Arduino 流水灯程序详解

本实验目标：完成一串LED灯流水闪烁

目标分析：为了完成这个实验，我们首先对于它进行架构的分析

传感器：无
控制器：UNO/ESP32
执行器：LED灯

接着是效果分析：

有五个小灯，它们能够做到依次亮起又熄灭，看起来就像水流一样，灯光逐个点亮并依次向前推进，就像是水流在流动一样。

明确目标系统与效果，开始正式学习---

代码全貌

```
void setup(){
  // 初始化设置区（此处为空）
}

void loop(){
  for (int i = 2; i <= 6; i = i + 1) {
    pinMode(i, OUTPUT);    // 设置引脚模式
    digitalWrite(i, HIGH); // 点亮LED
    delay(200);            // 保持200毫秒
    digitalWrite(i, LOW);  // 熄灭LED
  }
}
```

代码逐行解析

1. `void setup() { }`

- **功能**：Arduino启动时的初始化函数（本程序未使用）
- **存在问题**：
  - 未预先配置引脚模式，违反常规开发规范
  - **建议改进**：

```
void setup() {
  for (int i = 2; i <= 6; i++) {
    pinMode(i, OUTPUT); // 提前配置引脚模式
  }
}
```

2. `void loop() { ... }`

- **功能**：主循环体，持续重复执行

`for`循环结构解析

代码结构：

```
for (int i = 2; i <= 6; i = i + 1) {
  // 循环体
}
```

- **循环三要素**：

要素	说明	本程序参数
初始化	`int i = 2`	从引脚2开始
循环条件	`i <= 6`	包含引脚6
迭代操作	`i = i + 1`（等效i++）	每次增加1

- **执行流程**：

循环次数	i值	操作引脚	执行内容
1	2	引脚2	亮→延时→灭
2	3	引脚3	亮→延时→灭
3	4	引脚4	亮→延时→灭
4	5	引脚5	亮→延时→灭
5	6	引脚6	亮→延时→灭

循环体详解

行1： `pinMode(i, OUTPUT);`

- **问题**：在循环体内重复设置引脚模式（应仅在setup中设置一次）
- **作用**：将当前引脚配置为输出模式（控制LED）

行2： `digitalWrite(i, HIGH);`

- **功能**：向指定引脚输出高电平（5V）
- **效果**：点亮连接的LED

行3： `delay(200);`

- **参数**：200毫秒（0.2秒）
- **作用**：保持当前LED点亮状态

行4： `digitalWrite(i, LOW);`

- **功能**：向指定引脚输出低电平（0V）

- **效果**：熄灭LED

核心概念详解

1. 数字信号特性

参数	说明
电平范围	HIGH（5V），LOW（0V）
信号类型	离散信号（非连续变化）
控制对象	LED、继电器等开关型设备

2. 引脚模式配置原则

模式	配置时机	推荐做法
OUTPUT	setup()函数	提前统一配置
INPUT	setup()函数	根据传感器类型选择配置方式

3. 循环结构分析

- **循环变量i的变化**：

```
初始值 → 2
终止条件 → i <= 6
变化步长 → +1
```

- **执行次数计算**：

```
6 - 2 + 1 = 5次
```

- **引脚操作顺序**：

```
2 → 3 → 4 → 5 → 6（依次点亮）
```

硬件连接指南

标准流水灯电路

```
引脚2 → 220Ω电阻 → LED1正极 → LED1负极 → GND
引脚3 → 220Ω电阻 → LED2正极 → LED2负极 → GND
...
引脚6 → 220Ω电阻 → LED5正极 → LED5负极 → GND
```

优化方案（LED共阴极）

```
Arduino引脚 → LED阵列 →一个共同的电阻 →GND
优点：共阴控制起来和接线更加方便
```

关键问题解析

Q1：LED显示异常（多个同时亮）

- **原因排查**：
  1. 检查是否存在引脚短路
  2. 确认每个LED独立连接不同引脚
  3. 验证程序逻辑是否正确（应逐一点亮）

Q2：流水速度调整

- **修改参数**：

```
delay(100);  // 加快流动（100ms）
delay(500);  // 减慢流动（500ms）
```

Q3：如何实现逆向流动

- **修改循环参数**：

```
for (int i = 6; i >= 2; i--) {
  // 操作代码
}
```

扩展实验

实验1：双向流水灯

```
void loop(){
  // 正向流动
  for (int i = 2; i <= 6; i++) {
    digitalWrite(i, HIGH);
    delay(200);
    digitalWrite(i, LOW);
  }
  // 逆向流动
  for (int i = 6; i >= 2; i--) {
    digitalWrite(i, HIGH);
    delay(200);
    digitalWrite(i, LOW);
  }
}
```

实验2：跑马灯效果

```
void loop(){
  for (int i = 2; i <= 6; i++) {
    digitalWrite(i, HIGH);
    if(i > 2) digitalWrite(i-1, LOW);
    delay(200);
  }
  digitalWrite(6, LOW);
}
```

实验3：速度渐变效果

```
void loop(){
  int baseDelay = 100;
  for (int i = 2; i <= 6; i++) {
    digitalWrite(i, HIGH);
    delay(baseDelay + i*20); // 速度逐渐变慢
    digitalWrite(i, LOW);
  }
}
```

程序改进建议

1. 标准写法修正

```
void setup(){
  for(int i=2; i<=6; i++){
    pinMode(i, OUTPUT);
  }
}

void loop(){
  for(int i=2; i<=6; i++){
    digitalWrite(i, HIGH);
    delay(200);
    digitalWrite(i, LOW);
  }
}
```

2. 添加状态保持

```
void loop(){
  for(int i=2; i<=6; i++){
    digitalWrite(i, HIGH);
    delay(200);
  }
  for(int i=2; i<=6; i++){
    digitalWrite(i, LOW);
    delay(200);
  }
}
```

总结

本程序通过for循环实现了基本的流水灯效果，需重点掌握：

1. **循环结构**：理解初始化、条件判断、迭代操作的配合
2. **引脚操作规范**：正确配置和使用数字输出引脚
3. **时序控制**：通过delay函数调节流水速度
4. **硬件知识**：掌握多LED连接方法和限流电阻作用

通过修改循环参数和添加控制逻辑，可以扩展出多种灯光效果。

有问题找柯萌ㄑ・ᴥ・ʖ
mingyoufhh@outlook.com