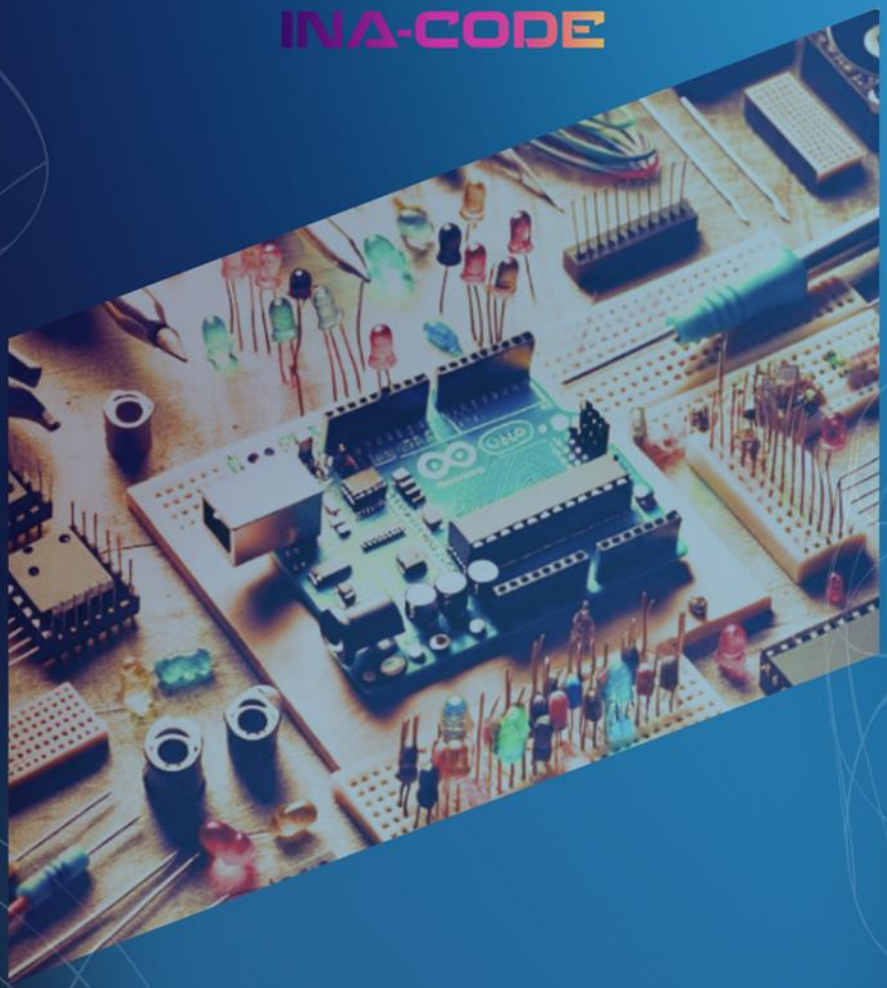
EXPERIMENT BOOKLET

INA-CODE



INNOVATIVE APPROACH FOR CODING IN DIGITAL ERA

EXPERIMENT BOOKLET

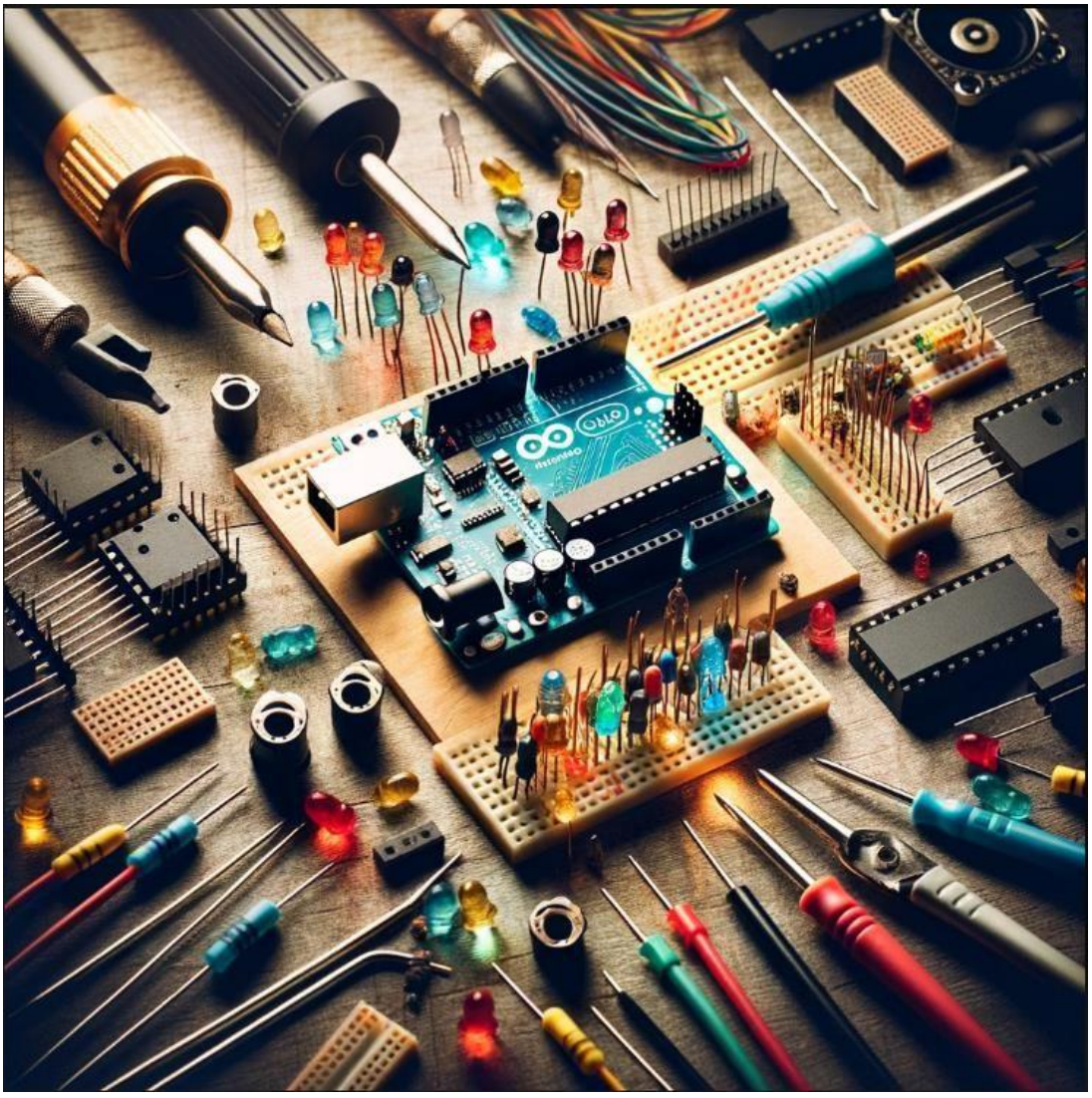
INNOVATIVE APPROACH FOR CODING IN DIGITAL ERA

2021-DE03-KA220-SCH-000024558

Co-funded by the European Union

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them

Deney Kitapçığı



İçindekiler tablosu

Önsöz

Günümüz dünyasında dijital dönüşümün hızı ve kapsamı, eğitim sistemlerimizin de bu değişimlere uyum sağlamasını zorunlu kılıyor. Bu bağlamda, çalışmamız "Deney Kitapçığı: Öğretmen Kitabı ve 25 Deneyi İçeren Öğrenci Kitabı" adlı çalışmamız, hem öğretmenlerin hem de öğrencilerin dijital becerilerini geliştirmeye yönelik önemli bir adımı temsil ediyor. Bu kitapçık, Avrupa Komisyonu ve Ulusal Ajanslar tarafından desteklenen, dijital uçurumu kapatmayı ve Avrupa genelinde daha kapsayıcı bir dijital eğitim ortamını teşvik etmeyi amaçlayan daha büyük bir girişimin parçasıdır.

Bu kitapçığın oluşturulması, özellikle COVID-19 salgınının vurguladığı zorlukların ardından, eğitimcileri dijital çağda gezinmek için gerekli araç ve bilgilerle donatmaya duyulan acil ihtiyaçtan kaynaklanmıştır. Öğretmenlerin kodlama ve robotiği müfredatlarına entegre etmeleri için bir kaynak görevi gören bu kitapçık, böylece sadece profesyonel profillerini geliştirmekle kalmıyor, aynı zamanda öğrencileri 21. yüzyıl işgücünün taleplerine hazırlıyor.

Bu çabadaki ortaklarımız arasında, dijital eğitimi ilerletme ortak hedefiyle bir araya gelen Avrupa'daki okullar, mesleki eğitim kurumları ve üniversitelerden oluşan bir konsorsiyum yer almaktadır. Kitapçık pratik, erişilebilir ve çeşitli eğitim ortamlarına uyarlanabilir şekilde tasarlanmış olup, farklı geçmişlere sahip öğretmen ve öğrencilerin bundan faydalanabilmesini sağlamaktadır.

Bu kitapçığın amacı sadece kodlama ve robotik öğretiminin ötesine geçerek öğrenciler arasında eleştirel düşünme, yaratıcılık ve problem çözme becerilerini geliştirmeyi hedeflemektedir. Eğitimciler bu deneyleri öğretimlerine dahil ederek daha ilgi çekici ve etkileşimli bir öğrenme deneyimi sağlayabilir ve öğrencileri dijital teknolojinin geniş olanaklarını keşfetmeye teşvik edebilirler.

Sonuç olarak, bu kitapçık sadece bir öğretim aracı değil; dijital açıdan daha yetkin ve dirençli bir toplumu teşvik ederek eğitim sistemlerimizde değişim için bir katalizördür. Bu dönüştürücü yolculuğa katkıda bulunmaktan gurur duyuyor ve hem eğitimcilere hem de öğrencilere dijital çağın zorluklarını ve fırsatlarını kucaklamaları için ilham vermesini umuyoruz.

Deney Sayısı: 1

1- Aküler

Deney Hedefi

Bu deneyin amacı pillerin nasıl çalıştığını ve ne tür piller olduğunu açıklamaktır. Öğrenciler farklı pil türlerini nasıl tanıyacaklarını öğreneceklerdir.

Teorik Arka Plan

Bir akü seçerken, voltajını ve kapasitesini aklınızda bulundurmalısınız. Bu terimleri daha iyi anlamak için su ile bir benzetme yapabiliriz.

Su ile dolu bir su tankımız olduğunu düşünelim.

Voltaj, suyu boruya doğru iten su tankındaki basınçtır. Suyun yüksekliği basıncı artırır ve elektrik devrelerindeki gerilime eşdeğerdir.

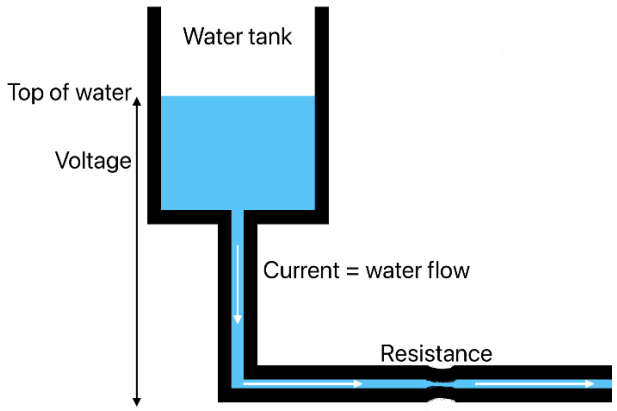
Bir tankın dibinde bir boru vardır. Borudan akan su bir elektrik akımını temsil eder. Basınç ne kadar yüksek olursa o kadar fazla su elde ederiz, yani voltaj ne kadar yüksek olursa akım da o kadar yüksek olur.

Borunun ucunda bir tıkanıklık bulunabilir. Su akışı azaldığı için direnci temsil eder. Boru çapı ne kadar küçükse (daha yüksek direnç), su akışı da o kadar düşük olacaktır. Bu, elektrik devresinde temsil edilir: direnç ne kadar büyükse, aynı voltajla akım daha küçük olacaktır.

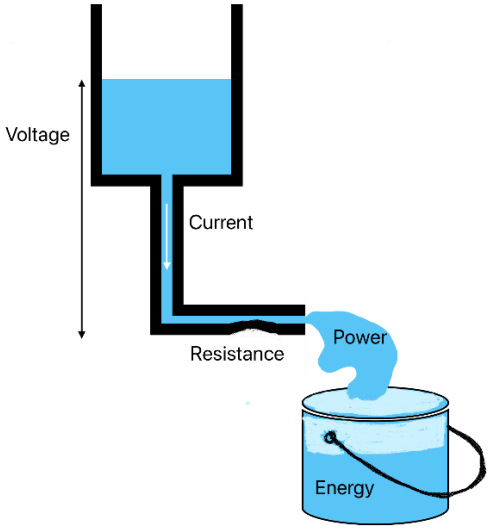
Matematiksel kelimelerle:

$$U = I \cdot R$$

Burada U gerilimi, I akımı ve R direnci temsil eder.



Bu analoji hem güç hem de enerji için genişletilebilir. Güç, su akış hızına benzer. Enerji ise kovada son bulan su miktarıdır. Güç Watt (W) veya kilowatt (kW) cinsinden, enerji ise Watt-saat (Wh), miliwattt saat (mWh) veya kilowatt saat (kWh) cinsinden ölçülür.



Şekilde bazı batarya örnekleri verilmiştir.

Gerekli Malzemeler

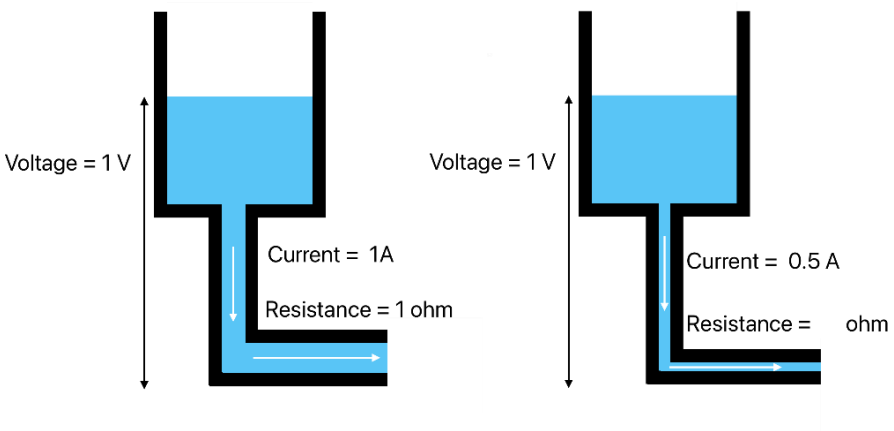
Farklı piller.

Görev 1.

Sınıfınızda farklı piller bulun ve model adını (örneğin AA veya AAA) ve voltajını belirleyin.

Görev 2.

Şekilde verilen gerilim ve akım için direnci belirleyin.



Çözüm 1.

Soldaki şekilde, direnç

$$R = \frac{U}{I} = \frac{1\,V}{1A} = 1\Omega$$

Sağdaki şekilde, direnç

$$R = \frac{U}{I} = \frac{1\,V}{0.5} = 2\Omega$$

Deney Sayısı: 2

2- Elektroniğe giriş: dirençler, LED, fotorezistörler ve esnek sensör

Deney Hedefi

Bu deneyin amacı LED, direnç, fotorezistör ve esnek sensörlerin elektrik devrelerine nasıl bağlanacağını açıklamaktır. Öğrenciler bu elemanları kullanım sırasında yanmamaları için güvenli bir şekilde kullanmayı öğreneceklerdir.

Teorik Arka Plan

Işık yayan diyot veya LED, içinden akım geçtiğinde ışık yayan yarı iletken bir elektronik elemandır.

Diyotun kısa bacağına katot denir ve - kutbunu temsil eder. Akülerin - kutbuna veya güç kaynağına bağlanır.

Diyotun uzun bacağına anot denir ve + kutbu temsil eder. Aküye veya güç kaynağına bağlanır.

Diyot yanlış bağlanırsa, içinden akım geçmeyecek ve yanmayacaktır.

Bir LED'i bir devreye bağlarken, bu LED'den geçen akımı dikkate almalıyız. Diyottan çok fazla akım geçerse, diyot yanacaktır. LED üzerinden önerilen maksimum akım 20 mA'dir.

Yanmayı önlemek için LED ile elektrik devresine bir direnç bağlamak gerekir. Hangi direncin kullanılacağı Ohm yasasına göre belirlenir:

$$U = I \cdot R$$

Burada U gerilimi, I akımı ve R direnci temsil eder.

LED'in gerilimi 3 V olsun. 9 V pil kullanırsak bir dirence ihtiyacımız olur:

$$R = \frac{U}{I} = \frac{9V - 3V}{20mA} = \frac{6V}{0.02A} = 300\Omega$$

Tipik olarak, hesaplanan değere en yakın standart değerlere sahip bir direnç kullanılır. Bu örnekte 330Ω'luk bir direnç kullanılmıştır.

Görev 1

LED'i 4,5 V'luk bir pile bağlarsak hangi direnci kullanmamız gerektiğini belirleyin.

Çözüm 1

Kullanmamız gerek:

$$R = \frac{U}{I} = \frac{4,5V - 3V}{20mA} = \frac{1.5V}{0.02A} = 75\Omega$$

Görev 2

İnternette diğer renklerdeki LED'lerdeki voltaj düşüşünü araştırın ve bir tablo doldurun.

LED Rengi	Gerilim
Sarı	
Turuncu	
Kırmızı	
Mavi	
Yeşil	
Menekşe	

Çözüm 2.

LED Rengi	Gerilim
Sarı	1.9-2.1V
Turuncu	2.0-2.1V
Kırmızı	1.6-2.0V
Mavi	2.7-3.2C
Yeşil	1.9-2.2V
Menekşe	2.8-4.0V

Görev 3

Elektronik devre simülatörünü deneyin: https://www.falstad.com/circuit/e-voltdivide.html Gösterilen devre iki paralel kola sahiptir. Akü gerilimi 10V'tur.

İki adet 10 $k\Omega$ direnç, toplam 20 $k\Omega$ olan ilk daldadır. Bunlardan geçen akım dirençler $I = \frac{U}{R} = \frac{10V}{20k\Omega} = 0{,}5\ mA$'dır.

Toplam $40k\Omega$ olan ikinci kolda 10 kΩ'luk dört direnç bulunmaktadır. Bunlardan geçen akım dirençler = $\frac{U}{R} = \frac{10V}{40k\Omega} = 0.25\ mA$'dır.

Animasyonda, daha yüksek akıma sahip dal daha hızlı hareket eden noktalarla gösterilirken, daha düşük akıma sahip daldaki noktalar daha yavaş hareket eder.

Akım değerleri eşit olacak şekilde diğer daldaki direnç değerlerini belirleyin. Simülatörde test edin.

Çözüm 3

Toplam 20 $k\Omega$olan ikinci kolda 5 $k\Omega$'luk dört direnç bulunmalıdır. İçinden geçen akım bu dirençler daha sonra $I = \frac{U}{R} = \frac{10V}{20k\Omega} = 0{,}5\ mA$ olacaktır.

Görev 4

İlk kolda iki adet $10k\Omega$ direnç olduğunda, aralarındaki gerilim $\frac{10V}{2} = 5V$ olur.

Dirençlerden biri değiştiğinde, aralarındaki gerilim de değişecektir. Üstteki direnç $10k\Omega$ ve alttaki $30k\Omega$ ise, aralarındaki gerilim 7,5V olacaktır.

Aralarındaki gerilim 8V olacak şekilde alt direncin değerini belirleyin. Değerini

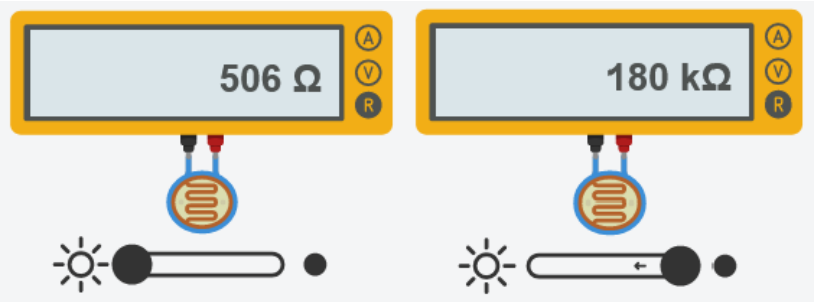
değiştiren bir dirence **potansiyometre** denir.

Çözüm 4

Üst direnç $10k\Omega$ ile birlikte, alt direnç $40k\Omega$ olmalıdır.

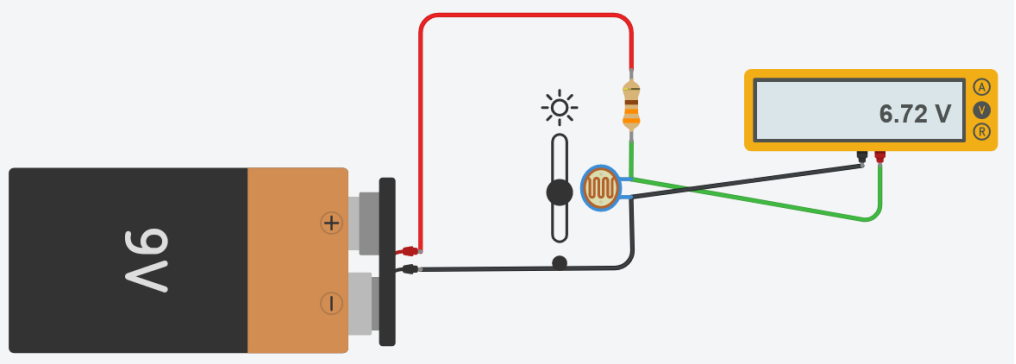
Görev 5

Fotorezistör, ışığı ölçmek için kullanılan bir dirençtir. Üzerine ne kadar çok ışık düşerse, direnci o kadar düşük olur. Üzerine ne kadar az ışık düşerse, direnci o kadar yüksek olur. Fotorezistör değişken bir direnç gibi davranır,
Yani bir potansiyometre. Tek fark, bunu kendi ellerimizle değil, odanın aydınlatmasıyla ayarlıyor olmamızdır.



Fotorezistörün direnci değiştikçe voltajı da değişir. Fotorezistörün direnci ne kadar yüksekse (ışık yok), voltajı da o kadar yüksektir.

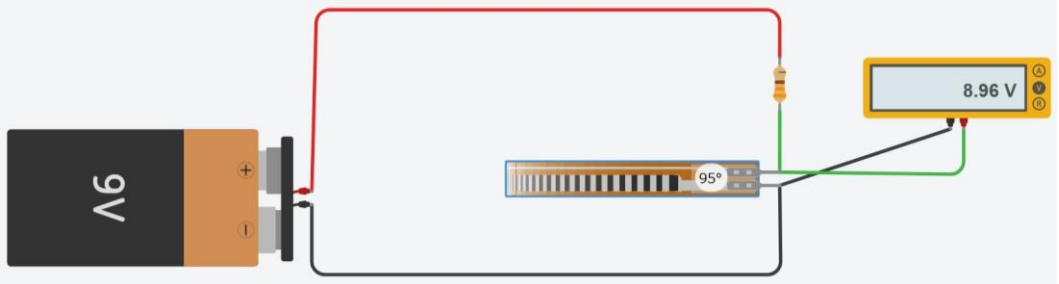
TINKERCAD'de 9V pil, direnç, fotorezistör ve voltmetreyi şekildeki örneğe göre bağlayın. Fotorezistör üzerindeki ışığı değiştirin ve direncinin ve voltajının nasıl değiştiğini gözlemleyin.



Görev 5

Esnek sensör, sapma veya bükülme miktarını ölçen bir sensördür. Genellikle insan eklemlerinde konum ve hareketlilik takibi için biyoteknoloji giyilebilir cihazlarında kullanılır.

Esnek sensörü şemaya göre bağlayın ve direnci ve voltajı ölçün.



Deney Sayısı: 3

3- Elektroniğe giriş: kapasite, endüktans, röle, transistör ve DC motorlu H-köprüsü

Deney Hedefi

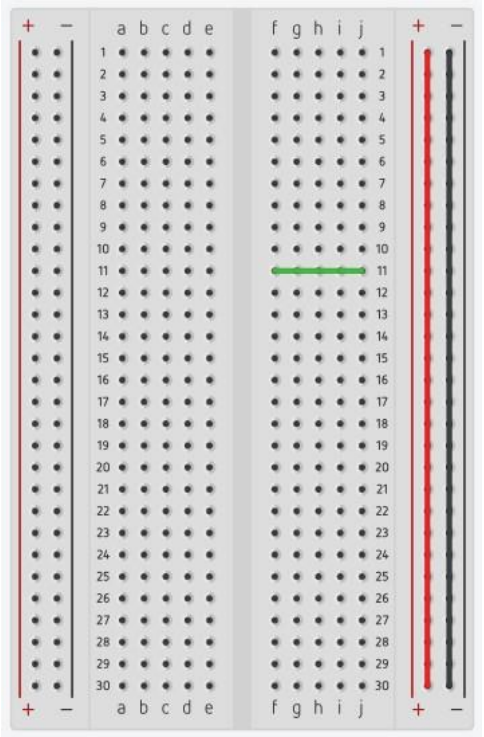
Bu deneyin amacı, kapasite, endüktans, röle, transistörler ve DC motorların elektrik devrelerine nasıl bağlanacağını açıklamaktır. Öğrenciler bu elemanları kullanım sırasında yanmamaları için güvenli bir şekilde kullanmayı öğreneceklerdir.

Teorik Arka Plan

Kondansatör, elektrik yükü depolayabilen elektronik bir elemandır. Yük depolama yeteneğini tanımlayan fiziksel miktar elektrik kapasitesidir. **Kapasite Farad (F)** cinsinden ölçülür. Voltajın artırılması depolanan şarj miktarını da artırır. Aynı zamanda kapasite değişmez. Çoğu zaman kondansatör, bir DC voltaj kaynağına bağlanarak elektriklendirilen iki paralel iletken plakaya sahiptir. Bir plaka pozitif kutba, diğeri ise kaynağın negatif kutbuna bağlanır.

Bobinden bir elektrik akımı geçtiğinde, bir manyetik alan oluşur. **Bobin,** elektromanyetik alanın enerjisini depolayan elektronik bir elemandır. Ne kadar enerji depolanacağı bobinin endüktansına ve içinden geçen akıma bağlıdır. Bobinin **endüktansı**, manyetik alanın enerjisini depolama yeteneğini tanımlamak için kullanılan bir niceliktir. Bobinden geçen akım ne kadar büyükse, bobinin manyetik indüksiyonu da o kadar büyük olur. Endüktans için ölçü birimi **Henri'dir (H)**.

Protoboard, prototip elektrik devrelerini bağlamak için kullanılan bir yapı levhasıdır. Bir sıradaki beş delik kısa devrelidir. Bunlardan ikisine bir elektrik elemanı veya tel bağlarsak, kontaklar doğrudan bağlıymış gibi davranırlar. Aynı durum protoboardın en sol ve en sağ tarafındaki + ve - işaretlerinin altındaki sütunlar için de geçerlidir.



Görev 1

Sadece bir kaynak, tek bir direnç, bir **kondansatör ve bir** anahtar içeren basit bir elektrik devresi düşünün. Anahtar açıldığında ve kapatıldığında akıma ne olduğunu inceleyin. Devreye bir ampul ekleyin ve ampulün ne zaman yandığını gözlemleyin.

Simülatör bağlantısı: https://www.falstad.com/circuit/e-cap.html

Görev 2

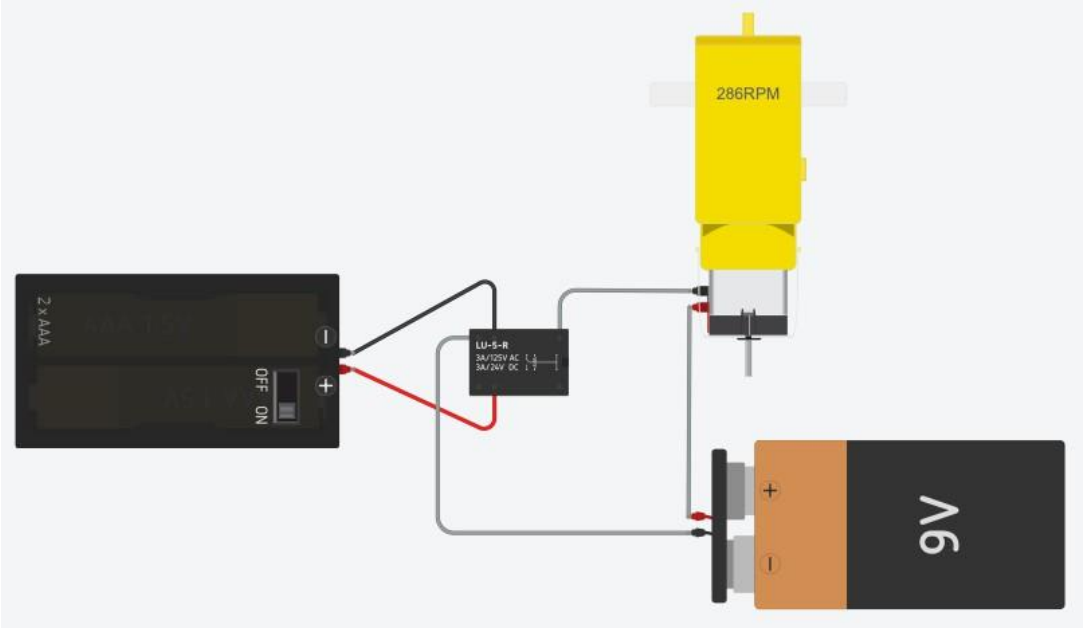
Sadece bir kaynak, tek bir direnç, bir **endüktans** ve bir anahtar içeren basit bir elektrik devresi düşünün. Anahtar açıldığında ve kapatıldığında akıma ne olduğunu inceleyin. Devreye bir ampul ekleyin ve ampulün ne zaman yandığını gözlemleyin.

Simülatör bağlantısı: https://www.falstad.com/circuit/e-induct.html

Görev 3

Röle bir elektrik güç anahtarıdır. İkinci elektrik devresini açmak veya kapatmak için ilk elektrik devresinden gelen akım tarafından etkinleştirilir. Rölelerle çalışırken, iki ayrı devre için iki ayrı güç kaynağı kullanmak zorunludur.

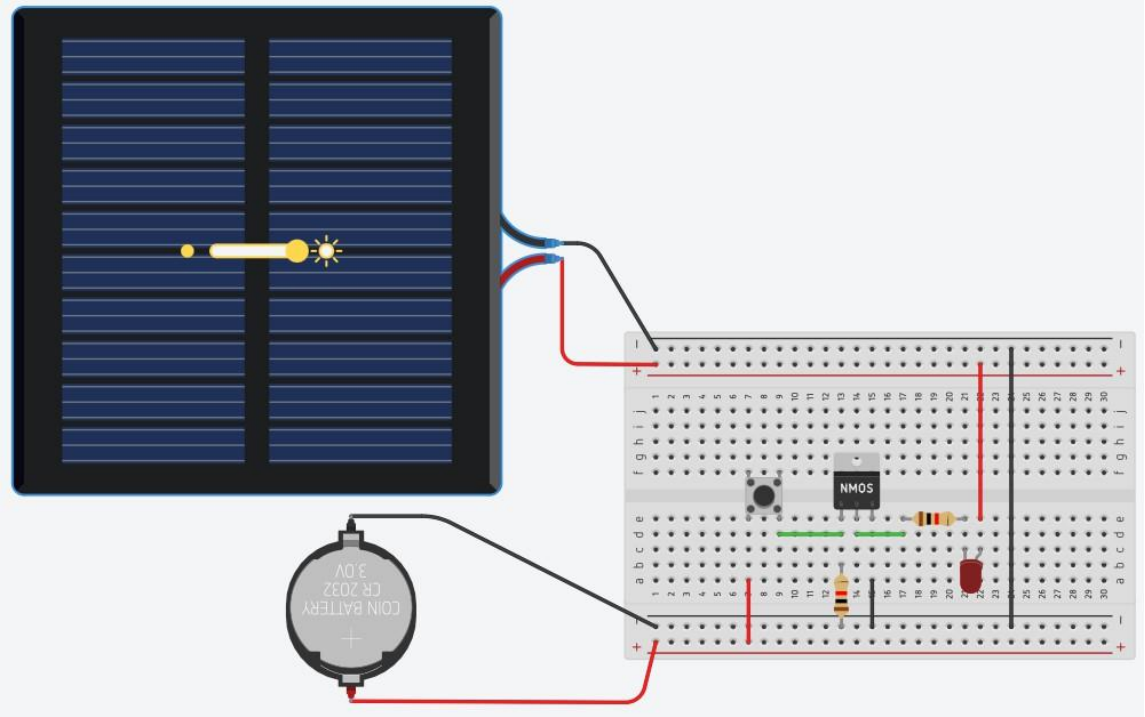
Şemayı şekle göre TINKERCAD'deki röleye bağlayın ve rölenin nasıl çalıştığını test edin.



Görev 4

Bir **NMOS transistör,** elektronik anahtar olarak kullanılan bir yarı iletkendir.

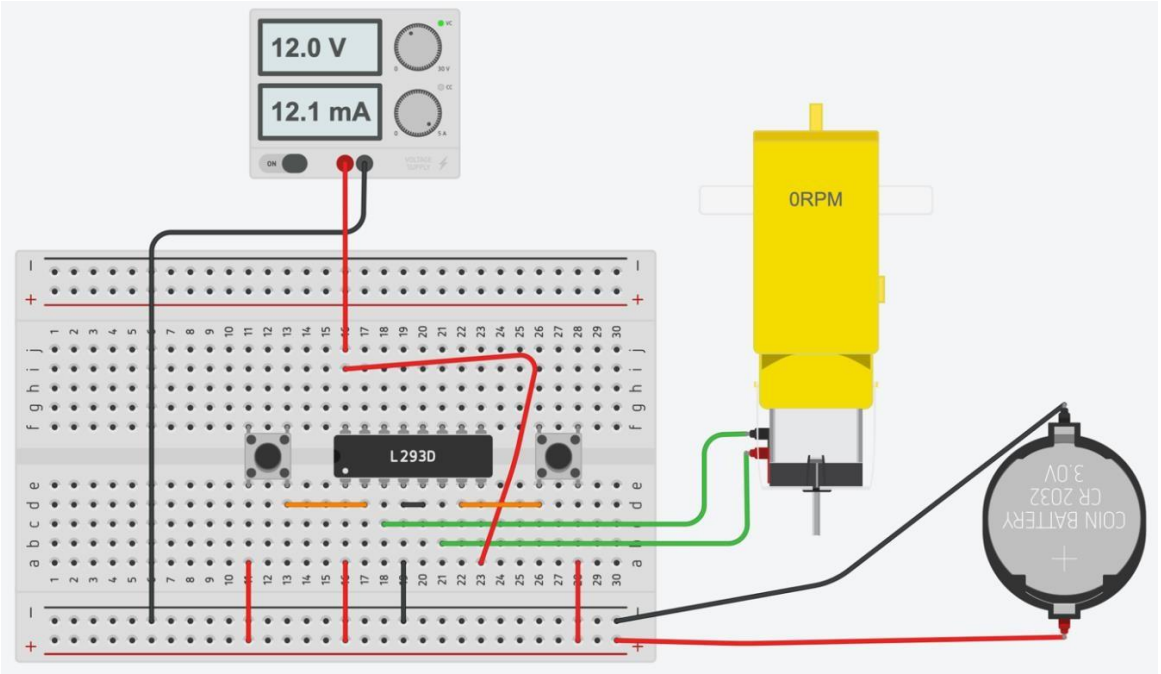
Şemayı şekle göre TINKERCAD'de NMOS ile bağlayın ve nasıl çalıştığını test edin. Güneş pili değerlerini değiştiriniz ve test ediniz. Direnç değerlerini değiştirin ve test edin.



Görev 5

DC motorların negatif (siyah) ve pozitif (kırmızı) olmak üzere iki kablosu vardır. Pozitif güç kaynağına ve negatif GND'ye bağlandığında, motor mili döner. Kablolar ters bağlandığında, pozitif GND'ye ve negatif güce, motor mili tekrar dönecektir, ancak ters yönde. Yön her değiştirildiğinde kabloları manuel olarak bağlamak zorunda kalmamak için H-köprüsü çipi kullanılır.

Şemayı H-köprüsü ve motor ile TINKERCAD'de şekle göre bağlayın ve test edin. Butonlara basınız ve motor milinin hangi yöne döndüğünü gözlemleyiniz.



Deney Sayısı: 4

4- Arduino'ya Giriş

Deney Hedefi

Bu deneyin amacı mikrodenetleyicinin ne olduğunu ve ne işe yaradığını açıklamaktır. Öğrenciler elektroniği Arduino'ya bağlamak için simülatör kullanmayı öğreneceklerdir.

Teorik Arka Plan

Bir mikrodenetleyici, ev ve ofis aletlerinin, robotların, araçların, tüketici elektroniğinin ve daha fazlasının işlevlerini kontrol etmek için kullanılabilen tek bir çip üzerinde küçük ve ucuz bir bilgisayardır. En ünlü mikrodenetleyici Arduino UNO'dur.

Görev 1

Tinkercad, Başlatıcılar menüsünde Arduino'yu seçin. Arduino ve elektronik ile 30'dan fazla demo şema ve program açılacaktır.

Bir demo programı seçin ve soruları yanıtlayın.

1. Hangi elektronik eleman kullanılıyor?
2. Eleman Arduino'nun hangi pinine bağlı?
3. Simülasyonu başlatın. Program ne yaptı?
4. Şemayı kopyalayın.
5. Kodu açın. Kullanılan kodu kopyalayın. Programın Blok ve Metin sürümlerinden aynı komutları bağlamayı deneyin.

Görev 2

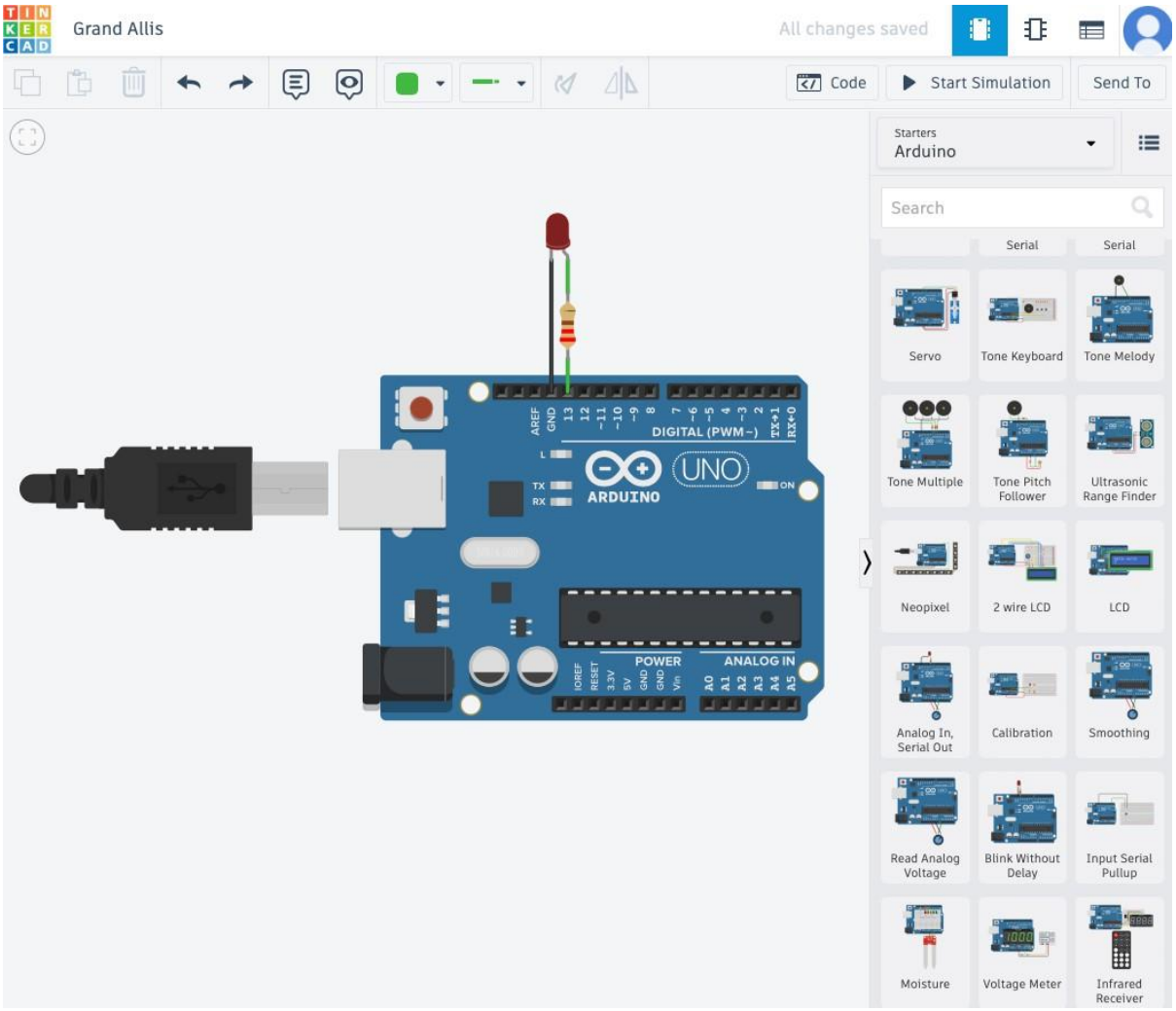
Tinkercad, Başlatıcılar menüsünde Arduino'yu seçin. Arduino ve elektronik ile 30'dan fazla demo şema ve program açılacaktır.

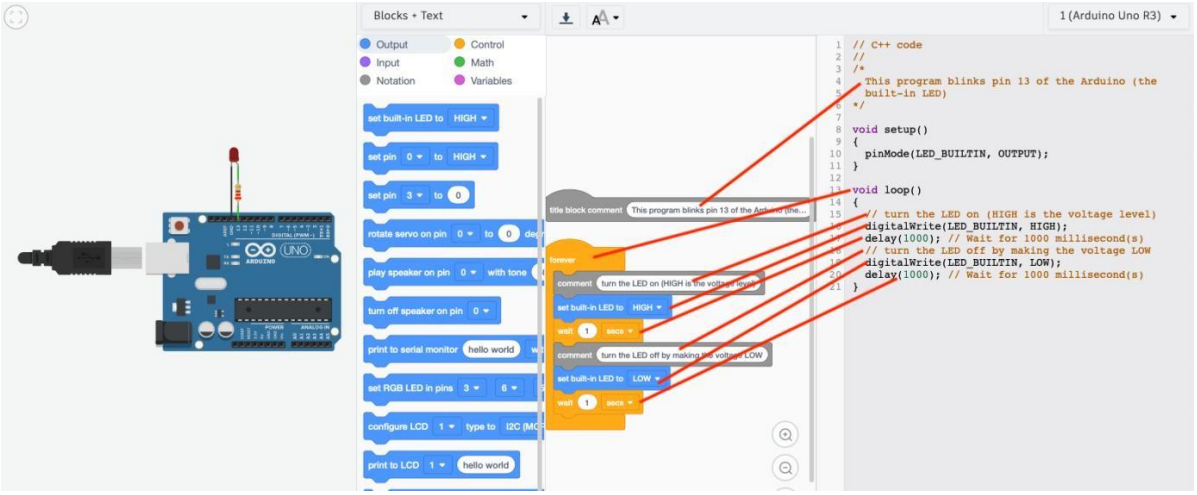
Bir demo programı seçin ve soruları yanıtlayın.

1. Hangi elektronik eleman kullanılıyor?
2. Eleman Arduino'nun hangi pinine bağlı?
3. Simülasyonu başlatın. Program ne yaptı?
4. Şemayı kopyalayın.
5. Kodu açın. Kullanılan kodu kopyalayın. Programın Blok ve Metin sürümlerinden aynı komutları bağlamayı deneyin.

Görev 1 ve Görev 2 için olası çözüm

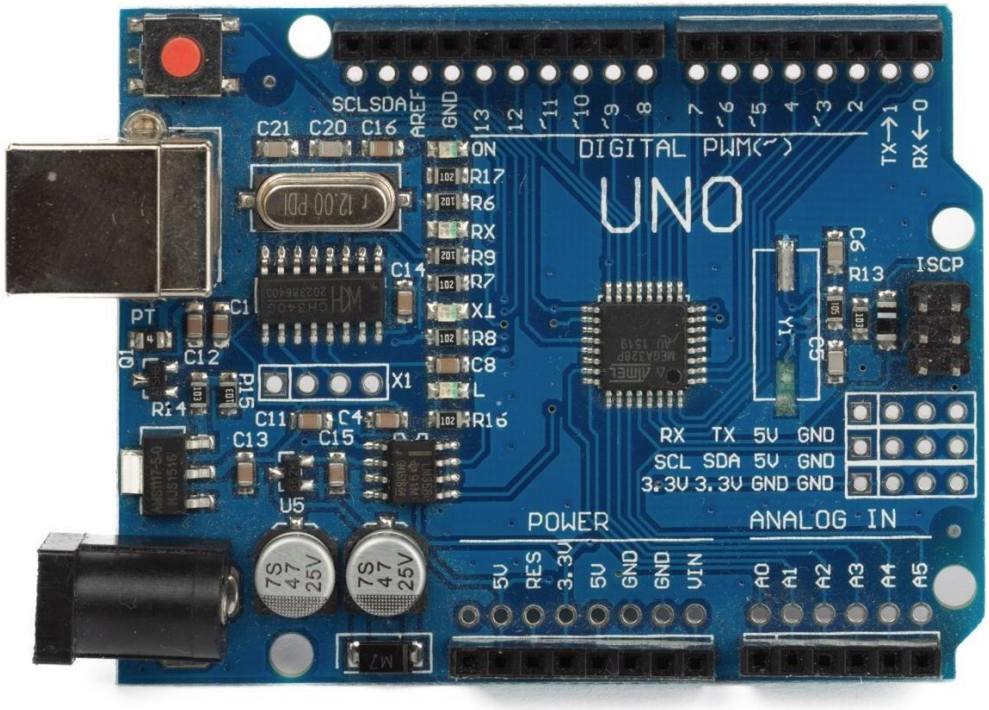
Bir örnek demo Blink'tir. LED ve direnç pin 13'e bağlanır. LED her 1 saniyede bir yanıp söner.


Grand Allis
All changes saved
Code
Start Simulation
Send To
Starters
Arduino
Search
Serial
Servo
Tone Keyboard
Tone Melody
Tone Multiple
Tone Pitch Follower
Ultrasonic Range Finder
Neopixel
2 wire LCD
LCD
Analog In, Serial Out
Calibration
Smoothing
Read Analog Voltage
Blink Without Delay
Input Serial Pullup
Moisture
Voltage Meter
Infrared Receiver
DIGITAL (PWM~)
POWER
ANALOG IN
UNO
ARDUINO


Blocks + Text
1 (Arduino Uno R3)
Output
Input
Notation
Control
Math
Variables
set built-in LED to HIGH
set pin 0 to HIGH
set pin 3 to 0
rotate servo on pin 0 to 0
play speaker on pin 0 with tone
turn off speaker on pin 0
print to serial monitor hello world
set RGB LED in pins 3 6
configure LCD 1 type to I2C
print to LCD 1 hello world
title block comment This program blinks pin 13 of the Arduino (the...
forever
comment turn the LED on (HIGH is the voltage level)
set built-in LED to HIGH
wait 1 secs
comment turn the LED off by making the voltage LOW
set built-in LED to LOW
wait 1 secs
// C++ code
//
/*
This program blinks pin 13 of the Arduino (the built-in LED)
*/
void setup()
{
pinMode(LED_BUILTIN, OUTPUT);
}
void loop()
{
// turn the LED on (HIGH is the voltage level)
digitalWrite(LED_BUILTIN, HIGH);
delay(1000); // Wait for 1000 millisecond(s)
// turn the LED off by making the voltage LOW
digitalWrite(LED_BUILTIN, LOW);
delay(1000); // Wait for 1000 millisecond(s)
}

Görev 3

Arduino UNO'nun fotoğrafında en önemli parçaları işaretleyin.



5- Arduino platformları

. Arduino Deneyi Nedir

Amaç

Amaç, sensörler, aktüatörler veya diğer bileşenleri içeren pratik bir uygulamayı sergilemek için Arduino veya benzer bir teknolojiyi kullanmaktır. Bu proje, bu teknolojilerin gerçek dünya senaryolarında nasıl uygulanabileceğini göstererek bir sorunu çözmeyi veya belirli bir görevi yerine getirmeyi amaçlamaktadır.

Teorik Arka Plan

Temel kavramlar, deneyle ilgili elektronik, programlama ve sistem tasarım ilkelerini içerir. Konular devre teorisi, dijital mantık, mikrodenetleyici mimarisi ve Arduino geliştirmede C/C++ kullanımını kapsayabilir ve başarılı proje uygulaması için gerekli kapsamlı bir anlayış sağlar.

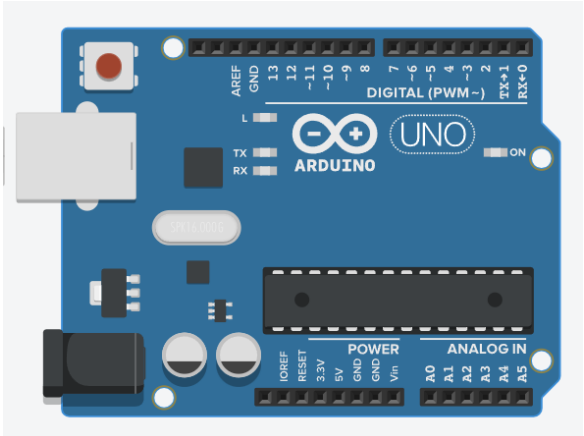
Görev 1

İşlevselliğini daha iyi anlamak için bir Arduino kartı kullanarak basit bir sıcaklık izleme sistemi oluşturun.

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 1 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Çözüm Prosedür

Adımları

1-Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2-Kodu yazın ve Arduino'ya yükleyin.

3-Seri bağlantı noktası ekranını izleyin.

4-Ekrandaki değişiklikleri gözlemleyin.

Kodlar

```
// Sıcaklık sensörünün bağlı olduğu giriş pinini tanımlayın int
sensorPin = A0;
void setup() {
  Serial.begin(9600); // Seri iletişimi başlatın
}
void loop() {
  int reading = analogRead(sensorPin); // Sensör değerini okuyun
  float voltage = reading * 5.0;
  voltage /= 1024.0; // Okunan değeri voltaja dönüştürün
  float temperatureC = (voltage - 0.5) * 100; // Voltajı Celsius cinsinden sıcaklığa dönüştürün
  Serial.print("Sıcaklık: ");
  Serial.print(sıcaklıkC);
  Serial.println(" C");
    delay(1000); // Döngüyü tekrarlamadan önce bir saniye bekleyin
}
```

Kuralların Açıklanması

Kod, sıcaklık sensörünün çıkışını bir Arduino üzerinde okur, gerilime dönüştürür, sıcaklığı Santigrat cinsinden hesaplar ve bunu Seri Monitörde sürekli olarak görüntüler.

6- Arduino programlama dili ve editörü-1

Deney Hedefi

Arduino ile yapılan deneyin amacı, çeşitli LED'leri kontrol etmek için basit bir program yazmak ve bunları belirli aralıklarla sırayla açıp kapatmaktır. Bu deney, Arduino'nun temel programlama yapılarını ve programlama yoluyla fiziksel dünya ile nasıl etkileşime girileceğini öğretmeyi amaçlamaktadır.

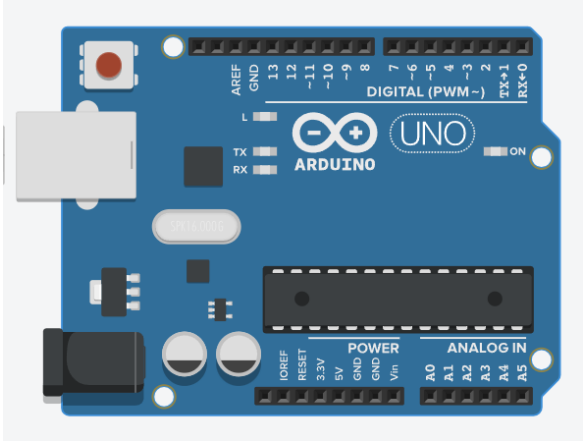
Teorik Arka Plan

Arduino'da kullanılan programlama dili, ortak giriş/çıkış işlemlerinde kullanım kolaylığı için tasarlanmış özelleştirilmiş bir C++ sürümü olan Wiring'e dayanmaktadır. Bu programlama dili ile ilgili önemli noktalar arasında nesne yönelimli programlama (OOP), taşınabilirlik, yüksek performans, standart kütüphane, çok yönlülük, çok paradigmalı programlama ve bellek yönetimi bulunmaktadır. Arduino programlama setup() ve loop() fonksiyonları etrafında döner. setup() pin modları gibi konfigürasyonları ayarlamak için programın başında bir kez çağrılırken, loop() zaman içinde tekrar eden ana program mantığını içerir ve programın mantığına göre LED'ler gibi donanımı kontrol eder.

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 2 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Prosedür Adımları

1- Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2- Kodu yazın ve Arduino'ya yükleyin.

3- Seri bağlantı noktası ekranını izleyin.

4- Ekrandaki değişiklikleri gözlemleyin.

Kodlar

İlk Kod Örneği: Led Kontrolü

```
int led1 = 4, led2 = 5, led3 = 6, led4 = 7;
void setup() {
  Serial.begin(9600);
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);
  pinMode(led4, OUTPUT);
}
void loop() {
  digitalWrite(led1, HIGH);
  delay(1000); // 1 saniye bekle
 digitalWrite(led1, LOW);
 delay(500); // 0,5 saniye bekle
 digitalWrite(led2, HIGH);
 delay(1000);
  digitalWrite(led2, LOW);
 delay(500);
 digitalWrite(led3, HIGH);
 delay(1000);
 digitalWrite(led3, LOW);
 delay(500);
 digitalWrite(led4, HIGH);
 delay(1000);
 digitalWrite(led4, LOW);
 delay(500);
}
```

Kuralların Açıklanması

Kod örneği, bir Arduino'ya bağlı dört LED'in sırayla nasıl açılıp kapatılacağını ve basit bir görsel desen oluşturulacağını gösterir. Her bir LED'in zamanlamasını ve durumunu kontrol etmek için pinMode(), digitalWrite() ve delay() gibi temel fonksiyonları kullanır.

İkinci Kod Örneği: Sensör Değer Okuması

```
void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorValue = analogRead(A0);
  Serial.println(sensorValue);
  delay(1); // Okunabilirlik için kısa gecikme
}
```

Kuralların Açıklanması

Kod örneği, Arduino'nun analog pini A0'a bağlı bir sensörden değerleri okumaya ve bu değerleri Seri Monitöre yazdırmaya odaklanır. Sensör verilerinin alınması ve kaydedilmesi için analogRead() ve Serial.println() işlevlerinin nasıl kullanılacağı gösterilmektedir.

7- Arduino programlama dili ve editör-2

Deney Hedefi

Amaç, Arduino'nun çeşitli sensörler ve bileşenlerle arayüz oluşturma yeteneklerini keşfetmektir. Bu, Arduino'nun çevresel izleme, robotik ve etkileşimli sanat enstalasyonları gibi pratik uygulamalar için nasıl kullanılabileceğini gösteren projeler oluşturmayı içerir.

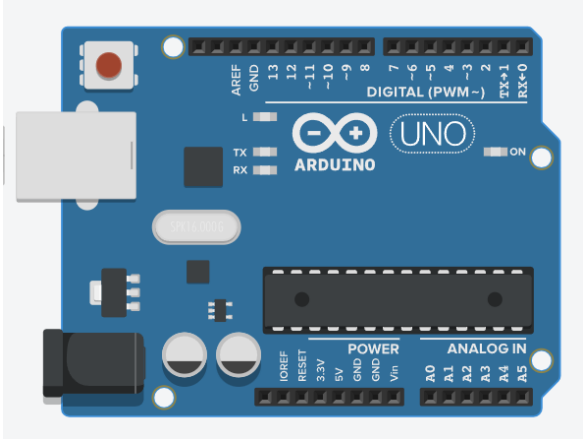
Teorik Arka Plan

Teorik temel, mikrodenetleyicilerin ilkelerini, özellikle Arduino'nun mimarisini ve programlamasını anlamayı içerir. Dijital ve analog giriş/çıkış, elektrik ve devrelerin temelleri, programlama yapıları (değişkenler, döngüler, koşullular) ve Seri ve I2C gibi iletişim protokollerini kapsar. Arka planda ayrıca elektronik ve programlama alanında yenilikçiliği ve eğitimi teşvik etmede açık kaynaklı donanım ve yazılımın önemine değinilmektedir.

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 3 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Görev 1.

Arduino IDE kullanarak elektronik devre tasarımı ve programlama ilkelerini gösteren Arduino tabanlı bir proje oluşturun.

Çözüm 1.

Çevre koşullarını ölçmek veya ışıkları veya motorları kontrol etmek gibi belirli işlevleri yerine getirmek için Arduino ile programlanan çeşitli sensörler ve aktüatörlerle etkileşime giren bir devre tasarlama.

Deney Sayısı: 8

8- Arduino IDE'de matematik işlem komutları: Aritmetik operatörler deneyi nasıl kullanılır

Deney Hedefi

Bu deneyin amacı, toplama, çıkarma, çarpma, bölme ve kalan gibi temel aritmetik operatörlerin Arduino Entegre Geliştirme Ortamında (IDE) nasıl kullanıldığını göstermektir. Öğrenciler bu operatörleri kullanmayı ve sonucu Arduino IDE'nin seri port ekranına yazdırmayı öğreneceklerdir.

Teorik Arka Plan

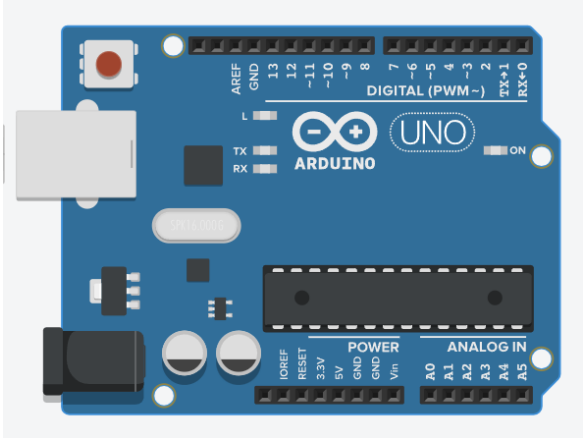
Tüm programlama dilleri bazı aritmetik operatörler içerir ve Arduino IDE de temel aritmetik operatörleri içerir. Bu operatörler Arduino'da bazı sensör verilerini işlemek gibi bazı matematiksel işlemleri yürütmek için kullanılabilir. Bu uygulamada gösterilecek 5 temel aritmetik operatör vardır.

Operatörler	Operatör Sembol	Değişkenler	Örnek	Sonuç
İlave	+	Sen1 = 21	Sonuç = Sen1 + Sen2	24
Çıkarma	-	Sen2 = 3	Sonuç = Sen1 - Sen2	18
Çarpma İşlemi	*	Sonuç = 0	Sonuç = Sen1 * Sen2	63
Bölüm	/		Sonuç = Sen1 / Sen2	7
Kalan	%		Sonuç = Sen1 % Sen2	0

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 4 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Prosedür Adımları

1-Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2-Kodu yazın ve Arduino'ya yükleyin.

3-Seri bağlantı noktası ekranını izleyin.

4-Ekrandaki değişiklikleri gözlemleyin.

Kodlar

```
/* Global değişken tanımı */ int
sensor1 = 21;
int sensör2 = 3;
int Sonuç = 0;
void setup(){
 Serial.begin(9600);
 //Ekleme
 Sonuç = sensör1 + sensör2;
 Serial.print("Sensör1 ve sensör2'nin toplamı ");
 Serial.println(Sonuç);
 gecikme (2000);
 //Çıkarma
 Sonuç = sensör1 - sensör2;
 Serial.print("Sensör1 ve sensör2'nin çıkarılması ");
 Serial.println(Sonuç);
 gecikme (2000);
 //Çarpma
 Sonuç = sensör1 * sensör2;
 Serial.print("Sensör1 ve sensör2'nin çarpımı ");
 Serial.println(Sonuç);
 gecikme (2000);
 //Bölüm
 Sonuç = sensör1 / sensör2;
 Serial.print("Sensör1 ve sensör2'nin bölümü ");
 Serial.println(Sonuç);
 gecikme (2000);
 //Remainder
 Sonuç = sensör1 % sensör2; Serial.print("Sensör1
 ve sensör2'nin kalanı "); Serial.println(Sonuç);
 gecikme (2000);
}

void loop() {
}
```

Kuralların Açıklanması

Kurulum: Seri monitör 9600 baud hızında etkinleştirilir. Önceden tanımlanmış değişkenler için ilgili matematiksel işlemler yalnızca bir kez gerçekleştirilir.

Döngü: -

Deney Sayısı: 9

9- Arduino IDE'de matematik işlem komutları: Aritmetik operatörler nasıl kullanılır deney 2

Deney Hedefi

Bu deneyin amacı, toplama, çıkarma, çarpma, bölme ve kalan gibi temel aritmetik operatörlerin Arduino Entegre Geliştirme Ortamında (IDE) nasıl kullanıldığını göstermektir. Öğrenciler bu operatörleri kayan noktalı sayı değişkenleri ile kullanmayı öğreneceklerdir.

Teorik Arka Plan

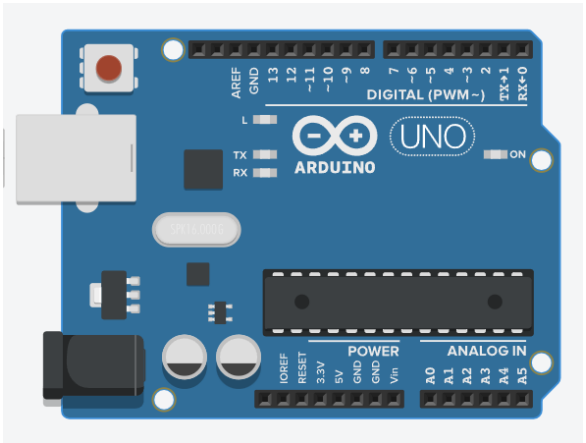
Aritmetik operatörlerin davranışı, kullanılan sayı değişkeninin türüne bağlı olarak değişir. Özellikle bölme operatörü kayan noktalı sonuç üretebilir. Bu nedenle tamsayı değişkenler üzerinde bölme operatörü kullanıldığında beklenen sonuç elde edilemeyebilir. Arduino kartları ile yapılan sensör uygulamalarında işlenen sayılar çoğunlukla kayan nokta tipindedir. Ayrıca modulo operatörü (kalan) sadece tamsayılar ile kullanılabilen bir operatördür.

Operatörler	Operatör Sembol	Değişkenler	Örnek	Sonuç
İlave	+	Sen1 = 24,5	Sonuç = Sen1 + Sen2	28.00
Çıkarma	-	Sen2 = 3,5	Sonuç = Sen1 - Sen2	21.00
Çarpma İşlemi	*	Sonuç = 0.0	Sonuç = Sen1 * Sen2	85.75
Bölüm	/	Sen3 = 12	Sonuç = Sen1 / Sen2	7.00
Kalan	%	Sen4 = 5	Sonuç = Sen3 % Sen4	2.00

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 5 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Prosedür Adımları

1-Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2-Kodu yazın ve Arduino'ya yükleyin.

3-Seri bağlantı noktası ekranını izleyin.

4-Ekrandaki değişiklikleri gözlemleyin.

Kodlar

```
//Global değişken
float sensor1 = 24.5;
float sensor2 = 3.5;
float Result = 0.0;
int seed = 123;
void setup()
{
  Serial.begin(9600);

  //Ekleme işlemi
  Sonuç = sensör1 + sensör2;
  Serial.print("Sensör1 ve sensör2'nin toplamı ");
  Serial.println(Sonuç);
  gecikme (2000);

  //Çıkarma işlemi Sonuç =
  sensör1 - sensör2;
  Serial.print("Sensör1 ve sensör2'nin çıkarılması ");
  Serial.println(Sonuç);
  gecikme (2000);

  //Çoğullama işlemi Sonuç =
  sensör1 * sensör2;
  Serial.print("Sensör1 ve sensör2'nin çarpımı ");
  Serial.println(Sonuç);
  gecikme (2000);

  / Bölme işlemi
  Sonuç = sensör1 / sensör2;
  Serial.print("Sensör1 ve sensör2'nin bölünmesi ");
  Serial.println(Sonuç);
  gecikme (2000);

  int sensör3 = 12, sensör4 = 5;
  //Toplama (modulo) işlemi Sonuç =
  sensör3 % sensör4;
  Serial.print("Sensör3 ve sensör4'ün kalanı ");
  Serial.println(Sonuç);
  gecikme (2000);

}

void loop()
{
  randomSeed(seed);
  sensor1 = random(1, 300);
```

```
  sensor2 = random(1, 300);
  Serial.println(sensor1);
  Serial.println(sensor2); seed
  = sensor1 * sensor2; Result
  = sensor1 + sensor2;
  Serial.print("Sensör verilerinin toplanması = ");
  Serial.println(Sonuç);
  Sonuç = sensör1 - sensör2; Serial.print("Sensör
  verilerinin çıkarılması = ");
  Serial.println(Sonuç);
  Sonuç = sensör1 * sensör2; Serial.print("Sensör
  verilerinin çarpımı = "); Serial.println(Sonuç);
  Sonuç = sensör1 / sensör2; Serial.print("Sensör
  verilerinin bölünmesi = ");
  Serial.println(Sonuç);
  Sonuç = (int)sensör1 % (int)sensör2;
  Serial.print("Sensör verilerinin geri kalanı = ");
  Serial.println(Sonuç);
}
```

Kuralların Açıklanması

Kurulum: Seri monitör 9600 baud hızında etkinleştirilir. Önceden tanımlanmış değişkenler için ilgili matematiksel işlemler yalnızca bir kez gerçekleştirilir.

Döngü: Basit bir simülasyon oyunu tasarlanmıştır. Bunun için Arduino editörünün random() ve randomSeed() yerleşik metotları kullanılmıştır. Her döngü fonksiyonu çalıştığında sensör 1 ve sensör 2 verileri belirli bir aralıkta tekrar rastgele üretilir. Her işlemde 5 matematik işlemi tekrar yapılır.

Deney Sayısı: 10

10- Arduino IDE'de kontrol yapıları: Kontrol yapıları nasıl kullanılır-1

Deney Hedefi

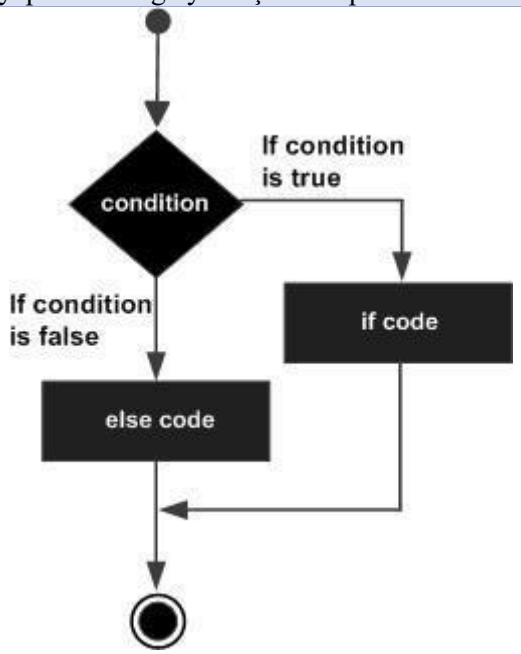
Bu deneyin amacı, Arduino Entegre Geliştirme Ortamında (IDE) kontrol yapılarının nasıl kullanıldığını göstermektir. Öğrenciler **karşılaştırma operatörlerini** ve if ***ifadelerini*** ("*if*", "if ... *else*", "if ... *else if* ... *else*") kullanmayı öğreneceklerdir.

Teorik Arka Plan

Tüm programlama dillerinde belirli koşullara göre karar vermek için bazı kontrol deyimleri vardır. Örneğin, belirli bir sıcaklık değerine göre yapılacak işleme karar verirken veya bir öğrencinin aldığı nota göre yapılacak işleme karar verirken bu kontrol deyimlerini kullanırız.

Diğer dillerde olduğu gibi Arduino IDE'de de kullandığımız çeşitli kontrol yapıları bulunmaktadır. Bu kontrol yapıları ve deyimler aşağıdaki gibi sıralanabilir:

Kontrol beyanı	Açıklama
"if" ifadesi	Parantez içinde bir ifade ve bir deyim veya blok alır. ifadeler. İfade doğruysa, deyim veya deyim bloğu yürütülür, aksi takdirde bu deyimler atlanır.
"if ... else" ifadesi	Bir **if** deyiminin ardından, ifade yanlış olduğunda çalıştırılan isteğe bağlı bir **else** deyimi gelebilir.
"if ... else if ... else" ifadesi	**if** deyiminin ardından, çeşitli koşulları test etmek için kullanılabilecek isteğe bağlı bir **else if** deyimi gelebilir.
"switch case" ifadesi	**if** deyimlerine benzer şekilde, **switch...case** akışı kontrol eder Programcıların çeşitli durumlarda çalıştırılması gereken farklı blokları belirtmesine izin vererek programlar.
Koşullu işleç "?:"	Bu, kullanıcının tek bir satırda iki durumlu bir karşılaştırma yapmasını sağlayan üçlü bir operatördür.



Şekil 6 if deyimlerinin akış diyagramı

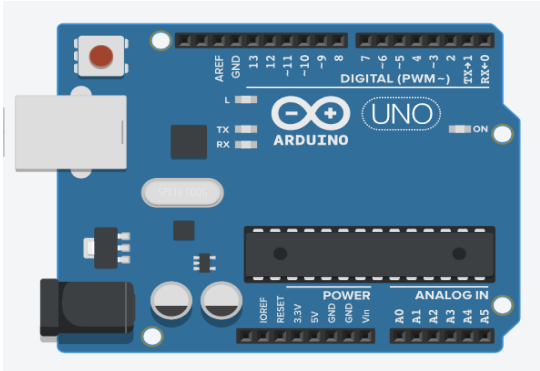
Kontrol deyimleri yazılırken iki değeri karşılaştırmak için bazı karşılaştırma operatörleri kullanılır. Karşılaştırma sonucunda **doğru (1)** veya **yanlış (0)** mantıksal bir çıktı elde edilir. PortB değişkeninin 63 ve portC değişkeninin 127 değerini tuttuğunu varsayalım,

Operatör isim	Operatör sembolü	Açıklama	Örnek
eşittir	= =	İki operandın değerinin eşit olup olmadığını kontrol eder, evet ise koşul doğru olur.	(portB == portC) **yanlıştır**
eşit değil	! =	İki operandın değerinin eşit olup olmadığını kontrol eder, değerler eşit değilse koşul doğru olur.	(portB != portC) **doğrudur**
daha az	<	Sol operandın değerinin sağ operandın değerinden küçük olup olmadığını kontrol eder, evet ise koşulu doğru hale gelir.	(portB < portC) **doğrudur**
daha büyük	>	Sol operandın değerinin daha büyük olup olmadığını kontrol eder sağ operandın değerinden büyükse, evet ise koşul doğru olur.	(portB > portC) **yanlıştır**
daha az veya eşit	< =	Sol operandın değerinin daha küçük olup olmadığını kontrol eder sağ operandın değerinden büyük veya eşitse, evet ise koşul doğru olur.	(portB <= portC) **doğrudur**
daha büyük veya eşittir	> =	Sol operandın değerinin sağ operandın değerinden büyük veya eşit olup olmadığını kontrol eder, eğer evet ise koşul doğru olur.	(portB >= portC) **yanlıştır**

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 7 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Prosedür Adımları

1-Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2-Kodu yazın ve Arduino'ya yükleyin.

3-Seri bağlantı noktası ekranını izleyin.

4-Ekrandaki değişiklikleri gözlemleyin.

Kodlar

```
/* Global değişken tanımı */ int
portB = 63;
int portC = 127;

void kurulum () {
  Serial.begin(9600);
  Serial.println(portB == portC);delay(1000);
  Serial.println(portB != portC);delay(1000);
  Serial.println(portB < portC);delay(1000);
  Serial.println(portB > portC);delay(1000);
  Serial.println(portB <= portC);delay(1000);
  Serial.println(portB >= portC);
}

void loop () {
  /* eğer blok */
  if(portB < portC) /* koşul doğruysa aşağıdaki deyimi çalıştırın*/ portB++;
  delay(200);
  Serial.print("PORTB= ");Serial.println(portB);

  /* if ... else bloğu */
  if(portB == portC){
    Serial.println("PortB, PortC'ye eşittir");
    portB++;
  }
  else{
    Serial.println("PortB, PortC'ye eşit değil");
  }

  /* if ... else if ... else bloğu */
  if(portB > portC){
    Serial.println("PortB, PortC'den büyüktür");
  }
  else if (portB < portC){
    Serial.println("PortB, PortC'den
    küçüktür");
  }
  else{
    Serial.println("PortB, PortC'ye eşittir");
  }

}
```

Kuralların Açıklanması

Kurulum: Seri monitör 9600 baud hızında etkinleştirilir. Önceden tanımlanmış değişkenler için ilgili karşılaştırma işlemleri yalnızca bir kez gerçekleştirilir.

Döngü: Kod bloklarını sürekli olarak içinde çalıştıran sonsuz döngü.

Deney Sayısı: 11

11- Arduino IDE'de kontrol yapıları: Kontrol yapıları nasıl kullanılır-2

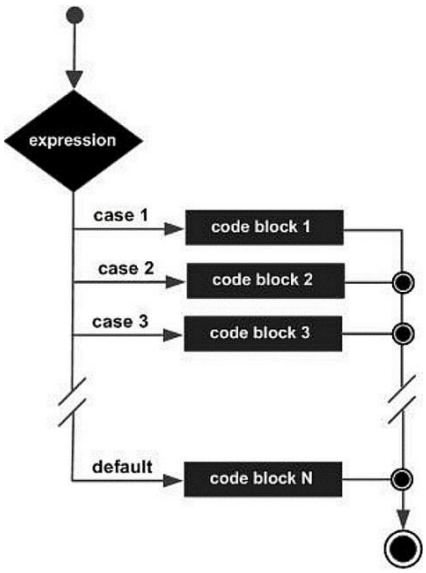
Deney Hedefi

Bu deneyin amacı, Arduino Entegre Geliştirme Ortamında (IDE) kontrol yapılarının nasıl kullanıldığını göstermektir. Öğrenciler ***switch case*** **deyimini** ve ***koşullu operatör*** "?:" kullanmayı öğreneceklerdir.

Teorik Arka Plan

if deyimlerine benzer şekilde **switch...case,** programcıların çeşitli durumlarda çalıştırılması gereken farklı kodları belirtmesine olanak tanıyarak programların akışını kontrol eder. Özellikle, **switch** deyimi, bir değişkenin değerini **case** deyimlerinde belirtilen değerlerle karşılaştırır.

Her **case** deyiminin sonuna bir **break** anahtar sözcüğü yazılır. Aksi takdirde, **case deyimindeki** koşuldan bağımsız olarak her **switch** deyimi çalıştırılır. Switch içinde case eşleşmesi yoksa çalıştırmak için **default** anahtar sözcüğü kullanılır.

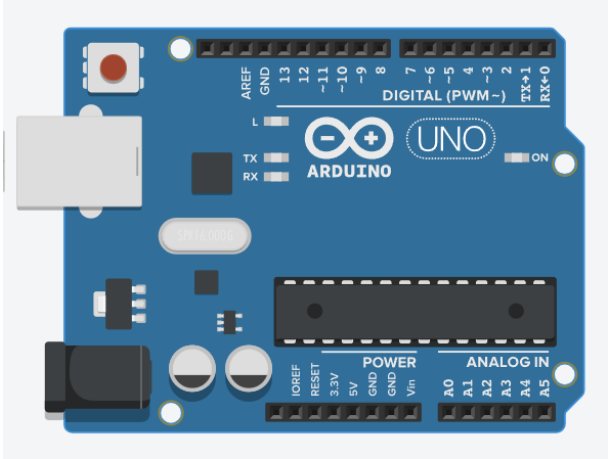


Şekil 8 switch case deyiminin akış diyagramı

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 9 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Prosedür Adımları

1-Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2-Kodu yazın ve Arduino'ya yükleyin.

3-Seri bağlantı noktası ekranını izleyin.

4-Ekrandaki değişiklikleri gözlemleyin.

Kodlar - 1

```
/* Global değişken tanımı */
byte portC = 1;
bool left = true;
void setup(){
Serial.begin(9600);
}
void loop(){
 switch (portC) {
  case 1: Serial.println("Birinci LED çalıştırıldı."); break;
  case 2: Serial.println("İkinci LED çalıştırıldı."); break;
  case 4: Serial.println("Üçüncü LED çalıştırıldı."); break;
  case 8: Serial.println("Dördüncü LED çalıştırıldı.");
  break; case 16: Serial.println("Beşinci LED çalıştırıldı.");
  break; case 32: Serial.println("Altıncı LED çalıştırıldı.");
  break; case 64: Serial.println("Yedinci LED çalıştırıldı.");
  break; case 128: Serial.println("Sekizinci LED
  çalıştırıldı."); break; default: Serial.println("Geçersiz
  durum!");
 }
 if(sol)
   portC = portC << 1; // Bit kaydırma operatörü. Tüm bitler bir kez sola kaydırılır.
 başka
   portC = portC >> 1; // Bit kaydırma operatörü. Tüm bitler bir kez sağa kaydırılır.
 if(portC == 0 && left){ // if bloğu ile çoklu karşılaştırma portC
   = 128;
   sol = yanlış;
   Serial.println("**********************");
 }
 else if(portC == 0 && !left){ // else if bloğu ile çoklu karşılaştırma portC =
   1;
```

```
    sol = doğru;
    Serial.println("**********************");
  }
  gecikme(1000);
}
```

Kuralların Açıklanması

Kurulum: Seri monitör 9600 baud hızında etkinleştirilir.

Döngü: Switch case bloğu çalıştırılır. Boolean değişkeni maksimum veya minimum bit değerine göre değiştirilir, böylece tarama soldan sağa veya sağdan sola sürekli olarak yapılır.

Kodlar - 2

```
/* Global değişken tanımı */ int
portB = 63;
int portC = 127;
char c = 'k';

void setup() {
  Serial.begin(9600);

  /* max(portB, portC) değerini bulun: */
  Serial.println((portB>portC)?portB:portC);/*PortB'nin değeri portC'den büyükse yazdırılır. Aksi takdirde
portC'nin değeri yazdırılır.*/

  /* Küçük harfi büyük harfe dönüştürün: */
  c = ( c >= 'a' && c <= 'z' ) ? ( c - 32 ) : c;
  Serial.println(c);
}

void loop() {

}
```

Kuralların Açıklanması

Kurulum: Seri monitör 9600 baud hızında etkinleştirilir. Hangi girişin diğerinden büyük olduğunu belirlemek için koşul operatörü kullanılır. Verilen bir küçük harfin ASCII karakter tablosu değerine göre büyük harfe dönüştürülmesi de koşul operatörü ile kolayca yapılır.

Döngü: -

Deney Sayısı: 12

12- Arduino IDE'de kontrol yapıları: Kontrol yapıları nasıl kullanılır-3

Deney Hedefi

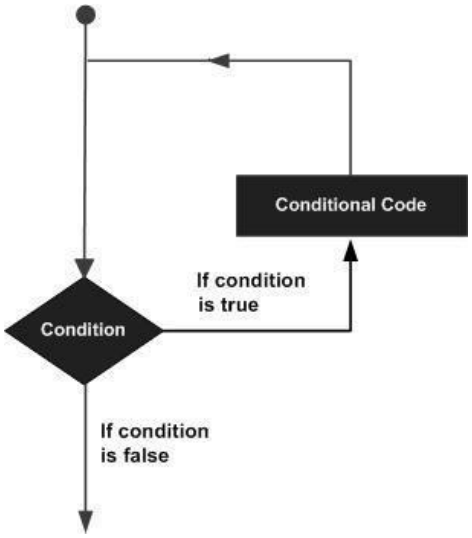
Bu deneyin amacı, Arduino Entegre Geliştirme Ortamında (IDE) kontrol yapılarının nasıl kullanıldığını göstermektir. Öğrenciler ***for***, ***while*** ve ***do while*** döngülerini kullanmayı öğreneceklerdir.

Teorik Arka Plan

for deyimi, küme parantezleri içine alınmış bir deyim bloğunu tekrarlamak için kullanılır. Döngüyü artırmak ve sonlandırmak için genellikle bir artış sayacı kullanılır. **for deyimi** tekrarlayan her türlü işlem için kullanışlıdır. Özellikle Arduino kartların port pinlerinin taranmasında bu yapı sıklıkla tercih edilir. ***for döngü*** başlığının üç bölümü vardır:

```
for (başlatma; koşul; artış) {
//ifade(ler);
}
```

"***while***" döngüleri, parantez () içindeki ifade yanlış olana kadar sürekli ve sonsuza kadar dönecektir. Koşul sağlandığı sürece döngü çalışır. Bu nedenle sonsuz döngüden kaçınmak için dikkatli kullanılmalıdır.



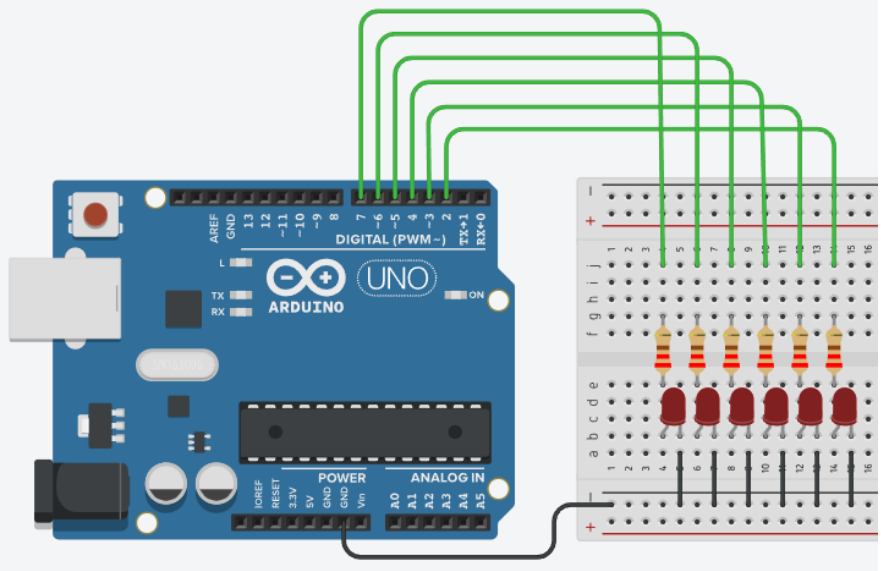
Şekil 10 Döngülerin çalışması

"***do ... while***" döngüsü ***while*** döngüsüne benzer, ancak koşul sağlansın ya da sağlanmasın en az bir kez çalışır.

Gerekli Malzemeler

Arduino UNO R3, 6 adet 220 Ω direnç, 6 LED, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 11 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Prosedür Adımları

1-Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2-Kodu yazın ve Arduino'ya yükleyin.

3-LED'lerin davranışlarını izleyin.

4-Seri port ekranındaki değişiklikleri gözlemleyin.

Kodlar

```
// Küresel değişkenler
// iki boyutlu diziler tanımlanır
int sensors01[3][3]={3,-7,7,8,5,-4,2,3,0};
int sensors02[3][3]={4,3,0,2,-1,-4,8,9,5};
int sensor3=50;
int sensor4=100;
int counter = 1;
void setup () {
  Serial.begin(9600);
}

 void loop () {
  PORTD = 4;
  gecikme(250);
  // for döngüsü başlar
  for(int i=0;i<6;i++){
    PORTD = PORTD << 1;
gecikme(250);
  }
  PORTD = 128;
  // diğer for döngüsü başlar
  for(int i=0;i<6;i++){
  PORTD = PORTD >> 1;
```

```
  gecikme(250);
 }

 //iç içe for döngüleri
 for(int i=0;i<=2;i++){//i değişkeni
   for(int j=0;j<=2;j++){// j değişkeni
     Serial.print(sensors01[i][j] + sensors02[i][j]);
     Serial.print("\t");//tab operatörü
   }
   Serial.println();//Taşıma dönüşü
 }

 // while döngüsü
 while(sensor3 < sensor4){
 Serial.print("I run ");
 Serial.print(counter);
 Serial.println(". times.");
 counter++;
   sensor3 += 10; // unary toplama operatörü
   delay(200);
 }

 //do...while döngüsü
 do{
   delay(200);
   Serial.println("En az bir kez çalıştırıyorum...");
 }while (sensör3 > sensör4);
}
```

Kuralların Açıklanması

Kurulum: Seri monitör 9600 baud hızında etkinleştirilir.

Döngü: Arduino kartının PORTD'sindeki LED'leri sürmek için iki ***for döngüsü*** çalıştırılır. İki boyutlu dizilerin nasıl taranacağını göstermek için iç içe ***for döngüsü çalıştırılır.*** "***while***" ve "***do...while***" döngüleri iki değerin karşılaştırılmasını test etmek için çalıştırılır.

Deney Sayısı: 13

Deney Adı: Arduino IDE'de kontrol yapıları: Kontrol yapıları nasıl kullanılır 4 Deneyin Amacı

Bu deneyin amacı, Arduino Entegre Geliştirme Ortamında (IDE) kullanıcı tanımlı ve yerleşik fonksiyonların nasıl kullanıldığını göstermektir. Öğrenciler kullanıcı tanımlı fonksiyonların bildirimini ve bazı önemli yerleşik fonksiyonların kullanımını öğreneceklerdir.

Teorik Arka Plan

Şimdiye kadar deneyimlediğiniz gibi, Arduino IDE'de setup(), loop() ve delay() gibi bazı temel yerleşik fonksiyonları kullandık. Bir fonksiyon, kullanıcının bireysel görevleri yerine getirmek için programları kod bölümleri halinde yapılandırmasına olanak tanır. Arduino geliştirme ortamı iki ana fonksiyon gerektirir setup() ve loop().

Bir fonksiyon tanımlamak için en yaygın sözdizimi aşağıdaki gibidir:

```
return_type function_name(argument1, argument2, ...) {
    statements;
    return_type_value;
}
```

return_type tamsayı, float, char vb. gibi bir veri türüdür. Her fonksiyon bir değer döndürmeyi gerektirmez. Böyle bir durumda fonksiyonun *return_type* değeri **void** olarak tanımlanır. Argümanlar, uygun veri tipine sahip herhangi bir değişken veya değerdir. Bir argümanın veri tipi tanımlanan tip ile eşleşmelidir.

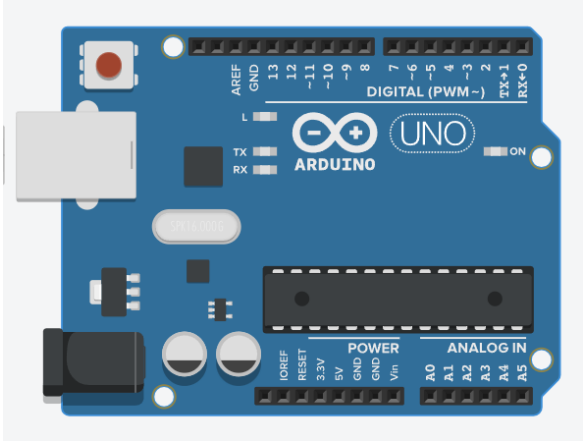
Bir fonksiyon diğer fonksiyonların dışında, *döngü* ve *kurulum* fonksiyonlarının üstünde veya altında bildirilir. Fonksiyonun kendisini yazıyorsanız, fonksiyonu *döngü* veya *kurulum fonksiyonlarından* önce (çağrıldığı yerde) bildirmeniz gerekir.

Kurulum veya döngü işlevlerinden sonra bir işlev bildirmek istiyorsanız, çağrıldığı diğer işlevlerin üzerine bir işlev **prototipi** yazmanız gerekir. Fonksiyon prototipinin ardından noktalı virgül (;) gelmelidir.

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 12 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Prosedür Adımları

1-Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2-Kodu yazın ve Arduino'ya yükleyin.

3-Seri bağlantı noktası ekranını izleyin.

4-Ekrandaki değişiklikleri gözlemleyin.

Kodlar

```
// Küresel değişkenler
int sum_func(int x, int y);// kullanıcı tanımlı bir fonksiyon
prototipi int sensor1=21;
int sensor2=45;

// Kullanıcı tanımlı bir işlevin çağrılmadan önce bildirilmesi
void printMessage(){
  Serial.println("Bu fonksiyon setup() fonksiyonunun üzerinde bildirilmiştir");
  Serial.print("Sensörlerin farkı ");
  Serial.println(sensör1-sensör2);
}

void kurulum () {
  Serial.begin(9600);
  printMessage(); //kullanıcı tanımlı fonksiyon çağrılır
}

void loop () {
  if(sum_func(sensor1, sensor2)>=50){
    Serial.print("Sensörlerin toplamı, ");
    Serial.println(sum_func(sensor1, sensor2));
  }
  else{
    Serial.println("Sensörlerin toplamı 50'den küçük");
  }
  sensör2 = 25;
  delay(1000); // Sensör verileri her saniye okunur.
```

```
}

// Çağrıldığı loop() işlevinden sonra işlevin
deklerasyonu. int sum_func(int x, int y){
  int z = 0;
  z = x + y;
  return z;
}
```

Kuralların Açıklanması

Kurulum: Seri monitör 9600 baud hızında etkinleştirilir. Kullanıcı tanımlı bir fonksiyon çağrılır.

Döngü: Kullanıcı tanımlı bir toplama işlevi çağrılır. Her saniye bir tarama gerçekleştirilir.

Deney Sayısı: 14

14- Arduino IDE'de Dizeler: Dize değişmezleri nasıl kullanılır

Deney Hedefi

Bu deneyin amacı, string veri tipinin Arduino Entegre Geliştirme Ortamında (IDE) nasıl kullanıldığını göstermektir. Öğrenciler string veri tipinin farklı bildirim yöntemlerini öğreneceklerdir.

Teorik Arka Plan

Diğer tüm programlama dillerinde olduğu gibi Arduino IDE'de de "string" adı verilen ve üzerinde matematiksel işlem yapılamayan önemli bir veri tipi vardır. Stringler metin tabanlı bir veri tipidir. Metin dizeleri iki şekilde temsil edilebilir. ***String*** veri tipini kullanabilir veya ***char tipinde*** bir diziden bir string oluşturabilir ve null-terminate ('\0') yapabilirsiniz.

- Dize Sözdizimi

Aşağıdakilerin tümü dizeler için geçerli bildirimlerdir.

```
char Str1[10];
char Str2[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o'};
char Str3[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o', '\0'};
char Str4[] = "arduino";
char Str5[7] = "arduino";
char Str6[10] = "arduino";
```

- Dizeleri bildirmek için olasılıklar
  - Str1'de olduğu gibi bir karakter dizisini başlatmadan bildirme
  - Bir karakter dizisi (fazladan bir karakter ile) bildirdiğinizde derleyici Str2'de olduğu gibi gerekli null karakterini ekleyecektir
  - Açıkça null karakterini ekleyin, Str3
  - Tırnak içinde bir dize sabiti ile başlatılır; derleyici diziyi dize sabitine ve sonlandırıcı null karakterine sığacak şekilde boyutlandırır, Str4
  - Diziyi açık bir boyut ve Str5 dize sabiti ile başlatın
  - Diziyi başlatın, daha büyük bir dize için fazladan alan bırakın, Str6
- Boş Sonlandırma

Genel olarak, dizeler null karakteri (ASCII kodu 0) ile sonlandırılır. Bu, fonksiyonların (**Serial.print()** gibi) bir dizenin sonunun nerede olduğunu anlamasını sağlar. Aksi takdirde, aslında dizenin bir parçası olmayan sonraki bellek baytlarını okumaya devam ederler.

- Tek Tırnak veya Çift Tırnak

Dizeler her zaman çift tırnak ("Arduino") içinde tanımlanır ve karakterler her zaman tek tırnak ('A') içinde tanımlanır.

- Uzun ipleri sarma

Uzun dizeleri bu şekilde sarabilirsiniz:

```
char myString[] = "Bu ilk satır"
                  "bu ikinci satır" "bu son
                  satır";
```

- Dizeler dizisi

Dize dizisi, LCD ekranlı projeler gibi büyük miktarda mesaj metinleriyle çalışırken kullanılır. Aynı mesajları projenin çeşitli yerlerine yazdırabilirsiniz. Bu gibi durumlarda bir mesaj serisi hazırlamak mantıklı olacaktır.

```
char *sensorStrings[] = {"Bu ilk sensör", "Bu
                         ikinci sensör", "Bu üçüncü sensör", "Bu dördüncü sensör", "Bu
                         beşinci sensör", "Bu altıncı sensör6"
                        };
```

Aşağıdaki kodda, **char** "**char***" veri türünden sonra gelen yıldız işareti, bunun bir "işaretçi" dizisi olduğunu gösterir.

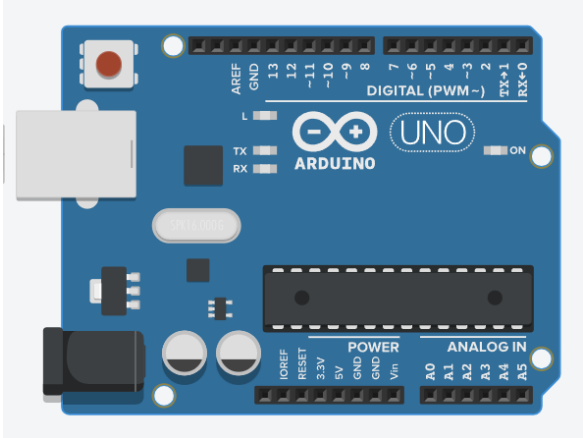
- Dize Birleştirme

Char veri tipi ile oluşturulan stringler aritmetik operatör ile birleştirilemez. Bu işlem için **String** veri yapısı kullanılmalıdır.

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 13 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Prosedür Adımları

1- Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2- Kodu yazın ve Arduino'ya yükleyin.

3- Seri bağlantı noktası ekranını izleyin.

4- Ekrandaki değişiklikleri gözlemleyin.

Kodlar

```
// Küresel değişkenler
char Str1[10]; // başlatıldı ancak ilk değer ataması yapılmadı
char Str2[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o'}; /* ilk değer atamasıyla başlatılır, null sonlandırma otomatik olarak
eklenir */
```

```
char Str3[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o', '\0'}; /* ilk değer ataması ile başlatılır, null sonlandırma manuel olarak
eklenir */
char Str4[] = "arduino"; //dizi boyutu atanan değer tarafından
oluşturulur. char Str5[8] = "arduino"; //dizi boyutu ve atama birlikte
yapılır. char Str6[10] = "arduino"; //dizi boyutu atanan değerden
büyüktür. char myString[] = "Bu ilk satır"
"bu ikinci satır" "bu son
satır";

char *sensorStrings[] = {"Bu birinci sensör", "Bu ikinci sensör", "Bu üçüncü sensör", "Bu dördüncü sensör",
"Bu beşinci sensör", "Bu altıncı sensör6"
                };

void setup () {
  Serial.begin(9600);
  Serial.println(Str1);delay(500);
  Serial.println(Str2);delay(500);
  Serial.println(Str3);delay(500);
  Serial.println(Str4);delay(500);
  Serial.println(Str5);delay(500)
  ;
  Serial.println(Str6);delay(500)
  ; Serial.println(myString);
}

void loop () {
  for (int i = 0; i < 6; i++) {
    Serial.println(sensorStrings[i]);
    delay(500);
  }
}
```

Kuralların Açıklanması

Kurulum: Seri monitör 9600 baud hızında etkinleştirilir. Farklı dize değişmezleri yazdırılır.

Döngü: Bir string dizisi for döngüsü kullanılarak tarama tekniği ile yazdırılır.

Deney Sayısı: 15

15- Arduino IDE'de Stringler: String veri yapısı nasıl kullanılır

Deney Hedefi

Bu deneyin amacı String() nesnesinin (veri yapısı) Arduino Entegre Geliştirme Ortamında (IDE) nasıl kullanıldığını göstermektir. Öğrenciler String() nesnesinin farklı bildirim yöntemlerini öğreneceklerdir.

Teorik Arka Plan

Arduino IDE nesne yönelimli bir programlama yapısı kullanır ve Arduino platformlarında kullanılan pek çok nesne türü vardır. String() nesnesi bunların en popülerlerinden biridir. Stirng() bir tür Sınıf veri yapısıdır. String sınıfının bir örneğini oluşturmanın birden fazla versiyonu vardır.

- çift tırnak içinde sabit bir karakter dizisi (yani bir char dizisi)
- tek tırnak içinde tek bir sabit karakter
- String nesnesinin başka bir örneği
- sabit bir tamsayı veya uzun tamsayı
- belirtilen bir tabanı kullanan sabit bir tamsayı veya uzun tamsayı
- bir tamsayı veya uzun tamsayı değişkeni
- belirtilen bir tabanı kullanan bir tamsayı veya uzun tamsayı değişkeni
- belirtilen ondalık basamakları kullanarak bir float veya double

Bir sayıdan bir String oluşturmak, o sayının ASCII gösterimini içeren bir string ile sonuçlanır. Varsayılan değer on tabanıdır, yani

```
String thisString = String(15);
```

size "15" dizesini verir. Ancak başka tabanlar da kullanabilirsiniz. Örneğin,

```
String thisString = String(15, HEX);
```

size ondalık değer olan 15'in onaltılık gösterimi olan "f" dizesini verir. Ya da ikiliyi tercih ederseniz,

```
String thisString = String(15, BIN);
```

size 15'in ikili gösterimi olan "1111" dizesini verir.

- **Sözdizimi**

```
String(val)
String(val, base)
String(val, decimalPlaces)
```

- **Parametreler**

val: String olarak biçimlendirilecek bir değişken. İzin verilen veri türleri: ***string***, ***char***, ***byte***, ***int, long, unsigned int***, ***unsigned long, float***, ***double***.
base: (isteğe bağlı) integral değerin biçimlendirileceği taban.
decimalPlaces: yalnızca val float veya double ise. İstenen ondalık
basamaklar.

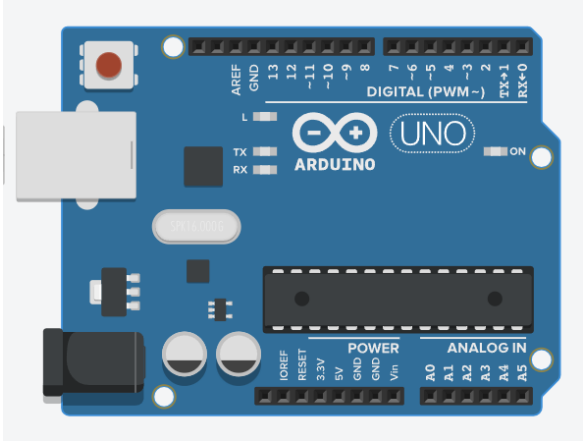
- Dize Birleştirme

Dizelerin birbirine eklenebilmesi için öncelikle bir String nesnesi olarak tanımlanmaları ve farklı veri türlerini birleştirmeye başlamadan önce bir ilk değer almaları gerekir.

Gerekli Malzemeler

Arduino UNO R3, Arduino IDE yazılımı veya Tinkercad simülasyon portalı.

Deney Devresi



Şekil 14 Arduino UNO R3 geliştirme kartı

Arduino kartını USB A/B tipi kablo ile bilgisayara bağlayın ve Arduino IDE'yi açın. Arduino'nun bağlı olduğu portu (Tools→Port) seçin. Araçlar menüsünden Seri Monitör Ekranını açın.

Prosedür Adımları

1-Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2-Kodu yazın ve Arduino'ya yükleyin.

3-Seri bağlantı noktası ekranını izleyin.

4-Ekrandaki değişiklikleri gözlemleyin.

Kodlar

```
// Global
değişkenler int
intSensor = 15;
float floatSensor = 13.495;
String stringString = "Hello String"; // sabit bir String kullanarak
String stringCharacter = String('a'); // sabit bir karakteri String'e dönüştürme
String stringObject = String("This is a string"); // sabit bir dizeyi String nesnesine dönüştürme
String stringConcatenation = String(stringString + " with more"); // iki dizeyi birleştirme String
stringInteger = String(15); // sabit bir tamsayı kullanma
String stringDEC = String(intSensor, DEC); // bir int ve bir taban kullanarak
String stringHEX = String(intSensor, HEX); // bir int ve bir taban (onaltılık) kullanarak
String stringBIN = String(intSensor, BIN); // bir int ve bir taban (ikilik) kullanarak
String stringLongDEC = String(millis(), DEC);// bir uzun ve bir taban kullanarak
String stringFloat = String(floatSensor, 2);// iki ondalık hassasiyete sahip bir float kullanarak

String stringOne, stringTwo, stringThree;
void setup () {
  Serial.begin(9600);
}

void loop () {
  Serial.println(stringString);delay(500);
```

```
  Serial.println(stringCharacter);delay(500);
  Serial.println(stringObject);delay(500);
  Serial.println(stringConcatenation);delay(500);
  Serial.println(stringInteger);delay(500);
  Serial.println(stringDEC);delay(500);
  Serial.println(stringHEX);delay(500);
  Serial.println(stringBIN);delay(500);
  Serial.println(stringLongDEC);delay(500);
  Serial.println(stringFloat);delay(500);
  stringLongDEC = String(millis(), DEC);
  Serial.println(stringLongDEC);

  stringOne = String("You added ");
  stringTwo = String("this string");
  stringThree = String();
  // bir String'e sabit bir tamsayı ekleme:
  stringThree = stringOne + 123;
  Serial.println(stringThree);delay(500);
  // bir String'e sabit bir uzun tamsayı ekleme:
  stringThree = stringOne + 123456789;
  Serial.println(stringThree);delay(500);
  // bir String'e sabit bir karakter ekleme:
  stringThree = stringOne + 'A';
  Serial.println(stringThree);delay(500);
  // bir String'e sabit bir string eklemek:
  stringThree = stringOne + "abc";
  Serial.println(stringThree);delay(500);
  stringThree = stringOne + stringTwo;
  Serial.println(stringThree);delay(500);
  // bir String'e değişken bir tamsayı
  ekleme: int sensorValue =
  analogRead(A0); stringOne = "Sensör
  değeri: "; stringThree = stringOne +
  sensorValue;
  Serial.println(stringThree);delay(500);
  // bir String'e değişken bir uzun tamsayı
  ekleme: stringOne = "millis() değeri: ";
  stringThree = stringOne + millis();
  Serial.println(stringThree);delay(500);
  while(true); //sonsuz döngü, hiçbir şey
  yapılmadı.
}
```

Kuralların Açıklanması

Kurulum: Seri monitör 9600 baud hızında etkinleştirilir.

Döngü: Kullanıcı tanımlı bir toplama işlevi çağrılır. Her saniye bir tarama gerçekleştirilir.

Deney Sayısı: 16

16- Dijital I/O işlemleri

Deney Hedefi

Bu deneyin birincil amacı, öğrencilere Arduino kullanarak temel dijital giriş/çıkış (I/O) işlemlerini tanıtmaktır. Katılımcılar, girişleri (düğmeler gibi) okumak ve çıkışları (LED'ler gibi) kontrol etmek için Arduino kartındaki dijital pinleri nasıl yapılandıracaklarını ve kullanacaklarını öğreneceklerdir.

Teorik Arka Plan

Arduino platformu, giriş veya çıkış olarak yapılandırılabilen dijital pinler ile donatılmıştır. Bu dijital pinler dijital sinyalleri okumak ve fiziksel cihazları kontrol etmek için kullanılır. Giriş olarak yapılandırıldıklarında, bir sinyalin varlığını veya yokluğunu tespit edebilirler (YÜKSEK veya DÜŞÜK). Çıkış olarak yapılandırıldıklarında, pimi YÜKSEK (genellikle 5V) veya DÜŞÜK (0V) olarak ayarlayarak LED'ler, motorlar vb. gibi cihazları kontrol etmelerini sağlarlar.

Dijital Giriş: Dijital bir sinyalin durumunu okuma. Genellikle düğmeler, anahtarlar ve sensörlerle kullanılır.

Dijital Çıkış: LED'ler veya röleler gibi cihazları kontrol etmek için bir dijital pinin YÜKSEK veya DÜŞÜK olarak ayarlanması.

Arduino'nun pinMode(), digitalWrite() ve digitalRead() fonksiyonları dijital I/O işlemleri için temeldir:

pinMode(pin, mode): Bir pini INPUT, OUTPUT veya INPUT_PULLUP

olarak ayarlar. digitalWrite(pin, value): Bir dijital pine YÜKSEK veya

DÜŞÜK değer yazar. digitalRead(pin): Bir dijital pinden değeri okur.

Gerekli Malzemeler

Arduino UNO R3 kartı

Breadboard

LED

220 ohm direnç Buton

anahtarı Jumper

kabloları Arduino

IDE yazılımı

Deney Devresi

1-Akımı sınırlamak için LED'i 220 ohm'luk bir direnç aracılığıyla dijital pinlerden birine bağlayın.

2-Butonun bir tarafını başka bir dijital pime ve diğer tarafını toprağa bağlayın. Düğmeye basılmadığında sabit bir YÜKSEK sinyal sağlamak için bir pull-up direnci veya dahili INPUT_PULLUP modu kullanın.

Prosedür Adımları

1- Tüm bağlantıların sağlam olduğundan emin olarak devreyi açıklandığı şekilde monte edin.

2- LED'i buton ile kontrol etmek için Arduino kodunu yazın.

3- Kodu Arduino kartına yükleyin.

4- LED kontrolünü test etmek için düğmeye basın ve bırakın.

5- LED'in butona verdiği tepkiyi gözlemleyin.

Örnek Kod

```
// Pin numaralarını tanımlayın
        const int buttonPin = 2; // buton pininin numarası const int
        ledPin = 13; // LED pininin numarası
// Değişkenler değişecektir:
        int buttonState = 0;     //buton durumunu okumak için kullanılan
        değişken void setup() {
  // LED pinini bir çıkış olarak başlatın:
          pinMode(ledPin, OUTPUT);
  // buton pinini bir giriş olarak başlatın: pinMode(buttonPin,
  INPUT);
}
void loop() {
  // buton değerinin durumunu okuyun:
  buttonState = digitalRead(buttonPin);
  // butonun basılı olup olmadığını kontrol edin.
  // eğer öyleyse, buttonState
  YÜKSEK'tir: if (buttonState ==
  YÜKSEK) {
  // LED'i açın:
    digitalWrite(ledPin, HIGH);
  } else {
    // LED'i kapatın:
    digitalWrite(ledPin, LOW);
  }
}
```

Kuralların Açıklanması

- Kurulum: pinMode() fonksiyonu LED pinini bir çıkış ve düğme pinini bir giriş olarak yapılandırır.
- Döngü: Program digitalRead() kullanarak sürekli olarak düğmenin durumunu okur. Düğmeye basılırsa (pull-up yapılandırması nedeniyle YÜKSEK okunur), LED açılır. Aksi takdirde LED kapatılır. LED'in durumu düğmeye basılmasına göre değişir.

Bu deney, sensörler, motorlar ve diğer bileşenleri içeren daha karmaşık Arduino projeleri için temel olan temel dijital I/O işlemlerini göstermektedir.

Deney Sayısı: 17

17- Analog I/O işlemleri

Deney Hedefi

Bu deneyin amacı, öğrencileri bir Arduino kartı kullanarak analog giriş ve çıkış işlemleri kavramı ve uygulaması ile tanıştırmaktır. Katılımcılar sensörlerden analog sinyallerin nasıl okunacağını ve kısılabilir LED'ler veya Darbe Genişlik Modülasyonu (PWM) ile motorlar gibi analog cihazların nasıl kontrol edileceğini öğreneceklerdir.

Teorik Arka Plan

YÜKSEK veya DÜŞÜK olan dijital sinyallerin aksine, analog sinyaller bir dizi değeri temsil edebilir. Arduino kartları Analog-Dijital Dönüştürücüler (ADC) kullanarak analog sinyalleri okuyabilir ve Darbe Genişlik Modülasyonu (PWM) kullanarak analog benzeri çıkış üretebilir.

Analog Giriş: Uno gibi Arduino kartları, A0'dan A5'e kadar etiketlenmiş bir dizi analog giriş pimine sahiptir. Bu pinler, sıcaklık veya ışık gibi sensörlerden gelen analog sinyalleri okuyabilir ve bunları mikrodenetleyici tarafından işlenebilecek dijital bir değere dönüştürebilir.

Analog Çıkış (PWM): Arduino gerçek analog çıkış üretemezken, PWM aracılığıyla bunu simüle edebilir. PWM, LED'ler veya motorlar gibi cihazlara verilen gücü değiştirmek için dijital bir sinyalin göreceli görev döngüsünü kontrol eder.

Analog işlemler için anahtar fonksiyonlar:

- analogRead(pin): Bir analog pinden değeri okur ve 0 (0V) ile 1023 (5V) arasında bir değer döndürür.
- analogWrite(pin, değer): Analog çıkışı simüle etmek için bir pine analog bir değer (PWM dalgası) yazar. Değer 0 (her zaman kapalı) ile 255 (her zaman açık) arasında olabilir.

Gerekli Malzemeler

Arduino UNO R3 kartı

Breadboard

Potansiyometre (örn.

10kΩ) LED

220 ohm direnç

Jumper kabloları

Arduino IDE yazılımı

Deney Devresi

1-Potansiyometreyi analog giriş pinlerinden birine (örn. A0) bağlayın. Bir tarafı 5V'a, orta pimi A0'a ve diğer tarafı GND'ye bağlayın.

2-Akımı sınırlamak için LED'i 220 ohm'luk bir direnç aracılığıyla PWM yapabilen bir dijital pime (örneğin Uno'da 9, 10, 11) bağlayın.

Prosedür Adımları

1-Tüm bağlantıların güvenli olduğundan emin olarak devreyi açıklandığı şekilde kurun.

2-Potansiyometreden analog değeri okuyan ve buna göre LED parlaklığını kontrol eden Arduino kodunu yazın ve yükleyin.

3-Potansiyometreyi döndürün ve LED parlaklığındaki değişimi gözlemleyin.

4-Potansiyometrenin konumu ile LED'in parlaklığı arasındaki ilişkiyi anlayın.

Örnek Kod

```
// Pin numaralarını tanımlayın
const int potPin = A0; // potansiyometre pin numarası const int
ledPin = 9; // LED pin numarası
// Potansiyometre değerini saklamak için
değişken int potValue = 0;
void setup() {
  // LED pinini bir çıkış olarak başlatın:
  pinMode(ledPin, OUTPUT);
  // seri iletişimi saniyede 9600 bitte başlatın:
  Serial.begin(9600);
}
void loop() {
  // değeri potansiyometreden okuyun:
```

```
  potDeğer = analogRead(potPin);
  // potansiyometre değerini 0-1023'ten 0-255'e eşleyin:
  int ledBrightness = map(potValue, 0, 1023, 0, 255);
  // LED'in parlaklığını ayarlayın:
  analogWrite(ledPin, ledBrightness);
  // sonuçları seri monitöre yazdırın:
  Serial.print("Potansiyometre: ");
  Serial.print(potDeğer);
  Serial.print("\t LED Parlaklığı: ");
  Serial.println(ledBrightness);
  delay(10); // kararlılık için küçük
  gecikme
}
```

Kuralların Açıklanması

Kurulum: LED pinini çıkış olarak başlatır ve izleme için seri iletişimi başlatır.

Döngü: analogRead() işlevi potansiyometreden analog değeri okur ve bu değer daha sonra uygun bir PWM değerine (0-255) eşlenir. Bu değer analogWrite() fonksiyonunda LED'in parlaklığını ayarlamak için kullanılır. Potansiyometre ve LED değerleri gözlem için seri monitöre yazdırılır.

Bu deney, çok çeşitli sensörlerle arayüz oluşturmak ve daha karmaşık Arduino projelerinde çeşitli aktüatörleri kontrol etmek için gerekli olan analog girişlerin nasıl okunacağını ve PWM çıkışlarının nasıl üretileceğini göstermektedir.

INA-CODE

robykode

web & mobile