SunFounder

お選びいただき、誠にありがとうございます

Inventor Lab Kit

Inventor Lab キットでエレクトロニクスの旅を始めましょう。本キットは、学習者や愛好者向けに設計されており、Arduino Uno R3 を中心に、LED やブザーといった基本コンポーネントから、RFID システムや超音波センサーなどの高度なモジュールまで、回路分析用のミニマルチメーターも含まれています。

このキットは、教育的な明瞭さに優れ、初心者にプログラミングと回路設計を段階的に導入します。あらかじめ書かれたコードではなく、独自のスクリプトを書く手順を一歩ずつガイドすることで、理解を深め、記憶に定着させます。プロジェクトは、LED の点灯やジョイスティックの使用といった簡単な作業から、レーダーシステムや自動石鹸ディスペンサーの構築といった複雑なアプリケーションまで幅広くカバーしています。

初心者からスキルを広げたい方まで、あらゆるレベルに適しており、Inventor Lab キットは学習を手軽かつ魅力的なものにし、エレクトロニクスの世界を探求し革新するために必要なツールをすべて提供します。

お問い合わせやサポートが必要な場合は、service@sunfounder.com までご連絡ください。Beginner's Lab Kit と共に学びの旅に飛び込み、エレクトロニクスのエキサイティングな世界で構築、コーディング、探求を始めましょう!

SunFounder

目次

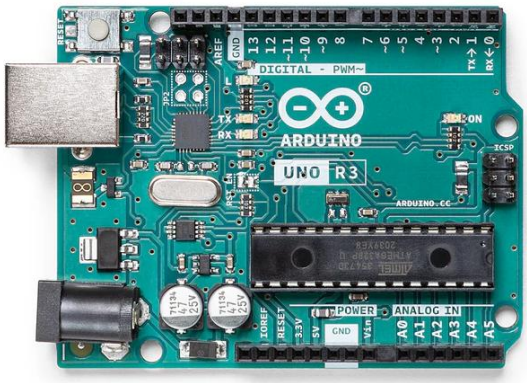
キットに含まれているもの

このキットには、コースを通して回路を組み立てる際に使用するさまざまなコンポーネントや部品が含まれています。以下は、その内容の簡単なガイドです。

基本コンポーネント

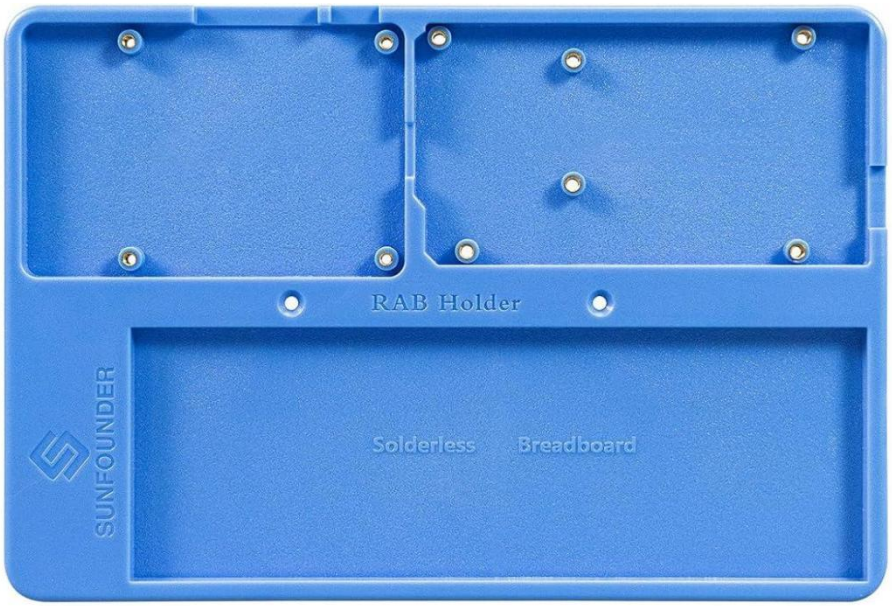
1 x オリジナル Arduino Uno R3

マイクロコントローラーボードは、回路の「頭脳」です。マイクロコントローラーをサポートするために必要なすべてのものが揃っており、USB ケーブルでコンピューターに接続するか、AC-DC アダプターやバッテリーで電源を供給するだけで始められます。



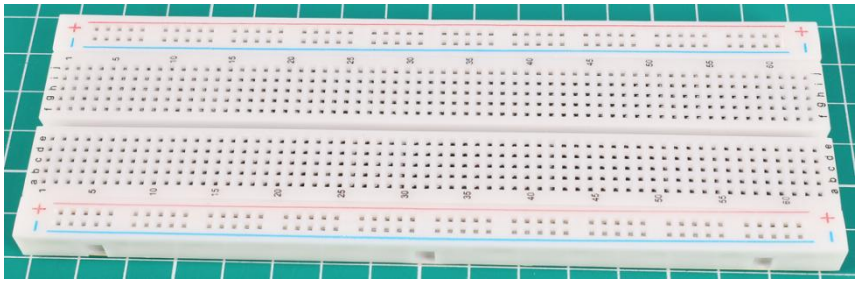
1 x RAB ホルダー

Arduino および Raspberry Pi ユーザーの両方に対応しており、すべてのデバイスの安定性を確保します。単なる保持装置ではなく、誰にでも適した革新のためのツールです。



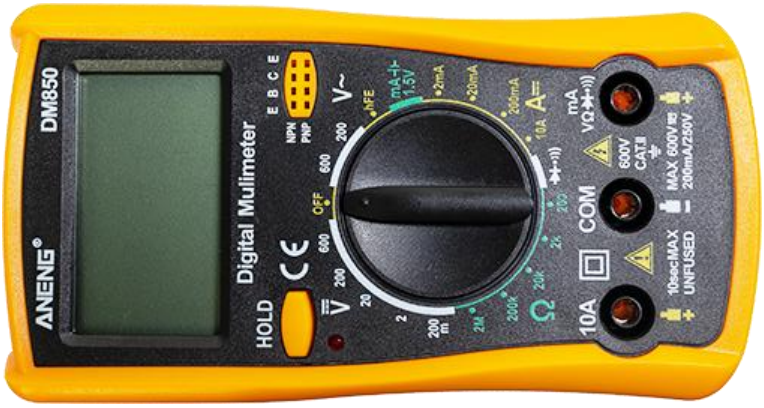
1 x 830-穴ブレッドボード

ハンダ付け不要で簡単に電子回路を組み立てることができるボードです。ワイヤーやコンポーネントを接続するための穴が列状に配置されています。



1 x マルチメーター（赤・黒リード付き）

この多機能マルチメーターは、電圧、電流、抵抗を測定できるだけでなく、その他の電気テストも行えるため、エレクトロニクスや電気作業に欠かせないツールです。

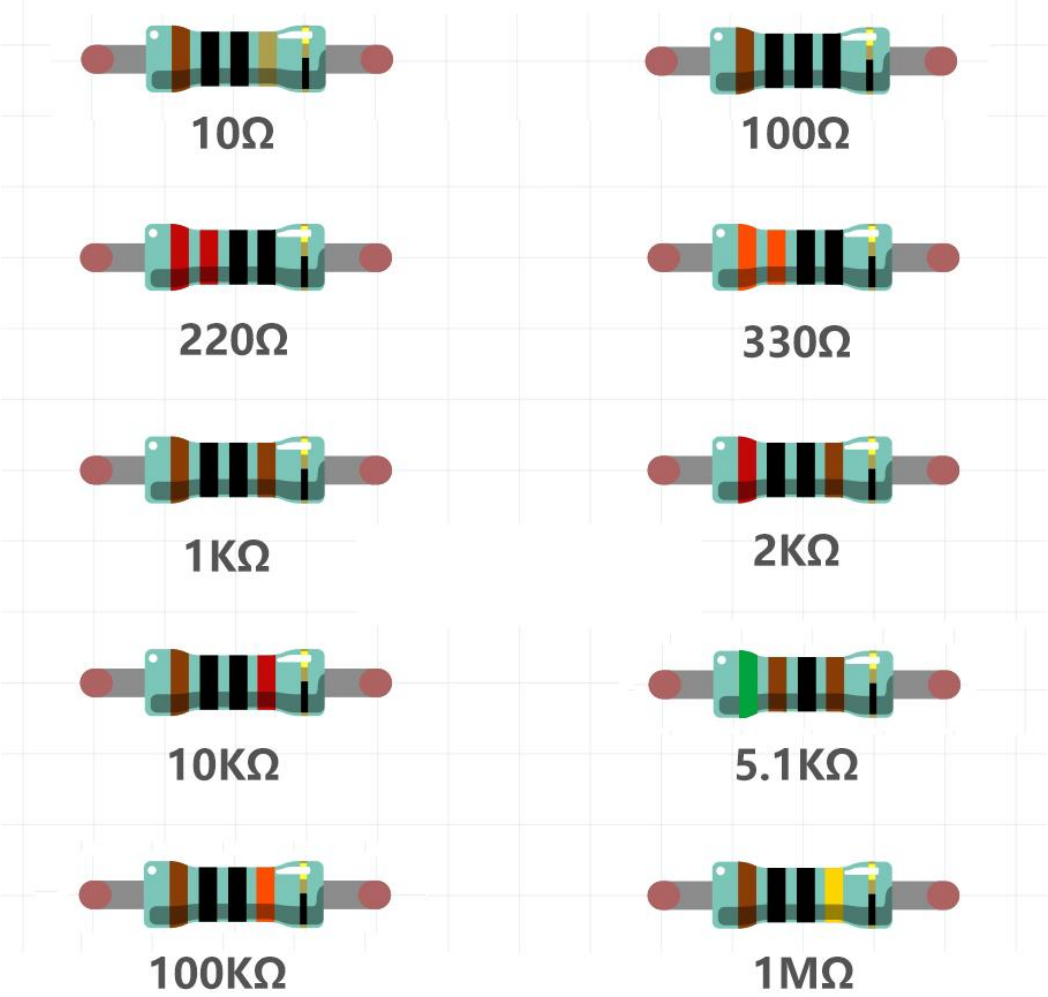


1 x ブレッドボード用電源モジュール

ブレッドボード用電源モジュールは、プロトタイプ作成時に便利なアクセサリーで、DC アダプターや USB から安定した 3.3V または 5V の電源を供給します。標準的なブレッドボードに対応し、電源レールに差し込むだけで使用可能です。オン/オフスイッチと電圧レギュレーターが付属しており、安定した出力を提供するため、エレクトロニクスプロジェクトには欠かせないアイテムです。

120 x 抵抗器（各 10 本、220Ω抵抗器は 30 本）

抵抗器は、電流の流れを妨げ、回路内の電圧と電流を調整するコンポーネントです。抵抗器の値はオーム（Ω）で測定され、ギリシャ文字のオメガ（Ω）で表されます。抵抗器に描かれた色の帯は、その抵抗値と許容誤差を示しています。



2 x 9V バッテリー

これは、充電できないアルカリ 9V バッテリーです。マルチメーターに装着するか、バッテリーケーブルを使用して Arduino Uno R3 またはブレッドボード用電源モジュールに電力を供給する際に使用します。

65 x ジャンパーワイヤー

ブレッドボード上のコンポーネント同士や、Arduino ボードとの接続に使用します。

20 x オス-メス DuPont ワイヤー

オス-メスの DuPont ワイヤーは、超音波モジュールのようなオスピンヘッダーを持つモジュールをブレッドボードに接続するために特別に設計されています。これらのワイヤーは、電子プロジェクトでブレッドボード対応のオス-メス接続が必要な場合に、さまざまなコンポーネントを接続するために欠かせないものです。

1 x USB ケーブル

Arduino ボードをコンピュータに接続します。これにより、プログラムの作成、コンパイル、および Arduino ボードへの転送が可能になります。また、ボードへの電源供給も行います。

1 x バッテリーケーブル

このケーブルは、9V バッテリーをブレッドボード用電源モジュールや Arduino Uno R3 の DC 入力に接続し、電子プロジェクトに便利で持ち運び可能な電源を提供します。

ディスプレイ

25 x LEDs（各色 5 個）

このカラフルな LED セットには、赤、緑、青、黄色、白の 5 色が含まれており、さまざまな照明や信号用途に対応しています。シンプルなステータスインジケーターから複雑な装飾照明プロジェクトまで、これらの LED は、電子プロジェクトの視覚的な魅力を高める豊かな色の選択肢を提供します。

1 x RGB LED

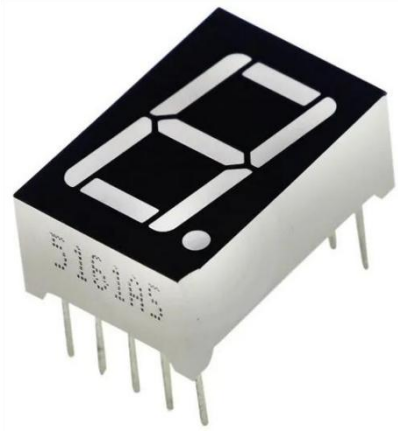
赤、緑、青の LED を 1 つのケースに組み合わせたものです。入力電圧を調整することで、さまざまな色を表示でき、数百万色の表現が可能です。

1 x 74HC595 チップ

74HC595 は、シフトレジスタであり、シリアル入力をパラレル出力に変換することで、デジタル回路の入出力ポートを拡張し、必要な接続ピンの数を減らします。このチップは、7 セグメントディスプレイなど、多数の出力デバイスを制御する際に、マイクロコントローラのピンをあまり占有せずに使用するのに適しています。

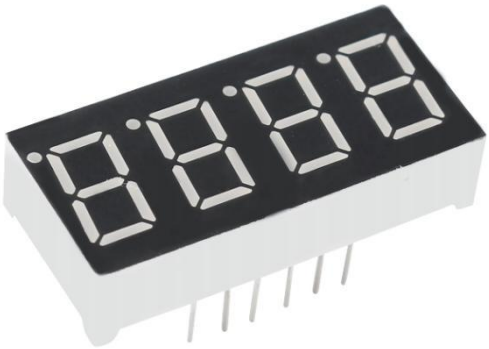
1 x 7 セグメントディスプレイ（共通カソード）

7 セグメントディスプレイは、7 つの LED をパッケージ化した 8 の字型のコンポーネントです。各 LED は「セグメント」と呼ばれ、通電すると、1 つのセグメントが表示する数字の一部を形成します。



1 x 4 桁 7 セグメントディスプレイ（共通カソード）

4 桁ディスプレイは、4 つの 7 セグメントディスプレイを組み合わせたもので、それぞれが 1 桁を表示します。必要なピン数を減らすために、各ディスプレイのセグメントは多重化されており、各セグメントピンは他のディスプレイの対応するセグメントピンと接続されています。



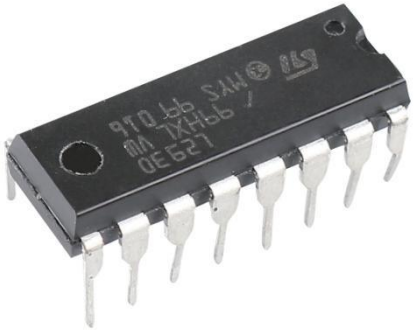
1 x I2C LCD1602

I2C LCD1602 は、I2C 通信プロトコルを使用する 16x2 文字表示モジュールです。このモジュールは、センサーデータやステータスメッセージなどのテキスト表示に最適です。

アクチュエーター

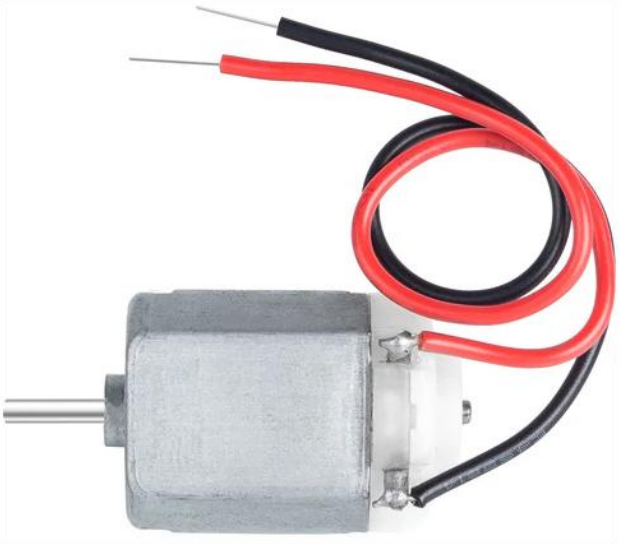
1 x L293D チップ

L293D は、2 つの DC モーターの速度と方向を同時に制御できるデュアル H ブリッジモータードライバ IC です。ロボティクスや自動化プロジェクトに最適で、信頼性が高く効率的なモーター制御を提供します。



1 x モーター

この 3V モーターは、低電圧アプリケーション向けに設計されたコンパクトで効率的な DC モーターです。小型の電子プロジェクト、玩具、ホビーロボティクスに最適で、低消費電力で信頼性の高いパフォーマンスを提供します。



1 x ポンプ

これは、DC 2.5-6V の小型水中ポンプで、卓上噴水、水槽、ハイドロポニックシステムなどの小規模プロジェクトに最適です。このポンプは、電動モーターを使用して回転エネルギーを流体力学的エネルギーに変換し、効率的に水を移動させる遠心機構を採用しています。

1 x サーボ

サーボは、角度や直線位置、速度、加速度の精密かつ多用途な制御を行うためのモーターです。ロボティクス、自動化、リモートコントロールシステムなどで広く使用され、さまざまな用途において信頼性が高くスムーズな動きを実現します。

1 x チューブ

これは、水中ポンプの出口から水を導くために使用される、長さ 20cm、直径 6mm の透明チューブです。

1 x ステッピングモーター

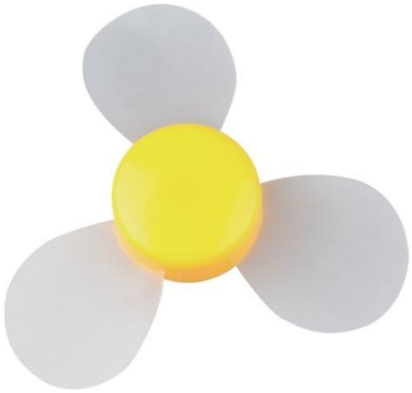
28BYJ-48 は、5V で動作する 5 線式ユニポーラステッピングモーターです。ロボティクスや 3D プリンター、自動化プロジェクトなど、回転を精密に制御する必要があるアプリケーションに最適です。

1 x ULN2003 モジュール

ULN2003 モジュールは、高電圧・高電流対応のダーリントントランジスタアレイで、ステッピングモーター、リレー、その他の誘導負荷を駆動するために使用されます。7 つのオープンコレクタダーリントンペアを備えており、さまざまな制御アプリケーションで TTL や CMOS ロジックレベルとインターフェースするのに最適です。

1 x 3 枚羽根ファンブレード

柔らかい 3 枚羽根のファンブレードは、3V モーターと併用するために設計された柔軟で安全なファンアクセサリーです。耐久性のある柔らかい素材で作られており、怪我のリスクを最小限に抑えます。



サウンド

1 x アクティブブザー & 1 x パッシブブザー

ブザーは、アクティブタイプとパッシブタイプがあり、電流を流すことで音を発する音響信号装置です。アラームやタイマー、通知システムで一般的に使用されます。

センサー

1 x フォトレジスタ

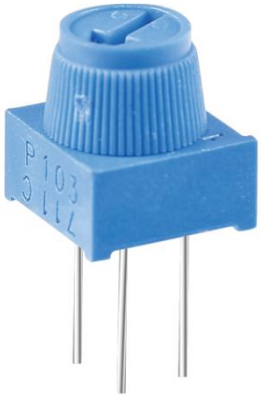
フォトレジスタは、光の強度に応じて抵抗値が変化する感光性コンポーネントで、光に反応する制御装置やセンサーを作成するのに最適です。

1 x NTC サーミスタ

サーミスタは、温度変化に敏感な抵抗器です。NTC サーミスタは、温度が上昇すると抵抗が減少し、PTC サーミスタは温度が上昇すると抵抗が増加します。

1 x ポテンショメータ

ポテンショメータは、3 つのピンを持つ可変抵抗器です。2 つのピンは抵抗器の両端に接続され、中央のピンは可動式のワイパーに接続され、抵抗器を 2 つの部分に分けます。ポテンショメータは、回路内の電圧を調整するためによく使用され、ラジオの音量調節つまみのような役割を果たします。



10 x 小型ボタン

小型プッシュボタンは、押されたときに物理的な反応を提供するもので、電子機器でアクションを開始したりコマンドを入力したりする際によく使用されます。

1 x 超音波モジュール

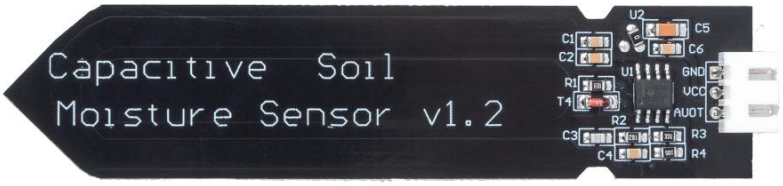
この超音波モジュールは、超音波を利用して距離を測定し、物体の位置や距離を正確に検出・測定します。ロボティクス、障害物回避システム、自動制御分野で広く使用されており、環境認識や空間ナビゲーションに欠かせない重要なコンポーネントです。

1 x ジョイスティックモジュール

ジョイスティックモジュール（ジョイスティックセンサーとも呼ばれる）は、ノブの水平 (X 軸) および垂直 (Y 軸) の 2 方向の動きを測定する入力デバイスです。

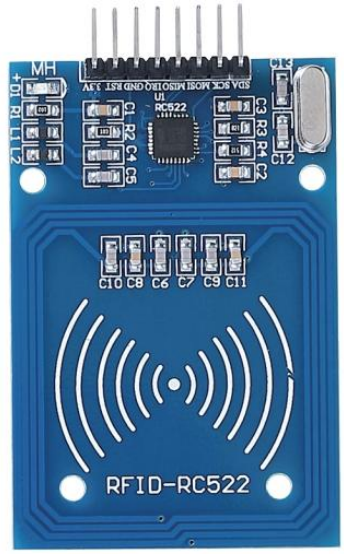
1 x 土壌湿度モジュール

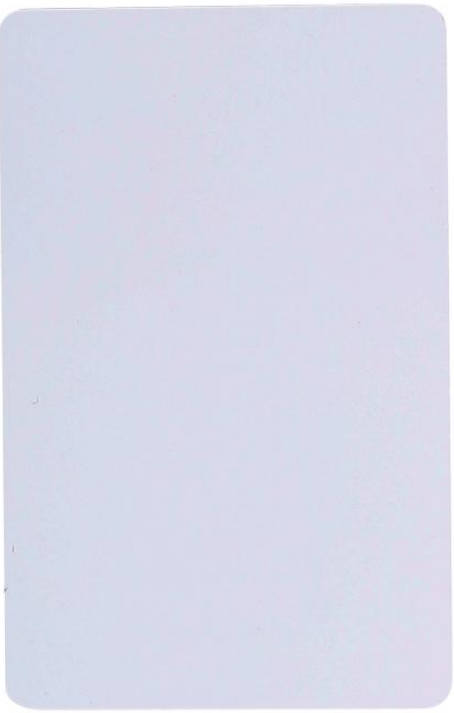
土壌の湿度を検出するための容量性センサーで、耐腐食性があり、3.3V ~ 5.5V で動作します。湿度値を出力し、土壌が湿っているほどアナログ値が小さくなります。



1 x RC522-RFID モジュール（タグとホワイトカード付き）

RC522 RFID リーダーモジュールは、13.56MHz の周波数で動作し、ISO 14443A 規格に準拠した RFID タグと通信するために設計されています。このコンパクトで多用途なデバイスは、4 ピンの SPI 接続を介してマイクロコントローラとインターフェースし、最大 10 Mbps のデータレートをサポートするため、アクセス制御、在庫管理、非接触決済システムなどに最適です。

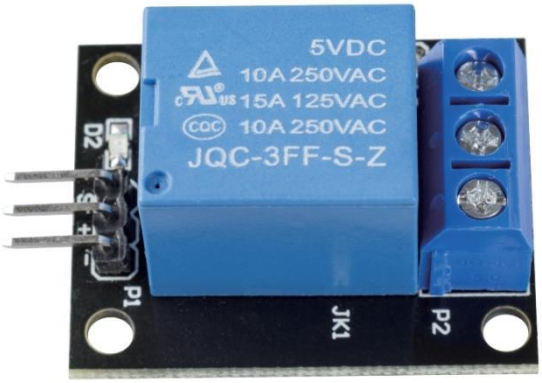




その他

1 x リレーモジュール

リレーモジュールは、マイクロコントローラーが高電圧デバイスを制御するための電気的に絶縁されたスイッチを提供します。低電圧のデジタル信号から AC や大電流負荷を制御する必要があるアプリケーションに最適です。



1 x 赤外線受信モジュール

SL838 赤外線受信モジュールは、赤外線信号を受信し、TTL レベルに対応した信号を独立して出力できるコンポーネントです。通常のプラスチックパッケージトランジスタと同程度のサイズで、さまざまな赤外線リモコンや赤外線通信に適しています。

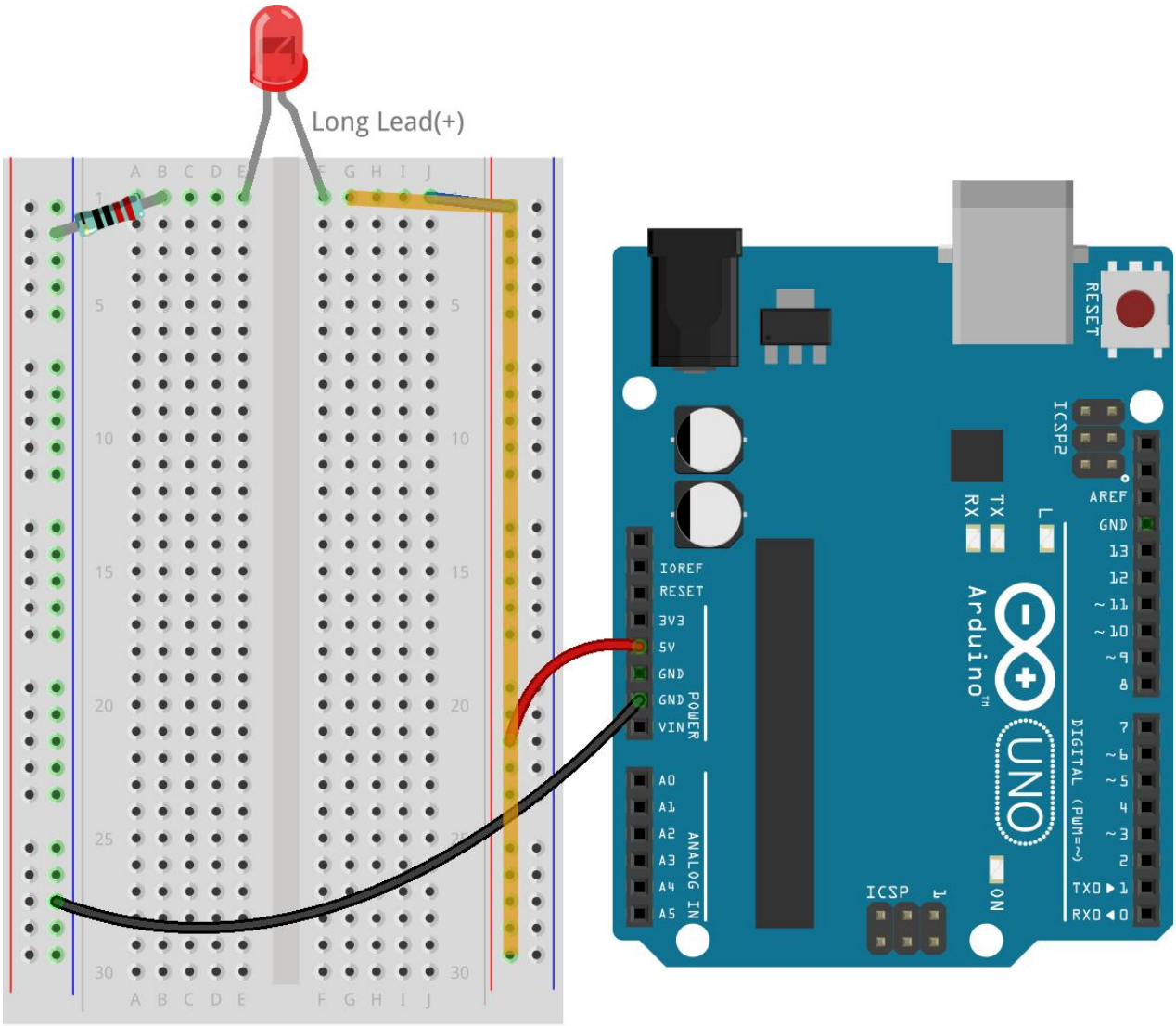
1 x リモコン

この 21 キーリモコンは、85x39x6mm のコンパクトサイズで、8 メートルの範囲を持ち、3V リチウム電池で動作します。38KHz の赤外線周波数と耐久性のある PET 表面を備え、20,000 回以上の使用が可能なため、さまざまなデバイスに最適です。

レッスン 2: 初めての回路

レッスンを完了した後、次の質問に答えてください。

1. ブレッドボードから赤いワイヤーを取り外し、別の穴に挿して実験してください。LED に変化があるか観察しましょう。LED が点灯する穴の位置をスケッチしてください。



1. LED のピンを逆にするとどうなりますか？点灯しますか？その理由は?

LED は点灯しません。なぜなら、LED は一方向の導電性を持っており、電流がアノードからカソードへ流れないと動作しないからです。

L レッスン 3: マルチメーターで測定する

「**マルチメーターについてもっと知ろう！**」を完了した後、この質問に答えてください。

マルチメーターの使用方法について詳しく理解したところで、次の電気的な値を測定する際にどのマルチメーター設定を使用するか考えてみてください。

測定対象	マルチメーター設定
9V volts DC	*20V*
1K ohms	*2kΩ*
40 milliamps	*200mA*
110 volts AC	*200V~*

「**マルチメーターで測定する**」間にこの表を記入してください。

種類	単位	測定結果	備考
Voltage	Volts	*≈5.13 volts*	
Current	Milliamps	*≈13.54 milliamps*	
Resistance	Ohms	*≈378.88 ohms*	

レッスン 4：オームの法則

「**オームの法則を実験で探る**」間に、次の表を記入してください。

1. 220Ωの抵抗を、以下に記載されている他の異なる値の抵抗に交換してください。それぞれの交換時に LED の明るさの変化を記録し、抵抗が電流に与える影響、ひいては光の出力にどのように影響するか観察しましょう。

抵抗値	観察結果
100Ω	*より明るい*
1KΩ	*明るい*
10KΩ	*暗い*
1MΩ	*ほとんど消えている*

100Ωの抵抗を使用した場合のみ、以前の 220Ωの抵抗よりも LED が明るくなることに気づくでしょう。抵抗値が高くなると、LED の明るさは次第に減少し、1MΩでは完全に消えてしまいます。これはなぜでしょうか?

オームの法則（I = V/R）によると、電圧が一定のまま抵抗が増加すると、LED を流れる電流が減少し、結果として LED が暗くなります。1MΩでは、電流が非常に小さくなり、LED が点灯しなくなります。

2. 抵抗を 220 オームに保ったまま、回路の電圧供給を 5V から 3.3V に変更し、LED の明るさの変化を記録してください。

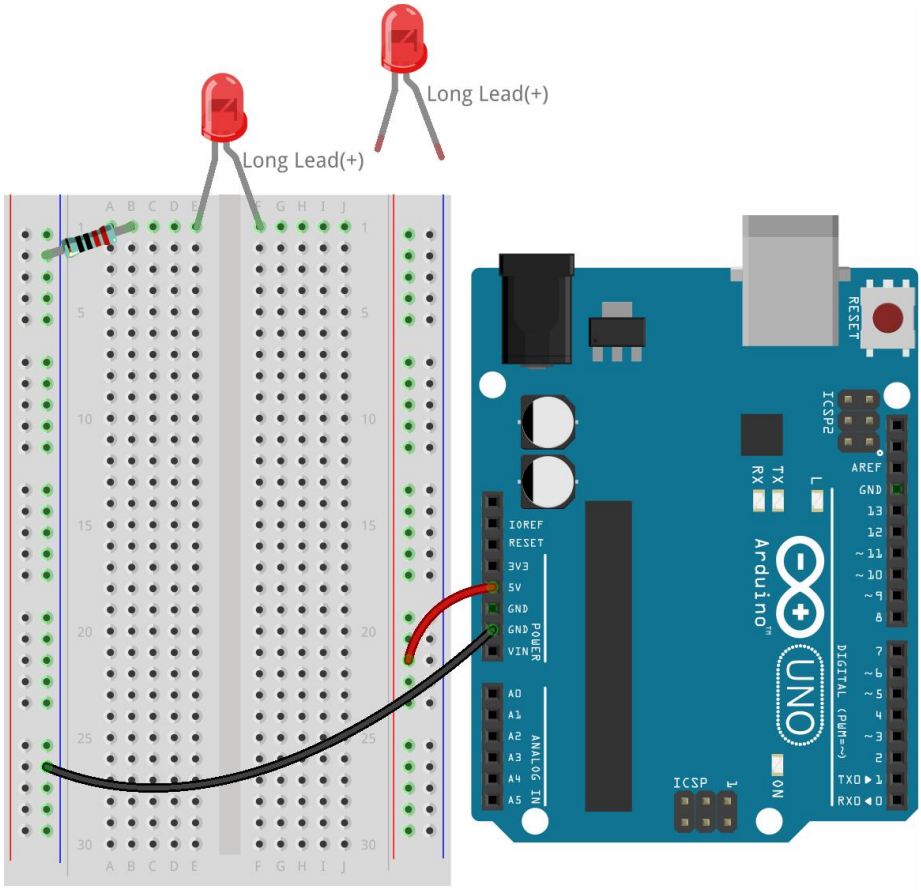
LED は、3.3V では 5V の時よりもわずかに暗くなることに気づくでしょう。なぜでしょうか?

オームの法則に基づき、抵抗と新しい電圧がわかっている場合、電流は I = V/R で計算されます。抵抗が同じままで電圧が下がると、電流が減少し、LED が暗くなります。

レッスン 5： 直列回路と並列回路

「**直列回路を深く学ぶ**」間に、次の質問に答えてください。

1. LED を 1 つ取り外すとどうなりますか？ それはなぜですか？



直列回路では、1 つの LED を取り外すと、他の LED も点灯しなくなります。これは、直列回路では電流が経路内のすべてのコンポーネントを通過する必要があるためです。1 つの LED を取り外すと回路が途切れ、残りの LED に電流が流れなくなります。

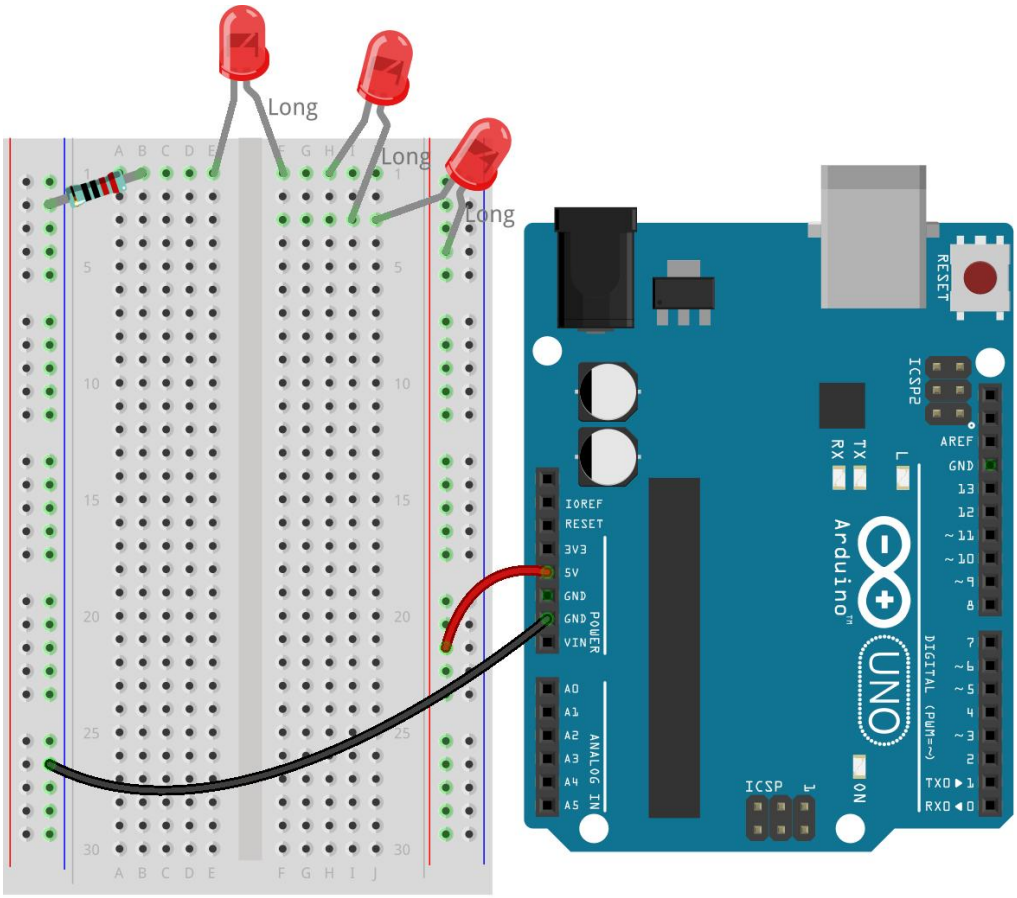
1. 直列回路内の各コンポーネントの電圧を測定してください。

回路	抵抗器の電圧	LED1 の電圧	LED2 の電圧	合計電圧
2 LEDs	*≈1.13 volts*	*≈1.92 volts*	*≈1.92 volts*	*≈4.97 volts*

2. 直列回路内の各コンポーネントの電流を測定してください。

回路	LED1 の電流	LED2 の電流
2 LEDs	*≈4.43 milliamps*	*≈4.43 milliamps*

1. この回路にもう 1 つ LED を追加して 3 つの LED にすると、LED の明るさはどう変わりますか？ その理由は何ですか？ また、3 つの LED にかかる電圧はどう変わりますか？

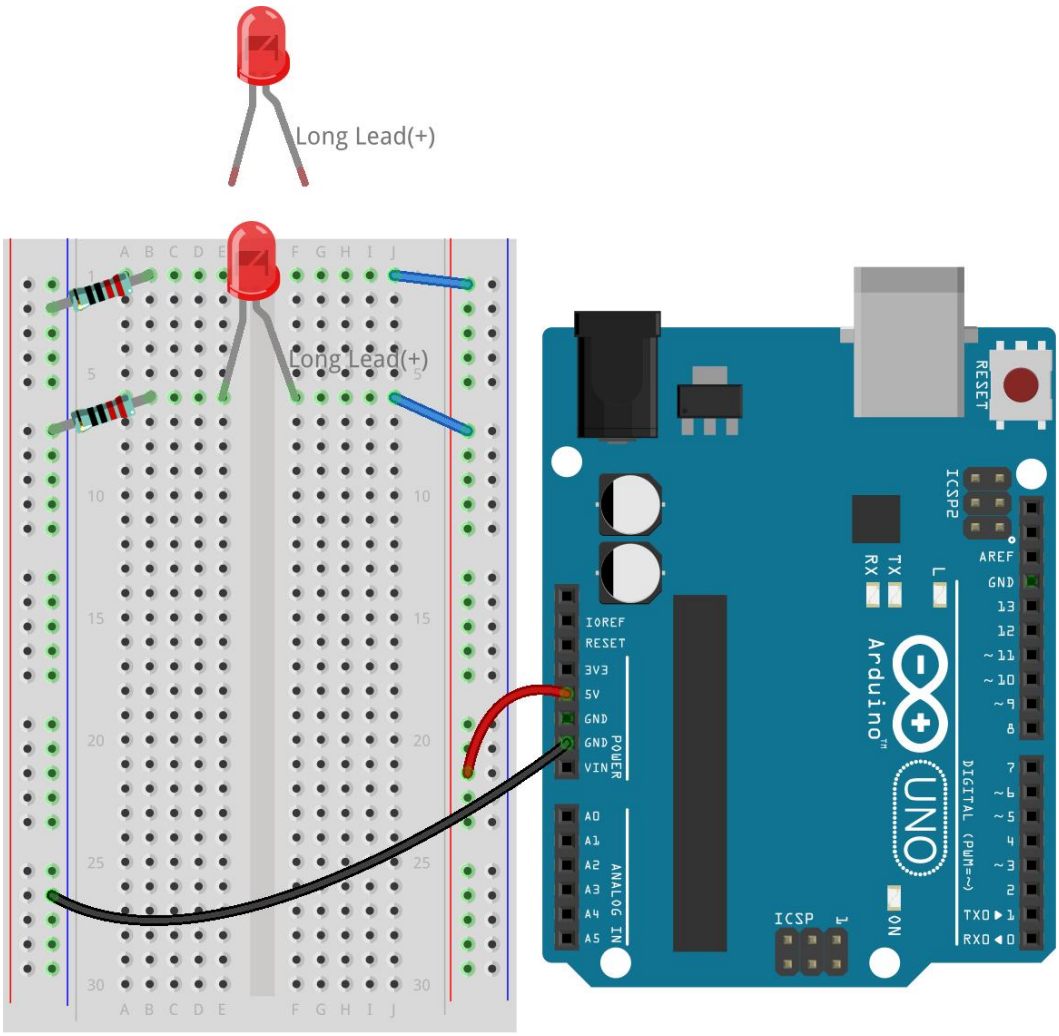


2 つの LED がすでにある直列回路にもう 1 つ LED を追加すると、各 LED の明るさは一般的に減少します。これは、電源の総電圧がより多くのコンポーネントに分配されるため、2 つだけだった場合よりも各 LED にかかる電圧降下が小さくなるためです。その結果、各 LED を流れる電流が減少し、明るさが低下します。

3 つの LED にかかる電圧については、各 LED には回路全体の電圧のうち、より小さな部分がかかることになります。電源の電圧が同じままであれば、この電圧は 3 つの LED で分割されるため、すべての LED が同様の電気特性を持っていると仮定すると、各 LED にかかる電圧はおおよそ電源電圧の 3 分の 1 になります。

「**並列回路を深く学ぶ**」間に、次の質問に答えてください。

1.この並列回路で、1 つの LED を取り外すとどうなりますか？ なぜそうなるのでしょうか？



並列回路では、1 つの LED を取り外しても、回路内の他の LED は引き続き点灯します。これは、並列回路の各 LED が電源への独立した経路を持っているためです。1 つの LED を取り外しても、他の LED への電流の流れは途切れないため、影響を受けずに通常通り動作し続けます。この構成により、並列回路内の各コンポーネントは他のコンポーネントから独立して動作することができます。

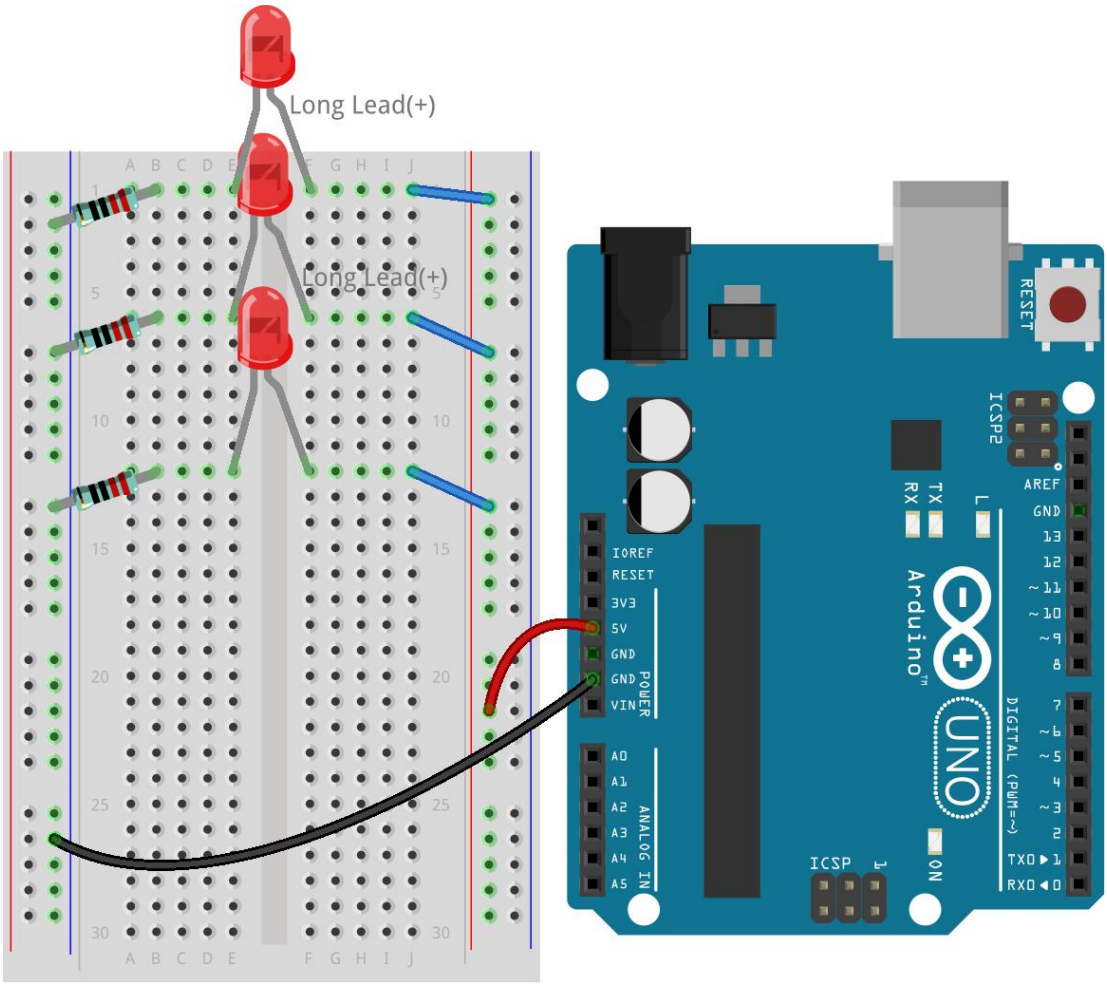
1. 測定した電圧を表に記録してください。

回路	パス 1 電圧	パス 2 電圧
2 LEDs	*≈5.00 volts*	*≈5.00 volts*

2. 測定した電流を表に記録してください。

回路	LED1 電流	LED2 電流	合計電流
2 LEDs	*≈12.6 milliamps*	*≈12.6 milliamps*	*≈25.3 milliamps*

1. この回路にもう一つ LED を追加すると、LED の明るさはどうなりますか？その理由は何ですか？あなたの手帳に答えを記録してください。



別の LED を並列回路に追加すると、既存の LED の明るさは通常変わりません。これは、並列回路の各 LED が電源への直接的な経路を持っているため、追加される LED の数に関係なく、各 LED にかかる電圧が一定であるからです。各 LED は、所定の明るさで動作するために必要な全電圧を受け取ります。したがって、追加の LED を加えても、すでに存在する LED の明るさには影響しません。ただし、電源が回路の総電流需要を維持できることが条件です。このことを将来の参考のために手帳に記録してください。

レッスン 6：ブリンク LED

「**LED を生き生きとさせる**」間に、以下の表を完成させてください。

1. 1. ピン 3 の測定電圧を表に記録してください。

状態	ピン 3 電圧
HIGH	≈4.95 volts
LOW	0.00 volts

レッスンを終えたら、次の質問に答えてください。

1. 上記のコードをアップロードすると、LED が 3 秒間隔で繰り返し点滅するのが確認できます。もし一度だけオン・オフさせたい場合は、どうすれば良いでしょうか?

LED をオン・オフするコマンドを loop()関数から setup()関数に移動できます。setup()関数はプログラムが開始されるときに一度だけ実行されるため、この変更により LED は一度だけ点灯し、消灯します。コードを調整する方法は以下の通りです:

```
void setup() {
  // Setup code here, to run once:
  pinMode(3, OUTPUT);  // set pin 3 as output

  digitalWrite(3, HIGH);  // Light up the LED on pin 3
  delay(3000);            // Wait for 3 seconds
  digitalWrite(3, LOW);   // Switch off the LED on pin 3
}

void loop() {
  // Main code here, to run repeatedly:
}
```

レッスン 7：信号機を作ろう！

「信号機の擬似コードを書く」中に次の質問に答えてください。

信号機のように回路が機能するために必要なことを考えてください。ログにあるスペースに、信号機の動作を説明する擬似コードを英語で書き留めてください。

Arduino を使用して信号機をシミュレーションするには、赤、黄、緑の 3 つの LED を用意し、実際の信号機のように点灯を制御するシーケンスが必要です。以下は、この信号機回路がどのように機能するかを説明するためのシンプルな擬似コードの概要です。ログに書き留めてください：

```
Setup:
   Define pins for the red, yellow, and green LEDs.
   Set all these pins as outputs.
Main Loop:
   Turn on the red LED for 5 seconds.
   Turn off the red LED.
   Turn on the yellow LED for 2 seconds.
   Turn off the yellow LED.
   Turn on the green LED for 5 seconds.
   Turn off the green LED.
   Repeat the cycle.
```

レッスンが終了したら、次の質問に答えてください。

自宅周辺の交差点を見てみましょう。通常、いくつの信号機がありますか？それらはどのように連携していますか?

都市部では、交差点に信号機が設置されており、車両と歩行者の流れを効率的に管理しています。交差点の信号機の数は、その規模や複雑さに応じて大きく異なります。シンプルな四差路では、通常、各方向に 1 つずつ、少なくとも 4 つの信号機があります。より複雑な交差点では、右折レーンや歩行者用横断歩道、その他の交通管理のための追加の信号機が設置されることがあります。

レッスン 8：歩行者用ボタン付き信号機

「**回路の構築**」を完了したら、次の質問に答えてください。

1. あなたの信号機は直列回路と並列回路の混合です。回路のどの部分が直列になっているか、その理由について議論してください。その後、どの部分が並列になっているか、その理由を説明してください。

回路において、ボタンと 10K のプルダウン抵抗は直列に接続されています。この構成により、ボタンが押されると、押されていないときにピン 8 が直接接地され、浮遊入力を防ぐことで、状態が適切に変わります。

ピン 3、4、5 に接続された 3 つの LED は互いに並列に接続されています。各 LED は独立して動作し、それぞれ異なる制御ピンに接続されており、共通の電源を共有しています。この構成により、各 LED は他の LED に影響を与えることなく機能できるため、信号機システムには非常に重要です。

「**コード作成**」中にこの表を記入してください。

1. ボタンが押されているときと押されていないときのピン 8 での測定電圧を表に記入してください。その後、対応する高レベルと低レベルの状態を記入してください。

ボタンの状態	ピン 8 の電圧	ピン 8 の状態
放す	*0.00 volts*	*LOW*
押す	*≈4.97 volts*	*HIGH*

このレッスンを終えたら、次の質問に答えてください。

1. テスト中に、歩行者ボタンが押されている間だけ緑色の LED が点滅することに気付くかもしれません。しかし、歩行者はボタンを押し続けることができないため、道路を渡ることができません。歩行者ボタンが押された後、緑色の LED が安全に渡れる時間だけ点灯するように、コードをどのように修正しますか？ ボタンを押し続ける必要がないようにしてください。擬似コードの解決策を手帳に書き留めてください。

歩行者がボタンを押し続けることなく緑色の LED が点灯し、その後通常の信号サイクルを続けるためには、擬似コードを調整してボタンの押下をチェックし、その押下に基づいて動作状態を変更する必要があります。以下は、これらの変更を反映した最適化された明確な擬似コードのバージョンです:

```
Setup:
    Define pins for red, yellow, and green LEDs as output
    Define the button pin as input
Main Loop:
    Check if the button is pressed
    If button is pressed:
        Turn off all LEDs
        Turn on green LED for pedestrians
        Delay 10 seconds
    Else:
        Execute normal traffic light cycle:
            Turn on green LED (for vehicles), turn off other LEDs
            Delay 10 seconds
            Turn on yellow LED, turn off other LEDs
            Delay 3 seconds
            Turn on red LED, turn off other LEDs
            Delay 10 seconds
```

レッスン 9：調光可能なデスクランプ

「**回路を組み立てる**」際に、この表を記入してください。

1. ポテンショメーターを位置 1 から 3 まで時計回りに回し、各点での抵抗値を測定して結果を表に記録してください。

測定ポイント	抵抗値（キロオーム）
1	*1.52*
2	*5.48*
3	*9.01*

2. ポテンショメーターを時計回りおよび反時計回りに回すと、A0 の電圧はどのように変化すると考えますか?

ポテンショメーターは回路内で直列に接続された二つの抵抗器として考えることができます。抵抗値の測定によれば、ポテンショメーターを時計回りに回すと、A0 と GND の間の抵抗が増加します。直列回路では電流が一定であるため、オームの法則（電圧 = 電流 × 抵抗）に従い、抵抗が増加すると A0 の電圧も増加します。したがって、ポテンショメーターを時計回りに回すと A0 の電圧が増加し、反時計回りに回すと抵抗が減少するため、電圧が減少します。

このレッスンが終了したら、次の質問に答えてください。

1.LED を別のピン（例えばピン 8）に接続し、ポテンショメータを回すと、LED の明るさは変わりますか？ その理由を説明してください。

LED を Arduino UNO のピン 8 に接続し、ポテンショメータを回しても、LED の明るさは変わりません。これは、ピン 8 が PWM（パルス幅変調）をサポートしていないためで、明るさを調整するには analogWrite()関数が必要です。Arduino UNO では、PWM をサポートし、analogWrite()を通じて LED の明るさを制御できるピンは、3、5、6、9、10、11 です。

レッスン 10: モールス信号

「回路を構築する」際に、以下の質問に答えてください。

1. アクティブブザーのカソードを直接 GND に接続し、アノードを 5V に接続すると何が起こりますか？ その理由は何ですか？

アクティブブザーのカソードを直接 GND に接続し、アノードを 5V に接続すると、ブザーは連続音を発生させます。これは、ブザー内部の発振器が 5V 電源によって動作し、回路が切断されるまで音を生成し続けるためです。

レッスンが終了したら、次の質問に答えてください。

1. 提供されたモールス信号表を使用して、「Hello」というメッセージを送信するコードを書いてください。

モールス信号では、「Hello」は以下のように符号化されます。

- **H**: ....
- **E**: .
- **L**: .-..
- **L**: .-..
- **O**: ---

まとめると、「Hello」のモールス信号は次のようになります:

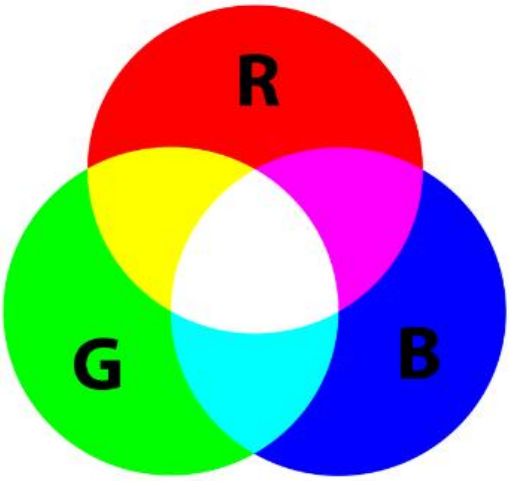
```
.... . .-.. .-.. ---
```

実際のコミュニケーションでは、単語を明確に区別するために通常は単語間に長めのポーズがありますが、「Hello」は一つの単語なので、コードは連続しており、各文字の間にはスペースだけが入ります。

レッスン 11: 虹の色

「**コード作成 - RGB LED の点灯**」の際に、この表を記入してください。

1. 他の色を希望する場合は、何をすべきですか? 下の図を参照し、あなたのアイデアを手帳に記入してください。



色	赤ピン	緑ピン	青ピン
赤	*HIGH*	*LOW*	*LOW*
緑	*LOW*	*HIGH*	*LOW*
青	*LOW*	*LOW*	*HIGH*
黄	*HIGH*	*HIGH*	*LOW*
ピンク	*HIGH*	*LOW*	*HIGH*
シアン	*LOW*	*HIGH*	*HIGH*
白	*HIGH*	*HIGH*	*HIGH*

「**コード作成 - 色の表示**」の間に、この表を記入してください。

1. これで、ピン 9、10、および 11 の値を個別に調整し、観察した色をハンドブックに記録できます。

赤ピン	緑ピン	青ピン	色
0	128	128	*ダークブルー*
128	0	255	*パープル*
128	128	255	*ライトブルー*
255	128	0	*オレンジ*

「**コード作成 - パラメータ化された関数**」の間に、この表を埋めてください。

1. お気に入りの色をいくつか選び、それらの RGB 値を表に記入してください。

色	赤	緑	青

レッスン 12: サイレン音

「**回路を組み立てる**」時に以下の質問に答えてください。

1. パッシブブザーのカソードを直接 GND に接続し、アノードを 5V に接続した場合、何が起こりますか？ その理由は何ですか？

パッシブブザーのカソードを直接 GND に接続し、アノードを 5V に接続すると、アクティブブザーとは異なり、パッシブブザーは内蔵オシレーターを持っていないため、自身では音を出しません。

パッシブブザーは音を生成するために外部信号を必要とします。通常、聞こえる音を作るためには、希望する周波数の矩形波（振動する電圧）で駆動する必要があります。

「**コード作成 - パッシブブザーを鳴らす**」の間に、以下の質問に答えてください。

1.コードと回路のピンを PWM ピンでない 7 または 8 に切り替えた場合、ブザーはまだ音を出しますか？ テストを行った後、手帳に答えを書き込んでください。

ピン 8 は PWM ピンではありませんが、tone()関数を使用すると、正確な方形波を生成することができ、パッシブブザーを駆動して音を出すことができます。この柔軟性により、PWM 対応ピンに制限されることなく、任意のデジタルピンを音の出力に使用できます。tone(pin, frequency)関数を呼び出すと、Arduino はタイマーを設定し、指定された周波数でピンの状態を切り替え（HIGH から LOW、再び HIGH へ）て方形波を生成します。この方形波がパッシブブザーを駆動し、生成された波の周波数で音を出すようになります。

1. tone(pin, frequency, duration)における周波数と持続時間がブザーの音にどのように影響するかを探るために、2 つの条件でコードを修正し、観察した現象を手帳に記入してください:

- 周波数を 1000 に保ちながら、持続時間を 100、500、1000 と段階的に増やします。ブザーの音はどのように変化し、なぜそうなるのでしょうか?
    - 100ms の持続時間: 音は短いビープ音です。
    - 500ms の持続時間: 音は長いビープ音で、はっきりと聞こえ、約 0.5 秒続きます。
    - 1000ms の持続時間: 音はさらに長く、1 秒間続きます。

    持続時間を増加させると、ブザーから出る音はより長く続きます。音の高さや周波数は一定（1000 Hz に設定されているため）であり、音色の「ノート」は変わりませんが、聞こえる時間が長くなります。これは、緊急度や警告の種類を音の長さで区別できるように、異なる持続時間の警告信号を送る際に役立ちます。

- 持続時間を 100 に保ちながら、周波数を 1000、2000、5000 と徐々に増加させます。ブザーの音はどのように変化し、その理由は何ですか?
    - 1000 Hz の周波数: 音は中音のビープ音です。
    - 2000 Hz の周波数: 音は 1000 Hz と比べて高い音調です。
    - 5000 Hz の周波数: 音は非常に高音で、近くで聞くと鋭く、不快に感じることもあります。

周波数を上げて持続時間を一定に保つと、音の音調が変化します。高い周波数は高音の音を生成します。この原理は、緊急性や重要性に基づいて異なる種類の通知や信号を区別するのに役立ちます。高音は通常、より緊急なアラートに使用されます。

レッスン 13: ジョイスティック LED ナビゲーター

「コード作成 - ジョイスティックモジュールからの読み取り」中に次の質問に答えてください。

1. ジョイスティックモジュールの X 軸と Y 軸はアナログ値を返し、SW ピンはデジタル値を返します。前のステップでは、これらの値をシリアルモニターで確認しました。

Arduino プログラミングにおけるデジタル値とアナログ値の違いを要約してください。

Arduino プログラミングにおいて:

- **アナログ値**: 0 から 1023 までの連続した値で、0V から 5V に対応し、Arduino のアナログ-デジタルコンバータ（ADC）によってキャプチャされます。これらの値は、ジョイスティックの X 軸と Y 軸に見られるように、信号の大きさに関する詳細な情報を提供します。
- **デジタル値**: 0 または 1 の離散的な値で、低（0V）または高（5V）状態を表します。ジョイスティックの SW ピンは、押されているかどうかを示すデジタル信号を使用しており、バイナリのオン/オフ決定に最適です。

アナログ入力は信号の強度を詳細に提供し、デジタル入力は単純な存在または不在の検出に使用されます。

「ジョイスティックの動きに基づく LED 制御」の**「コード作成」**中に、次の質問に答えてください。

1. 前回のコードでは、ジョイスティックの X 軸と Y 軸の値に基づいて対応する LED を制御しました。これらの LED が点灯している間に、明るさを調整するためにはコードをどのように修正しますか?

ジョイスティックの X 軸と Y 軸の値に基づいて LED の明るさを制御するには、digitalWrite()の代わりに analogWrite()関数を使って PWM (パルス幅変調) を利用する必要があります。この方法により、ジョイスティックの位置に応じて LED の明るさを調整できるため、単に LED をオンまたはオフにするだけではなく、さまざまな明るさレベルを設定できます。

この概念を示すための改善されたコードスニペットは以下の通りです:

コードを以下のように修正できます:

```
void loop() {
  // Read the joystick values
  int xValue = analogRead(xPin);
  int yValue = analogRead(yPin);
  int swValue = digitalRead(swPin);

  // First, turn off all LEDs
  analogWrite(ledLeft, 0);
  analogWrite(ledRight, 0);
  analogWrite(ledUp, 0);
  analogWrite(ledDown, 0);

  // Check joystick positions and set LEDs accordingly
  if (yValue < 412) {
    // Joystick up
    analogWrite(ledUp, yValue / 4);
  } else if (yValue > 612) {
    // Joystick down
    analogWrite(ledDown, yValue / 4);
  } else if (xValue < 412) {
    // Joystick left
    analogWrite(ledLeft, xValue / 4);
  } else if (xValue > 612) {
    // Joystick right
    analogWrite(ledRight, xValue / 4);
  }
  // Add a small delay to stabilize readings
  delay(100);
}
```

1. ピン 8 に接続された LED の動作は、他のピンに接続された LED と比べてどのように異なり、その理由は何ですか?

Arduino Uno R3 のピン 8 に接続された LED は、明るさの変化なくオンとオフの状態を切り替えるだけです。

これは、ピン 8 が PWM (パルス幅変調) をサポートしていないためです。Arduino Uno R3 では、PWM はピン 3、5、6、9、10、および 11 でのみ利用可能です。ピン 8 に PWM 機能がないため、LED の明るさを電圧レベルで調整する analogWrite() 関数はこのピンでは使用できません。そのため、LED を完全にオンまたはオフにするために、digitalWrite()のみを使用し、中間状態は存在しません。

レッスン 14: ダイナソーゲームをプレイする

「2. サーボの準備」の際に次の質問に答えてください。

1. サーボが Arduino のピン 8 または他の非 PWM ピンに接続されている場合、正常に動作しますか? その理由は何ですか?

まずはこれをテストしてから、回答してください。

Arduino のピンにサーボモーターを接続する際は、一般的に PWM (パルス幅変調) ピンを使用することが推奨されます。ただし、Arduino の Servo ライブラリは、PWM ピンに限らず、ソフトウェアエミュレーションを使用して必要な信号を生成することでサーボ制御を容易にするように設計されています。

このソフトウェアベースのアプローチにより、サーボ制御パルスはライブラリ内で管理され、マイコンのハードウェア PWM 機能には依存しません。その結果、Servo ライブラリは、非 PWM ピン (例えばピン 8 など) を含む任意のデジタルピンを使用して、サーボを効果的に制御することを可能にします。

「3. フォトレジスタの準備」の際に、以下の質問に答えてください。

1. 現在の周囲光の下で抵抗値を読み取り、以下の表に記録してください。

環境	抵抗値（キロオーム）
通常光	≈5.48
明るい光	≈0.16
暗闇	≈1954

レッスン 15: クールカラーまたはウォームカラー

「コード作成」の際に、この表を記入してください。

1. Paint やその他のカラー選択ツールを開き、最も暖かい色と涼しい色を見つけ、それらの RGB 値をハンドブックに記録してください。

色の種類	赤	緑	青
暖色	*246*	*52*	*8*
寒色	*100*	*150*	*255*

レッスンを終えたら、次の質問に答えてください。

1. 範囲の「下限」が「上限」よりも大きい場合や小さい場合があることに注意してください。そのため、map() 関数を使用して数値の範囲を逆にすることができます。例えば:

```
y = map(x, 1, 50, 50, 1);
```

この関数は負の数も適切に処理するため、この例も有効であり、問題なく動作します。

```
y = map(x, 1, 50, 50, -100);
```

For `y = map(x, 1, 50, 50, -100);`, if x equals 20, what should y be? Refer to the following formula to calculate it.

`y = map(x, 1, 50, 50, -100);` の場合、x が 20 のとき、y は何になるべきでしょうか。以下の式を参照して計算してください。

$$\frac{\text{Value - From Low}}{\text{From High - From Low}} = \frac{\text{Y - To Low}}{\text{To High - To Low}}$$

$$\text{Y} = \frac{\text{Value - From Low}}{\text{From High - From Low}} \times (\text{To High - To Low}) + \text{To Low}$$

x=20 を使ってマッピング式 `y = map(x, 1, 50, 50, -100);` を適用すると、y の値は約 -8.16 になります。

レッスン 16: サマーファン

「**回路を組み立てる**」際に、この表に記入してください。

1. 1,2EN を 5V に接続した状態で、ピン 2 (1A) を 5V に接続したときと GND に接続したときのピン 3 (1Y) の電圧をマルチメーターで測定し、測定値を表に記録してください。

1,2EN	1A	1Y
5V	5V	≈5.04V
5V	0V	≈0V

レッスンが終了したら、次の質問に答えてください。

1. モーターの回転方向も制御したい場合、コードはどのように修正すべきですか?

- motorRotate() 関数を修正して、両方のモーター制御ピンを使用した方向制御を含める。
- motorRotate() 関数に方向を指定するためのパラメータを追加する。
- メインループ内の motorRotate() の呼び出しを修正し、ボタンの押下に基づいて方向制御を含める。

```
void loop() {
  if (digitalRead(button1) == LOW) {
    motorRotate(0, 0);  // Turn off the motor
  } else if (digitalRead(button2) == LOW) {
    motorRotate(150, -1);  // Motor runs backward at low speed
  } else if (digitalRead(button3) == LOW) {
    motorRotate(200, 1);  // Motor runs forward at medium speed
  } else if (digitalRead(button4) == LOW) {
    motorRotate(250, 1);  // Motor runs forward at high speed
  }
}
void motorRotate(int speed, int direction) {
  if (direction >= 0) {
    analogWrite(motor1A, speed);
    analogWrite(motor2A, 0);
  } else {
    analogWrite(motor1A, 0);
    analogWrite(motor2A, speed);
  }
}
```

レッスン 17: I2C LCD1602 ディスプレイの探求

レッスンを終えたら、以下の質問に答えてください。

1. I2C LCD1602 の 1 行目の 5 列目から「Let's count」を表示したい場合、コードをどのように調整すればよいですか？ また、どのような視覚効果が得られますか？

「Let's count」を I2C LCD1602 の 1 行目の 7 列目から表示するには、lcd.setCursor()関数を調整してカーソルを 6 列目に設定します（列のインデックスは 0 から始まります）。以下のようにコードを修正することができ、観察される結果は次の通りです:

```
void loop() {
  lcd.setCursor(6, 0);        // Sets cursor to the fifth column (index
6) of the first row.
  lcd.print("Let's count");   // Displays "Let's count".
  lcd.setCursor(0, 1);        // Moves cursor to the first column of
the second line.
  lcd.print("Count: ");       // Displays "Count: ".
  lcd.print(count);           // Prints current count next to "Count:
".
  delay(1000);                // Pauses for one second.
  count++;                    // Increments counter.
  lcd.clear();                // Clears the display for the next
iteration.
}
```

7 列目から「Let's count」というテキストが表示され始めると、ディスプレイの右側にさらに進みます。「Let's count」は 11 文字あるため、17 列目まで延びますが、LCD1602 は 16 列しかないため、最後の文字は表示されません。このため、表示されるのは「Let's coun」のみで、最後の「t」はディスプレイの端で切れてしまいます。

したがって、ディスプレイのレイアウトを慎重に計画することが、LCD スクリーンにすべての希望するテキストが適切に表示されるために重要です。

レッスン 18: ON/OFF デスクランプ

「**回路を構築する**」際に、この表に記入してください。

1. では、マルチメーターを使用して、COM と NC 間の導通を測定し、リレー モジュールの動作原理を確認してください。

状態	COM 端子に接続されているのは NO または NC ですか?
デフォルト	*NC*
S ピン高	*NO*

このレッスンが終わったら、次の質問に答えてください。

1. デジタルピン 7 を INPUT のみに設定した場合、何が起こるでしょうか? その理由は何ですか?

```
void setup() {
  pinMode(9, OUTPUT);  // Set pin 9 as output
  pinMode(7, INPUT);   // Set pin 7 as input with an internal pull-up resistor
  Serial.begin(9600);  // Serial communication setup at 9600 baud
}
```

デジタルピン 7 を Arduino スケッチで INPUT モードに設定すると、INPUT_PULLUP とは異なり、ピンから読み取る信号が不安定になる可能性があります。ピンが INPUT としてのみ設定され、外部回路を通じて明確な高電圧または低電圧に接続されていない場合、それは「浮動」状態になります。浮動ピンは安定した高または低の状態にはなく、電気的ノイズや環境からの干渉によって状態が変動する可能性があります。この変動は、デジタル入力関数を使用してピンの状態を読み取ろうとした際に予測不可能な読み取り結果を引き起こし、マイコンが受信するデータに誤りや不整合が生じる原因となります。

2. ピン 7 を INPUT のみに設定した場合、回路にどのような調整が必要ですか?

もし Arduino のピン 7 が INPUT モードに設定されていて、安定した予測可能な読み取りを保証したい場合は、回路に外部プルアップ抵抗を追加する必要があります。これは、ピン 7 と Arduino の 5V 電源の間に 10kΩの抵抗を接続することを含みます。プルアップ抵抗は、他の入力信号が存在しないときに入力ピンが高い状態（論理レベル 1）になることを保証します。

レッスン 19: スマートゴミ箱

「コード作成 - 距離を測る」中に以下の質問に答えてください。

1.このデバイスで検出される距離を小数点までより正確にするには、コードをどのように修正すればよいですか?

距離測定の精度を向上させ、小数点以下の精度を含めるためには、Arduino コードの measureDistance()関数内の計算を調整する必要があります。現在、距離は整数演算を用いて計算されているため、小数点以下の値が切り捨てられています。小数点以下の精度を実現するために、整数演算の代わりに浮動小数点演算を使用するように計算を修正できます。

以下のようにコードを修正できます:

- 距離のデータ型を long から float に変更し、小数値を扱えるようにします。
- 計算式を調整し、浮動小数点演算を実行するようにします。

以下は measureDistance()関数の修正部分です:

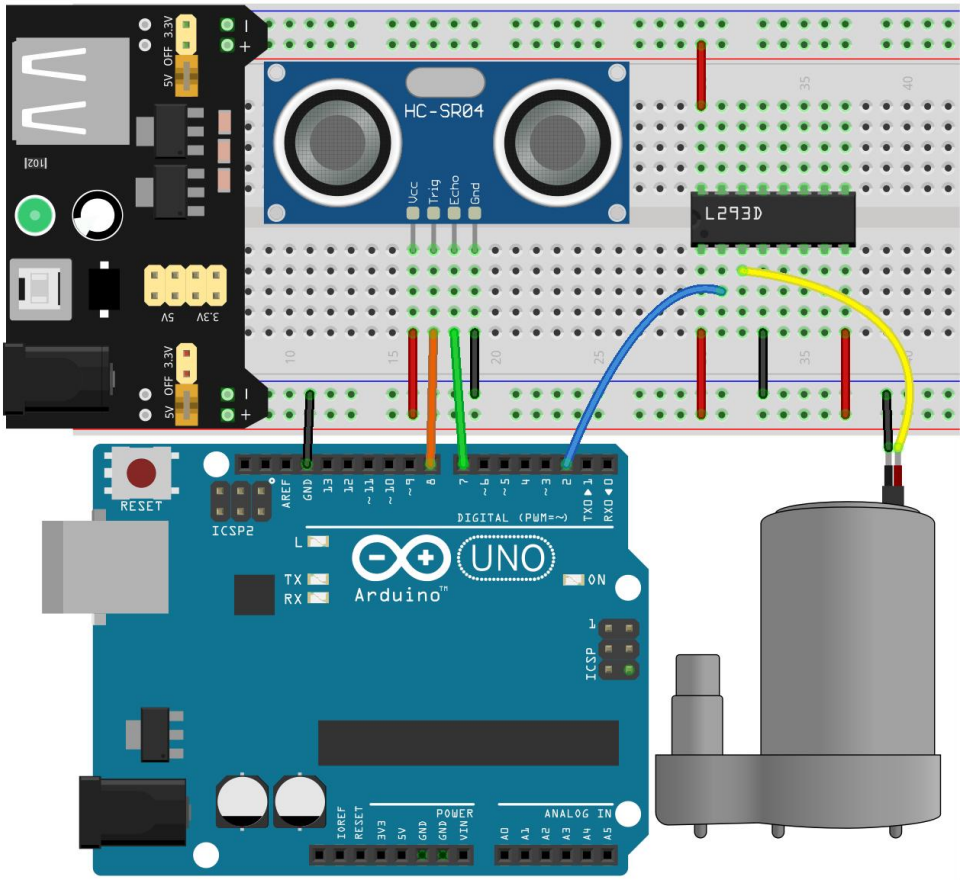
```
float measureDistance() {
  digitalWrite(TRIGGER_PIN, LOW);  // Ensure Trig pin is low before
a pulse
  delayMicroseconds(2);
  digitalWrite(TRIGGER_PIN, HIGH);  // Send a high pulse
  delayMicroseconds(10);            // Pulse duration of 10
microseconds
  digitalWrite(TRIGGER_PIN, LOW);   // End the high pulse

  long duration = pulseIn(ECHO_PIN, HIGH);  // Measure the duration
of high level on Echo pin
  float distance = duration * 0.034 / 2.0;   // Calculate the distance
(in cm) with decimal precision
  return distance;
}
```

レッスン 20: 自動ソープディスペンサー

「**コード作成 - 水ポンプを動かす**」中に以下の質問に答えてください。

1.このプロジェクトでは、特定のドライバーとセットアップを使用して水ポンプを接続しました。ポンプの接続を逆にした場合、どうなると思いますか? ポンプは逆に動作するのか、動作しなくなるのか、それとも他の何かが起こるのか? 試してみて、その結果を考えてみてください。



ポンプの接続を逆にしても、プロジェクト内のポンプは効果的に動作し続け、水をポンプし続けることがわかります。逆に動作したり、動作しなくなったり、損傷を引き起こすことはありません。

接続を逆にしても、ポンプの双方向機能のためにその機能は変わりません。ポンプは設計通り、入口から出口へと水を移動させ続けます。

レッスン 21: 温度アラーム

「**回路の構築**」中にこの表を記入してください。

1. 異なる温度下での抵抗値を測定し、以下の表に記録してください。

環境	抵抗値（キロオーム）
現在の温度	*9.37*
高温	*6.10*
低温	*12.49*

レッスンを終えたら、次の質問に答えてください。

1. コードではケルビンと摂氏の温度が計算されています。もし華氏の温度も知りたい場合は、どうすればよいですか?

これは摂氏から華氏に変換する標準的な方法であり、既に計算で得た摂氏の値に基づいて華氏の温度を算出します。

```
F = C * 1.8 +32
```

1. 今日私たちが作ったような温度監視システムが役立つ他の状況や場所を考えられますか?

温度監視システムは、日常的な状況やさまざまな環境で幅広く利用できます。以下はいくつかの簡単な例です:

- **家庭の快適さ**: リアルタイムの温度測定に基づいて、家庭の暖房や冷房を自動的に調整し、快適な居住空間を維持します。
- **ガーデニング**: 温室の温度を監視し、植物が最適な条件で育つようにします。必要に応じて温度を調整する自動システムを追加します。
- **食品の安全性**: 冷蔵庫や冷凍庫の温度を追跡し、特にレストランや食品輸送中に食品が安全に食べられる状態を保ちます。

医療: 温度に敏感な薬やワクチンの保管エリアの温度を監視し、効果を維持できるように記録します。

レッスン 22: リモートコントロールカラフルライト

「コード作成 - キー値の取得」の際に、この表を記入してください。

1. リモコンの各キーを慎重に押し、マニュアルの表に対応するキー値を記録してください。

キー名	キー値	キー名	キー値
POWER	0x45	0	0x16
モード	0x46	1	0xC
ミュート	0x47	2	0x18
再生/一時停止	0x44	3	0x5E
戻る	0x40	4	0x8
進む	0x43	5	0x1C
EQ	0x7	6	0x5A
-	0x15	7	0x42
+	0x9	8	0x52
サイクル	0x19	9	0x4A
U/SD	0xD		

レッスン 23: 「きらきら星」を演奏する

「**コード作成 - 配列**」の際に、次の質問に答えてください。

1. 配列の要素に対しても操作を行うことができます。例えば、Serial.println(melody[i] * 1.3),に変更した場合、どのようなデータが得られ、その理由は何ですか?

1.3 という数値は浮動小数点数です。melody 配列の整数（型は int）を 1.3 で乗算すると、演算結果は自動的に浮動小数点数（float）に昇格します。この配列内の各音符の周波数に対して 1.3 を掛けて、その結果を出力すると、次のようになります:

```
340.6
340.6
509.6
509.6
572.0
572.0
509.6
...
```

レッスンが終了したら、以下の質問に答えてください。

1. 回路内のパッシブブザーをアクティブブザーに置き換えた場合、「きらきら星」を確実に再生できますか? その理由は何ですか?

「きらきら星」を再生するためにパッシブブザーをアクティブブザーに置き換えると、意図した通りには動作しません。アクティブブザーは内蔵オシレーターのため、単一の音しか出すことができません。そのため、メロディーを正確に再現するために音程を制御することができず、曲のリズムに合わせた繰り返しのビープ音しか聞こえず、実際の音符は再生されません。

レッスン 24: ポモドーロタイマー

「**コーディング作成 - millis()**」の際に、以下の質問に答えてください。

1.delay(100) を delay(1000)に変更すると、プログラムにはどのような影響がありますか? その理由も教えてください。

```
if (currentMillis - previousMillis >= interval) {
  previousMillis = currentMillis;  // Save the last time the buzzer beeped
  digitalWrite(buzzerPin, HIGH);   // Make a voice
  delay(100);
  digitalWrite(buzzerPin, LOW);  // silence
}
```

元のコードでは、ブザーは 1000 ミリ秒（interval 変数で設定された 1 秒）ごとに約 100 ミリ秒鳴り、その後 900 ミリ秒の静寂が続きます。修正後は、ブザーは 1000 ミリ秒ごとに 1000 ミリ秒鳴り、次の間隔がほぼすぐに始まるため、ほぼ連続して鳴ることになります。したがって、delay を 100 から 1000 ミリ秒に変更すると、ブザーは短いビープ音を出すのではなく、連続的な音を発生させるようになり、元の意図から外れ、より煩わしく不適切になります。

コード内の delay(100)を delay(1000)に変更すると、ブザーは短いビープ音ではなく、1 秒間鳴るようになります。これは、ブザーが鳴っているときの間隔が長くなるためです。この変更により、ブザーの音が長くなり、プログラムのサイクルが減るため、これらの間隔中に他のタスクへの応答が鈍くなる可能性があります。

レッスンを完了したら、次の質問に答えてください。

あなたの生活の中で「時間を感じる」ことができる他の場所について考えてみてください。いくつかの例を挙げて、ハンドブックに書きましょう!

時間を「聞く」というのは興味深い概念で、私たちが日常生活の中で体験できるシーンはいくつかあります。以下に、メモしておくと良い例をいくつか挙げます:

- **時計や腕時計**: アナログ時計のティック音や、デジタル時計が毎秒または毎分を知らせる特定のビープ音。
- **キッチンタイマー**: 料理や焼き時間をカウントダウンする機械式またはデジタ

ルキッチンタイマーのティック音と最終アラーム。

- **学校のベル**: 授業や休憩の始まりと終わりを知らせる学校のベルの音。
- **公共交通機関のアナウンス**: 駅やバスでのアナウンス前のビープ音やチャイム音、出発や到着の迫ることを示す。
- **電子レンジ**: タイマーが終了したときのビープ音、加熱が完了したことを知らせる。
- **フィットネストラッカーまたはスポーツウォッチ**: ワークアウトやインターバル中に設定した時間の終了を知らせるビープ音やアラーム。

レッスン 25: 逆レーダーシステム

レッスンを終えたら、次の質問に答えてください。

1. このプロジェクトでは、アクティブブザーをアラート機構として使用しましたが、パッシブブザーを使用しても同様の機能を実現できます。アクティブブザーをパッシブブザーに置き換える場合、コードはどのように修正すべきでしょうか?

ア Arduino プロジェクトでアクティブブザーをパッシブブザーに置き換えることを決定した場合、ブザーを制御するコードを修正する必要があります。パッシブブザーは音を出すために交流信号を必要とし、アクティブブザーは高信号だけでビープ音を鳴らすためです。

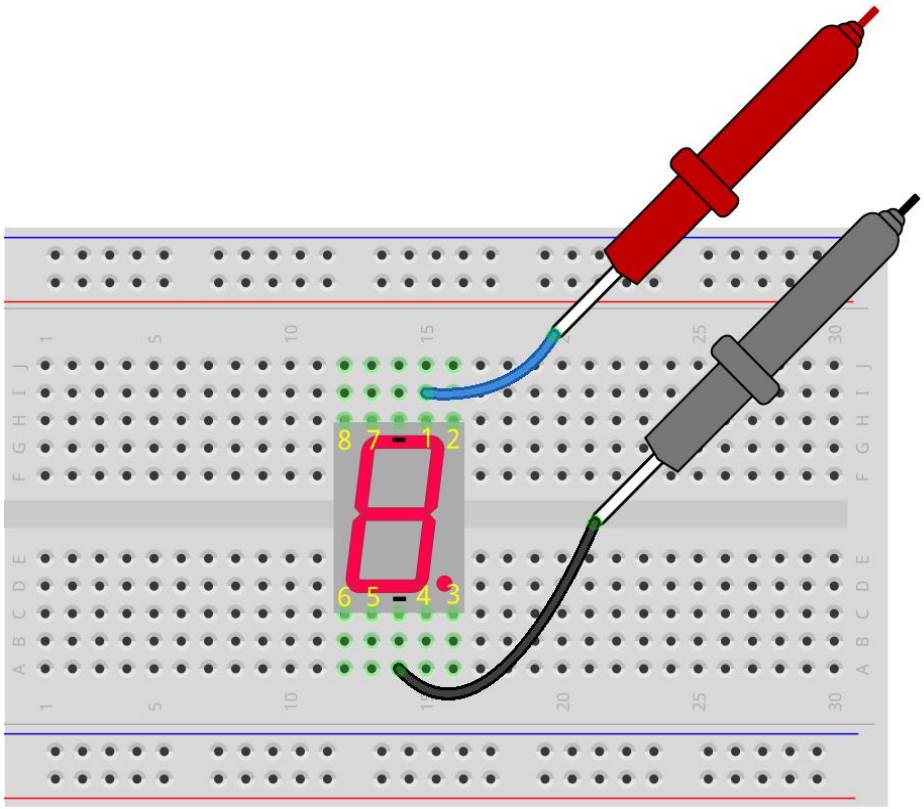
パッシブブザーに対応するために、コード内の beep()関数を次のように修正できます:

```
// Function to make passive buzzer beep
void beep() {
  // Generate a tone on the buzzer pin
  tone(BUZZER_PIN, 2000);  // 2000 Hz frequency
  delay(100);              // Beep duration: 100 milliseconds
  noTone(BUZZER_PIN);      // Stop the tone
}
```

レッスン 26: サイバーダイス

「**7 セグメントディスプレイを学ぶ**」セクション中に、以下の質問に答えてください。

1. セグメントが点灯した場合、この図を参照して、そのセグメントのピン番号とおおよその位置をハンドブックの表に記録してください。



ピン	セグメント番号	位置
1	A	*上部*
2	B	*右上部*
3	C	*右下部*
4	D	*下部*
5	E	*左下部*
6	F	*左上部*
7	G	*中央*
8	*小数点*	*ドット*

1. 上記のテストから、キットのディスプレイは共通カソードであることがわかります。つまり、共通ピンを GND に接続し、他のピンに高電圧を供給することで、対応するセグメントを点灯させる必要があります。ディスプレイに数字の 2 を表示させたい場合、どのピンに高電圧を供給する必要がありますか？ その理由は何ですか?

数字 2 を表示するためには、セグメント a、b、d、e、g をアクティブにする必要があります（高電圧に設定）。これらのセグメントがディスプレイ上で数字 2 を形成するためです。セグメント f、c、および dp（小数点がある場合）は、数字 2 の表示には含まれないため、オフ（低電圧）にしておく必要があります。

したがって、数字 2 を正しく表示するために高電圧を供給すべきピンは、セグメント a、b、d、e、g に接続されているものです。

レッスン 27: 74HC595 を使った流れる光

「コード作成 - LED を点灯させる」中に以下の質問に答えてください。

1. shiftOut(DS, SHcp, MSBFIRST, B11101110);の MSBFIRST を LSBFIRST に変更すると何が起こりますか？ その理由は何ですか？

MSBFIRST を LSBFIRST に変更すると、ビットの順序が逆になり、最下位ビット（右端のビット）からバイトがシフトアウトされます。シフトレジスタを使って LED を制御している場合、ビットの順序を変更すると LED が点灯する順序も逆になります。元々プログラムされた順序で点灯するのではなく、逆の順序で点灯します。

レッスンが終了したら、以下の質問に答えてください。

1. 3 つの LED を同時に点灯させ、それらが「流れる」ように見せたい場合、 datArray[]配列の要素をどのように変更すればよいでしょうか?

最初は最初の 3 つの LED を点灯させ、次のパターンごとに 1 つずつ LED を右に移動させ、最後は最後の 3 つの LED が点灯するようにします。これらのパターンは以下のように 2 進数で定義できます:

```
B11100000: LEDs 1, 2, 3 are on; others are off.
B01110000: LEDs 2, 3, 4 are on; others are off.
B00111000: LEDs 3, 4, 5 are on; others are off.
B00011100: LEDs 4, 5, 6 are on; others are off.
B00001110: LEDs 5, 6, 7 are on; others are off.
B00000111: LEDs 6, 7, 8 are on; others are off.
```

```
byte dataArray[] = { B11100000, B01110000, B00111000, B00011100,
B00001110, B00000111 };
```

レッスン 28: 数字の表示

「**0 から 9 までの数字に対応する 2 進数**」のセクション中に、この表を埋めてください。

1. 0 と 2 の数字に対応する 2 進数がわかったので、以下の表に残りの数字に対応する 2 進数を埋めてください。

数字	2 進数
0	*B00111111*
1	*B00000110*
2	*B01011011*
3	*B01001111*
4	*B01100110*
5	*B01101101*
6	*B01111101*
7	*B00000111*
8	*B01111111*
9	*B01101111*

「**2 進数変換**」のセクション中に、この表を埋めてください。

1. 0 から 9 までの数字を表す 2 進数を、電卓を使って 10 進数と 16 進数に変換し、表に記入してください。これにより、基数変換のための簡単なリファレンスガイドが得られます。

数字	2 進数	10 進数	16 進数
0	B00111111	63	0x3F
1	B00000110	6	*0x06*
2	B01011011	*91*	*0x5B*
3	B01001111	*79*	*0x4F*

4	B01100110	*102*	*0x66*
5	B01101101	*109*	*0x6D*
6	B01111101	*125*	*0x7D*
7	B00000111	*7*	*0x07*
8	B01111111	*127*	*0x7F*
9	B01101111	*111*	*0x6F*

レッスン 29: 植物モニター

「**コード作成 - 土壌水分の読み取り**」のセクション中に以下の質問に答えてください。

1. 提供されたコードでは、土壌の水分含有量が高いほどセンサー値が低くなり、水分は通常パーセンテージで表されます。土壌の水分レベルをパーセンテージで表示するようにコードをどのように変更すればよいでしょうか?

土壌の水分レベルをパーセンテージで表示するには、まず、水分センサーが完全に乾燥しているときと完全に湿っているときに返す値の範囲を理解する必要があります。これらの値を使用して、センサーのアナログ出力をパーセンテージスケールに変換できます。

0%は完全に乾燥していることを示し、100%は完全に湿っていることを示します。以下のようにコードを修正し、水分をパーセンテージで計算・表示できます:

```
const int moisturePin = A1;  // Define the pin where the soil moisture sensor is connected

void setup() {
  Serial.begin(9600);  // Initialize serial communication at 9600 baud rate
}

void loop() {
  int moistureValue = analogRead(moisturePin);  // Read the analog value from the moisture sensor
  Serial.print("Raw Moisture Value: ");
  Serial.println(moistureValue);  // Output the raw sensor value to the serial monitor for observation

  // Map the moisture value from the sensor's possible range (0 to 1023) to a percentage (0% to 100%)
  float moisturePercent = map(moistureValue, 0, 1023, 100, 0);
  Serial.print("Moisture Percentage: ");
  Serial.print(moisturePercent);
  Serial.println("%");  // Output the calculated moisture percentage
  delay(1000);  // Delay for one second before the next reading to reduce data flooding
}
```

レッスン 30: Arduino レーダーシステム

「**3. 超音波モジュールの準備**」のセクション中に以下の質問に答えてください。

1. 上記のコードでは、超音波モジュールが 1 度ごとに測定を行います。もし測定が頻繁すぎると感じ、5 度ごとに測定したい場合、コードをどのように修正すればよいでしょうか?

超音波モジュールが 1 度ごとではなく 5 度ごとに測定を行うようにコードを修正するには、サーボの動きを制御するループを調整する必要があります。具体的には、for ループの増分を 1 から 5 に変更します。これにより、サーボはループの各イテレーションで 5 度ずつ動き、測定の頻度が減少します。

以下のようにループをコード内で調整できます:

```
void loop() {
  // rotates the servo from 15 to 165 degrees in steps of 5 degrees
  for (int i = 15; i <= 165; i += 5) {
    ...
  }
  // rotates the servo from 165 to 15 degrees in steps of 5 degrees
  for (int i = 165; i >= 15; i -= 5) {
    ...
  }
}
```

レッスン 31: 数当てゲーム

レッスンが終了したら、以下の質問に答えてください。

ゲームの楽しさを高めるために、どのような追加コンポーネントを追加できますか？それらはゲーム内でどのような役割を果たしますか？

数当てゲームの楽しさを高めるために、以下のコンポーネントを追加することを検討してください:

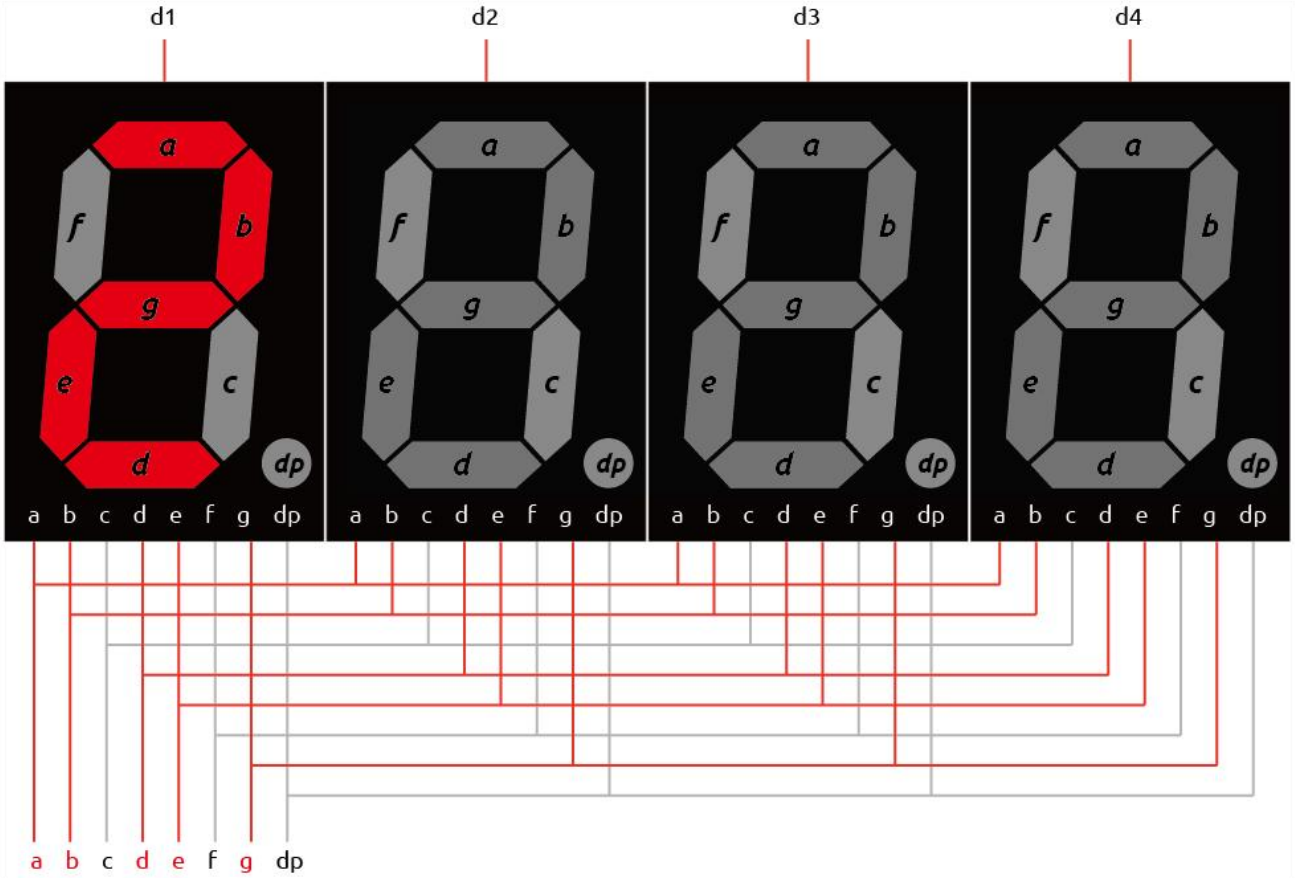
- **スピーカーやブザーを使ったサウンドエフェクト**: 正解や不正解に対して音声フィードバックを提供し、ゲームをより魅力的にします。
- **RGB LED や LED ストリップ**: 色の点滅などの視覚的なフィードバックを使って、ゲームを視覚的に盛り上げます。
- **追加ディスプレイによるスコアボード**: スコアやタイマーを表示し、競争性を高めます。
- **Wi-Fi または Bluetooth モジュール**: 遠隔プレイやマルチプレイヤーモードを可能にし、異なる場所からの競技を可能にします。
- **タッチスクリーンディスプレイ**: インターフェースをアップグレードし、よりインタラクティブで現代的なゲーム体験を提供します。

これらの追加要素により、ゲームはよりインタラクティブで、競争性が高まり、楽しさも増します。

レッスン 32: ストップウォッチ

「**4 桁 7 セグメントディスプレイの学習**」のセクション中に、以下の質問に答えてください。

1. 4 桁 7 セグメントディスプレイの最左端の桁（d1）に「2」を表示させたい場合、d1～d4 および a～g ピンのレベルをどのように設定すればよいでしょうか?



ピン	HIGH または LOW	7 セグメントディスプレイ	HIGH または LOW
d1	*LOW*	a	*HIGH*
d2	*HIGH*	b	*HIGH*
d3	*HIGH*	c	*LOW*
d4	*HIGH*	d	*HIGH*
		e	*HIGH*
		f	*LOW*
		g	*HIGH*
		dp	*LOW*

「**コード作成 - 1 桁での数字スクロール**」のセクション中に、以下の質問に答えてください。

In programming, bitwise operations like AND and OR are crucial for manipulating individual bits of data. The bitwise AND operation (&), compares each bit of its operands, resulting in 1 if both bits are 1, and 0 otherwise. Conversely, the bitwise OR operation (|), results in 1 if at least one of the bits is 1, and 0 only if both bits are 0.

プログラミングにおいて、ビット単位の操作（AND や OR）はデータの個々のビットを操作するために非常に重要です。ビット単位 AND 演算（&）は、各オペランドのビットを比較し、両方のビットが 1 の場合にのみ 1 を返し、それ以外は 0 を返します。逆に、ビット単位 OR 演算(|）は、少なくとも一方のビットが 1 の場合に 1 を返し、両方のビットが 0 の場合にのみ 0 を返します。

この情報を基に、(B01011011 >> 2) | 1 という式を考えてみましょう。2 ビット右にシフトした後、ビット単位の OR 演算で 1 を適用した場合、結果はどうなりますか?

1) **右シフト**: B01011011 >> 2 の操作は、B01011011 のビットを 2 つ右にシフトします。

- 元の値: 01011011
- シフト後: 00010110

2) **ビット単位 OR との演算**: ビット単位 OR 演算 (00010110 | 1) は、右シフトした値と 1 を組み合わせます。結果は以下の通りです。

- OR 前: 00010110
- OR 後: 00010111

結論

(B01011011 >> 2) | 1 の最終結果は 00010111 で、10 進数では 23 です。| 1 の演算は、最下位ビット（LSB）を 1 に設定するため、LSB が最初に 0 だった場合に数値がわずかに増加します。これにより、右シフトは値の大きさを減少させ（2 のべき乗で割る）、一方で OR 演算は特定のビットパターン、例えば LSB を設定するために使用できることがわかります。

レッスン 33: RF522-RFID モジュールの探究

レッスンが終了したら、以下の質問に答えてください。

1. RC522-RFID モジュールを使用してカードやタグの情報を読み書きし、それを LCD に表示する方法を理解したので、日常使用のための一般的なアクセス制御システムをどのように設計しますか? 設計アプローチを説明してください。

日常使用に適した実用的なアクセス制御システムを設計する際、私のアプローチは、使いやすさ、セキュリティ、スケーラビリティに重点を置きます。以下は、私の設計戦略の概要です:

システムコンポーネント

- **RFID リーダー (RC522)**: RFID カードやタグを通じてユーザー識別を行い、配布と管理が簡単です。
- **マイクロコントローラー (Arduino)**: RFID リーダーを制御し、入力を処理し、出力アクションを管理する中央処理ユニットとして機能します。
- **LCD ディスプレイ (I2C LCD1602)**: ユーザーにリアルタイムのフィードバックを提供し、アクセス状況や指示を表示します。
- **ブザー**: アクセスの許可や拒否を示す聴覚フィードバックを提供し、ユーザーエクスペリエンスを向上させます。
- **キーパッド**: RFID タグを忘れた場合や紛失した場合のバックアップとして、PIN コードによる代替エントリーを可能にします。

ユーザーインタラクション

- ユーザーは RFID カードをスワイプするか、PIN コードを入力します。システムは LCD に視覚的なフィードバックを、ブザーで聴覚的なフィードバックを即座に提供します。
- 認証に成功すると、ドアは一定時間解錠され、その後自動的に再ロックされます。
- アクセスが拒否された場合、システムはメッセージを表示し、警告音を鳴らします。

レッスン 34: アクセス制御システム

「コード作成 - ステッピングモーターを回転させる」のセクション中に以下の質問に答えてください。

1. 1 回転を一方向に行い、その後反対方向に 1 回転し、このサイクルを繰り返すようにしたい場合、コードをどのように修正すればよいでしょうか?

ステッピングモーターが一方向に 1 回転し、その後反対方向に 1 回転するようにし、これを連続して繰り返すためには、Arduino スケッチの loop()関数を調整する必要があります。以下のように実装できます:

- 1 回転するためのステップ数を設定します。ステッピングモーターが 1 回転に 2048 ステップ必要な場合 (STEPS 定数で定義されています) 、この数を使ってモーターを 1 回転させます。
- stepper.step() 関数を正負の値で使用します。正の値はモーターを一方向に回転させ、負の値は反対方向に回転させます。

以下は修正された loop()関数です:

```
void loop() {
  // Set the speed of the stepper
  stepper.setSpeed(5);

  // Rotate clockwise
  stepper.step(STEPS);  // Move 2048 steps (one full revolution)
  delay(1000);          // Wait for 1 second

  // Rotate counterclockwise
  stepper.step(-STEPS);  // Move -2048 steps (one full revolution in
the opposite direction)
  delay(1000);           // Wait for 1 second
}
```

「**コード作成 - アクセス制御システム**」のセクション中に、以下の質問に答えてください。

1. 基本的なアクセス制御システムが設定されたので、機能性と柔軟性を高めるために追加できるコンポーネントにはどのようなものがありますか?

アクセス制御システムの機能性と柔軟性を高めるために統合できる追加コンポーネントには以下のものがあります:

- **オーディオフィードバック用のブザー**: アクセスイベントに対して音声フィードバックを提供し、アクセスの許可や拒否、システムアラートを音でユーザーに伝え、インタラクションを向上させます。
- **コード入力用のマトリックスキーパッド**: PIN コード入力を可能にし、RFID カードを持っていないユーザーや、二要素認証を必要とするシステムに便利です。
- **視覚的な識別用のカメラ**: 追加のセキュリティ確認のために画像や動画をキャプチャし、アクセスイベントのビジュアルログを保持します。
- **モーションセンサー**: 制御ポイントに到達する前に存在を検知し、システムを起動したり、不正アクセスの試みを警告したりします。