Работа призера заключительного этапа

командной инженерной олимпиады школьников

Олимпиада Национальной технологической инициативы

Профиль «АВТОНОМНЫЕ ТРАНСПОРТНЫЕ СИСТЕМЫ»

Серянин Александр Константинович

Класс: 11

Город: Москва

Школа: Физмат школа № 2007

Регион: Москва

Уникальный номер участника: 198

Команда на заключительном этапе: Вжух и в продакшн

Результаты заключительного этапа:

№	Индивидуальный этап													Командный этап		ИТОГ
	Физика				Информатика							За задачи	Коэффициент 40%	баллы	Коэффициент 60%	
198	0	2	0	0	6	0	0	0	0	0	0	8	2,133,333	88,5	21,24	23,373

Индивидуальная часть

Персональный лист участника с номером 198:

Олимпиада НТИ

ФИО Серяжин Александр Константинович

Город Москва

Школа № 2007

Физика

Командная инженерная олимпиада «Олимпиада НТИ»

Направление АТС

Предмет физика

Номер участника 198

1	2	3	4	Σ
~~X~~	2	X	X	2

№1

спут

$S = 20$ км

$V = 1{,}77$ км/с

спут

?

№2

$R = 5$ м

$L = 11{,}78$ м

$\frac{11{,}78}{31{,}4} = \frac{L}{2\pi R}$

Длина части окружности составляет половину отношения $\frac{L}{2\pi R}$

I, II, III, IV

Найдем L по модулю $\frac{2\pi R}{2}$

$11{,}8 - 7{,}8 = 4$

$\frac{4}{7{,}8} \cdot \frac{2}{2} \Rightarrow$ $X_0 = 2$ метра относительно точки старта. Т.к. часть L лежит отрицательно

$X(-R; h) = (-5; h)$

Ответ: 2 метра влево относительно точки старта

2 балла

Командная инженерная олимпиада «Олимпиада НТИ»

Направление АТС

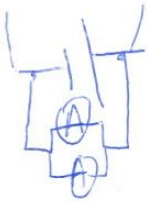
Предмет физика

Номер участника 198

№1

АВ, цилиндр по ходу

№2



3,14 · 2,5

3,14

7,14 5

2

15,56

15,6

1,6

11,6 - 7,8 =

4

Информатика

Задача 1.1

Код программы на языке Python:

```
def rfind(a, elem):
     pos = len(a) - 1
     while a[pos] != elem:
     pos -= 1
     return pos

n, k, m = map(int, input().split())
m -= 1
INF, k1, v1, v2 = 10 ** 18, k, 0, 0
 a = list(map(int, input().split()))
 while k < m:
     v = min(a[:k + 1])
     v1 += v
     a[a.index(v)] = INF
     k += 1
 while n - k1 > m:
     v = min(a[n - k:])
     v2 += v
     a[rfind(a, v)] = INF
     k1 += 1
 print(min(v1, v2) + a[m])
```

Программа верно решает задачу (6 баллов).

Задача 1.2

Код программы на языке Python:

```
def rfind(elem):
     ind = len(a) - 1
     while a[ind] != elem:
     ind -= 1
     return ind

n, k, m = map(int, input().split())
m -= 1
INF, k1, v1, v2 = 10 ** 18, k, 0, 0
 a = list(map(int, input().split()))
 while k < m:
     v = min(a[:k + 1])
     v1 += v
```

```
14          a[a.index(v)] = INF
15          k += 1
16      while n - k1 > m:
17          v = min(a[n - k:])
18          v2 += v
19          a[rfind(v)] = INF
20          k1 += 1
21      print(min(v1, v2) + a[m])
```

Ошибка при выполнении теста №14 (неверный ответ) (0 баллов).

Задача 2.1

Код программы на языке Python:

```
1     n = int(input())
2     a = list(map(int, input().split()))
3     dp = [[0 for i in range(2)] for j in range(n - 1)]
4     dp[0][0] = a[0] ^ a[1]
5     dp[0][1] = 0
6     for i in range(1, n - 1):
7         dp[i][0] = max(dp[i - 1][0], dp[i - 1][1]) +
(a[i] ^ a[i + 1])
8         dp[i][1] = dp[i - 1][0] + (a[i] ^ a[i - 1])
9     for i in dp:
10        print(i)
11    print(max(dp[-1][0], dp[-1][1]))
```

Ошибка при выполнении программы (0 баллов).

Задача 2.3

Код программы на языке Python:

```
1     n = int(input())
2     a = [list(map(int, input().split())) for i in
range(n)]
3     d = 0
4     for i in range(n):
5         for j in range(i + 1, n):
6         d = max(d, (a[i][0] - a[j][0]) ** 2 + (a[i][1] -
a[j][1]) ** 2)
7     print((d ** 0.5 - 1) / 2)
```

Ошибка при выполнении программы (0 баллов).

Командная часть

Результаты были получены в рамках выступления команды: Вжух и в продакшн

Личный состав команды:

- Белавенцев Валерий
- Васильев Валерий
- Литвинов Лев
- Лямаев Михаил
- Серянин Александр

Фото команды с собранными устройствами:

Рис.1 - Команда с четырьмя устройствами в сборке

Рис.2 - Отладка на стенде “Маринет” с устройством, на заднем плане - судьи

Баллы командного этапа:

Название команды	МАРИНЕТ	АВТОНЕТ	АЭРОНЕТ	СПЕЙСНЕТ	КОМПЛЕКС	ИТОГО
Вжух и в продакшн	13	17,5	28	30		88,5

Маринет

Название команды	Участники		№ подзадач	База	Номер попытки										лучшая
					1	2	3	4	5	6	7	8	9	10	
Вжух и в продакшн	Белавенцев	Валерий	1	9	9										9
	Васильев	Валерий	2.	5	4										4
	Литвинов	Лев	3.	5											0
	Лямаев	Михаил	4.	17											0
	Серянин	Александр	5.	5											0
			6.	9											0
	РЕЗУЛЬТАТ														13

Автонет

Название команды	Участники		№ подзадач	База	Номер попытки										лучшая
					1	2	3	4	5	6	7	8	9	10	
Вжух и в продакшн	Белавенцев	Валерий	1	10	0	9,5									9,5
	Васильев	Валерий	2.	2	2										2
	Литвинов	Лев	3.	4	4										4
	Лямаев	Михаил	4.	4											0
	Серянин	Александр	5.	2	2										2
			6.	8											0
			7.	20											0
	РЕЗУЛЬТАТ														17,5

Аэронет

Название команды	Участники		№ подзадач	База	Номер попытки										лучшая
					1	2	3	4	5	6	7	8	9	10	
Вжух и в продакшн	Белавенцев	Валерий	1	10	8										8
	Васильев	Валерий	2.	8	8										8
	Литвинов	Лев	3.	12	12										12
	Лямаев	Михаил	4.	15											0
	Серянин	Александр	5.	10											0
			6.	5											0
	РЕЗУЛЬТАТ														28

Спейснет

Название команды	Участники		№ подзадач	База	Номер попытки										лучшая
					1	2	3	4	5	6	7	8	9	10	
Вжух и в продакшн	Белавенцев	Валерий	1	10	10										10
	Васильев	Валерий	2.	5	5										5
	Литвинов	Лев	3.	15	15										15
	Лямаев	Михаил													0
	Серянин	Александр													0
															0
	РЕЗУЛЬТАТ														30

Комплексный проезд

Не приступали.

Решения командного этапа:

Маринет

Код программы на языке C++, обеспечивающий решение задач подтрека:

.LodkaProg.vsarduino.h

```
#ifndef _VSARDUINO_H_

#define _VSARDUINO_H_

#define __AVR_ATmega32u4__

#define __AVR_ATmega32U4__

#define F_CPU 16000000L
```

```
#define ARDUINO 10608
#define ARDUINO_AVR_MICRO
#define ARDUINO_ARCH_AVR
#define USB_VID 0x2341
#define USB_PID 0x8037
#define __cplusplus 201103L
#define __AVR__
#define __inline__
#define __asm__(x)
#define __extension__
#define __inline__
#define __volatile__
#define GCC_VERSION 40801
#define volatile(va_arg)
#define _CONST
#define __builtin_va_start
#define __builtin_va_end
#define __attribute__(x)
#define NOINLINE __attribute__((noinline))
#define prog_void
#define PGM_VOID_P int
#ifndef __builtin_constant_p
     #define __builtin_constant_p __attribute__((__const__))
#endif
#ifndef __builtin_strlen
     #define __builtin_strlen  __attribute__((__const__))
#endif
```

```
#define NEW_H

typedef void *__builtin_va_list;

extern "C" void __cxa_pure_virtual() {;}




#include <arduino.h>

#include <pins_arduino.h>

#undef F

#define F(string_literal) ((const PROGMEM char
*)(string_literal))

#undef PSTR

#define PSTR(string_literal) ((const PROGMEM char
*)(string_literal))

#include "LodkaProg.ino"

#endif
```

LodkaProg.ino

```
#include <NewPing.h>

#include <Servo.h>


#define MAX_DISTANCE 200

#define OFFSET_IN_STEP 500


#define START 5

#define LED_PIN 13

#define  L_wheel 9

#define  R_wheel 10
```

```
#define GEO_CHANEL "AT+C085"
#define SAT_CHANEL "AT+C025"
#define volt_pin A0
#define HC12_SetupPin 4
#define IR_PIN A4
#define P 20
//********* ***********
#define FFtrig 6
#define FFecho 6
NewPing sonarFF(FFtrig, FFecho, MAX_DISTANCE);

//********* ****** * ****** ****** **********
#define FRtrig  12
#define FRecho  12
#define RRtrig  8
#define RRecho  8
NewPing sonarFR(FRtrig, FRecho, MAX_DISTANCE);
NewPing sonarRR(RRtrig, RRecho, MAX_DISTANCE);

//********* ****** * ****** ****** **********
#define FLtrig  11
#define FLecho  11
#define RLtrig  7
#define RLecho  7
NewPing sonarFL(FLtrig, FLecho, MAX_DISTANCE);
NewPing sonarRL(RLtrig, RLecho, MAX_DISTANCE);
float distFF;
```

```
float distFL;
float distFR;
float distRL;
float distRR;
float LightIR;
int Voltage;
float vLeft;
float vRight;
int offset = 0;

Servo servoLeft, servoRight;
struct GEO {
     int x;
     int y;
     int grad;
};
char strbuf[40];
GEO geo_data;
GEO points[10];
float TwoPointAngle(GEO point1, GEO point2);
// the setup function runs once when you press reset or power
the board
void setup() {
     setPoints();
     servo_attach();
     Serial.begin(9600);
     Serial1.begin(9600);
     Serial1.flush();
```

```
    pinMode(13, OUTPUT);
    pinMode(START, INPUT);//►►►►►►►►►► ►►►►►►
    pinMode(HC12_SetupPin,OUTPUT);
    delay(500);
    set_transmiter_chanel(GEO_CHANEL);
    Voltage = analogRead(volt_pin);
    Serial.print("Voltage:");
    Serial.println(Voltage*10.00 / 1023.00);

    while (!digitalRead(START)){
    if (analogRead(volt_pin) < 720) {
         digitalWrite(LED_PIN,HIGH);
         delay(250);
         digitalWrite(LED_PIN, LOW);
         delay(250);
    }
    else {
         digitalWrite(LED_PIN, LOW);
    }

    read_geo_data();

    sprintf(strbuf, "X:=%d Y:=%d grad:=%d
offset:=%d",geo_data.x,geo_data.y,geo_data.grad, offset);
    Serial.println(strbuf);

    }
    digitalWrite(LED_PIN, HIGH);
```

```
     set_transmiter_chanel(GEO_CHANEL);
     //Serial.println((atan(5/4)*180)/3.14);

}

// the loop function runs over and over again until power down or reset
void loop() {
     read_geo_data();
     setServo(20, 20);
     while (!(geo_data.y + P >= points[0].y && points[0].y >= geo_data.y - P)) {
     read_geo_data();
     }
     setServo(-70, -70);
     delay(900);
     setServo(0, 0);
     if (geo_data.grad - 180 < 0) {
     Right();
     }
     else {

     Left();
     }

     setServo(0, 0);
```

```
     setServo(20, 20);
     while (!(geo_data.x + P >= points[1].x && points[1].x >=
geo_data.x - P)) {
     read_geo_data();
     }
     setServo(-80, -80);
     delay(900);
     setServo(0, 0);

//   SensorScan();
     while (1)
     {

     }

}
void Right()
{
     setServo(-20, 20);
     while (!(geo_data.grad + 30 >= points[1].grad &&
points[1].grad >= geo_data.grad - 30)) {
     read_geo_data();
     }
     setServo(80, -80);
     delay(500);
}
void Left()
{
```

```
     setServo(20, -20);
     while (!(geo_data.grad + 30 >= points[1].grad &&
points[1].grad >= geo_data.grad - 30)) {
     read_geo_data();
     }
     setServo(-80, 80);
     delay(500);
}
void SensorScan()
{

     delay(2);
     distFF = sonarFF.ping_cm();
     Serial.print("FF:");
     Serial.println(distFF);

     delay(2);
     distFR = sonarFR.ping_cm();
     Serial.print("FR:");
     Serial.println(distFR);

     delay(2);
     distFL = sonarFL.ping_cm();
     Serial.print("FL:");
     Serial.println(distFL);
```

```
delay(2);
     distRL = sonarRL.ping_cm();
     Serial.print("RL:");
     Serial.println(distRL);
     delay(2);
     distRR = sonarRR.ping_cm();
     Serial.print("RR:");
     Serial.println(distRR);



     LightIR = analogRead(IR_PIN);
     Serial.print("IR:");
     Serial.println(LightIR);

     Voltage = analogRead(volt_pin);
     Serial.print("Voltage:");
     Serial.print(Voltage);
     Serial.print(" ()");
     Serial.println(Voltage*10.00 / 1023.00);

     Serial.println("---");

}
void setServo(float vLeft, float vRight)
{
     if (vLeft > 80) { vLeft = 80; }
     if (vRight > 80) { vRight = 80; }
```

```
     servoLeft.write(90 - vLeft);
     servoRight.write(90 + vRight);

}
void servo_attach() {

     servoLeft.attach(L_wheel);
     servoRight.attach(R_wheel);
     servo_stop();
}

void servo_stop() {

     servoLeft.write(90);
     servoRight.write(90);
}

void read_geo_data() {
     String str;
     //str = "rer";
     while (1)
     {
     if (Serial1.available() > 0) {
          str = Serial1.readStringUntil('\n');
          //str = "00;1019;0609;192";
          if (str.substring(0, 1)=="22") {
               offset = str.substring(3, 7).toInt();
```

```
          }
          else {
               geo_data.x = str.substring(3, 7).toInt();
               geo_data.y = str.substring(8,
12).toInt()+(offset*OFFSET_IN_STEP);
               geo_data.grad = str.substring(13, 16).toInt();
          }
          Serial.println(str);
     }
     }
}

void set_transmiter_chanel(char* set_chanel) {

     digitalWrite(HC12_SetupPin, LOW);
     delay(50);
     Send_command(set_chanel);
     digitalWrite(HC12_SetupPin, HIGH);

}

void Send_command(char* ATcommand) {
     String str;
     Serial1.println(ATcommand);
     Serial.println(ATcommand);
     delay(200);
     if (Serial1.available() > 0)
     {
```

```
	str = Serial1.readStringUntil('\n');// LF
	Serial.println(str);
	}
	delay(100);
}
float TwoPointAngle(GEO point1,GEO point2) {

	return atanf((point2.y-point1.y)/(point2.x-point1.x));
}
void setPoints() {
	points[0].x = 380;
	points[0].y = 180;
	points[0].grad = 270;

	points[1].x = 1000;
	points[1].y = 100;
	points[1].grad = 0;
}
```

Автонет

Код программы на языке C++, обеспечивающий решение задач подтрека:

.MASHINA.vsarduino

```
/*
	Editor: http://www.visualmicro.com
		visual micro and the arduino ide ignore this code during compilation. this code is automatically maintained by visualmicro, manual changes to this file will be overwritten
		the contents of the Visual Micro sketch sub folder can be deleted prior to publishing a project
		all non-arduino files created by visual micro and
```

```
all visual studio project or solution files can be freely
deleted and are not required to compile a sketch (do not delete
your own code!).

            note: debugger breakpoints are stored in '.sln' or
'.asln' files, knowledge of last uploaded breakpoints is stored
in the upload.vmps.xml file. Both files are required to continue
a previous debug session without needing to compile and upload
again


     Hardware: Arduino Leonardo, Platform=avr, Package=arduino

*/


#ifndef _VSARDUINO_H_

#define _VSARDUINO_H_

#define __AVR_ATmega32u4__

#define __AVR_ATmega32U4__

#define F_CPU 16000000L

#define ARDUINO 10608

#define ARDUINO_AVR_LEONARDO

#define ARDUINO_ARCH_AVR

#define USB_VID 0x2341

#define USB_PID 0x8036

#define __cplusplus 201103L

#define __AVR__

#define __inline__

#define __asm__(x)

#define __extension__

#define __inline__

#define __volatile__
```

```
#define GCC_VERSION 40801
#define volatile(va_arg)
#define _CONST
#define __builtin_va_start
#define __builtin_va_end
#define __attribute__(x)
#define NOINLINE __attribute__((noinline))
#define prog_void
#define PGM_VOID_P int
#ifndef __builtin_constant_p
     #define __builtin_constant_p __attribute__((__const__))
#endif
#ifndef __builtin_strlen
     #define __builtin_strlen  __attribute__((__const__))
#endif
#define NEW_H
typedef void *__builtin_va_list;
extern "C" void __cxa_pure_virtual() {;}



#include <arduino.h>
#include <pins_arduino.h>
#undef F
#define F(string_literal) ((const PROGMEM char
*)(string_literal))
#undef PSTR
#define PSTR(string_literal) ((const PROGMEM char
```

```
*)(string_literal))

#include "MASHINA.ino"

#endif
```

MASHINA.ino

```
// the setup function runs once when you press reset or power
the board


#include <IRremoteInt.h>

#include <IRremote.h>

#include <Servo.h>


#define SIGNAL_FORWARD  0xc00

#define SIGNAL_STOP  0xf00

#define SIGNAL_LEFT  0xa00

#define SIGNAL_RIGHT  0x900

#define MAX_DISTANCE 200

#define OFFSET_IN_STEP 500

#define MOTOR_PIN 5

#define SERVO_PIN 6

#define LINE_PIN A2

#define SONIC_PIN 3

#define IR_PIN A4

#define SAUND_PIN 7

#define LED_PIN A1

#define HC12_SetupPin 2

#define GEO_CHANEL "AT+C056"

#define SAT_CHANEL "AT+C025"
```

```
#define START A0

int offset = 0;

int dist = 0;

IRrecv irrecv(IR_PIN);//пин ик

Servo servo, motor;

//NewPing sonar(SONIC_PIN, SONIC_PIN, MAX_DISTANCE); // NewPing
setup of pins and maximum distance.

//Ultrasonic ultrasonic(SONIC_PIN, 5); // (Trig PIN,Echo PIN)

struct GEO {

     int x;

     int y;

     int grad;

};

char strbuf[40];

GEO geo_data;

decode_results results;

uint32_t ms, ms1 = 0;

bool led_stat = false;

void setup() {

     Serial.begin(9600);

     Serial1.begin(9600);

     irrecv.enableIRIn();

     servo.attach(SERVO_PIN);

     motor.attach(MOTOR_PIN);

     pinMode(START, INPUT_PULLUP);

     digitalWrite(START,HIGH);

     pinMode(HC12_SetupPin, OUTPUT);

     digitalWrite(HC12_SetupPin,HIGH);
```

```
	pinMode(LED_PIN, OUTPUT);
	digitalWrite(LED_PIN, HIGH);
	pinMode(SAUND_PIN, OUTPUT);
	digitalWrite(SAUND_PIN, LOW);
	//digitalWrite(SAUND_PIN, HIGH);
	delay(500);
	set_transmiter_chanel(GEO_CHANEL);
	servo.write(90);
	motor.write(90);
	while (!digitalRead(START)) {
	read_geo_data();
	Serial.println(analogRead(LINE_PIN));

	}

	cheks();
	while (1);
}
// the loop function runs over and over again until power down
or reset
void loop() {
	read_geo_data();

	if (geo_data.grad > 270) {
	servo.write(80);
	}
	if (geo_data.grad < 270) {
	servo.write(100);
```

```
        }


        motorF(110);
        delay(200);
        motorF(100);
}
void IR_read() {
        if (irrecv.decode(&results)) {
        Serial.println(results.value, HEX);
        switch (results.value) {
        case SIGNAL_FORWARD:
                Serial.println("Forward!");
                break;

        case SIGNAL_LEFT:
                Serial.println("Left!");
                break;

        case SIGNAL_RIGHT:
                Serial.println("Right!");
                break;

        case SIGNAL_STOP:
                Serial.println("Stop!");
                break;
```

```
      default:
            Serial.print("Unknown signal: ");
            Serial.print(results.value, HEX);
            Serial.println("!");
      }
      irrecv.resume(); // Receive the next value
      }
}
void motorS() {
      //Serial.print(motor.read());
      if (motor.read() > 90) {
      motor.write(40);
      }
      if (motor.read() < 90)
      {
      //   motor.write(140);
      }
      delay(150);
      motor.write(90);
      delay(150);
}
void motorF(int speed) {
      motor.write(speed);

}
void motorB(int speed) {
      if (motor.read() > 90) {
```

```
      motorS();
      }
      motor.write(speed);
}


void read_geo_data() {
      String str;
      //str = "rer";
      while (1)
      {
      if (Serial1.available() > 0) {
            str = Serial1.readStringUntil('\n');
            //str = "00;1019;0609;192";
            if (str.substring(0, 1) == "22") {
                  offset = str.substring(3, 7).toInt();
            }
            else {
                  geo_data.x = str.substring(3, 7).toInt();
                  geo_data.y = str.substring(8, 12).toInt() +
(offset*OFFSET_IN_STEP);
                  geo_data.grad = str.substring(13, 16).toInt();
            }
            Serial.println(str);
            return;
      }
      }
}
```

```
void set_transmiter_chanel(char* set_chanel) {

     digitalWrite(HC12_SetupPin, LOW);
     delay(50);
     Send_command(set_chanel);
     digitalWrite(HC12_SetupPin, HIGH);

}

void Send_command(char* ATcommand) {
     String str;
     Serial1.println(ATcommand);
     Serial.println(ATcommand);
     delay(200);
     if (Serial1.available() > 0)
     {
     str = Serial1.readStringUntil('\n');// LF
     Serial.println(str);
     }
     delay(100);
}
int Distanse() {
  pinMode(SONIC_PIN, OUTPUT); //Ультразвук

  unsigned int impulseTime = 0;
  digitalWrite(SONIC_PIN, HIGH);
  /* Подаем импульс на вход trig дальномера */
```

```
  delayMicroseconds(10); // равный 10 микросекундам
  digitalWrite(SONIC_PIN, LOW); // Отключаем
     pinMode(SONIC_PIN, INPUT); //Ультразвук
  impulseTime = pulseIn(SONIC_PIN, HIGH); // Замеряем длину
импульса
//Serial.print(impulseTime);
  return  impulseTime / 58; // Пересчитываем в сантиметры

}
void cheks() {

     motorF(105);
     delay(1000);
     motorF(100);
     //Serial.println("F");
     while (analogRead(LINE_PIN) > 500);
     //Serial.println("S");
     //motorS();

     motorF(105);
     delay(500);
     motorF(100);
     while (analogRead(LINE_PIN) > 500)
     {
     ms = millis();
     // Событие срабатывающее каждые 500 мс
     if ((ms - ms1) > 200 || ms < ms1) {
          ms1 = ms;
```

```
            // Инвертируем светодиод
            digitalWrite(LED_PIN, led_stat);
            led_stat = !led_stat;
      }
      }
      digitalWrite(LED_PIN, HIGH);
      motorS();
      digitalWrite(SAUND_PIN, HIGH);
      delay(500);
      digitalWrite(SAUND_PIN, LOW);

}
```

Аэронет

Код программы на языке C++, обеспечивающий решение задач подтрека:

Copter01.ino

```
#include <Servo.h>
#include <Ultrasonic.h>

Servo Roll;
Servo Pitch;
Servo Throt;
Servo Yaw;

Ultrasonic sForward(11, 12);

void setup()
```

```
{
  Serial.begin(9600);
  pinMode(2, INPUT);
  Roll.attach(3, 1000, 2000);
  Pitch.attach(5, 1000, 2000);
  Throt.attach(6, 900, 2000);
  Yaw.attach(9, 1000, 2000);
  delay(6000);

  while(!digitalRead(2))
  {
    Serial.println(sForward.Ranging(CM));
    delay(100);
  }
  Throt.writeMicroseconds(900);
  Roll.writeMicroseconds(1500);
  Pitch.writeMicroseconds(1500);
  Yaw.writeMicroseconds(2000);
  delay(4000);
  Yaw.writeMicroseconds(1900);
  delay(100);
  Yaw.writeMicroseconds(1800);
  delay(100);
  Yaw.writeMicroseconds(1700);
  delay(100);
  Yaw.writeMicroseconds(1600);
  delay(100);
```

```
  Yaw.writeMicroseconds(1500);
  delay(100);

}

void loop()
{
  Throt.writeMicroseconds(1000);
  delay(1000);
  Throt.writeMicroseconds(1100);
  delay(1000);
  Throt.writeMicroseconds(1200);
  delay(1000);
  Throt.writeMicroseconds(1300);
  delay(1000);
  Throt.writeMicroseconds(1400);
  delay(1000);
  Throt.writeMicroseconds(1500);
  delay(2000);
  //Roll.writeMicroseconds(1500);
  //Pitch.writeMicroseconds(1500);
  //Yaw.writeMicroseconds(1500);

}
```

Copter02.ino

```
#include <Servo.h>
#include <NewPing.h>
```

```
Servo Roll;
Servo Pitch;
Servo Throt;
Servo Yaw;

NewPing sForward(4, 4, 100);
NewPing sBack(7, 7, 100);
NewPing sLeft(8, 8, 100);
NewPing sRight(10, 10, 100);
NewPing sDown(11, 11, 100);

void setup()
{
  Serial.begin(9600);
  pinMode(2, INPUT);
  Roll.attach(3, 1000, 2000);
  Pitch.attach(5, 1000, 2000);
  Throt.attach(6, 900, 2000);
  Yaw.attach(9, 1000, 2000);
  delay(5000);

  while(!digitalRead(2))
  {
    delay(10);
  }

  Throt.writeMicroseconds(900);
```

```
  Roll.writeMicroseconds(1500);
  Pitch.writeMicroseconds(1500);
  Yaw.writeMicroseconds(2000);
  delay(4000);
  Yaw.writeMicroseconds(1900);
  delay(100);
  Yaw.writeMicroseconds(1800);
  delay(100);
  Yaw.writeMicroseconds(1700);
  delay(100);
  Yaw.writeMicroseconds(1600);
  delay(100);
  Yaw.writeMicroseconds(1500);
  delay(100);
}

void loop()
{
  Serial.println(sonar.ping_cm());
  delay(100);



}
```

Copter03.ino

```
#include <Servo.h>
#include <NewPing.h>

Servo Roll;
```

```
Servo Pitch;
Servo Throt;
Servo Yaw;

NewPing sForward(4, 4, 100);
NewPing sBack(7, 7, 100);
NewPing sLeft(8, 8, 100);
NewPing sRight(10, 10, 100);
NewPing sDown(11, 11, 100);

int rollVal = 1500;
int pitchVal = 1500;
int throtVal = 1000;
int yawVal = 2000;

void control(int roll, int pitch, int throt, int yaw)
{
  Roll.writeMicroseconds(roll);
  Pitch.writeMicroseconds(pitch);
  Throt.writeMicroseconds(throt);
  Yaw.writeMicroseconds(yaw);
}

void arming()
{
  rollVal = 1500;
  pitchVal = 1500;
```

```
  throtVal = 1000;
  yawVal = 2000;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(5000);
  yawVal = 1900;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1800;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1700;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1600;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1500;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
}

void takeOff()
{
  rollVal = 1500;
  pitchVal = 1500;
  throtVal = 1000;
  yawVal = 1500;
```

```
while(throtVal <= 1300)
{
  throtVal += 50;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
}
delay(1000);

while(throtVal <= 1500)
{
  throtVal += 50;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
}
delay(1000);

while(throtVal <= 1600)
{
  throtVal += 50;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
}
delay(1000);

while(throtVal <= 1750)
{
  throtVal += 50;
```

```
    control(rollVal, pitchVal, throtVal, yawVal);
    delay(100);
  }
  delay(1000);
}

void landing()
{
  rollVal = 1500;
  pitchVal = 1500;
  throtVal = 1500;
  yawVal = 1500;
  while(throtVal >= 1300)
  {
    throtVal -= 50;
    control(rollVal, pitchVal, throtVal, yawVal);
    delay(500);
  }
  delay(1000);
}

void setup()
{
  Serial.begin(9600);
  pinMode(2, INPUT);

  Roll.attach(3, 1000, 2000);
```

```
  Pitch.attach(5, 1000, 2000);
  Throt.attach(6, 1000, 2000);
  Yaw.attach(9, 1000, 2000);
  delay(3000);

  while(!digitalRead(2))
  {
    delay(10);
  }

  arming();
  takeOff();
  delay(1000);
  landing();

}

void loop()
{
  if (!digitalRead(2))
  {
    throtVal = 1000;
    control(rollVal, pitchVal, throtVal, yawVal);
    delay(10);
  }
  if (sDown.ping_cm() <= 5)
  {
```

```
    throtVal = 1000;
    control(rollVal, pitchVal, throtVal, yawVal);
    delay(10);
  }
  Serial.println(sDown.ping_cm());
  delay(100);


}
```

Copter04.ino

```
#include <Servo.h>
#include <NewPing.h>

Servo Roll;
Servo Pitch;
Servo Throt;
Servo Yaw;

NewPing sForward(4, 4, 100);
NewPing sBack(7, 7, 100);
NewPing sLeft(8, 8, 100);
NewPing sRight(10, 10, 100);
NewPing sDown(11, 11, 100);

int rollVal = 1500;
int pitchVal = 1500;
int throtVal = 1000;
int yawVal = 2000;
```

```
#define rollMid 1460
#define pitchMid 1460
#define throtMid 1950
#define yawMid 1500

#define throtMax 1950

#define toWall 50
#define toFlour 70

void control(int roll, int pitch, int throt, int yaw)
{
  Roll.writeMicroseconds(roll);
  Pitch.writeMicroseconds(pitch);
  Throt.writeMicroseconds(throt);
  Yaw.writeMicroseconds(yaw);
}

void arming()
{
  Serial.println("Arming");
  rollVal = rollMid;
  pitchVal = pitchMid;
  throtVal = 1000;
  yawVal = 2000;
  control(rollVal, pitchVal, throtVal, yawVal);
```

```
  delay(5000);
  yawVal = 1900;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1800;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1700;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1600;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1500;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
}

void takeOff()
{
  Serial.println("Take Off start");
  rollVal = rollMid;
  pitchVal = pitchMid;
  throtVal = 1000;
  yawVal = yawMid;
  while(throtVal <= 1300)
  {
```

```
  throtVal += 50;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
}
delay(1000);

while(throtVal <= 1500)
{
  throtVal += 50;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
}
delay(1000);

while(throtVal <= 1600)
{
  throtVal += 50;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
}
delay(1000);

while(throtVal <= 2000)
{
  throtVal += 50;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
```

```
  }
  delay(700);

  while(throtVal > throtMid)
  {
    throtVal -= 50;
    control(rollVal, pitchVal, throtVal, yawVal);
    delay(100);
  }
  delay(1500);
  Serial.println("Take Off end");

}

void landing()
{
  Serial.println("Landing");
  rollVal = rollMid;
  pitchVal = pitchMid;
  throtVal = throtMid;
  yawVal = yawMid;
  while(sDown.ping_cm() >= 5)
  {
    while(throtVal > 1250)
    {
      throtVal -= 50;
      control(rollVal, pitchVal, throtVal, yawVal);
```

```
      delay(500);
    }
    delay(100);
  }

  throtVal = 1200;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(500);
  throtVal = 1100;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(500);
  throtVal = 1000;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(500);
  Serial.println("End");
  while (1)
  {
    delay(100);

  }
}

void setup()
{
  Serial.begin(9600);
  pinMode(2, INPUT);
```

```
  Roll.attach(3, 1000, 2000);
  Pitch.attach(5, 1000, 2000);
  Throt.attach(6, 1000, 2000);
  Yaw.attach(9, 1000, 2000);
  delay(3000);

  while(!digitalRead(2))
  {
    delay(10);
  }

  arming();
  takeOff();
}

int t = 0;
# define delta 75
void loop()
{
  if (!digitalRead(2))
  {
    throtVal = 1000;
  }
  if (t >= 30000)
  {
    landing();
  }
```

```
//ВЫСОТА
if (sDown.ping_cm() <= toFlour - 10)
{
  throtVal += delta;
  throtVal = constrain(throtVal, 1200, throtMax);
}
if (sDown.ping_cm() >= toFlour + 10 or sDown.ping_cm() == 0)
{
  throtVal -= delta;
  throtVal = constrain(throtVal, 1200, throtMax);
}

//ЛЕВО ПРАВО
if (sLeft.ping_cm() <= toWall and sLeft.ping_cm() != 0)
{
  rollVal += delta;
  rollVal = constrain(rollVal, 1000, 2000);
}
if (sRight.ping_cm() <= toWall and sRight.ping_cm() != 0)
{
  rollVal -= delta;
  rollVal = constrain(rollVal, 1000, 2000);
}

//ПЕРЕД ЗАД
if (sForward.ping_cm() <= toWall and sForward.ping_cm() != 0)
```

```
  {
    pitchVal += delta;
    pitchVal = constrain(pitchVal, 1000, 2000);
  }
  if (sBack.ping_cm() <= toWall and sBack.ping_cm() != 0)
  {
    pitchVal -= delta;
    pitchVal = constrain(pitchVal, 1000, 2000);
  }

  control(rollVal, pitchVal, throtVal, yawVal);
  delay(30);
  t += 30;
  Serial.println(t);
  Serial.println(throtVal);



}
```

Copter05.ino

```
#include <Servo.h>
#include <NewPing.h>

Servo Roll;
Servo Pitch;
Servo Throt;
Servo Yaw;

NewPing sForward(4, 4, 100);
```

```
NewPing sBack(7, 7, 100);
NewPing sLeft(8, 8, 100);
NewPing sRight(10, 10, 100);
NewPing sDown(11, 11, 100);

int rollVal = 1500;
int pitchVal = 1500;
int throtVal = 1000;
int yawVal = 2000;

#define rollMid 1460
#define pitchMid 1450
#define throtMid 1950
#define yawMid 1500

#define throtMax 2000

#define toWall 50
#define toFlour 120

void control(int roll, int pitch, int throt, int yaw)
{
  Roll.writeMicroseconds(roll);
  Pitch.writeMicroseconds(pitch);
  Throt.writeMicroseconds(throt);
  Yaw.writeMicroseconds(yaw);
}
```

```
void arming()
{
  Serial.println("Arming");
  rollVal = rollMid;
  pitchVal = pitchMid;
  throtVal = 1000;
  yawVal = 2000;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(5000);
  yawVal = 1900;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1800;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1700;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1600;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
  yawVal = 1500;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(100);
}
```

```
void takeOff()
{
  Serial.println("Take Off start");
  rollVal = rollMid;
  pitchVal = pitchMid;
  throtVal = 1000;
  yawVal = yawMid;
  while(throtVal <= 1300)
  {
    throtVal += 50;
    control(rollVal, pitchVal, throtVal, yawVal);
    delay(100);
  }
  delay(1000);

  while(throtVal <= 1500)
  {
    throtVal += 50;
    control(rollVal, pitchVal, throtVal, yawVal);
    delay(100);
  }
  delay(1000);

  while(throtVal <= 1600)
  {
    throtVal += 50;
    control(rollVal, pitchVal, throtVal, yawVal);
```

```
    delay(100);
  }
  delay(1000);

  while(throtVal <= 2000)
  {
    throtVal += 50;
    control(rollVal, pitchVal, throtVal, yawVal);
    delay(100);
  }
  delay(700);

  while(throtVal > throtMid)
  {
    throtVal -= 50;
    control(rollVal, pitchVal, throtVal, yawVal);
    delay(100);
  }
  delay(1000);
  Serial.println("Take Off end");

}

void forward()
{
  throtVal = 1950;
}
```

```
void landing()
{
  Serial.println("Landing");
  rollVal = rollMid;
  pitchVal = pitchMid;
  throtVal = throtMid;
  yawVal = yawMid;
  while(sDown.ping_cm() >= 5)
  {
    while(throtVal > 1250)
    {
      throtVal -= 50;
      control(rollVal, pitchVal, throtVal, yawVal);
      delay(500);
    }
    delay(100);
  }

  throtVal = 1200;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(500);
  throtVal = 1100;
  control(rollVal, pitchVal, throtVal, yawVal);
  delay(500);
  throtVal = 1000;
  control(rollVal, pitchVal, throtVal, yawVal);
```

```
  delay(500);
  Serial.println("End");
  while (1)
  {
    delay(100);

  }
}

void setup()
{
  Serial.begin(9600);
  pinMode(2, INPUT);

  Roll.attach(3, 1000, 2000);
  Pitch.attach(5, 1000, 2000);
  Throt.attach(6, 1000, 2000);
  Yaw.attach(9, 1000, 2000);
  delay(3000);

  while(!digitalRead(2))
  {
    delay(10);
  }

  arming();
  takeOff();
```

```
}


int t = 0;
# define delta 75
void loop()
{
  if (!digitalRead(2))
  {
    throtVal = 1000;
  }
  if (t >= 5000)
  {
    landing();
  }

  //ВЫСОТА
  if (sDown.ping_cm() <= toFlour - 10)
  {
    throtVal += delta;
    throtVal = constrain(throtVal, 1200, throtMax);
  }
  if (sDown.ping_cm() >= toFlour + 10 or sDown.ping_cm() == 0)
  {
    throtVal -= delta;
    throtVal = constrain(throtVal, 1200, throtMax);
  }
```

```
 /*//ЛЕВО ПРАВО
 if (sLeft.ping_cm() <= toWall and sLeft.ping_cm() != 0)
 {
   rollVal += delta;
   rollVal = constrain(rollVal, 1000, 2000);
 }
 if (sRight.ping_cm() <= toWall and sRight.ping_cm() != 0)
 {
   rollVal -= delta;
   rollVal = constrain(rollVal, 1000, 2000);
 }*/

/* //ПЕРЕД ЗАД
 if (sForward.ping_cm() <= toWall and sForward.ping_cm() != 0)
 {
   pitchVal += delta;
   pitchVal = constrain(pitchVal, 1000, 2000);
 }
 if (sBack.ping_cm() <= toWall and sBack.ping_cm() != 0)
 {
   pitchVal -= delta;
   pitchVal = constrain(pitchVal, 1000, 2000);
 }*/
 pitchVal = 1700;

 control(rollVal, pitchVal, throtVal, yawVal);
 delay(30);
```

```
  t += 30;
  Serial.println(t);
  Serial.println(throtVal);
}
```

Спейснет

Код программы на языке C++, обеспечивающий решение задач подтрека:

SputnikProg.ino

```
#define STEP_PIN 10
#define DIR_PIN 16
#define POWER_ENABLE_PIN 14
#define END_SWITCH1 3
#define END_SWITCH2 2
#define HC12_SetupPin 4
#define LEFT_DIR 0
#define RIGHT_DIR 1
#define GEO_CHANEL "AT+C056"
#define LED1 8
#define LED2 9

#define OFFSET_IN_STEP 100
#define STEP_IN_ROTATION 400
struct GEO {
  int x;
  int y;
  int grad;
};
```

```
GEO geo_data;
const int volt_pin = A0;
float Voltage;
int dir, rotation,offset;
long steps, i;
String str;
  char strbuf[15];
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  Serial1.begin(9600);
  pinMode(STEP_PIN, OUTPUT);
  pinMode(DIR_PIN, OUTPUT);
  pinMode(POWER_ENABLE_PIN, OUTPUT);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(HC12_SetupPin,OUTPUT);
  digitalWrite(POWER_ENABLE_PIN, HIGH);
  digitalWrite(LED1, HIGH);
  digitalWrite(LED2, HIGH);
  delay(1000);
  digitalWrite(LED1, LOW);
  digitalWrite(LED2, LOW);
  Voltage = analogRead(volt_pin);

if(Voltage < 480 ){ // мониторинг напряжения
  while(1){
```

```
  digitalWrite(LED2, HIGH);
  delay(1000);

  digitalWrite(LED2, LOW);
  delay(1000);
  }
  }

  set_transmiter_chanel(GEO_CHANEL);
    move(RIGHT_DIR,10);
    offset = 0;
}

void loop() {
  // put your main code here, to run repeatedly:
 read_geo_data();
 if(geo_data.y < 350 && geo_data.y > 10){
  move(LEFT_DIR,1);
  Serial.println("left");
  offset++;
  }
   if(geo_data.y > 700 && geo_data.y > 10 ){
  move(RIGHT_DIR,1);
  Serial.println("right");
  offset--;
  }
```

```
sprintf(strbuf,"22;%04d",offset);
Serial1.println(strbuf);
Serial.println(strbuf);
}

void move(int dir, unsigned int rotation) {

  digitalWrite(
    POWER_ENABLE_PIN, LOW);
  digitalWrite(DIR_PIN, dir);

  steps = rotation * STEP_IN_ROTATION;

  doSteps(steps);

  digitalWrite(POWER_ENABLE_PIN, HIGH);

}

boolean isEndSwitch() {

  if (digitalRead(END_SWITCH1) == 1) {
    Serial1.println("END_SWITCH1 (Start)");
    rollBack(RIGHT_DIR);
    return true;
  }
```

```
  if (digitalRead(END_SWITCH2) == 1) {
    Serial1.println("END_SWITCH2 (Finish)");
    rollBack(LEFT_DIR);
    return true;
  }

  return false;
}

boolean isStopInSerial() {

  if (Serial1.available() > 0) {
    if (Serial1.readStringUntil('\n').toInt() == 0) {
      return true;
    }
  }

  return false;

}

void rollBack(int dir) {

  digitalWrite(DIR_PIN, dir);
  for (i = 0; i <= 100; i++) {
    doStep();
```

```
  }
}

void doSteps(long steps) {

  for (i = 0; i <= steps; i++) {

    if (isEndSwitch()) {
      return;
    }

    if (isStopInSerial()) {
      return;
    }

    doStep();

  }

  Serial1.println("STOP");

}

void doStep() {

  digitalWrite(STEP_PIN, HIGH);
  delayMicroseconds(500);
```

```
  digitalWrite(STEP_PIN, LOW);
  delayMicroseconds(500);

}
void set_transmiter_chanel(String set_chanel) {

  digitalWrite(HC12_SetupPin, LOW);
  delay(50);
  Send_command(set_chanel);
  digitalWrite(HC12_SetupPin, HIGH);
 delay(100);
}

void Send_command(String ATcommand) {

  Serial1.println (ATcommand);
  Serial.println (ATcommand);
  delay(100);
  if (Serial1.available() > 0)
  {
    str = Serial1.readStringUntil('\n');// LF
    Serial.println (str);
  }
}
GEO read_geo_data() {

while(1){
```

```
    if (Serial1.available() > 0) {

      str = Serial1.readStringUntil('\n');
      //str = "00;1019;0609;192";
      Serial.println(str);

      geo_data.x    = str.substring(3, 7).toInt();
      geo_data.y    = str.substring(8, 12).toInt();
      geo_data.grad = str.substring(13, 16).toInt();

      return geo_data;
    }
  }
}
```