3) {
            if (dir == 0) {
                currentX++;
                forwardGyro(dir * 90, 12, 1);
            } else
            if (dir == 1) {
                currentY++;
                forwardGyro(dir * 90, 12, 1);
            }
            if (dir == 2) {
                currentX--;
                forwardGyro(dir * 90, 12, 1);
            }
        }

    } else {
        currentY--;
        forwardGyro(-90, 12, 1);
    }
    script.wait(750);
} else {
    if (dir == 3) {

```

```

        currentY--;
        forwardGyro(-90, 12, 1);
    }
    if (dir == 1) {
        currentY++;
        forwardGyro(90, 12, 1);
    }
    if (dir == 2) {
        currentX--;
        forwardGyro(180, 12, 1);
    }
    if (dir == 0) {
        currentX++;
        forwardGyro(0, 12, 1);
    }
    script.wait(500);
}
brick.display().addLabel(currentX + " " + currentY, 20, 20);
brick.display().redraw();
if (currentX == 0 && currentY == 0) {
    isOnFinish = true;
}

}

mLeft(0);
mRight(0);

brick.stop();
script.quit();
}

```

Программа команды для решения задачи “локализация” второго этапа:

```

var pi = 3.1415926535897931;
var d = 8.25; //8.7; //diameter of wheel, cm
var l = 17.5; // base of robot
var degInRad = 180 / pi;

var alpha = 0;
var gyro;
gyro = brick.gyroscope();

```



```

gyro.calibrate(10000);
print("initialized gyro");
script.wait(11000);
var fullTurn = 4 * 87; // from experimets. Not 360 degrees, but ....

//кнопки
var stopKey = KeysEnum.Esc;
var startKey = KeysEnum.Left;

//диод
var led = brick.led();

//моторы
var mLeft = brick.motor(M3).setPower;
var mRight = brick.motor(M4).setPower;

//сенсоры ИК
sA1 = brick.sensor(A1).read;
sA2 = brick.sensor(A2).read;

//энкодеры
var eLeft = brick.encoder(E3);
var eRight = brick.encoder(E4);
var cpr = 374; //560; //374; // количество сигналов на оборот

//длина клетки
var cellLength = 40 * cpr / (pi * d);

//переопределение метода нажатия клавиши
if (0) {
    var key = 0;

    function wasPressed(_key) {
        key = 0;
        if (brick.keys().wasPressed(_key)) {
            key = _key;
            return 1;
        } else {
            return 0;
        }
    }
}

```

```

var direction = 0; // absolute angle of direction movement
var directionOld = 0;

print("-----");

led.red();

eLeft.reset();
eRight.reset();

var el = eLeft.readRawData();
var er = eRight.readRawData();
mLeft(0);
mRight(0);
print("Start");

//инициализация и калибровка гироскопа

var v = 20; // velocity

var ex = 0;
var ey = 0;
var encLeftOld = 0;
var encRightOld = 0;
var encLeft = 0;
var encRight = 0;
var n = 0;

// вычисление абсолютного угла относительно начального положения
function angle() {
    var sgn = 0;
    var _direction = readYaw(); // mgrad
    var dtDirection = _direction - directionOld;

    sgn = directionOld == 0 ? 0 : directionOld /
Math.abs(directionOld);
    n += sgn * Math.floor(Math.abs(dtDirection) / 320000));
    direction = _direction + n * 360000;
    directionOld = _direction;
}

var t = 0;

```

```
print("Run, robot, run!");
```

```
// делаем прерывание основной программы с частотой 200Гц, чтобы  
посчитать абсолютный угол с гироскопа
```

```
//var mtimer = script.timer(50);
```

```
//mtimer.timeout.connect(angle);
```

```
// вывод в консоль показаний гироскопа
```

```
function printGyro() {  
    print("direction=" + direction + " sA1" + sA1() + " sA2" + sA2());  
}
```

```
//var ptimer = script.timer(300);
```

```
//ptimer.timeout.connect(printGyro);
```

```
//поворот на угол по гироскопу
```

```
function turnDirection(_angle, _v) {  
    _angle = _angle * 1000; //toMGrad  
    var _vel = _v == undefined ? 10 : _v;  
    var angleOfRotate = _angle - direction;  
    print("ar=" + angleOfRotate);  
    var sgn = angleOfRotate == 0 ? 0 : angleOfRotate / Math.abs(  
        angleOfRotate);  
    mLeft(_vel * sgn);  
    mRight(-_vel * sgn);  
    while (Math.abs(angleOfRotate = _angle - direction) > 20000) {  
        script.wait(50);  
    }  
    brick.motor(M3).forceBreak(500);  
    brick.motor(M4).forceBreak(500);  
    script.wait(500);  
}
```

```
// проезд в перед на количество ячеек _kcell со скоростью _v выравниваясь  
по гироскопу на угол _alpha
```

```
function forward(_alpha, _v, _kcell) {  
    var _vel = _v == undefined ? 10 : _v;  
    var u = 0;  
    var e1 = Math.abs(eLeft.readRawData());  
    while (Math.abs(eLeft.readRawData()) < e1 + (_kcell * cellLength))
```

```

{
    u = -0.7 * (_alpha - direction / 1000);
    mLeft(_vel - u);
    mRight(_vel + u);
    script.wait(5);
}
brick.motor(M3).forceBreak();
brick.motor(M4).forceBreak();
script.wait(500);
}

```

// проезд назад, чтобы упереться в стенку с целью выравнивания

```

function backward() {
    mLeft(-50);
    mRight(-50);
    var eLOld = eLeft.readRawData();
    var eROld = eRight.readRawData();
    var el = eLOld;
    var er = eROld;
    do {
        eLOld = el;
        eROld = er;
        script.wait(1000);
        el = eLeft.readRawData();
        er = eRight.readRawData();
    }
    while (Math.abs(eLOld - el) > 200 && Math.abs(eROld - er) > 200)
    brick.motor(M3).forceBreak();
    brick.motor(M4).forceBreak();
    script.wait(300);
}

```

```

function get_odo() {
    tilt = (gyro.read()[6] / 1000) * pi / 180;
    var eL = brick.encoder("E3").read();
    var eR = brick.encoder("E4").read();

    block = ((eL - eoldL) * cmtodeg + (eR - eoldR) * cmtodeg);

    x = xold + (block / 2) * (Math.cos(told + (tilt - told) / 2));
    y = yold + (block / 2) * (Math.sin(told + (tilt - told) / 2));
}

```

```

eoldL = eL;
eoldR = eR;

xold = x;
yold = y;
told = tilt;

print("x= " + x + " y= " + y);
}

var dir = 0;
var isOnFinish = false;
var currentX = 0,
    currentY = 0;
var fwd = 0,
    right = 0;
var cmtodeg = (pi * 37) / 3600;

var forwardGyro = function(angle, speed, kcell) {
  if (angle == 180) {
    eLeft.reset();
    var e1 = Math.abs(eLeft.readRawData());
    while (Math.abs(eLeft.readRawData()) < e1 + (kcell * cellLength))
    {
      tilt = (gyro.read()[6] / 1000);
      var k = 0.7
      if (tilt > -180 && tilt < 0)
        k = -0.7
      else
        k = 0.7;
      var u = k * (180 - Math.abs(tilt))
      mLeft(speed - u);
      mRight(speed + u);
      script.wait(1);
    }
    mLeft(-10);
    mRight(-10);
    script.wait(250);
    mLeft(0);
    mRight(0);
  } else {

```

```

eLeft.reset();
var e1 = Math.abs(eLeft.readRawData());
while (Math.abs(eLeft.readRawData()) < e1 + (kcell * cellLength))
{
    tilt = (gyro.read()[6] / 1000);
    var u = 0.7 * (angle - tilt)
    mLeft(speed - u);
    mRight(speed + u);
    script.wait(1);
}
mLeft(-10);
mRight(-10);
script.wait(250);
mLeft(0);
mRight(0);
}
}

```

```

var rotate = function(deg) {
    var limit = 0;

    if ((deg <= 0) && (deg >= -180)) {
        var eR = eRight.readRawData();
        limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
        var target = Math.abs(eR) + limit;
        while (Math.abs(eR) < target) {
            mRight(25);
            mLeft(-25);
            eR = eRight.readRawData();
            script.wait(1);
        }
        mRight(-10);
        mLeft(10);
        script.wait(200);
        mRight(0);
        mLeft(0);
    }
    if ((deg >= 0) && (deg <= 180)) {
        var eL = eLeft.readRawData();
        limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
        var target = Math.abs(eL) + limit;
        while (Math.abs(eL) < target) {
            mRight(-25);

```

```

        mLeft(25);
        eL = eLeft.readRawData();
        script.wait(1);
    }
    mRight(10);
    mLeft(-10);
    script.wait(200);
    mRight(0);
    mLeft(0);
}
}

```

```

var count = 0;

```

```

var main = function() {

```

```

    mLeft(0);
    mRight(0);

```

```

while (count != 8) {
    right = brick.sensor("A1").read();
    fwd = brick.sensor("A2").read();
    if (right < 20 && fwd < 25) {
        //left

        dir = dir - 1;
        rotate(45);
        dir = dir + 4;
        dir = dir % 4;

    } else
    if (right > 30) {
        //right

        dir = dir + 1;
        dir = dir % 4;
        rotate(-45);
        if (dir != 3) {
            if (dir == 0) {
                currentX++;
                forwardGyro(dir * 90, 30, 1);
            } else
            if (dir == 1) {

```

```

        currentY++;
        forwardGyro(dir * 90, 30, 1);
    }
    if (dir == 2) {
        currentX--;
        forwardGyro(dir * 90, 30, 1);
    }

} else {
    currentY--;
    forwardGyro(-90, 30, 1);
}
count++;
script.wait(750);
} else {
    if (dir == 3) {
        currentY--;
        forwardGyro(-90, 30, 1);
    }
    if (dir == 1) {
        currentY++;
        forwardGyro(90, 30, 1);
    }
    if (dir == 2) {
        currentX--;
        forwardGyro(180, 30, 1);
    }
    if (dir == 0) {
        currentX++;
        forwardGyro(0, 30, 1);
    }
    count++;
    script.wait(500);
}
brick.display().addLabel(currentX + " " + currentY, 20, 20);
brick.display().redraw();
}

```

```

mLeft(0);
mRight(0);
script.wait(10000);
brick.stop();

```



```
    script.quit();  
}
```

Программа команды для решения задачи третьего этапа:

```
var pi = 3.1415926535897931;  
var d = 8.25; //8.7; //diameter of wheel, cm  
var l = 17.5; // base of robot  
var degInRad = 180 / pi;  
  
var alpha = 0;  
  
var fullTurn = 4 * 87; // from experimets. Not 360 degrees, but ....  
  
//кнопки  
var stopKey = KeysEnum.Esc;  
var startKey = KeysEnum.Left;  
  
//диод  
var led = brick.led();  
  
//моторы  
var mLeft = brick.motor(M3).setPower;  
var mRight = brick.motor(M4).setPower;  
  
//сенсоры ИК  
//sA1 = brick.sensor(A1).read;  
//sA2 = brick.sensor(A2).read;  
  
//энкодеры  
var eLeft = brick.encoder(E3);  
var eRight = brick.encoder(E4);  
var cpr = 374; //560; //374; // количество сигналов на оборот  
  
//длина клетки  
var cellLength = 40 * cpr / (pi * d);  
  
//переопределение метода нажатия клавиши  
if (0) {  
    var key = 0;  
  
    function wasPressed(_key) {
```

```

    key = 0;
    if (brick.keys().wasPressed(_key)) {
        key = _key;
        return 1;
    } else {
        return 0;
    }
}
}

var direction = 0; // absolute angle of direction movement
var directionOld = 0;

print("-----");

led.red();

eLeft.reset();
eRight.reset();

var el = eLeft.readRawData();
var er = eRight.readRawData();
mLeft(0);
mRight(0);
print("Start");

//инициализация и калибровка гироскопа

var v = 20; // velocity

var ex = 0;
var ey = 0;
var encLeftOld = 0;
var encRightOld = 0;
var encLeft = 0;
var encRight = 0;
var n = 0;

// вычисление абсолютного угла относительно начального положения
function angle() {
    var sgn = 0;
    var _direction = readYaw(); // mgrad

```

```

var dtDirection = _direction - directionOld;

sgn = directionOld == 0 ? 0 : directionOld /
Math.abs(directionOld);
n += sgn * Math.floor(Math.abs(dtDirection / 320000));
direction = _direction + n * 360000;
directionOld = _direction;
}

var t = 0;

print("Run, robot, run!");

// делаем прерывание основной программы с частотой 200Гц, чтобы
посчитать абсолютный угол с гироскопа
//var mtimer = script.timer(50);
//mtimer.timeout.connect(angle);

// вывод в консоль показаний гироскопа
function printGyro() {
    print("direction=" + direction + " sA1" + sA1() + " sA2" + sA2());
}

//var ptimer = script.timer(300);
//ptimer.timeout.connect(printGyro);

//поворот на угол по гироскопу
function turnDirection(_angle, _v) {
    _angle = _angle * 1000; //toMGrad
    var _vel = _v == undefined ? 10 : _v;
    var angleOfRotate = _angle - direction;
    print("ar=" + angleOfRotate);
    var sgn = angleOfRotate == 0 ? 0 : angleOfRotate / Math.abs(
        angleOfRotate);
    mLeft(_vel * sgn);
    mRight(-_vel * sgn);
    while (Math.abs(angleOfRotate = _angle - direction) > 20000) {
        script.wait(50);
    }
    brick.motor(M3).forceBreak(500);
    brick.motor(M4).forceBreak(500);
}

```

```
    script.wait(500);  
}
```

// проезд в перед на количество ячеек _kcell со скоростью _v выравниваясь по гироскопу на угол _alpha

```
function forward(_alpha, _v, _kcell) {  
    var _vel = _v == undefined ? 10 : _v;  
    var u = 0;  
    var e1 = Math.abs(eLeft.readRawData());  
    while (Math.abs(eLeft.readRawData()) < e1 + (_kcell * cellLength))  
    {  
        u = -0.7 * (_alpha - direction / 1000);  
        mLeft(_vel - u);  
        mRight(_vel + u);  
        script.wait(5);  
    }  
    brick.motor(M3).forceBreak();  
    brick.motor(M4).forceBreak();  
    script.wait(500);  
}
```

// проезд назад, чтобы упереться в стенку с целью выравнивания

```
function backward() {  
    mLeft(-50);  
    mRight(-50);  
    var eLOld = eLeft.readRawData();  
    var eROld = eRight.readRawData();  
    var e1 = eLOld;  
    var er = eROld;  
    do {  
        eLOld = e1;  
        eROld = er;  
        script.wait(1000);  
        e1 = eLeft.readRawData();  
        er = eRight.readRawData();  
    }  
    while (Math.abs(eLOld - e1) > 200 && Math.abs(eROld - er) > 200)  
    brick.motor(M3).forceBreak();  
    brick.motor(M4).forceBreak();  
    script.wait(300);  
}
```

```

function get_odo() {
    tilt = (gyro.read()[6] / 1000) * pi / 180;
    var eL = brick.encoder("E3").read();
    var eR = brick.encoder("E4").read();

    block = ((eL - eoldL) * cmtodeg + (eR - eoldR) * cmtodeg);

    x = xold + (block / 2) * (Math.cos(told + (tilt - told) / 2));
    y = yold + (block / 2) * (Math.sin(told + (tilt - told) / 2));

    eoldL = eL;
    eoldR = eR;

    xold = x;
    yold = y;
    told = tilt;

    print("x= " + x + " y= " + y);
}

var dir = 0;
var isOnFinish = false;
var currentX = 0,
    currentY = 0;
var fwd = 0,
    right = 0;
var cmtodeg = (pi * 37) / 3600;

var forwardGyro = function(angle, speed, kcell) {
    if (angle == 180) {
        eLeft.reset();
        var e1 = Math.abs(eLeft.readRawData());
        while (Math.abs(eLeft.readRawData()) < e1 + (kcell * cellLength))
        {
            tilt = (gyro.read()[6] / 1000);
            var k = 0.7
            if (tilt > -180 && tilt < 0)
                k = -0.7
            else
                k = 0.7;
            var u = k * (180 - Math.abs(tilt))

```

```

        mLeft(30 - u);
        mRight(30 + u);
        script.wait(1);
    }
    mLeft(-10);
    mRight(-10);
    script.wait(250);
    mLeft(0);
    mRight(0);
} else {
    eLeft.reset();
    var e1 = Math.abs(eLeft.readRawData());
    while (Math.abs(eLeft.readRawData()) < e1 + (kcell * cellLength))
    {
        tilt = (gyro.read()[6] / 1000);
        var u = 0.7 * (angle - tilt)
        mLeft(30 - u);
        mRight(30 + u);
        script.wait(1);
    }
    mLeft(-10);
    mRight(-10);
    script.wait(250);
    mLeft(0);
    mRight(0);
}
}

var rotate = function(deg) {
    var limit = 0;

    if ((deg <= 0) && (deg >= -180)) {
        var eR = eRight.readRawData();
        limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
        var target = Math.abs(eR) + limit;
        while (Math.abs(eR) < target) {
            mRight(25);
            mLeft(-25);
            eR = eRight.readRawData();
            script.wait(1);
        }
        mRight(-10);
        mLeft(10);
    }

```

```

    script.wait(200);
    mRight(0);
    mLeft(0);
}
if ((deg >= 0) && (deg <= 180)) {
    var eL = eLeft.readRawData();
    limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
    var target = Math.abs(eL) + limit;
    while (Math.abs(eL) < target) {
        mRight(-25);
        mLeft(25);
        eL = eLeft.readRawData();
        script.wait(1);
    }
    mRight(10);
    mLeft(-10);
    script.wait(200);
    mRight(0);
    mLeft(0);
}
}

```

```

var gyro;
gyro = brick.gyroscope();
gyro.calibrate(15000);
print("initialized gyro");
script.wait(16000);

```

```

var main = function() {

    mLeft(0);
    mRight(0);

    var data = brick.sensor("A6").read();
    print(data);
    var spd = 0;
    var tim = script.time();
    spd = 25;
    while (data < 40) {
        data = brick.sensor("A6").read();
        tilt = (gyro.read()[6] / 1000);
    }
}

```

```

    var u = (0 - tilt) * -1;
    mLeft(spdx + u);
    mRight(spdx - u);
    //if(script.time() - tim > 50 && spd < 15)
    //    spd++;
    print(data);
    script.wait(1);
}

mLeft(-10);
mRight(-10);
script.wait(200);
mLeft(0);
mRight(0);
var left, right;
left = brick.sensor("A5").read();
right = brick.sensor("A6").read();
/*while((left < 20 || left > 60) && (right < 20 || right > 60))
{
    var k = 0.6;
    var u1 = 35 - left;
    var u2 = 35 - right;

    mLeft((15+u1)*k);
    mRight((15+u2)*k);
    script.wait(1);
}*/

mLeft(0);
mRight(0);
script.wait(2000);
mLeft(0);
mRight(0);
var i = 0;
var t = script.time();
var summa = 0;
var step = 3;
spd = 25;
tim = script.time();
eLeft.reset();
while (i != 4) {
    left = brick.sensor("A5").read();
    right = brick.sensor("A6").read();

```



```

var u = eLeft.read() - eRight.read();
//if(script.time() - tim > 75 && spd < 15)
//spd++;
tilt = (gyro.read()[6] / 1000);
var u = (0 - tilt) * -1;
mLeft(spd + u);
mRight(spd - u);
//var m1 = spd - u*0.5;
//var m2 = spd - u*0.5;

var enc = eLeft.read();
if (Math.abs(enc) > 16) //script.time() - t > 200)
{
    mLeft(-5);
    mRight(-5);
    script.wait(100);
    mLeft(0);
    mRight(0);
    script.wait(1000);
    if ((left + right) / 2 > 50) {
        print("black");
        summa = summa + Math.pow(2, step);
        step--;
    } else {
        print("white");
        step--;
    }
    print(enc);
    t = script.time();
    i++;
    eLeft.reset();
}
}
mLeft(50);
mRight(50);
script.wait(500);
mLeft(0);
mRight(0);
print(summa);
brick.display().addLabel(summa, 20, 20);
brick.display().redraw();
script.wait(10000);
brick.stop();

```

```
script.quit();
}
```

Программа команды для решения задачи “построение маршрута” четвертого этапа:

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```

],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1
],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0
],
];

var map1 = [];
var map2 = [
  []
];
var map3 = [
  []
];

var deikstra = function(start, end) {
  var i, j, v = 0,
      min, z, index = 0;
  var st, c = "";
  var visited = [];
  var D = [];
  var P = [];

  for (i = 0; i < V; i++) {
    P[i] = start;
    visited[i] = false;
  }
  visited[start] = true;
  for (i = 0; i < V; i++) {
    D[i] = map[start, i];
  }
  D[start] = 0;
  P[start] = 0;

  for (i = 0; i < V - 1; i++) {
    min = 999999;
    for (j = 0; j < V; j++) {
      if (!visited[j] && D[j] <= min) {

```



```

        min = D[j];
        index = j;
    }
}
v = index;
for (j = 0; j < V; j++) {
    if ((D[j] > D[v] + map[v, j]) && (D[v] < 999999) && (map[v, j]
<
        999999)) {
        D[j] = D[v] + map[v, j];
        P[j] = v;
    }
}
start = v;
visited[v] = true;
}
st = "";
z = P[end];
print(end);
while (z != 0) {
    c = z;
    st = c + " -> " + st;
    z = P[z];
}
c = end;
st = st + c;
print(st);

return D[end];
}

```

```

var dfs = function(start, end) {
    var index = 0;
    var current = start;
    var stack = [];
    var marked = [];

    marked[current] = true;
    stack.push(current);
    while (stack.length != 0) {
        current = stack[stack.length - 1];
        var isGotFriend = false;
        for (var i = 0; i < 64; i++) {

```

```

        if (map[current][i] != 0 && !marked[i]) {
            isGotFriend = true;
            current = i;
            if (end == i) {
                stack.push(end);
                print(stack);
            }
            break;
        }
    }
    if (isGotFriend) {
        marked[current] = true;
        stack.push(current);
        //print("pushed " + current);
    } else {
        stack.pop();
        //print("popped " + current);
    }
}

}

var print_way = function(u) {
    if (pred[u] != u) {
        print_way(pred[u]);
    }
    way.push(u);
}

var pred = [];

var bfs = function(start, end) {

    var used = [];

    for (var i = 0; i < 64; i++) {
        used[i] = false;
    }

    used[start] = true;

```

```

pred[start] = start;
var stack = [];

stack.push(start);
var isOnEnd = false;
while (stack.length != 0) {

    var current = stack.shift();
    //print(stack);
    var sosedni = [];
    for (var i = 0; i < 64; i++) {
        if (map[current][i] != 0) {
            sosedni[sosedni.length] = i;
        }
    }
    for (var i = 0; i < sosedni.length; i++) {
        var v = sosedni[i];

        if (!used[v]) {
            used[v] = true;
            pred[v] = current;
            stack.push(v);
        }
    }
    /*for(var i = 0; i < stack.length; i++)
    {
        if(stack[i] != -1)
        {
            isOnEnd = false;
            break;
        } else
        {
            isOnEnd = true;
        }
    }
    */

}

}

var xStart = 0;

```

```

var yStart = 5;
var rotate = 1;
var xFinish = 0;
var yFinish = 6;
var rotateOld = rotate;

var start = yStart * 8 + xStart;
var finish = yFinish * 8 + xFinish;

//моторы
var mLeft = brick.motor(M3).setPower;
var mRight = brick.motor(M4).setPower;

//энкодеры
var eLeft = brick.encoder(E3);
var eRight = brick.encoder(E4);

var d = 8.25; //8.7; //diameter of wheel, cm
var l = 17.5; // base of robot
var pi = 3.141592653589793;
var degInRad = 180 / pi;

var cpr = 374;
//длина клетки
var cellLength = 38 * cpr / (pi * d);
var cmtodeg = (pi * 37) / 3600;

eLeft.reset();
eRight.reset();

var gyro;
gyro = brick.gyroscope();
gyro.calibrate(15000);
script.wait(16000);
print("initialized gyro");
var tilt = 0;

var forwardGyro = function(angle, speed, kcell) {
  if (angle == 180) {
    eLeft.reset();
    var e1 = Math.abs(eLeft.readRawData());

```

```

print(cellLength);
while (Math.abs(eLeft.readRawData()) < (e1 + (kcell *
    cellLength))) {
    print(eLeft.readRawData());
    tilt = (gyro.read()[6] / 1000);
    var k = 0.7
    if (tilt > -180 && tilt < 0)
        k = -0.7
    else
        k = 0.7;
    var u = k * (180 - Math.abs(tilt) + 5)
    mLeft(30 - u);
    mRight(30 + u);
    script.wait(1);
}
mLeft(-10);
mRight(-10);
script.wait(250);
mLeft(0);
mRight(0);
} else {
    eLeft.reset();
    var e1 = Math.abs(eLeft.readRawData());
    while (Math.abs(eLeft.readRawData()) < (e1 + (kcell *
        cellLength))) {
        print(eLeft.readRawData());
        tilt = (gyro.read()[6] / 1000);
        var u = 0.7 * (angle - tilt + 5)
        mLeft(30 - u);
        mRight(30 + u);
        script.wait(1);
    }
    mLeft(-10);
    mRight(-10);
    script.wait(250);
    mLeft(0);
    mRight(0);
}
}

var rotateMot = function(deg) {
    var limit = 0;

```

```

if ((deg <= 0) && (deg >= -180)) {
    var eR = eRight.readRawData();
    limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
    var target = Math.abs(eR) + limit;
    while (Math.abs(eR) < target) {
        mRight(25);
        mLeft(-25);
        eR = eRight.readRawData();
        script.wait(1);
    }
    mRight(-10);
    mLeft(10);
    script.wait(200);
    mRight(0);
    mLeft(0);
}
if ((deg >= 0) && (deg <= 180)) {
    var eL = eLeft.readRawData();
    limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
    var target = Math.abs(eL) + limit;
    while (Math.abs(eL) < target) {
        mRight(-25);
        mLeft(25);
        eL = eLeft.readRawData();
        script.wait(1);
    }
    mRight(10);
    mLeft(-10);
    script.wait(200);
    mRight(0);
    mLeft(0);
}
}
}

```

```

var main = function() {
    __interpretation_started_timestamp__ = Date.now();
    mLeft(0);
    mRight(0);

    bfs(start, finish);
}

```

```

print_way(finish);

var wayPath = [];

for (var i = 0; i < way.length - 1; i++) {
    var delta = way[i + 1] - way[i];

    switch (delta) {
        //VLEVO
        case -1:
        {
            var delRot = 3 - rotate;
            switch (delRot) {
                case 3:
                {
                    //povorot nalevo
                    wayPath.push('L');
                    break;
                }
                case 2:
                {
                    wayPath.push('L');
                    wayPath.push('L');
                    break;
                }
                case 1:
                {
                    wayPath.push('R');
                    break;
                }
            }
            rotate = 3;
            wayPath.push('F');
            break;
        }
        //VPRAVO
        case 1:
        {
            var delRot = 1 - rotate;
            switch (delRot) {
                case 1:
                {

```

```

        //povorot nalevo
        wayPath.push('R');
        break;
    }
    case -1:
    {
        wayPath.push('L');
        break;
    }
    case -2:
    {
        wayPath.push('L');
        wayPath.push('L');
        break;
    }

}
rotate = 1;
wayPath.push('F');
break;
}
//VVERH
case -8:
{
    var delRot = 0 - rotate;
    switch (delRot) {
        case -1:
        {
            //povorot nalevo
            wayPath.push('L');
            break;
        }
        case -2:
        {
            wayPath.push('L');
            wayPath.push('L');
            break;
        }
        case -3:
        {
            wayPath.push('R');
            break;
        }
    }
}

```



```

    }
    rotate = 0;
    wayPath.push('F');
    break;
}
//VNIZ
case 8:
{
    var delRot = 2 - rotate;
    switch (delRot) {
        case 2:
        {
            //povorot nalevo
            wayPath.push('L');
            wayPath.push('L');
            break;
        }
        case 1:
        {
            wayPath.push('R');
            break;
        }
        case -1:
        {
            wayPath.push('L');
            break;
        }
    }

    rotate = 2;
    wayPath.push('F');
    break;
}
}
}

```

```

print(wayPath);
brick.display().addLabel(wayPath, 5, 20);
brick.display().redraw();
script.wait(10000);

```

```

rotate = rotateOld;
var dir = 0;

for (var i = 0; i < wayPath.length; i++) {
    //print(rotate);
    switch (wayPath[i]) {
        case 'R':
            {
                rotateMot(-44);
                dir++;
                dir = dir % 4;
                break;
            }
        case 'L':
            {
                dir--;
                dir = dir + 4;
                dir = dir % 4;
                rotateMot(44);
                break;
            }
        case 'F':
            {
                if (dir != 3) {
                    forwardGyro(90 * dir, 25, 1);
                } else {
                    forwardGyro(-90, 25, 1);
                }
                break;
            }
    }
}

brick.playSound("media/beep.wav");

return;
}

```

Программа команды для решения полной финальной задачи:

```

var pi = 3.1415926535897931;
var d = 8.25; //8.7; //diameter of wheel, cm
var l = 17.5; // base of robot
var degInRad = 180 / pi;

var wasRotate = false;
var codePos = false;

var xStart = 6;
var yStart = 1;
var rot = 0;
var xFinish = 7;
var yFinish = 4;
var rotateOld = rot;

var start = yStart * 8 + xStart;
var finish = yFinish * 8 + xFinish;

var alpha = 0;
var gyro;
gyro = brick.gyroscope();
gyro.calibrate(15000);
print("initialized gyro");
script.wait(16000);
var fullTurn = 4 * 87; // from experimets. Not 360 degrees, but ....

//кнопки
var stopKey = KeysEnum.Esc;
var startKey = KeysEnum.Left;

//диод
var led = brick.led();

//моторы
var mLeft = brick.motor(M3).setPower;
var mRight = brick.motor(M4).setPower;

//сенсоры ИК
sA1 = brick.sensor(A1).read;
sA2 = brick.sensor(A2).read;

//энкодеры
var eLeft = brick.encoder(E3);

```

```

var eRight = brick.encoder(E4);
var cpr = 374; //560; //374; // количество сигналов на оборот

//длина клетки
var cellLength = 39.5 * cpr / (pi * d);

//переопределение метода нажатия клавиши
if (0) {
    var key = 0;

    function wasPressed(_key) {
        key = 0;
        if (brick.keys().wasPressed(_key)) {
            key = _key;
            return 1;
        } else {
            return 0;
        }
    }
}

var direction = 0; // absolute angle of direction movement
var directionOld = 0;

print("-----");

led.red();

eLeft.reset();
eRight.reset();

var el = eLeft.readRawData();
var er = eRight.readRawData();
mLeft(0);
mRight(0);
print("Start");

//инициализация и калибровка гироскопа

var v = 20; // velocity
var ex = 0;
var ey = 0;
var encLeftOld = 0;

```

```

var encRightOld = 0;
var encLeft = 0;
var encRight = 0;
var n = 0;

// вычисление абсолютного угла относительно начального положения
function angle() {
    var sgn = 0;
    var _direction = readYaw(); // mgrad
    var dtDirection = _direction - directionOld;

    sgn = directionOld == 0 ? 0 : directionOld /
Math.abs(directionOld);
    n += sgn * Math.floor(Math.abs(dtDirection) / 320000));
    direction = _direction + n * 360000;
    directionOld = _direction;
}

var t = 0;

print("Run, robot, run!");

// делаем прерывание основной программы с частотой 200Гц, чтобы
// посчитать абсолютный угол с гироскопа
//var mtimer = script.timer(50);
//mtimer.timeout.connect(angle);

// вывод в консоль показаний гироскопа
function printGyro() {
    print("direction=" + direction + " sA1" + sA1() + " sA2" + sA2());
}

//var ptimer = script.timer(300);
//ptimer.timeout.connect(printGyro);

//поворот на угол по гироскопу
function turnDirection(_angle, _v) {
    _angle = _angle * 1000; //toMGrad
    var _vel = _v == undefined ? 10 : _v;
    var angleOfRotate = _angle - direction;
    print("ar=" + angleOfRotate);
}

```

```

var sgn = angleOfRotate == 0 ? 0 : angleOfRotate / Math.abs(
    angleOfRotate);
mLeft(_vel * sgn);
mRight(-_vel * sgn);
while (Math.abs(angleOfRotate = _angle - direction) > 20000) {
    script.wait(50);
}
brick.motor(M3).forceBreak(500);
brick.motor(M4).forceBreak(500);
script.wait(500);
}

```

// проезд в перед на количество ячеек _kcell со скоростью _v выравниваясь по гироскопу на угол _alpha

```

function forward(_alpha, _v, _kcell) {
    var _vel = _v == undefined ? 10 : _v;
    var u = 0;
    var e1 = Math.abs(eLeft.readRawData());
    while (Math.abs(eLeft.readRawData()) < e1 + (_kcell * cellLength))
    {
        u = -0.7 * (_alpha - direction / 1000);
        mLeft(_vel - u);
        mRight(_vel + u);
        script.wait(5);
    }
    brick.motor(M3).forceBreak();
    brick.motor(M4).forceBreak();
    script.wait(500);
}

```

// проезд назад, чтобы упереться в стенку с целью выравнивания

```

function backward() {
    mLeft(-50);
    mRight(-50);
    var eLOld = eLeft.readRawData();
    var eROld = eRight.readRawData();
    var e1 = eLOld;
    var er = eROld;
    do {
        eLOld = e1;
        eROld = er;
        script.wait(1000);
        e1 = eLeft.readRawData();
    }
}

```

```

    er = eRight.readRawData();
}
while (Math.abs(eLold - e1) > 200 && Math.abs(eRold - er) > 200)
brick.motor(M3).forceBreak();
brick.motor(M4).forceBreak();
script.wait(300);

}

function get_odo() {
    tilt = (gyro.read()[6] / 1000) * pi / 180;
    var eL = brick.encoder("E3").read();
    var eR = brick.encoder("E4").read();

    block = ((eL - eoldL) * cmtodeg + (eR - eoldR) * cmtodeg);

    x = xold + (block / 2) * (Math.cos(told + (tilt - told) / 2));
    y = yold + (block / 2) * (Math.sin(told + (tilt - told) / 2));

    eoldL = eL;
    eoldR = eR;

    xold = x;
    yold = y;
    told = tilt;

    print("x= " + x + " y= " + y);
}

var isOnFinish = false;
var currentX = 0,
    currentY = 0;
var fwd = 0,
    right = 0;
var cmtodeg = (pi * 37) / 3600;

var forwardGyro = function(angle, speed, kcell) {
    if (angle == 180) {
        eLeft.reset();
        var e1 = Math.abs(eLeft.readRawData());
        print(eLeft.readRawData());
        while (Math.abs(eLeft.readRawData()) < e1 + (kcell * cellLength))
    {

```

```

    tilt = (gyro.read()[6] / 1000);
    print(eLeft.readRawData());
    var k = 0.7
    if (tilt > -180 && tilt < 0)
        k = -0.7
    else
        k = 0.7;
    var u = k * (180 - Math.abs(tilt))
    mLeft(30 - u);
    mRight(30 + u);
    script.wait(1);
    //delete here
    if (brick.sensor("A5").read() > 50) {
        mLeft(-10);
        mRight(-10);
        script.wait(100);
        mLeft(0);
        mRight(0);
        isOnFinish = true;
        break;
    }
}
mLeft(-10);
mRight(-10);
script.wait(250);
mLeft(0);
mRight(0);
} else {
    eLeft.reset();
    var e1 = Math.abs(eLeft.readRawData());
    print(eLeft.readRawData());
    while (Math.abs(eLeft.readRawData()) < e1 + (kcell * cellLength))
{
    print(eLeft.readRawData());
    tilt = (gyro.read()[6] / 1000);
    var u = 0.7 * (angle - tilt)
    mLeft(30 - u);
    mRight(30 + u);
    script.wait(1);
    //delete here
    if (brick.sensor("A5").read() > 50) {
        mLeft(-10);
        mRight(-10);

```


[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```

],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0
],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0
],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1
],
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0
],
];

var print_way = function(u) {
    if (pred[u] != u) {
        print_way(pred[u]);
    }
    way.push(u);
}

var pred = [];

var bfs = function(start, end) {

    var used = [];

    for (var i = 0; i < 64; i++) {
        used[i] = false;
    }

    used[start] = true;
    pred[start] = start;
    var stack = [];

    stack.push(start);
    var isOnEnd = false;

```

```

while (stack.length != 0) {

    var current = stack.shift();
    //print(stack);
    var sosed = [];
    for (var i = 0; i < 64; i++) {
        if (map[current][i] != 0) {
            sosed[sosed.length] = i;
        }
    }
    for (var i = 0; i < sosed.length; i++) {
        var v = sosed[i];

        if (!used[v]) {
            used[v] = true;
            pred[v] = current;
            stack.push(v);
        }
    }
    /*for(var i = 0; i < stack.Length; i++)
    {
        if(stack[i] != -1)
        {
            isOnEnd = false;
            break;
        } else
        {
            isOnEnd = true;
        }
    }
    */

}

}

var rotate = function(deg) {
    var limit = 0;

    if ((deg <= 0) && (deg >= -180)) {
        var eR = eRight.readRawData();
        limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
    }
}

```



```

    var target = Math.abs(eR) + limit;
    while (Math.abs(eR) < target) {
        mRight(25);
        mLeft(-25);
        eR = eRight.readRawData();
        script.wait(1);
    }
    mRight(-10);
    mLeft(10);
    script.wait(200);
    mRight(0);
    mLeft(0);
}
if ((deg >= 0) && (deg <= 180)) {
    var eL = eLeft.readRawData();
    limit = ((Math.abs(deg / 180) * pi * 20) / cmtodeg) / 2;
    var target = Math.abs(eL) + limit;
    while (Math.abs(eL) < target) {
        mRight(-25);
        mLeft(25);
        eL = eLeft.readRawData();
        script.wait(1);
    }
    mRight(10);
    mLeft(-10);
    script.wait(200);
    mRight(0);
    mLeft(0);
}
}

var dir = 0;

var main = function() {

    mLeft(0);
    mRight(0);

    print("started");

    while (!isOnFinish) {
        right = brick.sensor("A1").read();

```

```

fwd = brick.sensor("A2").read();
if (right < 20 && fwd < 25) {
    //left
    wasRotate = true;
    dir = dir - 1;
    rotate(45);
    dir = dir + 4;
    dir = dir % 4;
    //delete here
    /*backward();
    if(dir != 3)
        forwardGyro(dir * 90, 12, 0.5);
    else
        forwardGyro(-90, 12, 0.25);
    */
} else
if (right > 30) {
    //right

    dir = dir + 1;
    dir = dir % 4;
    rotate(-45);
    wasRotate = true;
    if (dir != 3) {
        if (dir == 0) {
            currentX++;
            forwardGyro(dir * 90, 12, 1);
        } else
        if (dir == 1) {
            currentY++;
            forwardGyro(dir * 90, 12, 1);
        }
        if (dir == 2) {
            currentX--;
            forwardGyro(dir * 90, 12, 1);
        }
    }

    } else {
        currentY--;
        forwardGyro(-90, 12, 1);
    }
    script.wait(750);
} else {

```

```

    if (dir == 3) {
        currentY--;
        forwardGyro(-90, 12, 1);
    }
    if (dir == 1) {
        currentY++;
        forwardGyro(90, 12, 1);
    }
    if (dir == 2) {
        currentX--;
        forwardGyro(180, 12, 1);
    }
    if (dir == 0) {
        currentX++;
        forwardGyro(0, 12, 1);
    }
    script.wait(500);
}
if (brick.sensor("A6") > 50) {
    isOnFinish = true;
}
if (!isOnFinish) {
    wasRotate = false;
}

}
mLeft(-25);
mRight(-25);
script.wait(400);
mLeft(0);
mRight(0);

mLeft(10);
mRight(10);
script.wait(100);
mLeft(0);
mRight(0);

var data = brick.sensor("A6").read();
print(data);
var spd = 0;
var tim = script.time();
spd = 25;

```

```

gyro.calibrate(15000);
print("initialized gyro");
script.wait(16000);

while (data < 40) {
    data = brick.sensor("A6").read();
    tilt = (gyro.read()[6] / 1000);
    var u = (0 - tilt) * -1;
    mLeft(spd + u);
    mRight(spd - u);
    //if(script.time() - tim > 50 && spd < 15)
    //    spd++;
    print(data);
    script.wait(1);
}

mLeft(-10);
mRight(-10);
script.wait(200);
mLeft(0);
mRight(0);
var left, right;
left = brick.sensor("A5").read();
right = brick.sensor("A6").read();
/*while((left < 20 || left > 60) && (right < 20 || right > 60))
{
    var k = 0.6;
    var u1 = 35 - left;
    var u2 = 35 - right;

    mLeft((15+u1)*k);
    mRight((15+u2)*k);
    script.wait(1);
}*/

mLeft(0);
mRight(0);
script.wait(2000);
mLeft(0);
mRight(0);
var i = 0;
var t = script.time();
var summa = 0;

```

```

var step = 3;
spd = 25;
tim = script.time();
eLeft.reset();
while (i != 4) {
    left = brick.sensor("A5").read();
    right = brick.sensor("A6").read();
    var u = eLeft.read() - eRight.read();
    //if(script.time() - tim > 75 && spd < 15)
    //spd++;
    tilt = (gyro.read()[6] / 1000);
    var u = (0 - tilt) * -1;
    mLeft(spd + u);
    mRight(spd - u);
    //var m1 = spd - u*0.5;
    //var m2 = spd - u*0.5;

    var enc = eLeft.read();
    if (Math.abs(enc) > 16) //script.time() - t > 200)
    {
        mLeft(-5);
        mRight(-5);
        script.wait(100);
        mLeft(0);
        mRight(0);
        script.wait(1000);
        if ((left + right) / 2 > 50) {
            print("black");
            summa = summa + Math.pow(2, step);
            step--;
        } else {
            print("white");
            step--;
        }
        print(enc);
        t = script.time();
        i++;
        eLeft.reset();
    }
}
mLeft(40);
mRight(60);
script.wait(700);

```

```
mLeft(0);
mRight(0);
print(summa);
brick.playSound("media/beep.wav");
brick.display().addLabel(summa, 20, 20);
brick.display().redraw();
script.wait(10000);
```

```
/*if(wasRotate)
{
    codePos = false;
} else
{
    codePos = true;
}
print(codePos);
```

```
//PROEZD
```

```
if(codePos)
{
    start = 44;
    finish = 15;
    rot = 1;
} else
{
    start = 15;
    finish = 44;
    rot = 2;
}
```

```
bfs(start,finish);
print_way(finish);
```

```
var wayPath = [];
```

```
for(var i = 0; i < way.length - 1; i++)
{
    var delta = way[i+1] - way[i];

    switch(delta)
    {
```

```

//VLEVO
case -1:
{
    var delRot = 3 - rot;
    switch(delRot)
    {
        case 3:
        {
            //povorot nalevo
            wayPath.push('L');
            break;
        }
        case 2:
        {
            wayPath.push('L');
            wayPath.push('L');
            break;
        }
        case 1:
        {
            wayPath.push('R');
            break;
        }
    }

    }
    rot = 3;
    wayPath.push('F');
    break;
}
//VPRAVO
case 1:
{
    var delRot = 1 - rot;
    switch(delRot)
    {
        case 1:
        {
            //povorot nalevo
            wayPath.push('R');
            break;
        }
        case -1:
        {

```

```

        wayPath.push('L');
        break;
    }
    case -2:
    {
        wayPath.push('L');
        wayPath.push('L');
        break;
    }

}
rot = 1;
wayPath.push('F');
break;
}
//VVERH
case -8:
{
    var delRot = 0 - rot;
    switch(delRot)
    {
        case -1:
        {
            //povorot nalevo
            wayPath.push('L');
            break;
        }
        case -2:
        {
            wayPath.push('L');
            wayPath.push('L');
            break;
        }
        case -3:
        {
            wayPath.push('R');
            break;
        }
    }

}
rot = 0;
wayPath.push('F');
break;

```



```

    }
    //VNIZ
    case 8:
    {
        var delRot = 2 - rot;
        switch(delRot)
        {
            case 2:
            {
                //povorot nalevo
                wayPath.push('L');
                wayPath.push('L');
                break;
            }
            case 1:
            {
                wayPath.push('R');
                break;
            }
            case -1:
            {
                wayPath.push('L');
                break;
            }
        }

        rot = 2;
        wayPath.push('F');
        break;
    }
}

print(wayPath);

rot = rotateOld;

dir = 0;

for(var i = 0; i < wayPath.Length; i++)
{
    //print(rotate);

```

```

switch(wayPath[i])
{
    case 'R':
    {
        rotate(-44);
        dir++;
        dir = dir % 4;
        break;
    }
    case 'L':
    {
        dir--;
        dir = dir + 4;
        dir = dir % 4;
        rotate(44);
        break;
    }
    case 'F':
    {

        if(dir != 3)
        {
            forwardGyro(90 * dir, 25,1);
        } else
        {
            forwardGyro(-90, 25,1);
        }
        break;
    }

}

}
*/
brick.stop();
script.quit();
}

```