Arduino IDE (Tümleşik Geliştirme Ortamı-Integrated Development Environment)

* karta yükleme işlemini de yapabilen, her platformda çalışabilen Java programlama dilinde yazılmış bir uygulamadır

* Arduino yazılımı bir tümleşik geliştirme ortamı (IDE) ve mikrodenetleyici ve elektronik konusunda detaylı bilgi sahibi olmayı gerektirmeden herkesin programlama yapabilmesini sağlayan kütüphanelerden oluşmaktadır. Arduino kütüphaneleri, geliştirme ortamı ile birlikte gelmekte ve “libraries” klasörünün altında bulunmaktadır

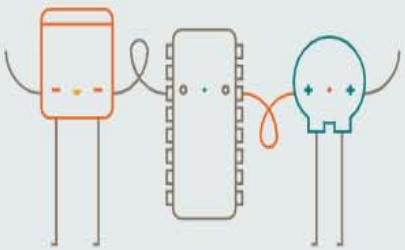
* Kütüphaneler yardımıyla uygulama geliştirilmesi de oldukça kolay ve hızlıdır.

Arduino Yazılımını Kurmak

Arduino yazılımının yüklenmesi için öncelikle internet tarayıcımıza https://www.arduino.cc/en/Main/Software linki girilir. Hangi işletim sistemine sahipsek Windows, Mac OS X, Linux dan birini seçerek indirme işlemini gerçekleştiririz.

HOME BUY SOFTWARE PRODUCTS EDU RESOURCES COMMUNITY HELP SIGN IN

is not tax deductible). Learn more on how your contribution will be used.

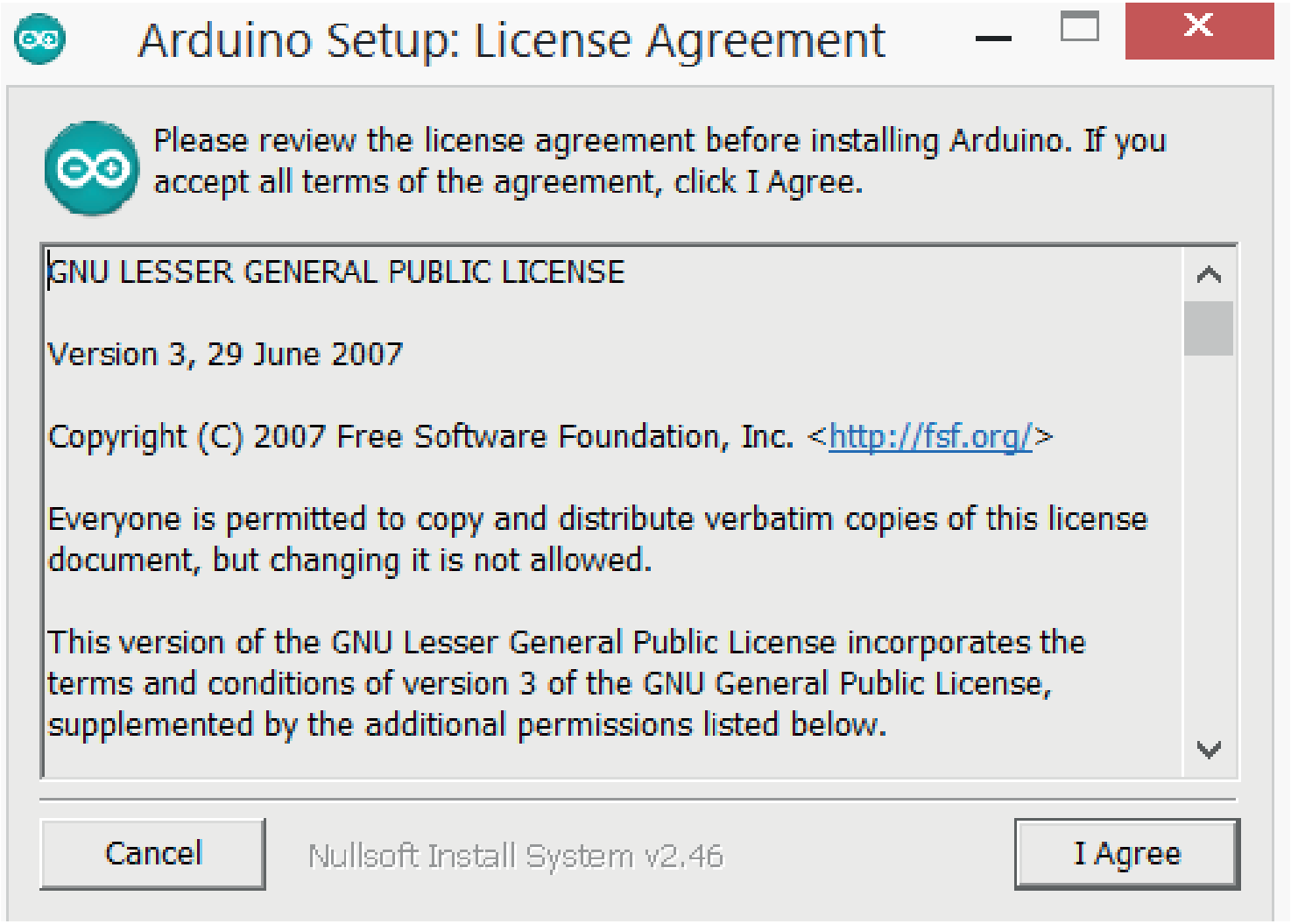


SINCE MARCH 2015, THE ARDUINO IDE HAS BEEN DOWNLOADED 26,392,731 TIMES. (IMPRESSIVE!) NO LONGER JUST FOR ARDUINO AND GENUINO BOARDS, HUNDREDS OF COMPANIES AROUND THE WORLD ARE USING THE IDE TO PROGRAM THEIR DEVICES, INCLUDING COMPATIBLES, CLONES, AND EVEN COUNTERFEITS. HELP ACCELERATE ITS DEVELOPMENT WITH A SMALL CONTRIBUTION! REMEMBER: OPEN SOURCE IS LOVE!

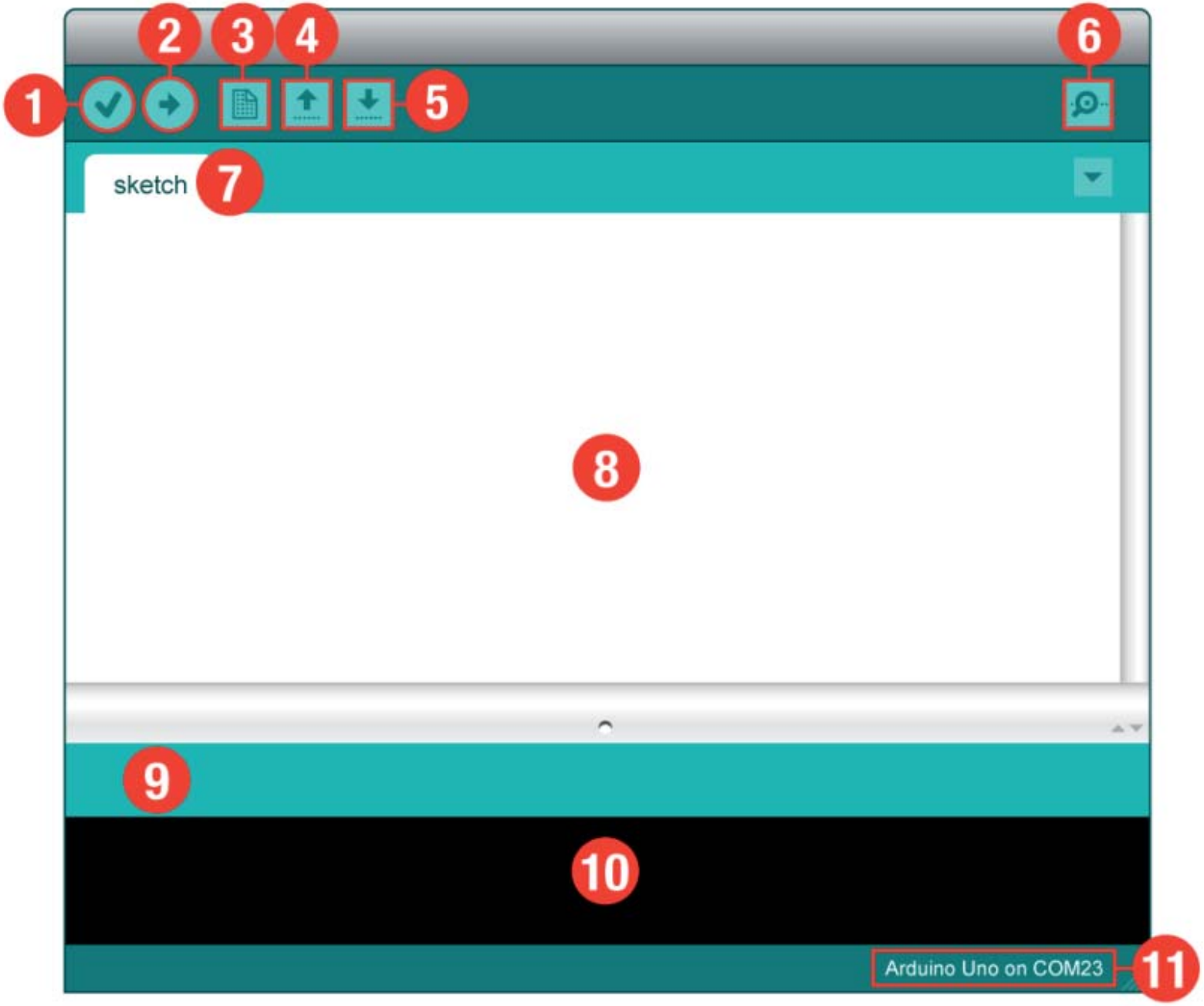
$3 $5 $10 $25 $50 OTHER

JUST DOWNLOAD CONTRIBUTE & DOWNLOAD

Program indirildikten sonra kurulum yapılır



Arduino IDE nin bölümleri

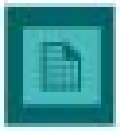


- **1.Derleme:** Yazdığımız programı derler hataları bulur.
- **2.Yükleme:** Yazdığımız kodu derler, Arduino içine atar.
- **3.Yeni:** Yeni çalışma sayfası açar.
- **4.Açma:** Kayıtlı bir programı açar.
- **5.Kaydetme:** Yazdığımız programı kaydeder.
- **6.Seri Monitör:** Arduino ile seri iletişim yaparak ekran açar.
- **7.Sketch:** Yazdığımız programın dosya ismi.
- **8.Boş alan:** Yazacağımız program alanı.
- **9.Gösterge:** Yaptığı işlemin ilerleme durumunu gösterir.
- **10.Rapor:** Derleme sonucu yapılan hataların veya programımızın yükleme sonrası mikro denetleyicide kapladığı alanı gösterir.
- **11.Gösterge:** Bilgisayarımıza usb ile bağladığımız Arduino'nun bağlandığı portu ve hangi Arduino modeli ile çalışıyorsak onu gösterir.

Arduino Ekranı


:Oluşturulan Skecth te mantıksal veya yapısal hataları bulmaya yarar.


:Bu buton oluşturulmuş olan Sketch'in derlenerek USB üzerinden Arduino'ya göndermek için kullanılır.

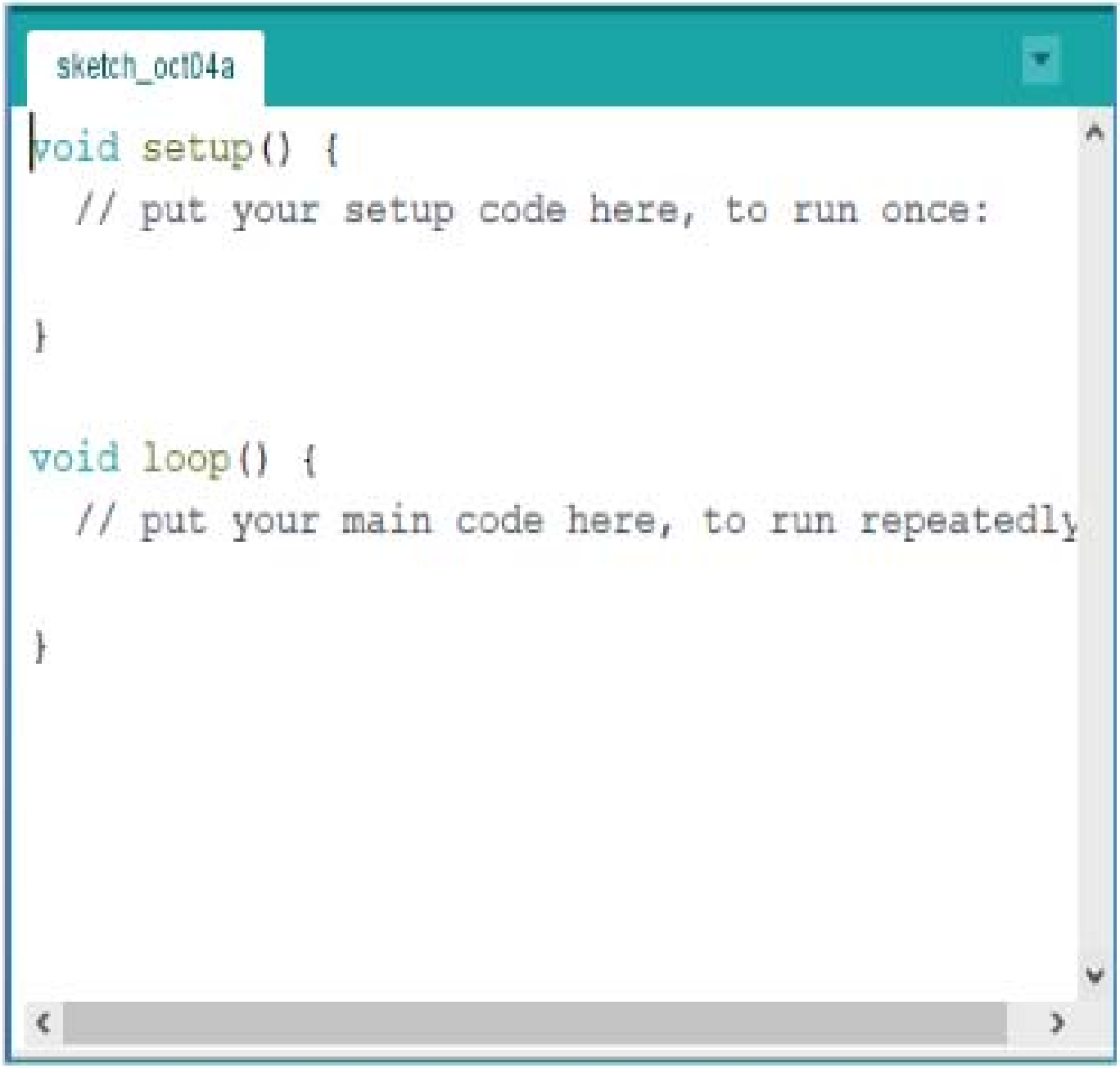

:Bu buton halihazırda açık olan Skecth haricinde yeni bir sketch sayfası açmaya yarar.


:Hali hazırda yapılmış olan Skecth dosyalarını açmaya yarar.


:Üzerinde çalıştığımız sketch dosyasını kaydetmemize yarar.


:Bu buton serial monitör penceresini açmaya yarar.

Arduino Ekranı



Kodlarımızı yazdığımız, programlarımızı tasarladığımız alan.

PROGRAM YAPISI

```
4  void setup() {
5
6    pinMode(13, OUTPUT);//13 nolu dijital pini çıkış olarak ayarla
7  }
8
```

1

```
10 void loop() {
11   digitalWrite(13, HIGH);   // 13 nolu pini YÜKSEK yap +5V verir dijital değeri 1(bir) dir
12   delay(1000);              // 1000 ms=1 sn bekle
13   digitalWrite(13, LOW);    // 13 nolu pini DÜŞÜK yap 0V verir dijital değeri 0(sıfır) dir
14   delay(1000);              // 1000ms=1 sn bekle
15 }
```

2

1-void setup()

Program yüklenip enerji verildikten veya reset atıldıktan sonra bir defa çalışır. Bu fonksiyon içine genellikle kurulum ve ayar için gerekli kodlar yazılır. Aynı zamanda bir defa çalışmasını istediğimiz kodları da yazabiliriz.

2. void loop()

Setup() fonksiyonumuz tamamlandıktan sonra loop fonksiyonumuza geçer ve burada sonsuz döngü içinde yazdığımız programı sürekli çalıştırır.

Arduino' da Kullanılan Temel Kodlar

Arduino Dijital Giriş / Çıkış Komutları

pinMode(pin, mode)

- Dijital giriş/çıkış pinlerinden herhangi birini giriş yada çıkış olarak tanımlamak için kullanılır.
- Pin alanına dijital pinlerden hangisini kullanacaksak onun numarası yazılır.
- "mode" ile ifade edilen alana ise veri girişi mi veri çıkışı mı olacağı yazılır.
- Giriş ise "INPUT", çıkış ise "OUTPUT" yazılır.

Örneğin: Nem sensörü gibi dışarıdan veri algılanacak bir materyal kullanılıyorsa INPUT, LED gibi sinyal verilecek bir materyal kullanılıyorsa OUTPUT yazılır.

Kullanım şekli:

pinMode(13, OUTPUT);

pinMode(*pinNo* , *Görev*)

pinMode(13,OUTPUT); => 13 Nolu pin Çıkış olarak tanımlandı.

pinMode(12,OUTPUT); => 12 Nolu pin Çıkış olarak tanımlandı.

digitalWrite(pin, value)

Dijital yazma anlamına gelen bu komut, çıkış olarak belirlenen pine değer aktarmak(yazmak) için kullanılır.

Kullanım şekli:

digitalWrite(13, 1);

Açıklama:

- İki farklı argüman vardır. İlki dijital yazma işleminin yapılacağı pin belirlenmesi, ikincisi ise yazılacak değerin belirlenmesi. Yazılacak değer 1 veya 0 olabilir.
- Yazılacak değer 1 veya 0 olarak yazılabileceği gibi 1 = HIGH, 0 = LOW olacak biçimde HIGH ve ya LOW da yazılabilir.
- pinMode komutu ile hangi pinler dijital çıkış olarak tanımlandı ise o pinlere yazma işlemi yapılabilir. Giriş olarak tanımlanan bir pine yazma işlemi yapılamaz.

digitalRead(pin)

Dijital okuma anlamına gelen komut belirtilen pinden dijital olarak okuma işlemi yapar.

Kullanım şekli:

digitalRead(5);

Açıklama:

- Read okuma işlemi olduğundan bu dışarıdan gelen bilgiyi okumak biçiminde olmalıdır.
- Burada dikkat edilecek husus şudur; eğer okunan değer 1 ise "digitalRead()" komutunun geri getirdiği değer 1 olur. Yani okuduğu değer 1 ise digitalRead(5) = 1 olur, okuduğu değer 0 ise digitalRead(5) = 0 olur.
- Dijital okuma işlemi pinMode() komutu ile Setup kısmında giriş olarak tanımlanan pinden okuma işlemi yapabilir.

Arduino Analog Komutları

analogRead();

Arduino Uno da analog pinlerin tamamı analog giriş olarak kullanılabilir. "analogRead" komutu analog pine dışardan gelecek 0-5 volt arasında analog değerleri okumak için kullanılır.

```
int analogdeger=0;   //Analog olarak okunan değerin kaydedileceği değişkenin 0
                     //degeri  alması
void setup()
{
pinMode(analog giriş , INPUT); //analog değerin okunacağı bacak numarası
}
void loop()
{
analogdeger = analogRead(analog giriş); //analog değer okunup değişken alanına
                                        //aktarılır
Serial.println(analogdeger);            //serial ekrana okunan değerin yazılması
}
```

Arduino Gecikme Komutları

delay(süre)

Delay, gecikme ya da bekleme anlamlarında kullanılır. Bu komut çalıştırıldığı sırada mikrodenetleyici veya Arduino hiçbir şey yapmadan belirttiğimiz süre kadar bekler.

Kullanım şekli:

```
delay(1000);
```

Açıklama:

- Parantez içine yazılacak sayı milisaniye cinsinden olmalıdır.
- 1000 milisaniye 1 saniyeye eşittir.

Arduino Gecikme Komutları

delayMicroseconds(süre)

Bekleme komutlarından diğeri ise delaymicroseconds() komutudur. Bu komutun delay komutundan farkı süre kısmına yazılan sayı mikrosaniye cinsinden olmalıdır.

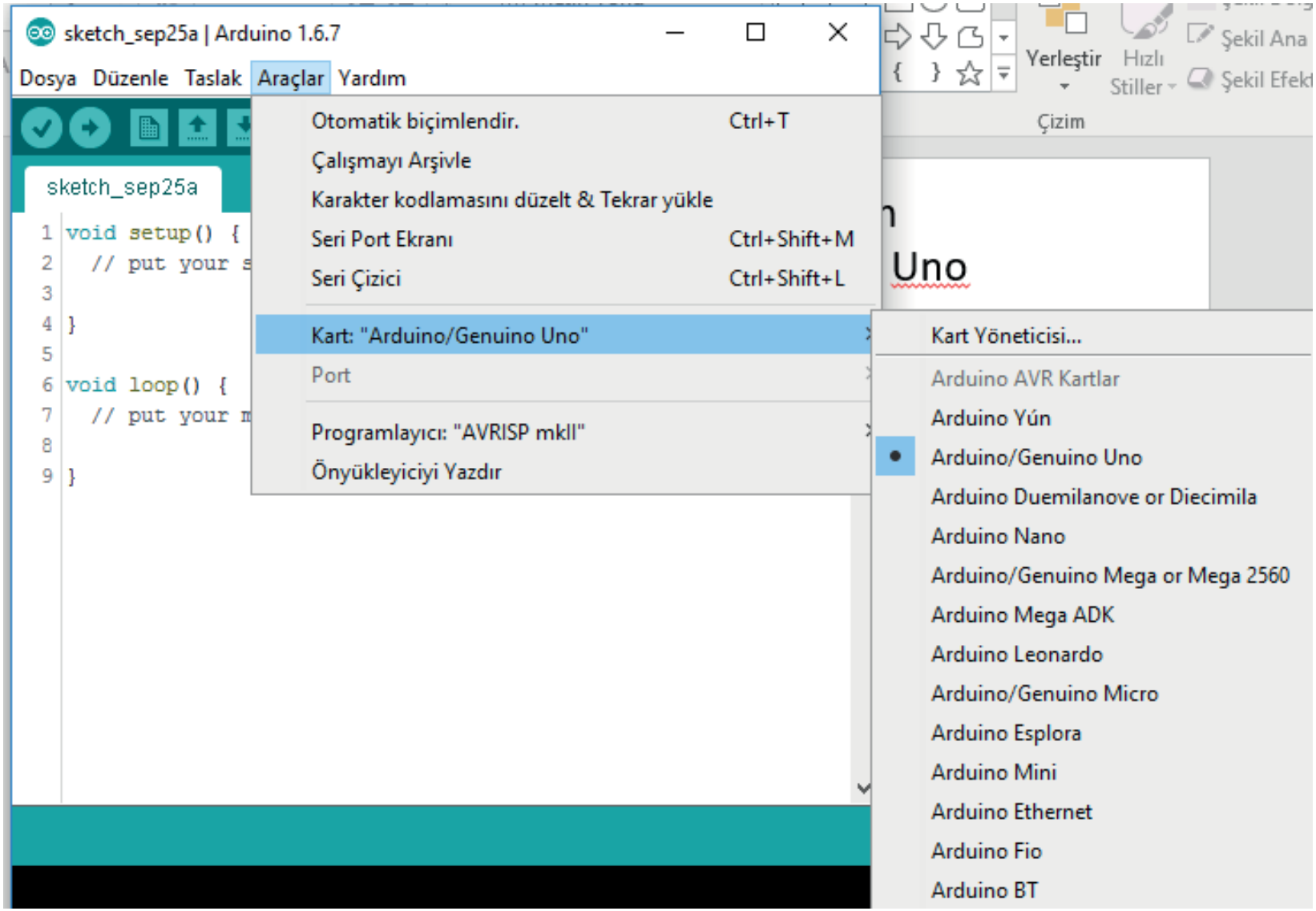
Kullanımı:

delayMicroseconds(1000);

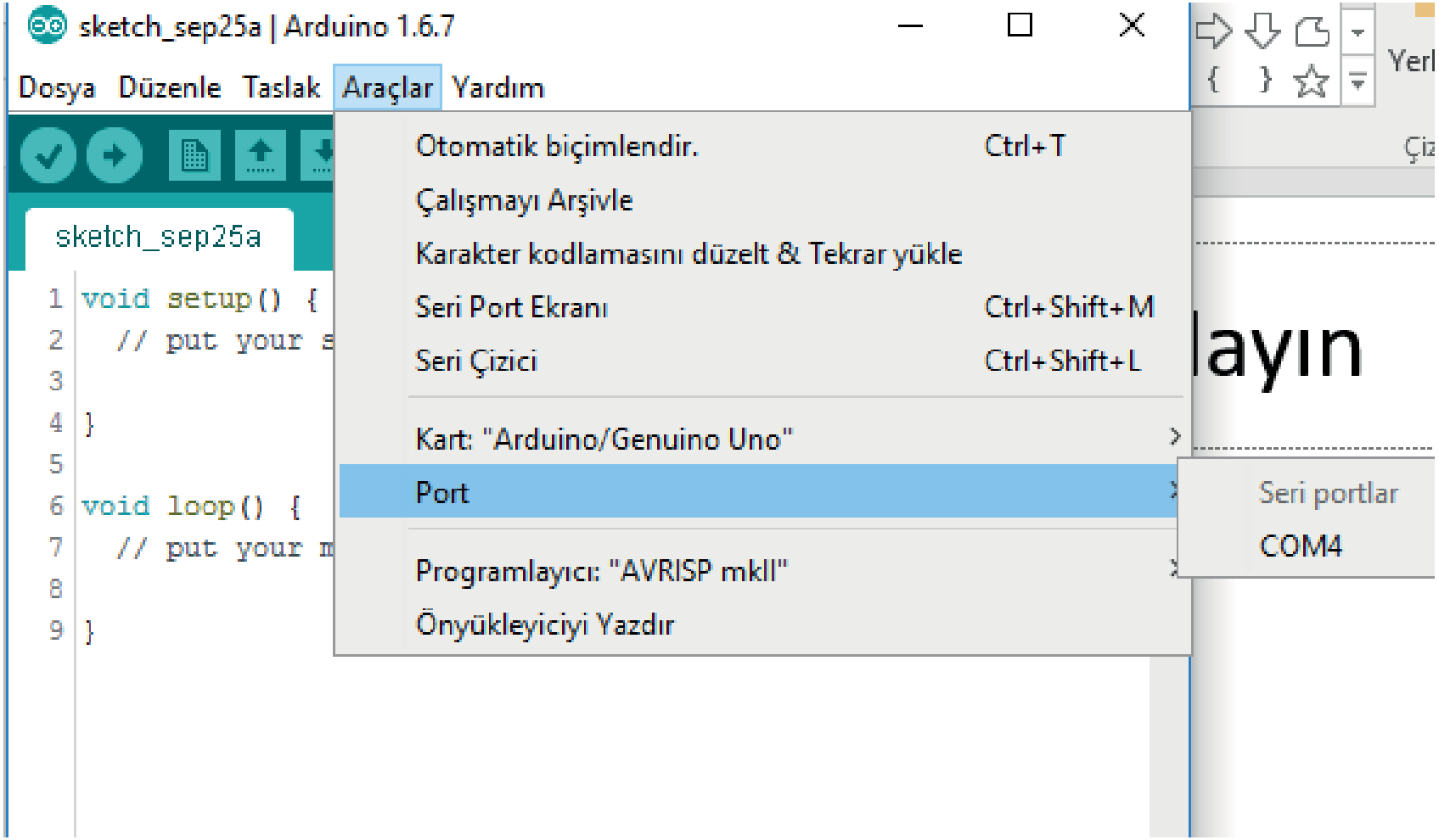
Açıklama:

- Mikro saniye olarak süre belirtilmeli.
- 1.000.000 mikro saniye 1 saniyeye eşittir.

Kullanacağımız Kartı Seçelim
Araçlar>Kart>Arduino/Geniuno Uno
menü yolu seçilmeli



Arduino kartı bilgisayara taktıktan sonra port seçilmelidir. Port seçimi yapılmazsa yazılım yüklemesi yapılamaz



Arduino Tümleşik Geliştirme Ortamının Temel Özellikleri

✓ Arduino IDE tümleşik geliştirme ortamında basitleştirilmiş C++ kullanılır.

✓ Arduino programları genellikle tanımlamalar, kurulum ve ana program bloğu olmak üzere üç bölümden oluşur.

✓ Program yazımı belirli kalıpta, bloklar halinde gerçekleştirilir.

✓ Program kodları renkli olarak gösterilir. Kodların bulunduğu yerlerde gri renkte olan yazılar kodun ne işe yaradığı hakkında bilgi vermek için kullanılır.

✓ Arduino'ya yüklenen programlar kaldırılana kadar Arduino içinde kalır. Yüklemeden sonra bağımsız olarak çalıştırılabilir.

✓ Bloklar, { } parantezleri ile oluşturulur.

✓ Komutlar aynı veya alt alta satırlara yazılabilir. Fakat programın anlaşılabilirliği açısından alt alta yazmak daha uygundur.

✓ Tüm komutlar noktalı virgül (;) ile biter. Fakat blok başlatan ifadelerden sonra noktalı virgül kullanılmaz.

✓ Programda kullanılan tüm değişkenler ve bilgi tipleri bildirilir.

✓ Programın başında kullanılacak kütüphaneler aktifleştirilir /çağrılır.

✓ Açıklamalar "//" ve "/* */ " (birden fazla satır için) ile yazılır.

✓ Türkçe karakter kullanılmamalıdır. Fakat açıklama satırları içerisinde (derleme işlemine dâhil edilmediğinden) kullanılabilir.

✓ Eşdeğer ifadeler #define ile atanır.

✓ Kütüphaneler #include ile çağrılır.

Arduino Tümleşik Geliştirme Ortamının Bölümleri

Arduino programları yapı, değişkenler (değişkenler ve sabitler) ve fonksiyonlar olmak üzere üç ana bölümden oluşmaktadır.

A. PROGRAM YAPISI

void setup()
void loop()

1. Kontrol Yapıları

if
if/else
for
switch/case
while
do/while
break
continue
return
goto

2. Söz Dizimi

; Noktalı Virgül
{} Süslü Parantez
// Çift Slash
/**/ Yıldızlı Slash
#define
#include

3. Aritmetik Operatörler

= Atama İşleci
+ Toplama
- Çıkarma
* Çarpma
/ Bölme
% Modulo

4. Karşılaştırma Operatörleri

== (eşit eşit)
!= (eşit değil)
< (küçük)
> (büyük)
<= (küçük eşit)
>= (büyük eşit)

5. Boolean Operatörleri

&& (ve)
|| (veya)
! (değil)

6. İşaretçi Operatörler

* Referan dışı operatör
& Referans operatörü

7. Bitsel Operatörler

& (Bitsel Ve)
| | (Bitsel Veya)
^ (Bitsel Xor)
~ (Bitsel Değil)
<< (Bitshift Sol)
>> (Bitshift Sağ)

8. Birleşik Operatörler

++ (arttırma)
-- (azaltma)
+- (Birleşik artırma)
-= (Birleşik çıkarma)
*= (Birleşik çarpma)
/= (Birleşik bölme)
%= (Birleşik mod)
&= (Bitsel Lojik Ve)
|= (Bitsel Lojik Veya)

B. DEĞİŞKENLER

1. Sabitler
- HIGH | LOW
- INPUT | OUTPUT | INPUT_PULLUP
- LED BUILTIN
- True | false
- integer constants
- floating point constants

2. Veri Tipleri
- void
- boolean
- char
- unsigned char
- byte
- int
- unsigned int
- word
- long
- unsigned long
- short
- dounle
- string – char array
- substring – object
- array

3. Dönüşümler
- char()
- byte()
- int()
- word()
- long()
- float()

4. Değişken Kapsamları
- static
- volatile
- const

5. Yardımcılar
- PROGMEM
- Sizeof()

C. FONKSİYONLAR

1. Dijital Giriş Çıkışlar
- pinMode(pin,mod)
- digitalWrite(pin,değer)
- digitalRead(pin)

2. Analog Giriş Çıkışlar
- analogRead(pin,mod)
- analogWrite(pin,değer)
- analogReference(tip)
- analogReadResolution ()
- analogWriteResolution ()

3. Gelişmiş Giriş Çıkışlar
- tone()
- noTone()
- shiftOut()
- shiftIn()
- pulseIn()

4. Gecikmeler
- delay(milisaniye)
- unsigned long milis()
- delay(Microse conds (mikrosaniye)

5. Matematiksel İşlevler
- min(x,y)
- max(x,y)
- abs(x)
- constrain(x, a, b)
- map(value, fromLow fromHigh, toLow, toHigh)
- pow(base, esponent)
- sqrt(x)

6. Trigonometri İşlevleri
- sin(raf)
- cos(rad)
- tan(rad)

7. Karakterler
- isAlphaNumeric()
- isAlpha()
- isAscii()
- isWhiteSpace()
- isDigit()
- isGraph()
- isPrintable()
- isPunct()
- isSpace()
- isUpperCase()
- isHexaDecimalDigit()

8. Rastgele Sayılar
- randomSeed()
- random()

9. Bit ve Bayt'lar
- lowByte()
- high(Byte()
- bitRead()
- bitWrite()
- bitSet()
- bitClear()
- bit()

10. Harici İnterruptlar (Kesmeler)
- attachInterruption(interrupt, function, mode)
- detachInterrupt(interrupt)

11. İnterruptlar (Kesmeler)
- interrupts()
- noInterrupts()

12. Seri Haberleşme
- Serial.begin(hız)
- int Serial.available()
- int Serial.read()
- Serial.flush()
- Serial.print(data)
- Serial.printIn(data)

13. Haberleşme Protokolleri
- I2C Veri Yolu
- SPI Veri Yolu