



ЗОЛОТОЕ
СЕЧЕНИЕ

ФОНД ПОДДЕРЖКИ
ТАЛАНТЛИВЫХ ДЕТЕЙ
И МОЛОДЕЖИ

Разбор заданий школьного этапа всероссийской олимпиады школьников по Труд (Технологии), Автоматизированные технические системы для 10-11 класса

2026/2027 учебного года в Свердловской области

Разработчики –
Кощеева Елена Сергеевна, доцент УрГПУ
Омельченко Никита Сергеевич,
магистрант УрГПУ
Портнягин Василий Петрович,
ассистент кафедры ТиММИВО

ВС{ }Ш



Этап 1. Технические условия

Перед проектированием автоматизированной системы необходимо исследовать работу ультразвукового датчика HC-SR04.

Измерить расстояние до плоской поверхности на четырёх фиксированных расстояниях, выполнить по три измерения для каждого расстояния, определить среднее значение и абсолютную погрешность измерений.

Назначение: Измерение расстояния до объекта в диапазоне от 2 до 400 см.

Технические характеристики:

Рабочее напряжение: 5 В

Потребляемый ток: 15 мА

Измеряемое расстояние: 2–400 см

Точность измерения: до 3 мм

Угол измерения: 15°

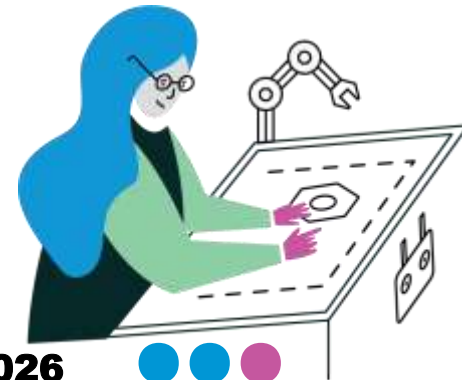
Контакты датчика:

VCC — питание (+5 В)

GND — земля

TRIG — вход для запуска измерения (сигнал от Arduino)

ECHO — выход с измеренным временем (сигнал на Arduino)



Этап 1. Технические условия

Для подключения датчика HC-SR04 используйте электрическую схему, изображенную на рисунке 1.

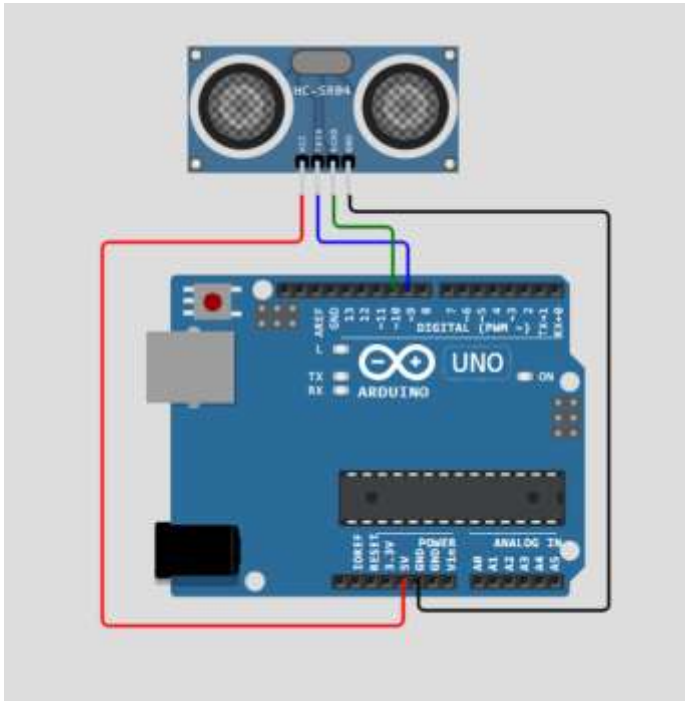


Рисунок 1

Подключение датчика к Arduino:

Контакт HC-SR04	Контакт Arduino	Назначение
VCC	5V	Питание датчика
GND	GND	Общий провод (земля)
TRIG	Pin 9	Запуск измерения
ECHO	Pin 10	Приём сигнала

При сборке схемы важно обеспечить надёжные соединения на монтажной плате. Плохой контакт может стать причиной нестабильных показаний. Также важно убедиться, что датчик расположен строго перпендикулярно поверхности для минимизации погрешности.

Этап 1. Технические условия

Анализ тестовой программы.

Загрузите тестовую программу измерения расстояния.

Переменная	Тип	Назначение
trigPin	const byte	Номер вывода для TRIG — 9
echoPin	const byte	Номер вывода для ECHO — 10
duration	long	Время распространения звука, мкс
distance	float	Вычисленное расстояние, см

Формула $distance = duration * 0.0343 / 2.0$: 0,0343 — это скорость звука в см/мкс (343 м/с = 0,0343 см/мкс). Деление на 2, потому что звук проходит путь до препятствия и обратно.

```
1 // Назначение выводов Arduino
2 const byte trigPin = 9;
3 const byte echoPin = 10;
4 // Переменные
5 long duration;
6 float distance;
7 void setup()
8 {
9     // Настройка выводов
10    pinMode(trigPin, OUTPUT);
11    pinMode(echoPin, INPUT);
12    // Инициализация последовательного порта
13    Serial.begin(9600);
14    Serial.println("Исследование характеристик HC-SR04");
15    Serial.println("-----");
16 }
17 void loop()
18 {
19     // Формирование ультразвукового импульса
20    digitalWrite(trigPin, LOW);
21    delayMicroseconds(2);
22    digitalWrite(trigPin, HIGH);
23    delayMicroseconds(10);
24    digitalWrite(trigPin, LOW);
25    // Измерение длительности отраженного сигнала
26    duration = pulseIn(echoPin, HIGH);
27    // Вычисление расстояния
28    distance = duration * 0.0343 / 2.0;
29    // Вывод результатов
30    Serial.print("Расстояние = ");
31    Serial.print(distance, 1);
32    Serial.println(" см");
33    delay(1000);
34 }
```

Этап 1. Технические условия

Выполнение измерений

Необходимое оборудование:

1. Собранная схема подключения датчика (Рисунок 1);
2. Загруженная тестовая программа;
3. Линейка;
4. Плоская поверхность (стена, книга);
5. Монитор последовательного порта.

Используя линейку, выполните измерение расстояния до плоской поверхности.
Для каждого значения проведите три измерения и определите среднее значение.
Полученные результаты занесите в таблицу.

№	Измеряемое расстояние, см (паспортное величина, устанавливается по линейке)	Измерение 1	Измерение 2	Измерение 3	Среднее значение	Абсолютная погрешность
1	20					
2	40					
3	60					
4	80					

Модуль разности
среднего и паспортного
значений

Среднее
арифметическое
трёх измерений

Показания датчика из монитора последовательного порта

Этап 1. Технические условия

Расчёт средне арифметического значения

Формула: $\bar{x} = \frac{x_1 + x_2 + x_3}{3}$

$\bar{x}$ - среднее значение, см

x_1, x_2, x_3 - результаты трёх измерений, см.

Этап 1. Технические условия

Расчёт абсолютной погрешности

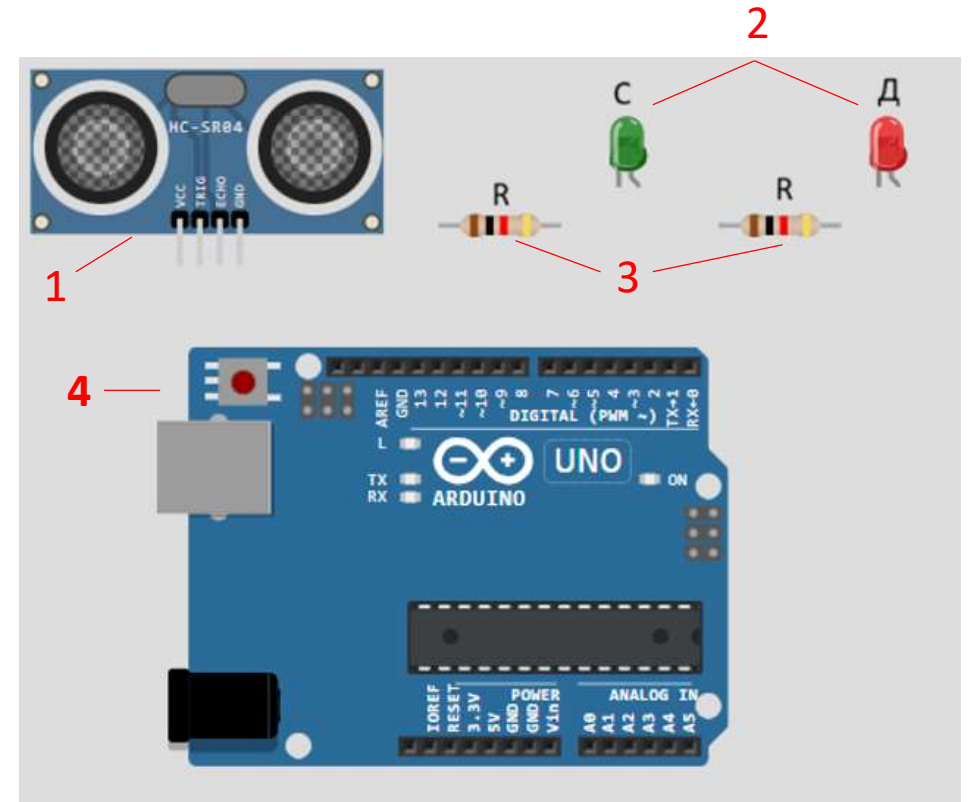
Формула:
$$\Delta(\%) = \frac{\text{измеренная величина} - \text{паспортная величина}}{\text{паспортная величина}} \times 100\%$$

Δ — абсолютная погрешность, см;

Этап 2. Технические условия

Проектирование схемы, включающей ультразвуковой датчик, красный и зелёный светодиоды, ограничивающие резисторы. Расчёт параметров для управления светодиодами в зависимости от расстояния.

1. Ультразвуковой датчик HC-SR04;
2. Красный и зелёный светодиоды, потребляемый ток светодиода: 10 мА;
3. Набор резисторов 220 Ом из стандартного набора Arduino;



Этап 2. Технические условия

Задание 1. Для указанных расстояний из 1 этапа определите у красного светодиода: значение PWM (0–255); среднее напряжение на светодиоде U , В; ток через светодиод I , мА.

Исходные данные для расчёта:

Параметр	Значение
Напряжение питания	5 В
Сопротивление резистора	220 Ом
Падение напряжения на красном светодиоде	2 В

Расчётные расстояния и режимы:

Расстояние	Режим работы
60 см	Зелёный светодиод включён, красный выключен (PWM = 0)
50 см	Яркость красного светодиода плавно увеличивается
40 см	Яркость красного светодиода плавно увеличивается
30 см	Красный светодиод горит максимально ярко (PWM = 255)

Этап 2. Технические условия

Задание 1. Расчёт значения PWM для красного светодиода

При расстоянии 60 см: PWM = 0 (красный светодиод выключен);

При расстоянии 30 см: PWM = 255 (красный светодиод максимально яркий).

Формула линейной интерполяции: $PWM = 255 \times \frac{60-d}{60-30} = 255 \times \frac{60-d}{30}$

где d — расстояние в см.

Расчёт для 50 см: $PWM = 255 \times \frac{60-50}{30} = 255 \times \frac{10}{30} = 255 \times 0,3333 \approx 85$

Расчёт для 40 см: $PWM = 255 \times \frac{60-40}{30} = 255 \times \frac{20}{30} = 255 \times 0,6667 \approx 170$

Задание 1. Расчёт среднего напряжения на выходе Arduino

Формула:

$$U_{Arduino} = \frac{PWM}{255} \times 5 \text{ В}$$

Обозначения:

- $U_{Arduino}$ — среднее напряжение на выходе Arduino, В;
- PWM — значение ШИМ (0–255);
- 5 В — напряжение питания.

Расчёт для каждого расстояния:

Расстояние	PWM	Расчёт	U_Arduino, В
60 см	0	$0/255 \times 5$	0
50 см	85	$85/255 \times 5$	$\approx 1,67$
40 см	170	$170/255 \times 5$	$\approx 3,33$
30 см	255	$255/255 \times 5$	5

Задание 1. Расчёт напряжения на светодиоде

В задании указано: «падение напряжения на красном светодиоде — 2 В».

Это означает, что: $U_{\text{диод}} = 2 \text{ В} = \text{const}$.

Независимо от расстояния и PWM напряжение на светодиоде остаётся постоянным и равным 2 В (при условии, что светодиод включён). Это физическое свойство полупроводникового диода: падение напряжения на нём практически не зависит от тока (в рабочем диапазоне).

Расстояние	PWM	$U_{\text{диод}}$, В
60 см	0	0 (светодиод выключен)
50 см	85	2
40 см	170	2
30 см	255	2

Задание 1. Расчёт тока через светодиод

Формула (по закону Ома):

$$I = \frac{U_{\text{резистор}}}{R} = \frac{U_{\text{Arduino}} - U_{\text{диод}}}{R}$$

Обозначения:

- I — ток через светодиод, А (затем переводим в мА);
- U_{Arduino} — напряжение на выходе Arduino, В;
- $U_{\text{диод}}$ — падение напряжения на светодиоде, В (2 В);
- R — сопротивление резистора, Ом (220 Ом).

Этап 2. Технические условия

Задание 1. Расчёт тока через светодиод для 50 см:

$$I = \frac{1,67 - 2}{220} = \frac{-0,33}{220}$$

При 50 см: напряжение на Arduino (1,67 В) меньше падения напряжения на светодиоде (2 В), поэтому светодиод не загорается при таком напряжении.

Это означает, что при PWM = 85 светодиод не будет гореть, так как напряжение недостаточно для открытия диода.

Этап 2. Технические условия

Задание 1. Расчёт тока через светодиод для 40 см:

$$U_{Arduino} = 3,33 \text{ В}$$

$$I = \frac{3,33-2}{220} = \frac{1,33}{220} = 0,00605 \text{ А} = 6,05 \text{ мА}$$

Расчёт для расстояния 30 см:

$$U_{Arduino} = 5 \text{ В}$$

$$I = \frac{5-2}{220} = \frac{3}{220} = 0,01364 \text{ А} \approx 13,6 \text{ мА}$$

Этап 2. Технические условия

Задание 1. Расчёт тока через светодиод для 40 см:

$$U_{Arduino} = 3,33 \text{ В}$$

$$I = \frac{3,33-2}{220} = \frac{1,33}{220} = 0,00605 \text{ А} = 6,05 \text{ мА}$$

Расчёт для расстояния 30 см:

$$U_{Arduino} = 5 \text{ В}$$

$$I = \frac{5-2}{220} = \frac{3}{220} = 0,01364 \text{ А} \approx 13,6 \text{ мА}$$

Расчёт для расстояния 60 см (красный светодиод
выключен):

$$I = 0 \text{ мА}$$

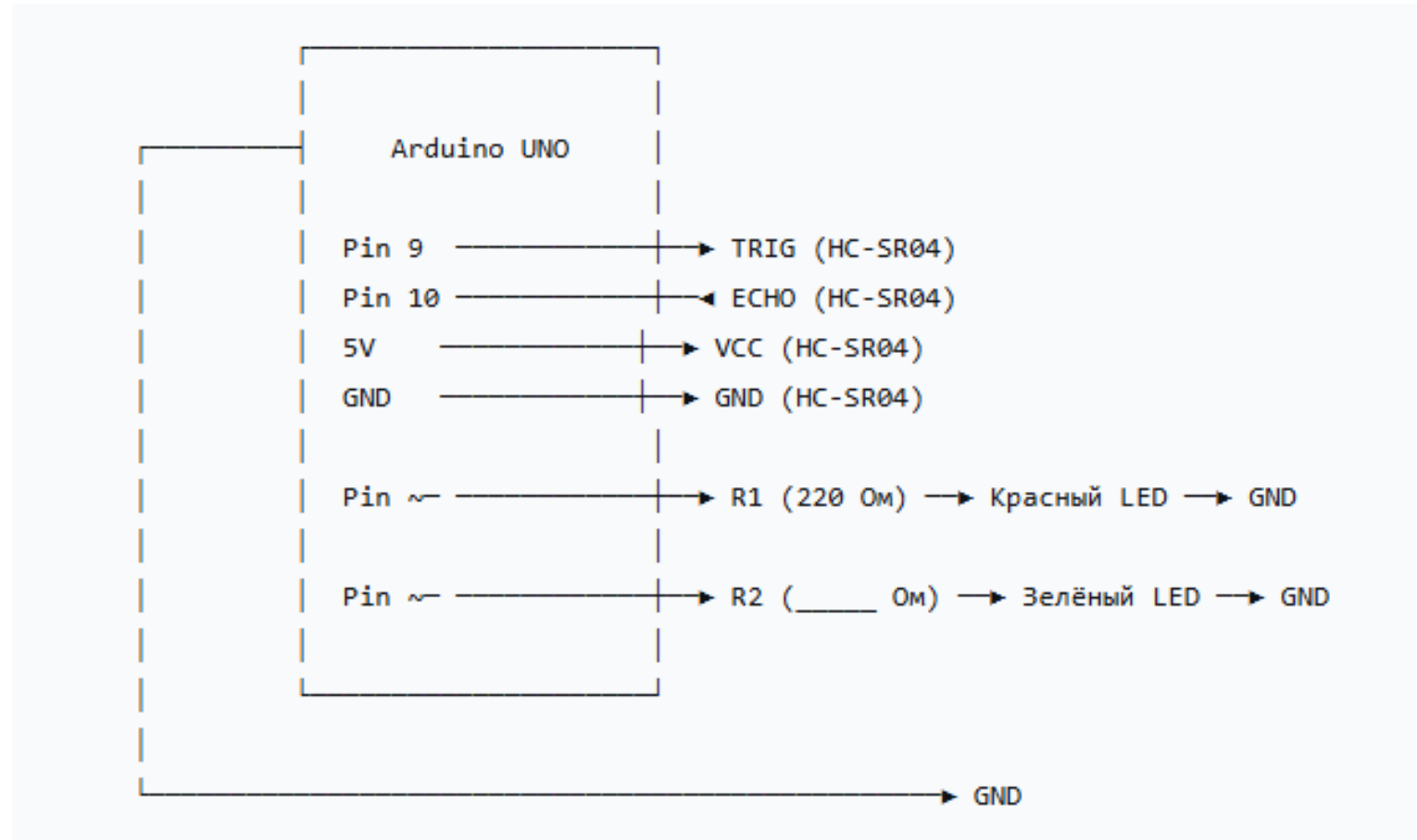
Этап 2. Технические условия

Задание 1. Результаты выполнения задания 1

Расстояние, см	PWM	Среднее напряжение на выходе Arduino, В	Ток через светодиод, мА
60	0	0	0
50	85	~1,67	0*
40	170	~3,33	~6,05
30	255	5	~13,6

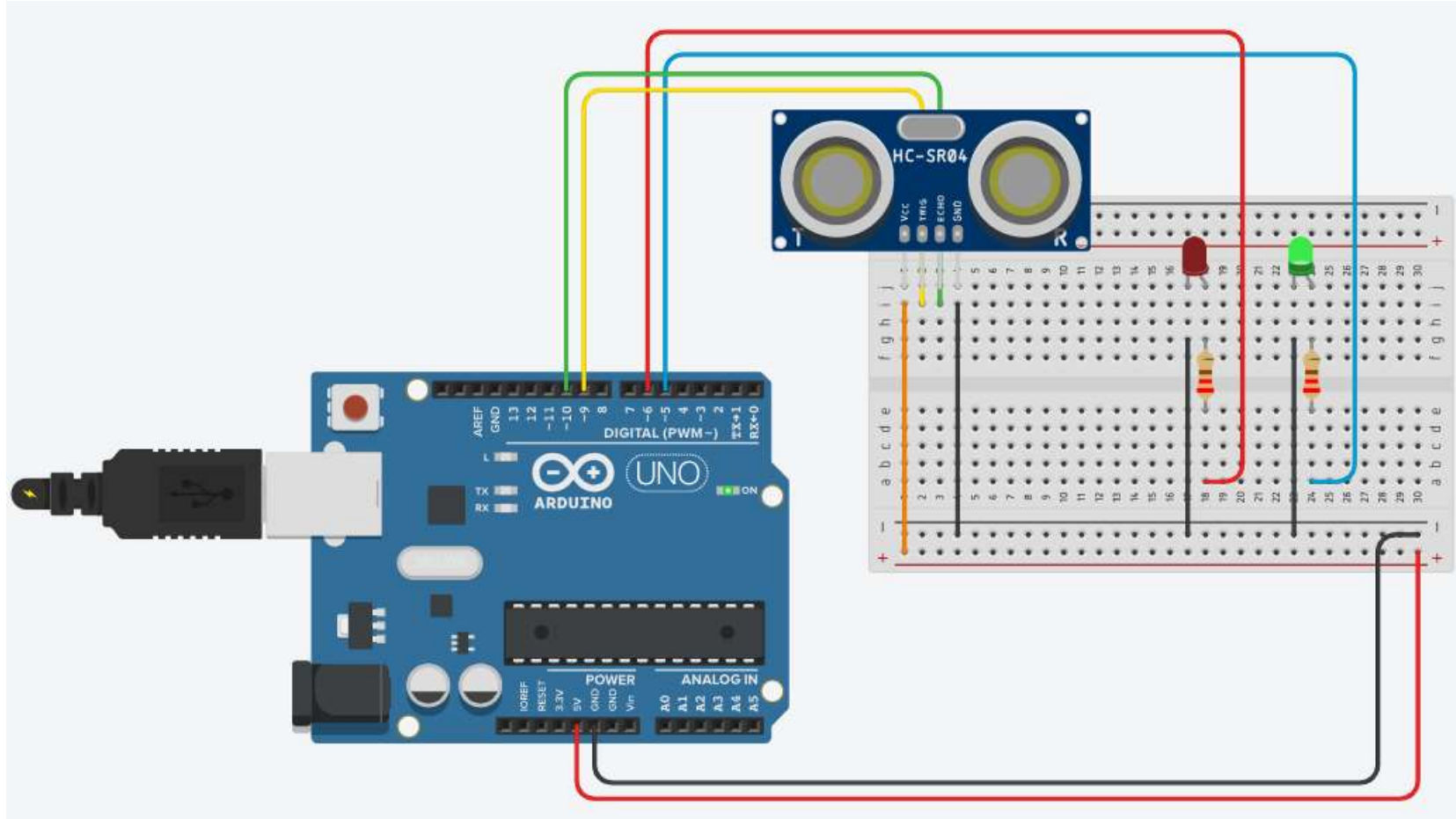
Этап 2. Технические условия

Задание 2. Создание принципиальной схемы



Этап 2. Технические условия

Задание 2. Создание принципиальной схемы



Этап 2. Технические условия

Задание 2. Анализ исходного скетча. Часть 1

Исходный скетч

```
const int trigPin = 10;    // Ошибка: TRIG должен быть отдельный вывод
const int echoPin = 10;    // Ошибка: ECHO не может быть на том же выводе
const int greenLedPin = 10; // Ошибка: вывод уже используется для ECHO
const int redLedPin = 10;   // Ошибка: вывод уже используется для ECHO
```

Необходимые исправления:

```
const int trigPin = 9;      // TRIG – на Pin 9
const int echoPin = 10;     // ECHO – на Pin 10
const int greenLedPin = 4;   // Зелёный LED – на Pin 4
const int redLedPin = 3;     // Красный LED – на Pin 3 (PWM)
```

Этап 2. Технические условия

Задание 2. Анализ исходного скетча. Часть 2

```
void setup() {  
  Serial.begin(9600);  
  pinMode(trigPin, OUTPUT);  
  pinMode(echoPin, INPUT);  
  pinMode(greenLedPin, OUTPUT);  
  pinMode(redLedPin, OUTPUT);  
}
```

функция setup настроена правильно:

Последовательный порт открыт на 9600 бод;

TRIG настроен как выход (OUTPUT) для отправки импульса;

ECHO настроен как вход (INPUT) для приёма сигнала;

Светодиоды настроены как выходы (OUTPUT).

Ошибок в setup нет (при условии, что исправлены объявления пинов).

Задание 2. Анализ исходного скетча. Часть 3

```
void loop() {  
    digitalWrite(trigPin, LOW);  
    delayMicroseconds(2);  
    digitalWrite(trigPin, HIGH);  
    delayMicroseconds(10);  
    digitalWrite(trigPin, LOW);  
  
    duration = pulseIn(echoPin, HIGH);  
    distanceCm = duration * 0.0343 / 2;  
  
    if (distanceCm > 100 || distanceCm <= 0) {  
        distanceCm = 100;  
    }  
    // ... вывод в порт ...  
}
```

Анализ: код формирования импульса и измерения расстояния корректен.

Особенности:

`duration = pulseIn(echoPin, HIGH)` — измеряет длительность высокого уровня на ECHO;

`distanceCm = duration * 0.0343 / 2` — расчёт расстояния в см;

Проверка `if (distanceCm > 100 || distanceCm <= 0)` — защита от некорректных значений.

Задание 2. Анализ исходного скетча. Часть 4

```
if (distanceCm > 60) {  
    analogWrite(redLedPin, 0);  
    digitalWrite(greenLedPin, HIGH);  
}  
else if (distanceCm >= 30 && distanceCm <= 60) {  
    digitalWrite(greenLedPin, LOW);  
    int brightness = map(distanceCm, 60, 30, 50, 255);  
    analogWrite(greenLedPin, 0); // Ошибка: должно быть analogWrite(redLedPin, brightness)  
}  
else {  
    digitalWrite(greenLedPin, LOW);  
    analogWrite(redLedPin, 255);  
}
```

Ошибки в коде управления светодиодами:

Строка	Ошибка	Исправление
<code>analogWrite(greenLedPin, 0)</code>	Управляется зелёный светодиод, а нужно красный	<code>analogWrite(redLedPin, brightness)</code>
Отсутствует обработка границ	При <code>distanceCm</code> точно 60 см — неясно	Использовать <code>></code> и <code><</code>

Задание 2. Исправление управления светодиодами

```
if (distanceCm > 60) {  
    analogWrite(redLedPin, 0);          // Красный выключен  
    digitalWrite(greenLedPin, HIGH);    // Зелёный включён  
}  
else if (distanceCm >= 30 && distanceCm <= 60) {  
    digitalWrite(greenLedPin, LOW);     // Зелёный выключен  
    int brightness = map(distanceCm, 60, 30, 0, 255); // Яркость растёт при уменьшении расстояния  
    analogWrite(redLedPin, brightness); // Красный – с изменяемой яркостью  
}  
else { // distanceCm < 30  
    digitalWrite(greenLedPin, LOW);     // Зелёный выключен  
    analogWrite(redLedPin, 255);        // Красный максимально яркий  
}
```

Объяснение функции map():

- `map(distanceCm, 60, 30, 0, 255)` — преобразует расстояние (60→0, 30→255);
- При 60 см → яркость 0;
- При 30 см → яркость 255;
- Промежуточные значения линейно интерполируются.