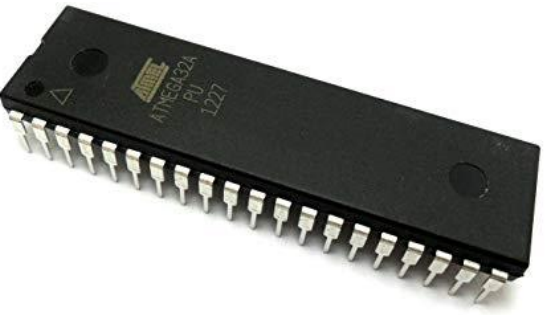
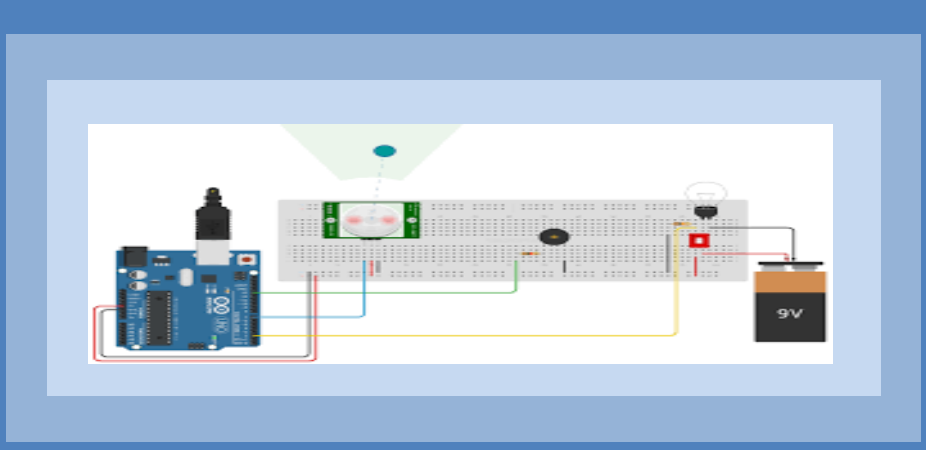


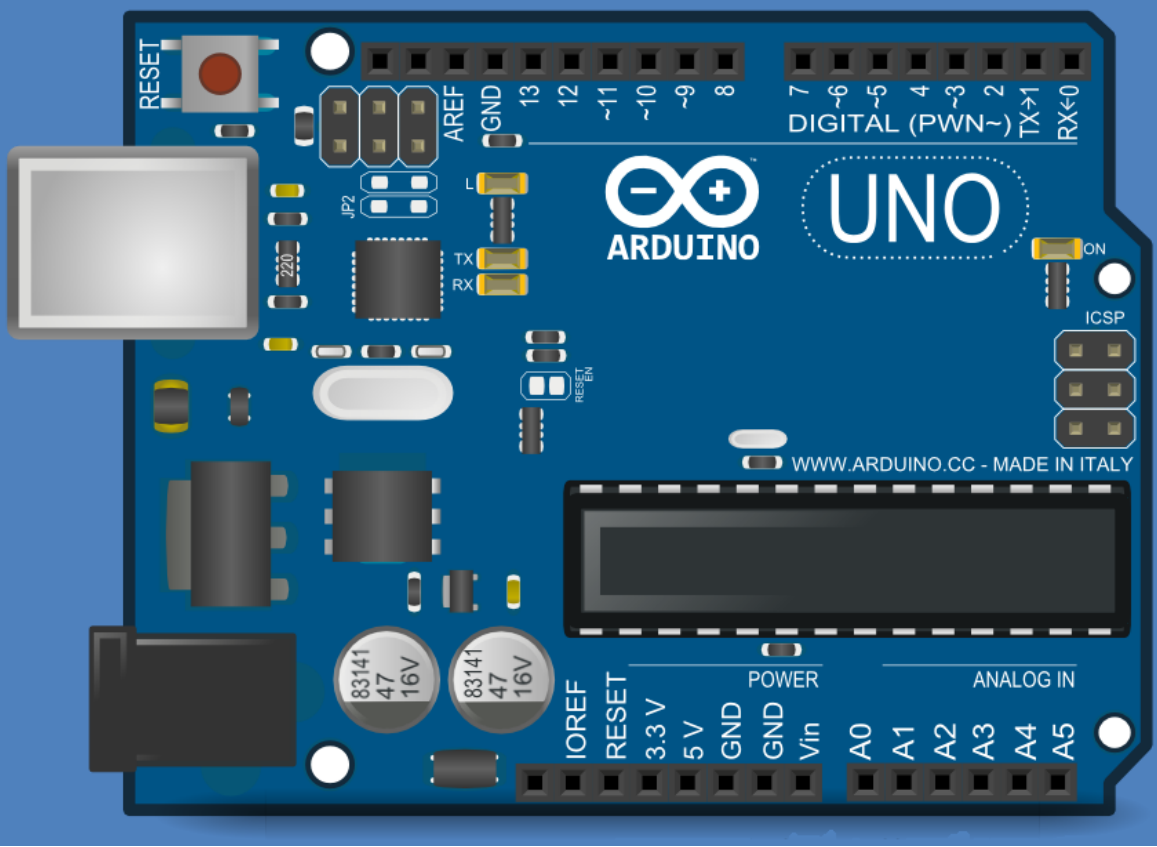


# اردوينو بالعربي


تأليف

حسن كريم صبيح & زهراء حسين علي


2020



# المقدمة

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

الحمد لله أولاً وأخيرًا ... قال الامام علي (عليه السلام) «زكاة العلم بذله لمستحقه»

ومن هذا المنطلق بدأنا تأليف هذا الكتاب الذي يحتوي في صفحاته على معلومات واساسيات يحتاجها كل شخص في تطوير نفسه في الاردوينو بشكل خاص والانظمة المدمجة بشكل عام

حيث يتطرق الى المفاهيم النظرية في اولى صفحاته والتي تحتوي على معلومات مهمة يجب معرفتها قبل الدخول الى الجانب العملي وتصميم المشاريع المختلفة ...

حيث تطرقنا الى شرح مفهوم الاردوينو والمتحكمات الدقيقة واهميتها وكذلك مميزات الاردوينو ومكوناته بشرح بسيط وملخص

وبعد ان تعرفنا على البيئة والهيكل العام لبرمجة الاردوينو تكلمنا عن بعض الاوامر البرمجية المهمة

حيث يحتوي الكتاب على مجموعة تجارب ومشاريع تم شرحها بطريقه مبسطة وحديثة من أجل ايصال المعلومة للمتلقي

ولأننا نؤمن بإن التعلم والخبرة تكون من خلال التطبيق العملي ومواجهة الاخطاء سوف يكون اعتمادنا على ايصال المعلومة من خلال التجارب العملية

المؤلفون

حسن كريم صبيح

زهراء حسين علي

# سوف نتعلم في هذا الكتاب

- ✓ مقدمة عن الأردوينو والمُتحَكِمات الدقيقة
- ✓ مكونات ال MICROCONTROLLERS الاساسية
- ✓ الأردوينو و مميزاته
- ✓ مكونات بطاقة الاردوينو
- ✓ تحميل بيئة برمجة الاردوينو والتعرف عليها
- ✓ الهيكل العام لكود البرمجة
- ✓ المحاكي وكيفية استخدامة
- ✓ وظيفة المقاومة في الدائرة الكهربائية
- ✓ انواع البيانات
- ✓ Project 1 تشغيل واطفاء LED
- ✓ Project 2 عمل تجربة اشارة المرور
- ✓ Project 3 تشغيل واطفاء 13 led بواسطة for
- ✓ توضيح الية عمل for بشكل عام

# سوف نتعلم في هذا الكتاب

- ✓ Project 4 تشغيل واطفاء led بواسطة push button
- ✓ Project 5 **تشغيل 3** LED **بالتتابع عن طريق** push putton
- ✓ توضيح الية عمل switch case بشكل عام
- ✓ تعديل عرض الموجة PWM
- ✓ Project 6 التحكم بشدة الاضاءة من خلال putton
- ✓ الحساسات , انواعها
- ✓ بعض الاوامر البرمجية المتقدمة
- ✓ Project 7 حساس درجة الحرارة
- ✓ Project 8 المقاومة الضوئية لتحسس شدة الضوء LDR

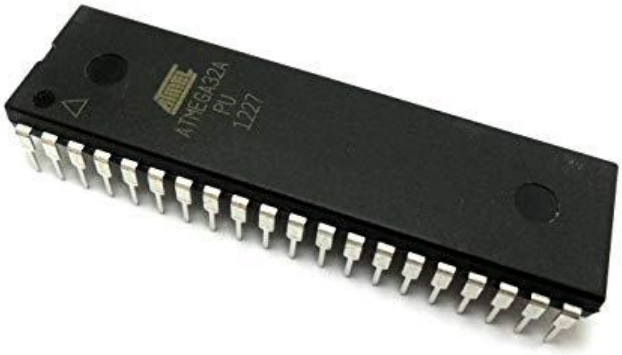
# سوف نتعلم في هذا الكتاب

# مقدمه عن اردوينو والمُتحَكِمات الدقيقة

## ما هو الأردوينو ؟

الأردوينو بالإنجليزية Arduino :هي عبارة عن لوحة تطوير إلكترونية Board Development تتكون من دائرة إلكترونية مفتوحة المصدر مع متحكم دقيق على لوحة واحدة يتم برمجتها عن طريق الكمبيوتر وهي مصممة لجعل عملية استخدام الإلكترونيات التفاعلية في مشاريع متعددة التخصصات أكثر سهولة. ويستخدم اردوينو بصورة أساسيه في تصميم المشاريع الإلكترونية التفاعلية أو المشاريع التي تستهدف بناء حساسات بيئية مختلفة (مثل درجات الحرارة، الرياح، الضغط...الخ) ويمكن توصيل اردوينو ببرامج مختلفة على الحاسب الشخصي. وتعتمد الاردوينو في برمجتها على لغة البرمجة مفتوحه المصدر وتتميز الأكواد البرمجية الخاصة بلغة اردوينو أنها تشبهه لغة (C++) وتعتبر من أسهل لغات البرمجة المستخدمة في كتابة برامج المتحكمات الدقيقة

## ما هي المتحكمات الدقيقة ؟



لا تخلو حياتنا اليومية من استخدام التكنولوجيا ووجود المتحكمات الدقيقة في كل مكان حولنا أصبح أمراً لابد منه ولكن ماهي هذه المتحكمات التي تتواجد في منازلنا وأماكن عملنا، سيارتنا والأماكن العامة وحتى في الفضاء؟!

**المتحكمات الدقيقة microcontrollers** هي عبارة عن كمبيوترات صغيرة قابلة للبرمجة لتؤدي مجموعة من المهام، الفرق الأساسي بين الكمبيوترات والمتحكمات الدقيقة هي المهام التي تؤديها، مثلاً تستخدم الكمبيوترات لتشغيل الأفلام ومشاهدتها او لتصفح الانترنت او قراءة الكتب والمقالات ... بينما تستخدم المتحكمات الدقيقة في المنتجات او الاجهزة التي يتم التحكم فيها أوتوماتيكيا وتوجد الكثير من المنتجات والاجهزة التي تجعل استخدامنا لها يومياً بدون ان ندرك ذلك مثل التلفزيون، الجوالات، أنظمة الأمن، الكاميرات، أجهزة ألعاب الفيديو، المايكروويف، الطابعات، السيارة ويمكن تفصيل بعضها كالآتي:

- المكيفات للتحكم بدرجة الحرارة أو مدة التشغيل...
- الغسالات للتحكم بسرعة واتجاه المحركات والمهام التي تقوم بها (غسل، شطف، تنشيف)
- التحكم بالإضاءة حيث يمكن او تعمل حين يتواجد شخص ما في الغرفة او أن تعمل في وقت محدد...
- محطات الطقس كقراءة درجة الحرارة والرطوبة ومستوى أشعة الشمس وسرعة الرياح ومستويات الغازات... ويمكن حفظها وعمل إحصاءات او مقارنات بينها...
- التحكم في الروبوتات: فمثلا التحكم في سرعته، مساره، حركة الأذرع، قراءة المعلومات (صوت أو فيديو)

## مكونات ال MICROCONTROLLERS الاساسية

1- cpu
2- ذاكرة ROM و RAM
3- وحدات الادخال والاخراج I\O
4- وحدة التحكم في الزمن Timer
5- وحدة التخاطب مع الاجهزة الخارجية
6- وحدة التحويل الى النظام التماثلي او الرقمي

ماهو النظام الرقمي ؟؟؟؟

ببساطة النظام الرقمي هو عبارة عن نظام ثنائي (1 \0 ) ال 0 يعني 0 volt وال 1 يعني 5 volt

ابسط مشروع ممكن نصنع بهذا النظام تشغيل واطفاء LED

يحتوي على الكثير من التفاصيل سوف نتعرف عليها لاحقا

ماهو النظام التماثلي ؟؟؟؟

الفرق بينه وبين النظام الرقمي انو بالتماثلي تختلف نوعية وقيمة البيانات او الاشارة مثلا عداد السرعة , مقياس درجة الحرارة , وغيرها من الادوات والاجهزة الي تستقبل اشارات متغيرة وليس ثابته

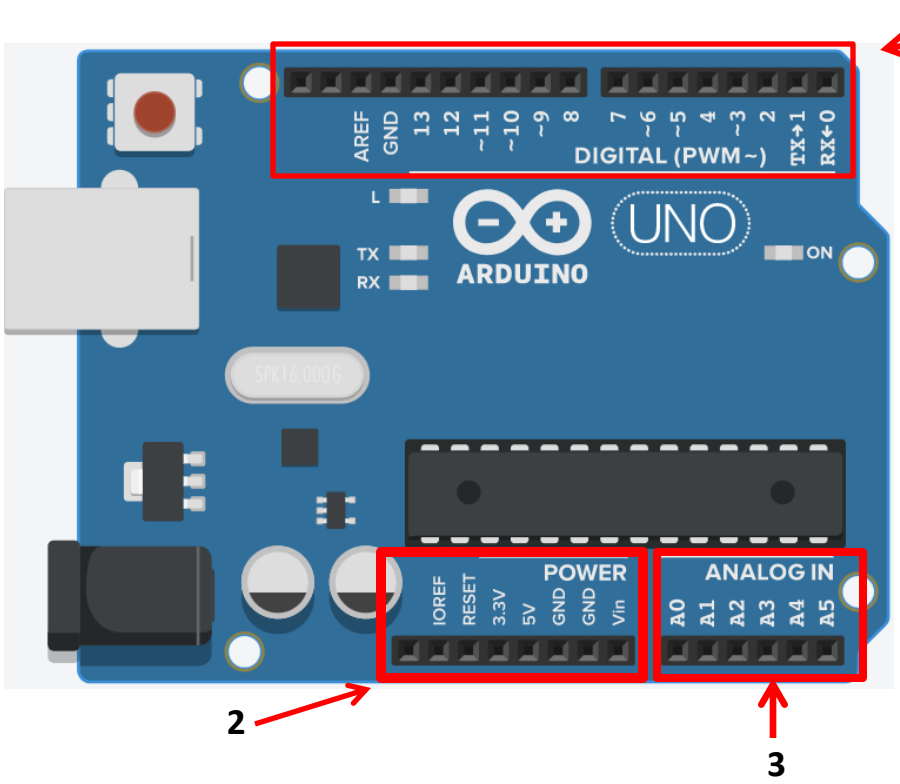
مع ذلك نستطيع تغيير النظام من تماثلي الى رقمي ...

## الاردوينو (ARDUINO)

### مميزات الاردوينو

انخفاض السعر
سهولة الاستخدام (مقارنة بغيرة من الدوائر المبرمجة)
كثرة الإضافات المتوافقة مع الأردوينو و التي تقوم بأعمال متنوعة
موقع الإنترنت الخاص بالأردوينو منظم ومفيد جداً arduino.cc
الشهرة الواسعة وآلاف المستخدمين و الدروس و المشاريع عبر العالم

# المنافذ ports




## 1- المنافذ الرقمية digital pins

المنافذ الرقمية وعددها 14 منفذ وهي مرقمه من (0-13)
ويمكنك في الكود تحديد عمل كل منفذ عندما تعمل المنافذ كمخارج يمكنك حسب كتابة الكود اخراج 5v او 0v

**العلامة ~** تعني أن هذا الطرف يصلح لإخراج قيمة جهد تماثلية. ويسمى أيضا **PWM**

**المنفذين(0,1)** يسميان **TX** و **RX** ويستخدمان للتواصل مع الكمبيوتر

**المنفذ GND** يعمل كأرضي للدائرة الإلكترونية 0v

**المنفذ AREF** نادر الاستخدام يستخدم لضبط اعلى قيمة في نقاط الجهود للمداخل التماثلية (0V – 5V)

## 2- منافذ الطاقة power pins

بعد تشغيل الأردوينو وتوصيل الطاقة المناسبة له يمكنك ان تستخدم هذه المجموعة لتمد دائرتك الالكترونية بالطاقة المناسبة **(5v او 3.3v)**

**الطرفين GND** تسمى الارضي وجهدها 0V لاحظ ايضا يمكنك ان تمد الأردوينو بالطاقة عبر توصيل جهد مناسب من (7V – 12V) إلى الطرف **Vin** كما يمكنك ان تعمل **Reset** إعادة التشغيل الأردوينو عبر استخدام المنفذ Reset**(لعمل هذا: وصل المنفذ reset ب GND)**

## 3- المداخل التماثلية Analog inputs

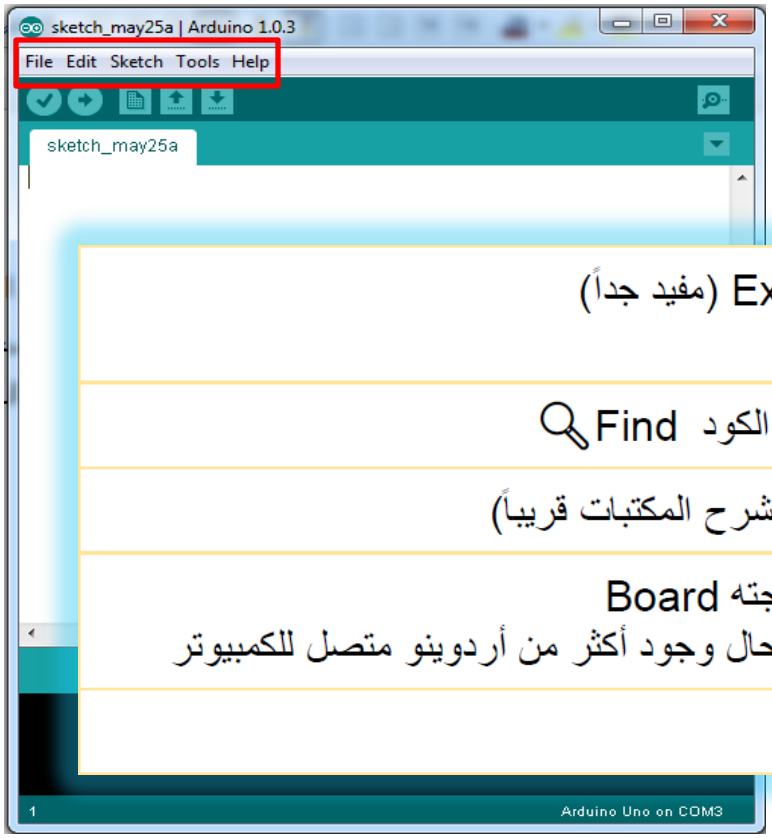
عددها 6 من **(0A – 5A)** ويمكنها قياس الجهد (تماثليا) ويكون التعامل معها بتوصيلها مع السلك المطلوب قياس الجهد عنده ثم التحكم بها في البرنامج **يمكن استخدام هذه الأطراف كمداخل رقمية أو مخارج رقمية . ستشرح لاحقاً.**

# البدء مع بيئة برمجة أردوينو

للحصول على بيئة البرمجة Arduino IDE يمكن تحميلها من الموقع الرسمي
بعد تثبيت برنامج ArduinoIDE من موقع المدرج اعلاه

تعتبر الواجهة سهلة الاستخدام وقليلة الخيارات انظر للصورة

سوف نشرح الخيارات الاساسية التي تمكنك من العمل على البرنامج اما بقية الخيارات المتطورة استكشفها بنفسك



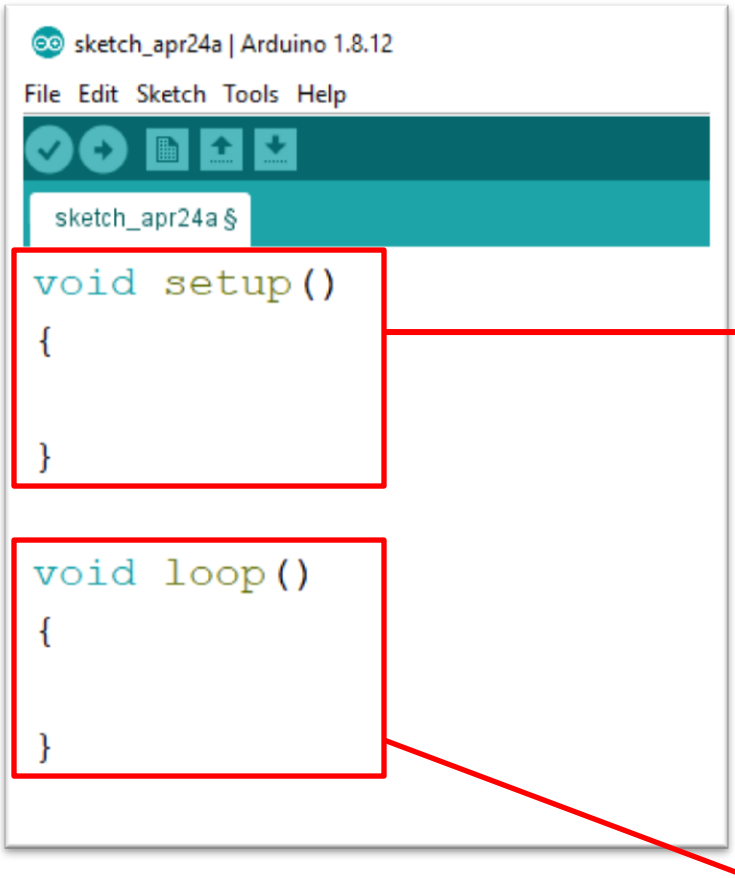

| | |
|---|---|
| **File** | فتح أمثلة كثيرة من الأكواد Examples (مفيد جداً)<br>ضبط خيارات الخط |
| **Edit** | قص نسخ لصق ، البحث عن كلمة في الكود Find |
| **Sketch** | إضافة مكتبات الأوامر Library ( سنشرح المكتبات قريباً) |
| **Tools** | اختيار موديل الأردوينو المطلوب برمجته Board<br>اختيار المنفذ شيء يسمى Port . في حال وجود أكثر من أردوينو متصل للكمبيوتر |
| **Help** | مجموعة روابط مساعدة. |

| | |
|---|---|
|  | فحص الكود<br>في حال وجود خطأ في القواعد الكتابية سينبهك أسفل الصفحة (المربع الأسود) |
|  | فحص الكود ثم رفعه إلى الأردوينو |
|  | فتح صفحة جديدة فارغة |
|  | فتح كود مخزن سابقاً |
|  | حفظ الكود الحالي |
|  | فتح شاشة السيريال ( مهم جداً ) Serial monitor |

# الهيكل العام لكود الأردوينو

يتم تقسيم الكود منطقيا الى قسمين كما موضح في الصورة

**Void loop و Void setup**




في هذا الجزء كل ما يكتب بين الاقواس يتم تنفيذه مره واحدة عند تشغيل الكود

وايضا يمكننا في هذا القسم من تحديد المدخلات و المخرجات

كما في المثال

pinMode( 13 , OUTPUT );

pinMode (13, OUTPUT );

بالاضافة الى الكثير من التفاصيل سوف نتعلمها لاحقا

اما هذا الجزء وهو الاهم الذي يحوي اغلب الكود المشغل للمشروع

ويتم تكار تنفيذه طوال فترة اتصاله بالطاقة

سوف ندرج مجموعة من الاوامر البرمجية الاساسية التي سوف

نكتبها في هذا القسم بكثرة

## مجموعة اوامر برمجية

**سوف نكتب لكم مجموعة اوامر برمجية مهم ان تعرفوها قبل البدء بالتطبيق العملي:**

**هذه الاوامر دائما تكتب في قسم التكرار LOOP**

```
digitalWrite( 13 , HIGH );
```

هنا نكتب اسم او رقم المنفذ

في هذا المثال استخدمنا 13 كمخرجات

اي يعطينا 5V

والامر HIGH يعني true او تشغيل

ويستخدما عادة لاعطاء امر تشغيل led

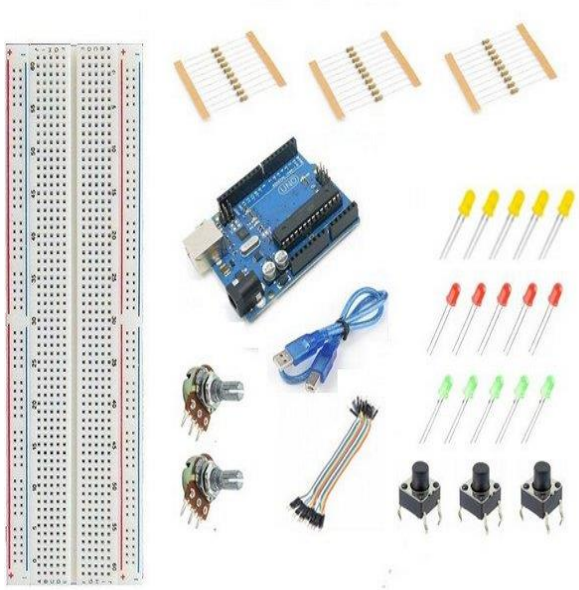
digitalRead( pin ); ← قراءة قيمة قطب "pin" معرف كدخل

analogRead(pin); ← قراءة القيم التماثلية على قطب الدخل pin التماثلي (A5 ~ A0) في لوحات Arduino UNO

هنالك الكثير من الاوامر البرمجية المهمة والتي سوف نستخدمها بأستمرار لكن يحتاج تطبيقها عمليا من اجل فهمها بصورة صحيحة سوف نبدأ بشرح مشاريع تدريجية من اجل التعرف على المواضيع التالية:

1- التعرف على الادوات المستخدمة في الاردوينو
2- التعرف على الاوامر البرمجية وتطبيقها بطرق مختلفة
3- التعرف على الحساسات والكثير من القطع المهمة

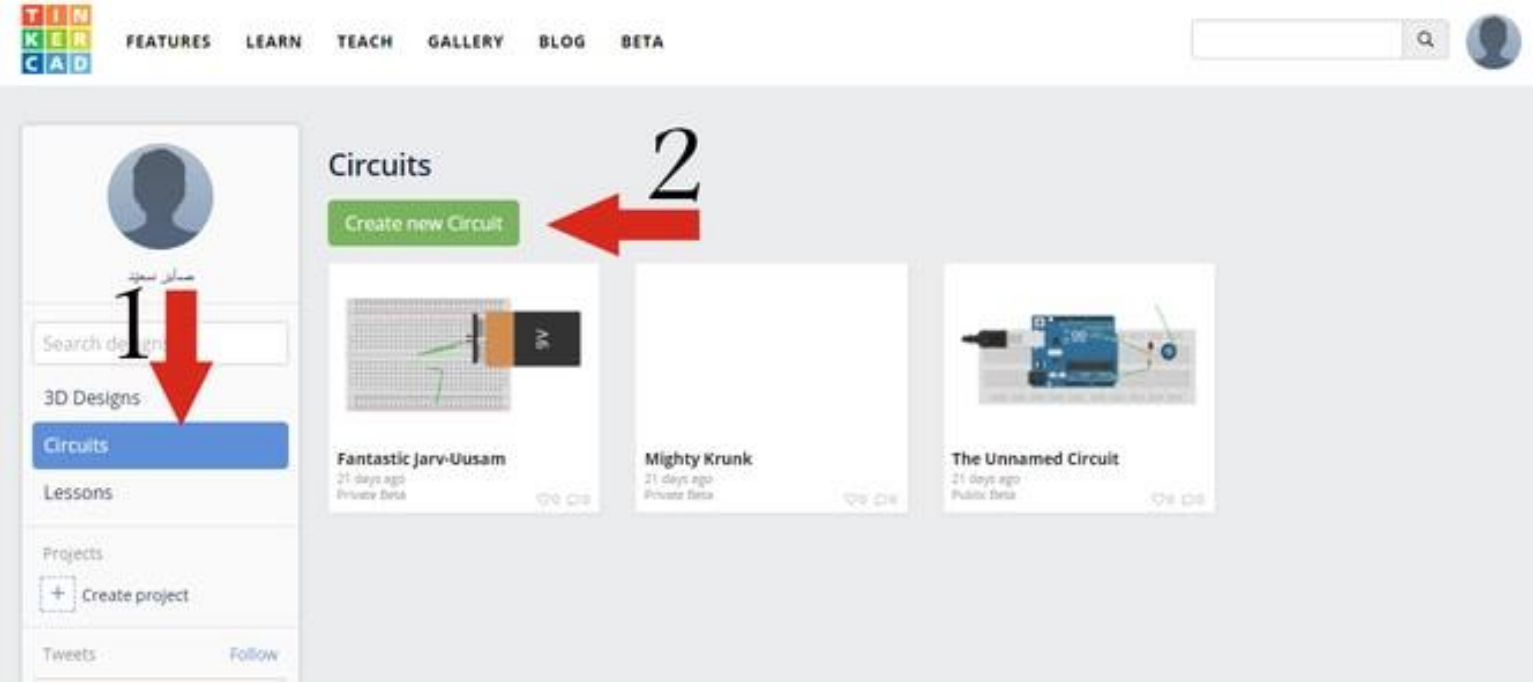
**هذه مجموعة صور للادوات التي سوف نستخدمها في المشاريع والتجارب القادمه**



## المحاكي الذي سوف نستخدم في هذا الكتاب

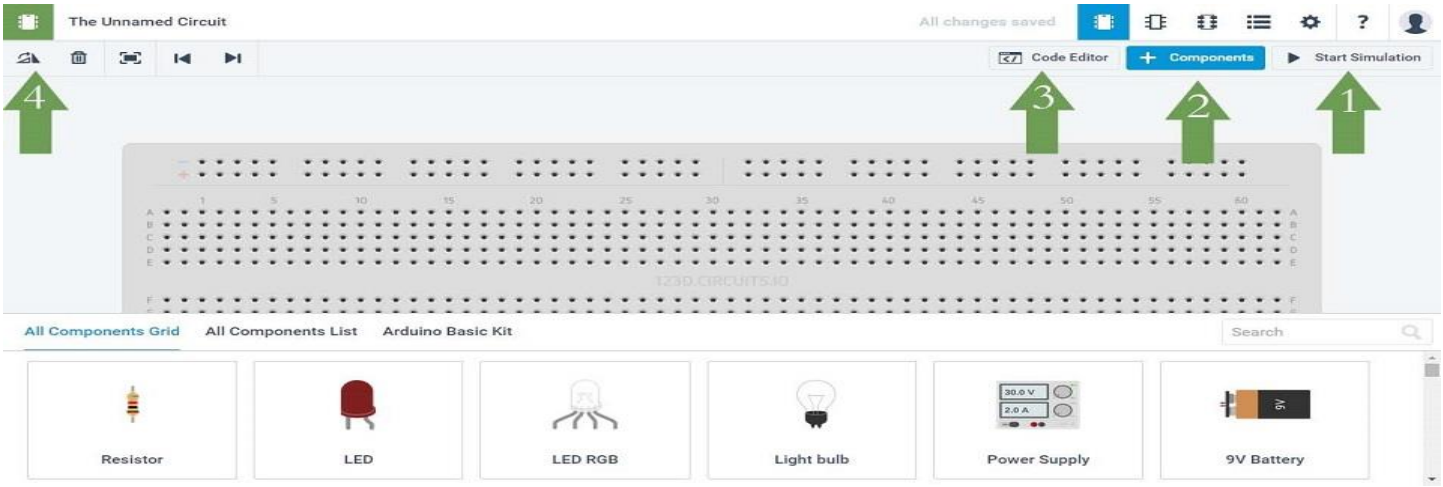
### موقع TINKERCAD

**بعد تسجيل الدخول في موقع TINKERCAD.COM اما باستخدام الايمال و كلمة المرور التي استخدمناها في انشاء حساب اوتوديسك او باستخدام مواقع خارجية كالفيسبوك, نقوم بالضغط على Circuits من القائمة اليسرى ثم نضغط على Create new Circuit كما هو موضح بالصورة التالية:**




فتظهر لنا بعد ذلك الواجهة التالية. و إليكم دور بعض الأزرار في هذا التطبيق:

1- من أجل البدأ في عملية المحاكاة و تجريب المشروع الذي قمت بإنشاءه.
2- من أجل فتح مجموعة المكونات الإلكترونية.
3- من أجل فتح نافذة للتعديل على كود البرمجة لاردوينو.
4- من أجل تدوير المكونات الإلكترونية.

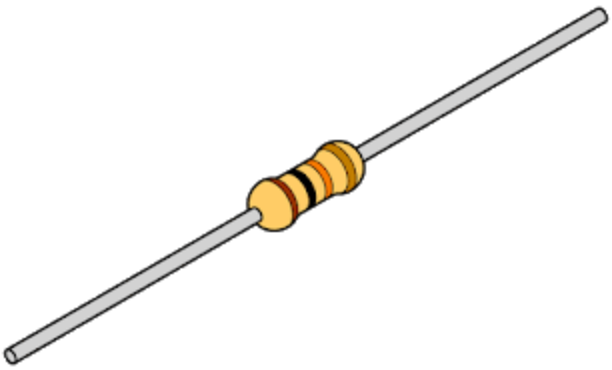


# القسم العملي ( التجارب او المشاريع )

سوف يكون الجانب العملي على قسمين القسم الاول خاص بالأساسيات والمشاريع البسيطة اما القسم الثاني سوف يكون فقط للمشاريع المتقدمة

**قبل البدء في شرح المشاريع يجب ان نشرح موضوع مهم جدا**

في جميع التجارب العملية سوف نستخدم المقاومة والتي تعتبر جزء مهم في الدائرة الكهربائي لذلك من المهم معرفة عملها وكيفية اختيار النوع المناسب في التجربة



## وظيفة المقاومة في الدائرة الكهربائية؟

تعتمد وظيفتها على المكان الذي ستوضع فيه فاذا ربطت مع حمل على التوازي فان تاثيرها سيكون على المصدر هو سحب تيار اكبر اما على الحمل فلن توثر اذا كان المصدر ذو سعة عالية اما اذا كان المصدر محدود السعة فقد يودي الى انخفاض التيار في الحمل اما ربطها على التوالي مع الحمل فان ذلك سيخفض التيار القادم من المصدر ويودي الى انخفاض الجهد عبر الحمل المربوطة معه هذا اكاديميا اما عمليا فغالبا تستخدم المقاومه للحد من قيمة التيار فمثلا المقاومة المستخدمة في الارضي تبع المحولات هو لتخفيض قيمة تيار الخط المتعادل كذلك توضع بعض الانواع من المقاومات في المحركات لتعمل كمتحسس للتيار العالي حيث تحمي الملفات من زيادة التار وفي الحقيقة للمقاومة تطبيقات كثيرة جدا

**الخلاصة!** المقاومة تعمل على التقليل من شدة التيار وزيادة فرق الجهد في الدائره الكهربائية و بالتالي تحميها من التيار العالي.

# انواع البيانات

**الكائنSTRING :** تنشئ الدالة STRING() نسخةً من الصنف STRING الذي يوفِّر الكثير من الأدوات للتعامل مع السلاسل النصية وإجراء عمليات عليها.

**ARRAY :** المصفوفة هي مجموعة من المتغيرات والثوابت الموضوعة في وعاء واحد والتي يمكن الوصول إليها والتعامل معها عبر رقم فهرس كلٍّ منها.

**BOOL :** يخزّن المتغير الذي يصرَّح عنه بأنَّه من النوع BOOL إحدى القيمتين المنطقيَّتين التاليتينTRUE :، أو FALSE عبر حجز بايتٍ واحدٍ من الذاكرة فقط.

BOOLEAN : النوع BOOLEAN هو نوع غير قياسي، وهو اسمٌ بديل للنوع BOOL المُعرَّف في أردوينو.

**BYTE :** يُخزِّن النوع BYTE عددًا عديم الإشارة بحجم 8 بت.

**CHAR :** يحجز النوع CHAR بايتًا واحدًا من الذاكرة ويخزِّن فيه قيمة محرف محدَّد.

**DOUBLE :** يحجز النوع DOUBLE أربعة بايتات من الذاكرة فقط في لوحات أردوينو) UNO والتي تعتمد على متحكمات (ATMEGA أو ثمانية بايتات في اللوحات DUE لتخزين عدد عشري فيها.

FLOAT : يحجز النوع FLOAT أربعة بايتات من الذاكرة لتخزين عدد عشري فيها.

**INT :** يحجز النوع INT حجمًا مقداره 2 بايت من الذاكرة في لوحات أردوينو) UNO وتلك التي تعتمد على متحكمات (ATMEGA أو 4 بايت في لوحات أردوينو) DUE وتلك التي تعتمد على متحكمات (SAMD لتخزين عدد صحيح فيه.

LONG : تحجز النوع LONG حجمًا كبيرًا من الذاكرة مقداره 4 بايت يُستعمَل لتخزين الأعداد التي تتسم بأنَّها طويلة.

**SHORT** : يحجز النوع SHORT في جميع لوحات أردوينو (التي تعتمد على المتحكمات ATMEGA و (ARM حجمًا من الذاكرة مقداره 2 بايت لتخزين عدد قصير فيه.

**STRING :** يمثِّل النوع STRING سلسلةً نصيةً مؤلفةً من عدة محارف مرتبطة مع بعضها بعضًا. تُستعمَل مصفوفة من المحارف لتخزين هذا النوع من البيانات واستدعائها والتعامل معها لاحقًا.

**UNSIGNED CHAR :** يحجز النوع UNSIGNED CHAR حجمًا من الذاكرة مقداره 1 بايت فقط.

**UNSIGNED INT :** يحجز النوع UNSIGNED INT حجمًا من الذاكرة مقداره 2 بايت في لوحات أردونيو UNO واللوحات التي تعتمد على متحكمات ATMEGA أو 4 بايت في لوحات أردوينو DUE لتخزين عدد صحيح عديم الإشارة فيها.

**UNSIGNED LONG :** يحجز النوع UNSIGNED LONG حجمًا كبيرًا من الذاكرة مقداره 4 بايت يُستعمَل لتخزين الأعداد عديمة الإشارة التي تتسم بأنَّها طويلة.

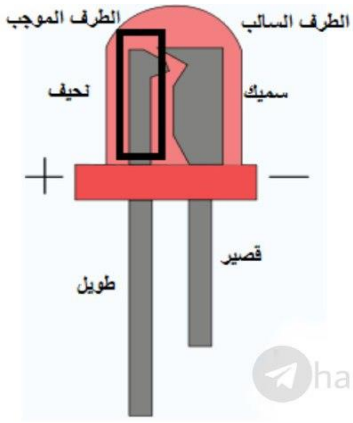
**VOID :** تُستعمَل الكلمة المفتاحية VOID مع الدوال التي يُعرِّفها المبرمج في الشيفرة لتُشير إلى أنَّه لا يُتوقَّع أن تعيد هذه الدالة بعد انتهاء تنفيذها أيَّة بيانات إلى من استدعاها.

**WORD :** تُخزِّن المتغيرات التي من النوع WORD عددًا عديم الإشارة بحجم 2 بايت.

# Project 1

## LEDتشغيل واطفاء

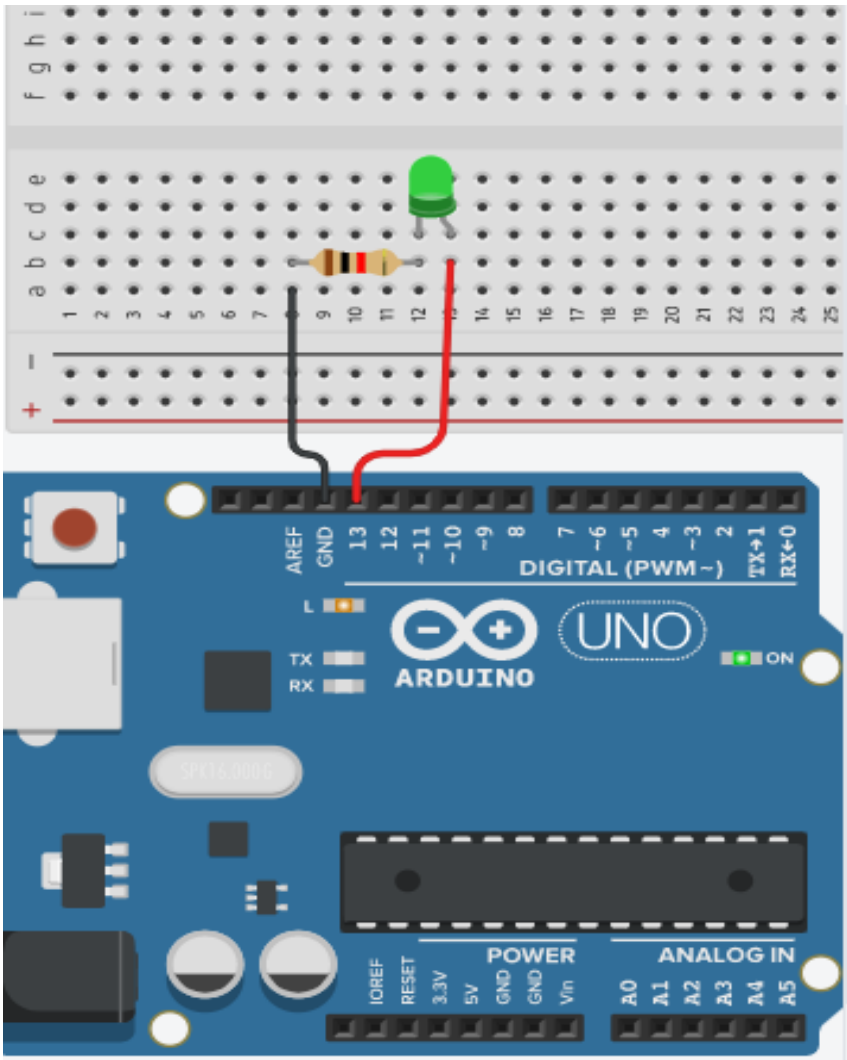
ملاحظة : دائما المنفذ السالب يوصل بإحد منافذ GND




لمعرفة المنفذ السالب أو الموجب في ال LED الصغيرة
يوجد طريقتين كما موضح في الصورة

المكونات

1- Led
2- مقاومة
3- اردوينو UNO
4- Breadboard
5- اسلاك توصيل




Text

```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
 digitalWrite(13,HIGH);
  delay(1000);

  digitalWrite(13,LOW);
  delay(1000);
}
```

Serial Monitor

# شرح التجربة

```
Text
1  void setup()
2  {
3    pinMode(13, OUTPUT);
4  }
5
6  void loop()
7  {
8   digitalWrite(13,HIGH);
9    delay(1000);
10
11   digitalWrite(13,LOW);
12   delay(1000);
13 }
Serial Monitor
```

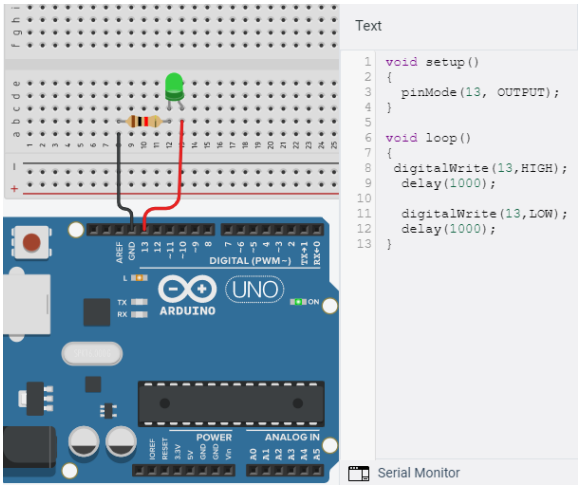
بالنسبة لدالة setup هي دالة التهيئة من خلالها يتم تهيئة المنفذ . ويتم تنفيذها مرة واحدة فقط عندما يتم تشغيل الكود على الاردوينو أول مرة. دالة pinMode(13,OUTPUT) تحمل بداخلها متغيرين الاول يشير إلى رقم المنفذ (13) والاخر يحدد نوعه (INPUT)أو(OUTPUT)

الأمر **digitalWrite** يعمل على اخراج اشارة كهربائية على الطرف 13 ( HIGH= 5v او LOW =0v ) وبما ان المخرج 13 متصل ب ضوء في الأردوينو فسوف ترى هذا الضوء يعمل

الأمر delay يعمل على تـخير زمني لمدة 1 ثانية
1000 ميللي ثانية = 1 ثانية

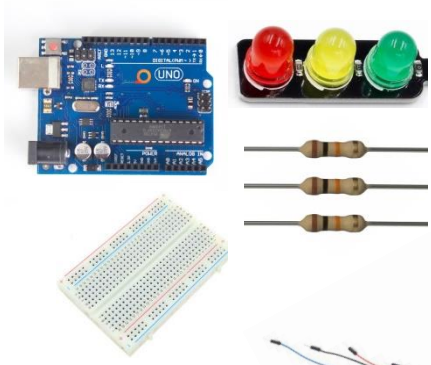
وذلك اننا لو شغلنا الضوء و أطفأناه بدون تأخير فلن نلاحظ الوميض بسبب سرعة الأردوينو
سوف يستمر هكذا الكود طوال فترة اتصاله بالطاقة وهو تشغيل واطفاء الليد كل ثانية

**لتشغيل التجربة او الاطلاع عليها وتعديلها اضغط على الصورة المصغرة ادناه**



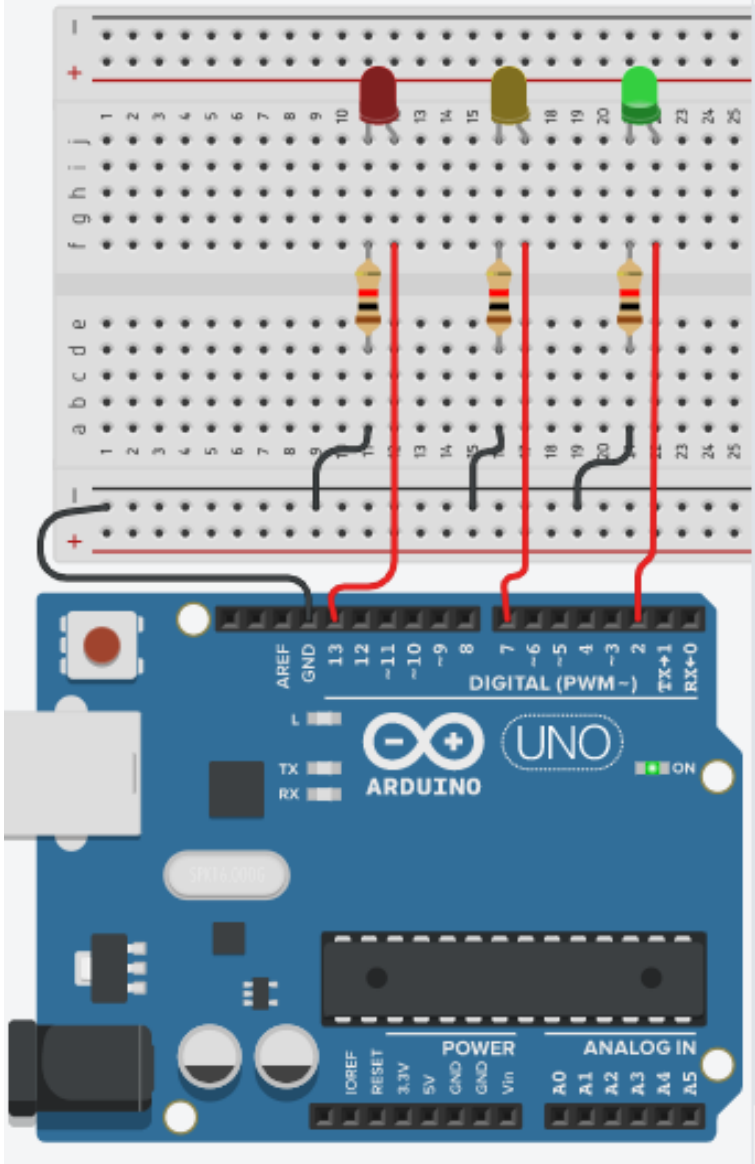
## تجربة عمل اشارة المرور

Project 2



المكونات

1- Led عدد 3
2- مقاومة عدد 3
3- اردوينو UNO
4- Breadboard
5- اسلاك توصيل




```ino
void setup()
{
  pinMode(13, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(2, OUTPUT);
}

void loop()
{
  //The red LED is on for 3 seconds
  digitalWrite(13, HIGH);
  delay(3000);

  //The yello LED is on for 1 seconds
  digitalWrite(7, HIGH);
  delay(1000);

  //off the led red
  //off the led yello
  digitalWrite(13, LOW);
  delay(1000);
  digitalWrite(7, LOW);
  delay(500);

  //The green LED is on for 3 seconds
  digitalWrite(2, HIGH);
  delay(3000);
   digitalWrite(7, HIGH);
  delay(1000);

  //off the green red
  digitalWrite(2, LOW);
  delay(500);
}
```

Serial Monitor

# شرح الكود

```
1  void setup()
2  {
3    pinMode(13, OUTPUT);
4    pinMode(7, OUTPUT);
5    pinMode(2, OUTPUT);
6  }
7
8  void loop()
9  {
10   //The red LED is on for 3 seconds
11   digitalWrite(13, HIGH);
12   delay(3000);
13
14   //The yello LED is on for 1 seconds
15   digitalWrite(7, HIGH);
16   delay(1000);
17
18   //off the led red
19   //off the led yello
20   digitalWrite(13, LOW);
21   delay(1000);
22   digitalWrite(7, LOW);
23   delay(500);
24
25   //The green LED is on for 3 seconds
26   digitalWrite(2, HIGH);
27   delay(3000);
28    digitalWrite(7, HIGH);
29   delay(1000);
30
31   //off the green red
32   digitalWrite(2, LOW);
33   delay(500);
34 }
```

1
2
3
4
5

دالة الإعداد , تعمل مرة واحدة عند التشغيل او اعادة التشغيل

تم تعريف ثلاث منافذ مخرجات OUTPUT التي سوف تعطينا اشارة كهربائية بقيمة 5V

**اما الأوامر البرمجية المتبقية تتكرر طوال فترة اتصال الأردوينو بالطاقة**

**هنا كال الاوامر متشابهة ومتكررة سوف نشرح الفكرة الاساسية من الاوامر وما عليك سوى تطبيقها والتعديل عليها من أجل الممارسة والتعلم من اخطائك**

1- هذا الامر من اجل **تشغيل** ال led الاحمر والذي متصل بمنفذ 13 ثم امر الانتظار 3 ثواني لتنفيذ الامر التالي

2- **تشغيل** ال led الاصفر والذي متصل بمنفذ 7 ثم امر الانتظار ثانية واحدة لتنفيذ الامر التالي

3- **إطفاء** المنفذ 13 و 7 اي ال led الاحمر والاصفر

4- **تشغيل** ال led الاخضر والذي متصل بمنفذ 2 ثم الانتظار 3 ثواني لينتقل الى المر التالي وهو تشغيل ال led الاصفر ثم الانتظار ثانية واحدة للانتقال الى الامر التالي

5- **إطفاء** المنفذ 2 اي ال led الاخضر والانتظار لمدة نصف ثانية من اجل العودة لبداية الكود لتنفيذه من جديد اي الى النقطة رقم 1 اعلاه

**لتشغيل التجربة او الاطلاع عليها وتعديلها اضغط على الصورة المصغرة ادناه**

# Project 3

تشغيل وإطفاء 13 led على التوالي

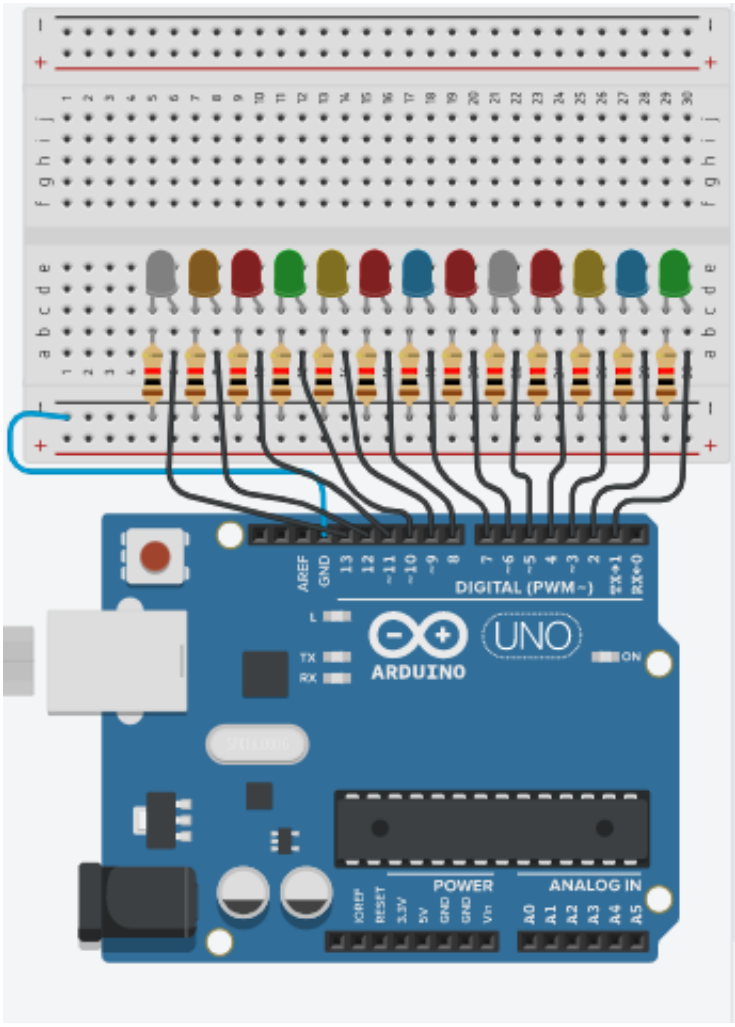
الفكرة من هذه التجربة توضيح الية عمل التكرار بواسطة for

**لو كتبنا كود هذه التجربة كما في التجربة السابقة لكتبنا 65 سطر برمجي**

لكن باستخدام هذه الطريقة كتبنا بعدد اسطر لا يتجاوز ال 10

## المكونات

1- **Led عدد 13**
2- **مقاومة عدد 13**
3- **اردوينو UNO**
4- **Breadboard**
5- **اسلاك توصيل**




```
void setup()
{
  for(int i=0; i<=13; i++)
  {
  pinMode(i, OUTPUT);
  }
}

void loop()
{
  for(int a=0; a<=13; a++)
  {
  digitalWrite(a, HIGH);
  delay(300);
  digitalWrite(a, LOW);
  delay(300);
  }
}
```

Serial Monitor

## شرح الكود

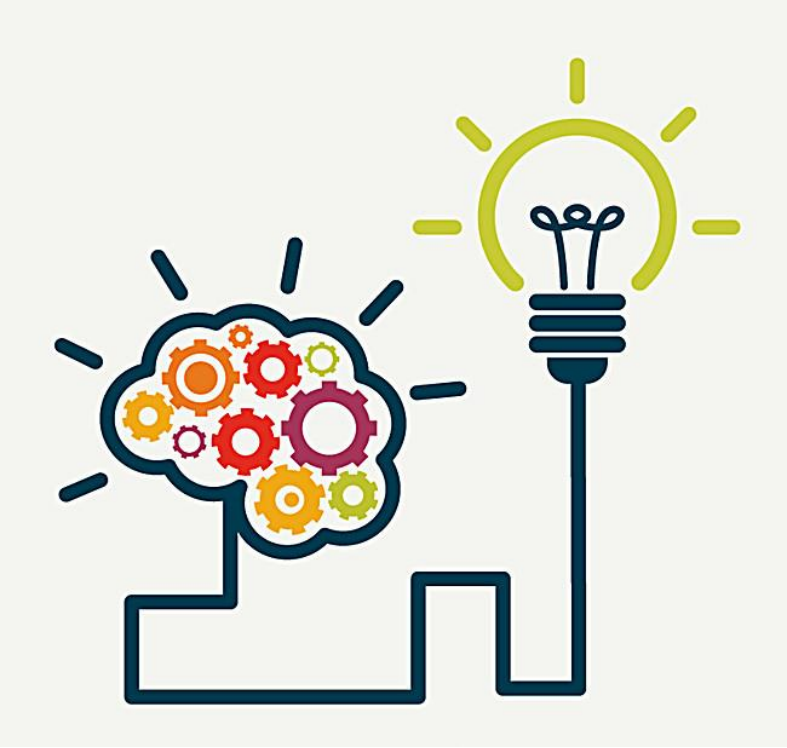
```
1  void setup()
2  {
3    for(int i=0; i<=13; i++)
4    {
5    pinMode(i, OUTPUT);
6    }
7  }
8
9  void loop()
10 {
11   for(int a=0; a<=13; a++)
12   {
13   digitalWrite(a, HIGH);
14   delay(300);
15   digitalWrite(a, LOW);
16   delay(300);
```

هذا الجزء من البرنامج ينفذ مره واحده عند اتصاله بالطاقة حيث تم تعريف متغير **(I)** والذي قيمته من (0 الى 13) ويزداد بمقدار 1 وذلك من اجل تعريف 13 متغير كمخرجات

اما هذا الجزء يتم تكرار تنفيذه باستمرار اتصاله بالطاقة

حيث كتبنا الامر الاول وهو تشغيل الليد ذو المنفذ **A** حيث **A** هو متغير من 0 – 13 وكما وضحنا اعلاه الدالة **FOR** تقوم بتكرار الكود ويزداد رقم المنفذ في كل مره حتى يتحقق الشرط والذي هو **A<=13** عند تحقق الشرط يذهب الى تنفيذ الامر الذي يليه

**لمشاهدة التجربة و تشغيلها او التعديل عليها اضغط على الصورة ادناه**



# الية عمل التكرار بواسطة FOR LOOP

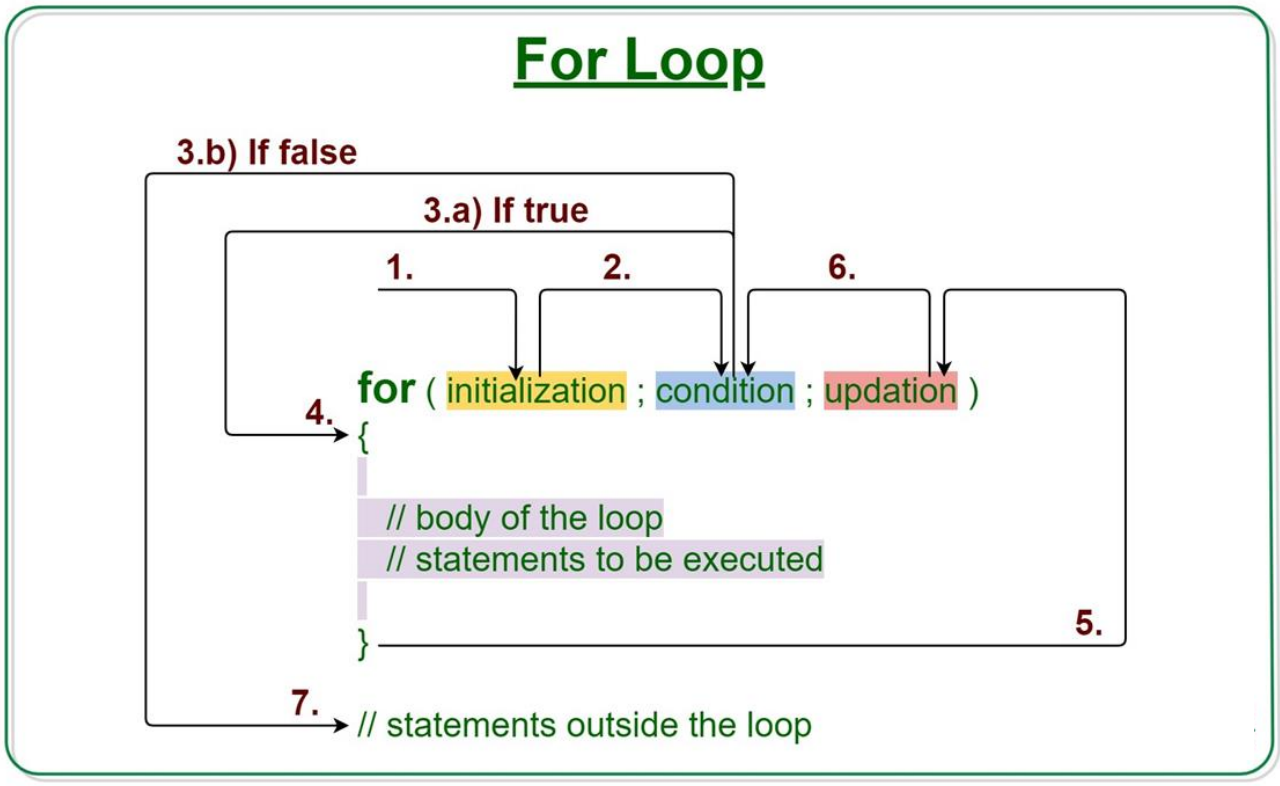
الحلقة for من الممكن تشبيهها بأنها عداد ينتهي عند وصول هذا العداد الى رقم معين

الصيغة العامة :

```
for ( expr1 ; expr2 ; expr3)
    {
    statement1;
    statement2;
    statement3;
    }
```

حيث ان :
expr1 : هو القيمة الابتدائية للتكرار
expr2 : وهو الشرط
expr3 : وهو الزيادة بعد كل دورة

## هذه الصورة توضح الية عملها ببساطة

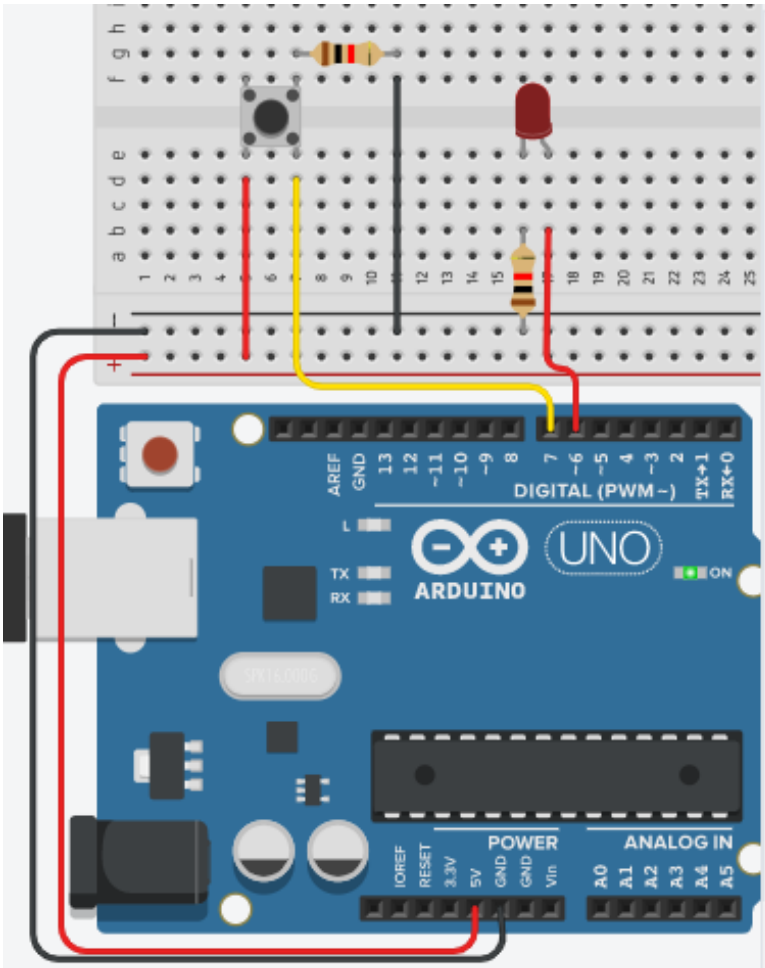


تشغيل واطفاء LED بواسطة PUSH BUTTON

# Project 4

## المكونات

1- Led عدد 1
2- مقاومة عدد 1
3- اردوينو UNO
4- Breadboard
5- اسلاك توصيل
6- Push button




```
byte button=7, led=6;
void setup()
{
  pinMode(button, INPUT);
  pinMode(led,OUTPUT);
}

void loop()
{
  bool push=digitalRead(button);
  if(push==true)
    digitalWrite(led,HIGH);
  else
    digitalWrite(led,LOW);
}
```

Serial Monitor

## شرح الكود

```
1  byte button=7, led=6;
2  void setup()
3  {
4    pinMode(button, INPUT);
5    pinMode(led,OUTPUT);
6  }
7
8  void loop()
9  {
10   bool push=digitalRead(button);
11   if(push==true)
12     digitalWrite(led,HIGH);
13   else
14     digitalWrite(led,LOW);
15 }
```

1

2

3

تم تعريف متغيرين من نوع BYTE لأنه يخزن بحجم 8 بت

المتغير الاول من اجل تعريف زر التشغيل والذي يكون ضمن المدخلات اي ان المنفذ 7 سوف يستقبل الاشارة عند الضغط على الزر اما المتغير الثاني وهو من المخرجات لتشغيل **LED**

1- في هذا الجزء تم اختيار منفذين المنفذ الاول للمدخلات اي عند الضغط على الزر تتولد اشارة تستلمها بطاقة الاردوينو عن طريق المنفذ رقم 7 يمكنك مشاهدة صورة الرابط اعلاه للفهم اكثر

2- في هذا الجز من الكود عرفنا متغير من نوع **BOOLEAN** والتي تحمل قيمتين فقط اما **TRUE** اي 1 وذلك عندما نضغط على الزر تتولد اشارة قيمتها **5V** اي تحقق الشرط في حال عدم الضغط على الزر اي ان قيمة المتغير **FALSE** اي 0 مما يعني عدم تولد اي اشارة يعني **0V**

3- كما ذكرنا في النقطة السابقة عندما يتحقق الشرط ينفذ الامر الخاص بتشغيل ال **LED** وعندما لم يتحقق اي عند عدم الضغط ينفذ الشرط بعد **ELSE** اي اطفاء ال **LED**

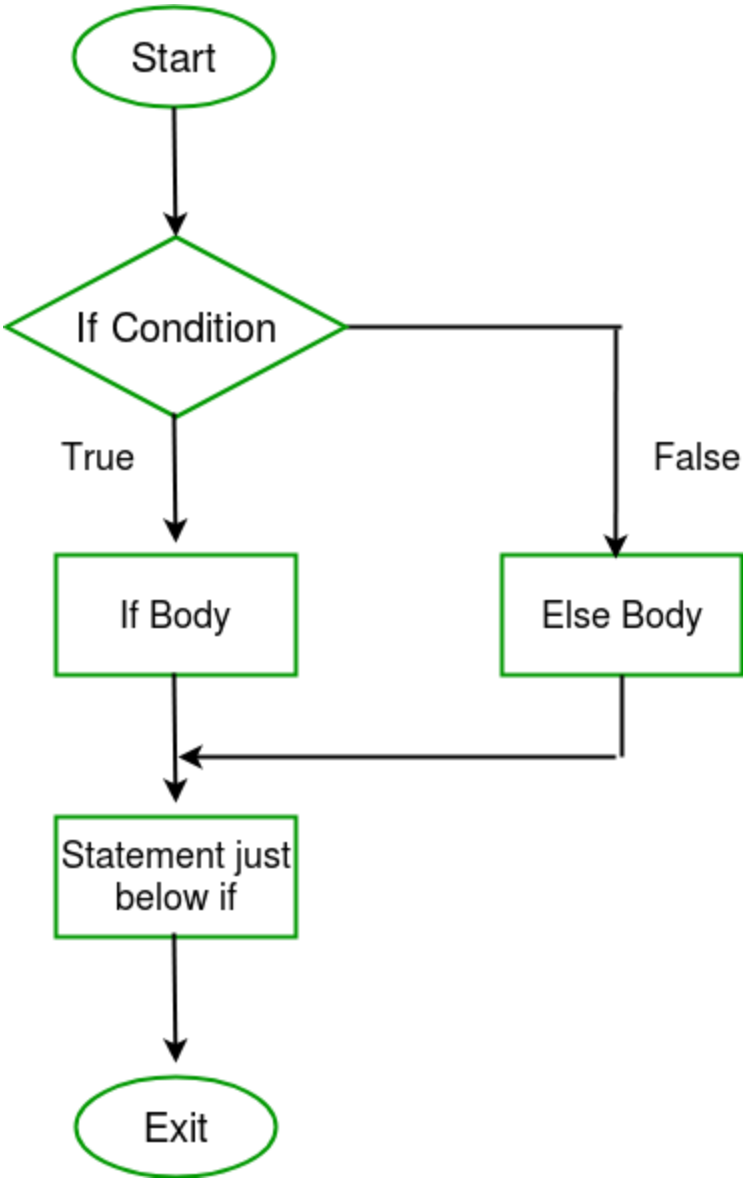
**لمشاهدة التجربة و تشغيلها او التعديل عليها اضغط على الصورة ادناه**

# الية عمل IF

هي عبارة شرطية مكونة من جزئيين من الاكواد البرمجية الذي نريده أن ينفذ بتحقق شرط (condition) نضعه داخل قوسي (***if***) والذي نريده أن ينفذ في حالة عدم تحقق الشرط نضعه بين قوسي (**else**)

شكل عبارة if --else الشرطية

```
If ( condition )
{
Statement1
}
else
{
Statement2
}
```



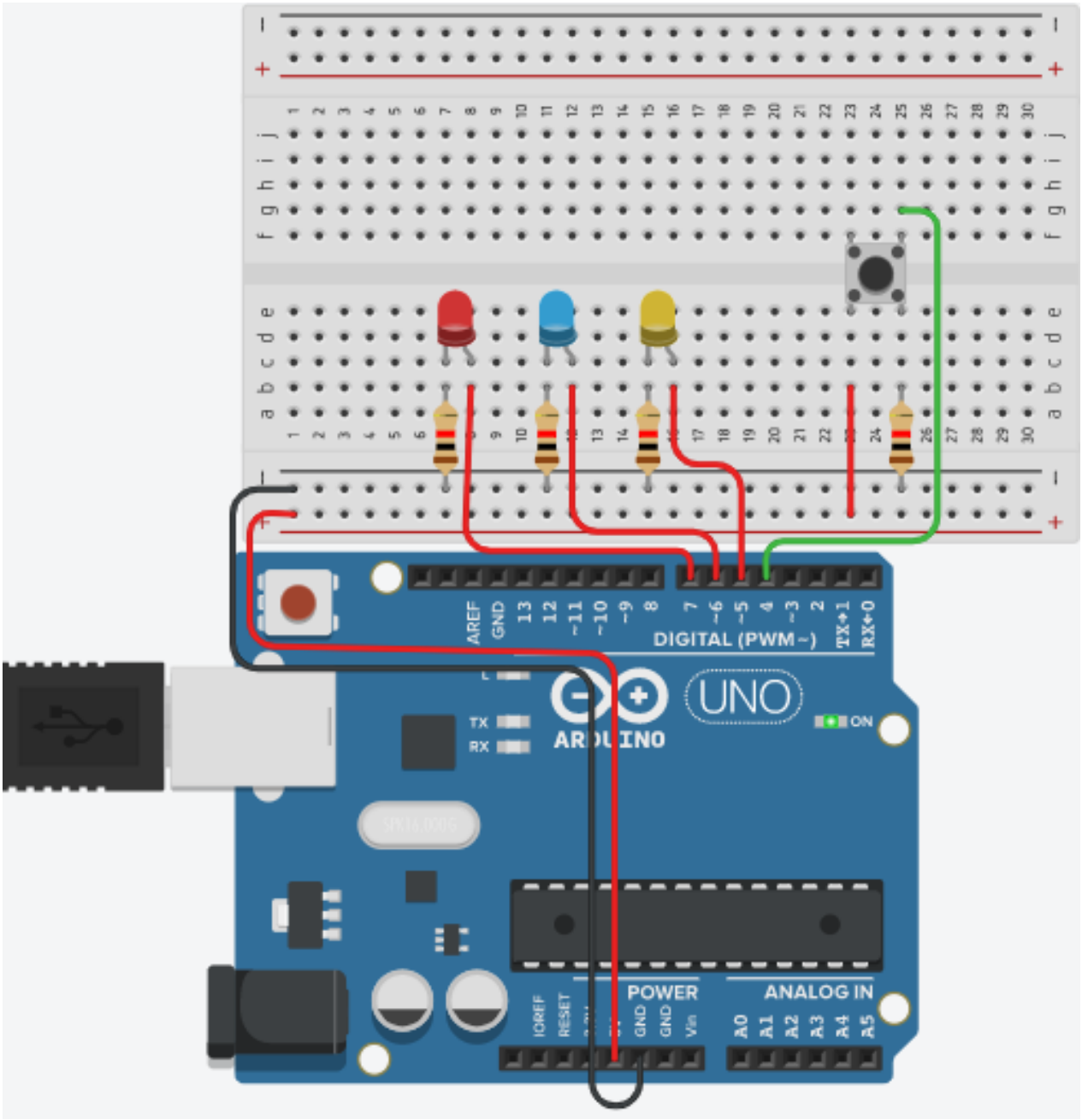
تشغيل LED 3 بالتتابع عن طريق PUSHPUTTON

# Project 5

الهدف من التجربة اعادة الممارسة للدالة IF وكذلك استخدام دالة جديدة وهي SWITCH CASE

## المكونات

1- Led عدد 3
2- مقاومة عدد 3
3- اردوينو UNO
4- Breadboard
5- اسلاك توصيل
6- Push button



## شرح الكود

```
1  byte button=4, yellow=5, blue=6, red=7;
2  byte counter=0;
3  void setup()
4  {
5    pinMode(yellow, OUTPUT);
6    pinMode(blue, OUTPUT);
7    pinMode(red, OUTPUT);
8    pinMode(button,INPUT);
9
10 }
11
12 void loop()
13 {
14   bool push= digitalRead(button);
15         if (push == true)
16       counter = counter+1;
17     if (counter == 4)
18       counter= 1;
19   switch (counter)
20   {
21     case 1: digitalWrite(yellow,HIGH); break;
22     case 2: digitalWrite(blue,HIGH); break;
23     case 3: digitalWrite(red,HIGH); break;
24   }
25   delay(1000);
26 }
```

1

2

3

4

1- في هذا القسم عرفنا 5 متغيرات من نوع BYTE والاول كان للزر الخاص بالتشغيل في المنفذ 4 والثاني والثالث والرابع في المنافذ 5 و 6 و7 لتشغيل ال LEDS اما المتغير الاخير COUNTER هو عداد اي قيمته متغيرة كما سوف نشرحه في النقطة 3 ادناه

2- هذا الجزء من البرنامج كما اتفقنا يتم تنفيذه مره واحدة ومن خلاله يتم تعريف المدخلات والمخرجات حيث تم تعريف مجموعة ال LEDS مخرجات بينما تم تعريف ال BUTTON كمدخلات في المنفذ 4

3- هنا الجزء المهم من الكود اما في السطر الأول تم تعريف متغير ذو قيمتين **true** او **false** هذا المتغير يقرأ قيمة المنفذ رقم 4 اي كما شرحنا في التجارب السابقة اما true والتي تمثل الضغط على الزر او false وتعني عدم الضغط

4- في حال ضغطنا على الزر يتحقق الشرط ويتم تنفيذ الامر الذي بعده وهو زيادة قيمة العداد **counter**

كما موضح اعلاه تم اعطاء قيمة 0 للعداد اي عندما نضغط على الزر يزداد العداد بقيمة +1

استخدمنا الدالة **switch case** واليكم الية عملها

الامر **switch (counter)** يعني وبسهولة اختبر قيمة ال **counter**

**case1** وتعني اذا كانت قيمة ال **counter** هي 1 شغل ال led الاصفر

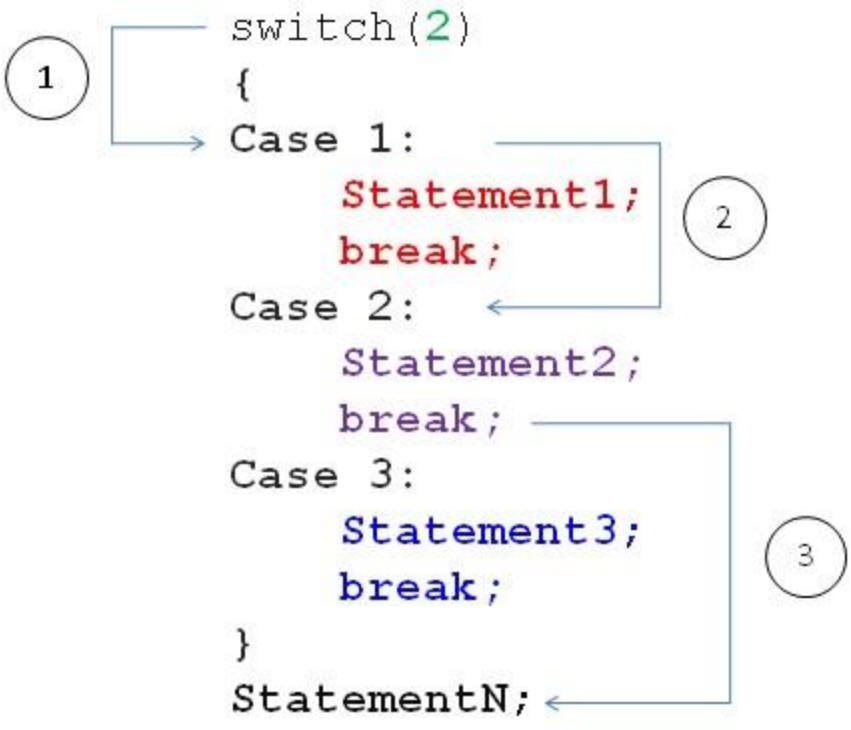
عند الضغط مرة ثانية على الزر سوف يزداد العداد كما موضح في هذه النقطة اعلاه لتصبح قيمة ال **counter** هي 2 ويتم تنفيذ الحلة الثانية اي **case2** وعند الضغط مرة ثالثة يزداد العداد ويتحقق الشرط لتنفيذ الحالة الثالثة **case3**

**مما يؤدي الى تشغيل ال LED الاول ثم الثاني ثم الثالث**

**لمشاهدة التجربة و تشغيلها او التعديل عليها اضغط هنا**

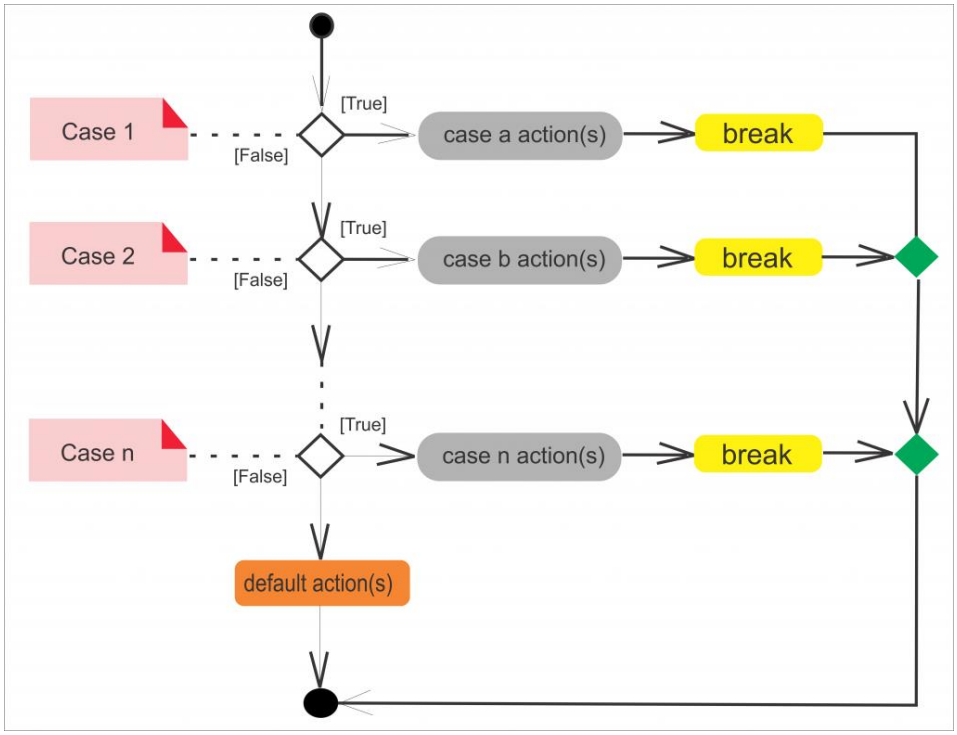
# عبارة SWITCH

إذا كانت لديك شجرة قرارات كبيرة , وكلها تعتمد علي قيم مختلفة لنفس المتغير , فإن صيغة العبارة switch تكون أكثر ملاءمة.




الكلمة الحاكمة break

بعد كل اختيار , توضع الكلمة الحاكمة , break وهي تدفع بالبرنامج للخروج من مجموعة الخيارات إلي خارجها , أي إلي أول سطر بعد القوس الحلزوني المحدد للكتلة الخاصة بها , وهو في برنامجنا هذا نهاية البرنامج . فمعني ذلك أنه بعد تنفيذ الخيار المطلوب ينتهي عمل مجموعة الخيارات , ويسير البرنامج بعد ذلك في طريقه . فإذا ما حدث ونسيت هذه الكلمة , ستجد أنك بعد تنفيذ خيارك , دخلت في الخيار التالي تلقائيا , ويسير البرنامج في غير الطريق الذي رسم له.

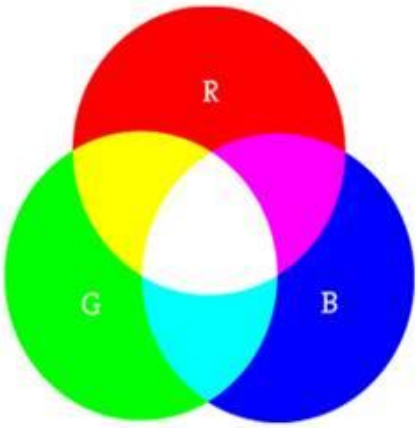


# تعديل عرض الموجة PWM

يحتوي الاردوينو على اكثر من مخرج من الممكن استخدامها للحصول على موجات **PWM.** يستخدم الأمر **analogWrite** للحصول على هذه الموجات على المخرج المحدد بالأمر وبـ **duty cycle** المحددة في الأمر أيضاً.

يمكن تحديد **duty cycle** تتراوح بين 0 و 255, حيث تمثل القيمة 255 النسبة 100% أي إن فولتية المخرج المحدد ستكون مساوية لفولتية الاردوينو في حين تثمل القيمة 0 النسبة 0% أي إن فولتية المخرج ستكون صفر فولت. للحصول على فولتية تساوي نصف فولتية المصدر على المخرج رقم 3 نستخدم الأمر **analogWrite (3,127).**

يُمكنك التحكم في شدة إضاءة **leds** المضيئة عن طريق تعديل قيمة دورة العمل. إذا كان لدينا ديود مضيء **RGB** (يصدر الألوان: أحمر (**red**)، أخضر (**green**)، أزرق (**blue**)) يُمكننا التحكم في كمية كل لون من الألوان الثلاثة ضمن مزيج الألوان عن طريق إعتام كل منها بدرجات متفاوتة. حسب القواعد الأساسية لمزج الألوان الرئيسية




في أغلب لوحات الأردوينو تعمل هذه الدالة على المنافذ 3 و 5 و 6 و 9 و 10 و 11. أي التي تحمل علامة (~)

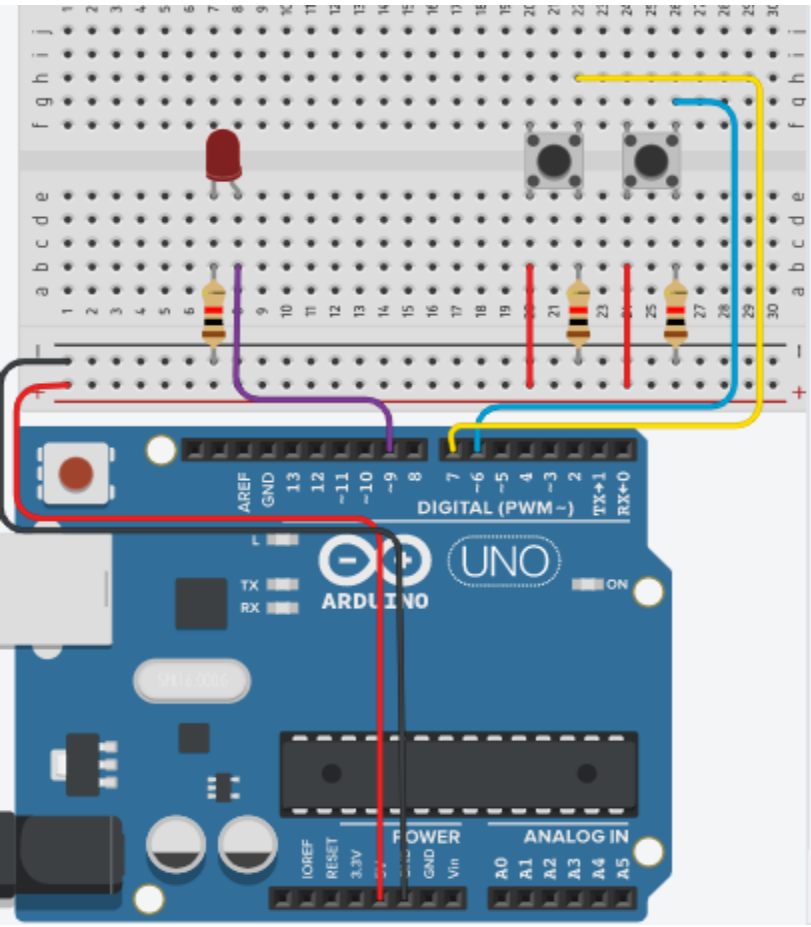

الكثير من المشاريع والتجارب التي سوف نستخدم فيها خاصية تعديل عرض النبضة او الموجة

# Project 6

## التحكم بشدة الاضاءة من خلال putton

### المكونات

1- Led عدد 1
2- مقاومة عدد 1
3- اردوينو UNO
4- Breadboard
5- اسلاك توصيل
6- Push button




```
byte b1=6, b2=7, led=9, light=25;
void setup()
{
  pinMode(led, OUTPUT);
  pinMode(b1, OUTPUT);
  pinMode(b2, OUTPUT);

}

void loop()
{
  if(digitalRead(b1))
    light=light+20;
  if(digitalRead(b2))
    light=light-20;

  analogWrite(led,light);
  delay(1000);
}
```

Serial Monitor

## شرح الكود

```
1  byte b1=6, b2=7, led=9, light=25;
2  void setup()
3  {
4    pinMode(led, OUTPUT);
5    pinMode(b1, OUTPUT);
6    pinMode(b2, OUTPUT);
7
8  }
9
10 void loop()
11 {
12   if(digitalRead(b1))
13     light=light+20;
14   if(digitalRead(b2))
15     light=light-20;
16
17   analogWrite(led,light);
18   delay(1000);
19 }
```

1

2

3

4

**1-** في هذا القسم تم تعريف 4 متغيرات الاول والثاني في المنفذين 6 و 7 كمخرجات لل button اما ال led من اجل تشغيل الضوء في المنفذ 9 واخيرا المتغير light ولبذي يأخذ قيمة 25 سوف يتم شرح الفائدة منه

**2-** تم تعريف المخرجات وهما ال led وال الازرار التي مهمتهما زيادة وتقليل شدة الضوء

**3-** في هذا القسم من الكود استخدمنا الشرط if اي ان في حال ضغطنا على الزر الاول والذي رمزه b1 سوف يتم زيادة قيمة ال light بمقدار 20 واذا ضغطنا على الزر b2 سوف تقل قيمة ال light بمقدار 20 كما اتفقنا ان

**يمكن تحديد duty cycle تتراوح بين 0 و 255, حيث تمثل القيمة 255 النسبة 100% أي إن فولتية المخرج المحدد ستكون مساوية لفولتية الاردوينو في حين تثمل القيمة 0 النسبة 0% أي إن فولتية المخرج ستكون صفر فولت. للحصول على فولتية تساوي نصف فولتية المصدر على المخرج رقم 3 نستخدم الأمر.analogWrite (3,127)** راجع صفحة 27

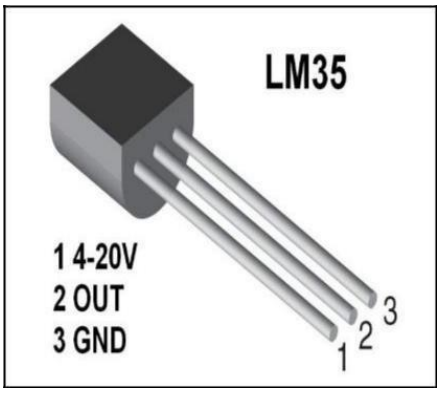
**4- الامر analogWrite ( led , hight )** لقراءة مقدار الزيادة او النقصان ومن خلالها تنتج شدة او ضعف الاضاءة في ال **led** ثم ينتظر ثانية واحدة لتنفيذ الامر التالي والتي يكون اما زيادة او نقصان في شدة الاضاءة تبعا لضغطة الزر

## لمشاهدة التجربة و تشغيلها او التعديل عليها اضغط هنا

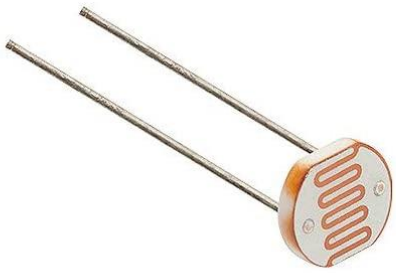
## انواع الحساسات والقطع

### 1- الحساسات

حساس درجة الحرارة مثلا **lm35** يمكننا التعرف على درجة حرارة الغرفة او قياس اي حرارة في التجارب المختلفة




حساس الضوء او المقاومة الضوئية وتسمى **LDR** يمكننا معرفة نسبة الاضاءة من خلال هذا الحساس



حساس المسافة التراسونيك يمكن تحسس بعد الاجسام بالسنتيمتر بواسطة ارسال الموجات الفوق الصوتية قد يستخدم في انظمة الإنذار او في روبوت تخطي العقبات

بالإضافة الى الكثير من الحساسات سوف نتعرف عليها لاحقا

# بعض الدوال و الاوامر البرمجية التي سوف نستخدمها في المشاريع

## الدالة begin

تستخدم هذه الدالة لفتح منفذ الاتصال، وهي تأخذ بين قوسيها رقم يمثل رقم المنفذ حيث ان لكل نوع من الاتصال له رقم خاص به، وهذه الدالة اجبارية إذا كنا نريد ان نعمل اتصال ويجب ان تكتب في المرحلة الثانية أي في داخل الدالة **void setup** ، مثال:

**Serial.begin(9600);**

الرقم 9600 يمثل منفذ الاتصال بالحاسوب عبر ال **serial monitor** ويستخدم نفس هذا المنفذ للاتصال عبر البلوتوث **HC-06** اما قطعة البلوتوث **HC-05** رقم المنفذ لها هو. 38400

## الدالة PRINT

تستخدم هذه الدالة لإرسال قيم الى الحاسب الالي ) او أي جهاز سيستقبل البيانات ( لعرضها على بيئة

التطوير باستخدام **SERIAL MONITOR** او حسب نوع الجهاز المستلم، مثال:

**SERIAL.PRINT(VOLTAGE);**

حيث ان VOLTAGE هو متغير معرف مسبقا يحمل قيمة فولتية، وهنا سترسل قيمة هذا المتغير ليستقبلها

الجهاز الاخر ( وإذا كانت عبر ال **SERIAL MONITOR** فان قيمة المتغير ستعرض على الشاشة ).

## الدالة PRINTIN

تستخدم هذه الدال ة لأرسال قيمة عبر منفذ الاتصال للجهاز المتصل بالاردوينو من خلال ال USB على بيئة

التطوير **SERIAL MONITOR** او عبر البلوتوث، فرقه عن الدالة السابقة هو ان هذا الامر يطبع في سطر جديد،

مثال:

**SERIAL.PRINTIN(“ HELLO“ );**

## الدالة read

يستخدم هذا الامر لاستلام قيم من الجهاز الاخر المتصل بالاردوينو ( الحاسب الالي او غيره ) الى الاردوينو عبر المنفذ المفتوح، مثال:

```
int X = Serial.read();
```

هنا سيتم تخزين القيمة المرسلة من الجهاز الاخر ( الحاسوب او غيره ) في المتغير X ، لاحظ ان المتغيرx هو من النوع int لأنه أي حرف او رقم يرسله الجهاز الاخر سيتم تحويله الى رقم ال ASCII الخاص به، وسيتم استقبال كل حرف على حدة أي حرف بعد حرف.

## الدالة readString

يستخدم هذا الامر لاستلام قيم من الجهاز الاخر المتصل بالاردوينو ( الحاسب الالي او غيره ) الى الاردوينو عبر المنفذ، مثال:

```
String X = Serial. readString ();
```

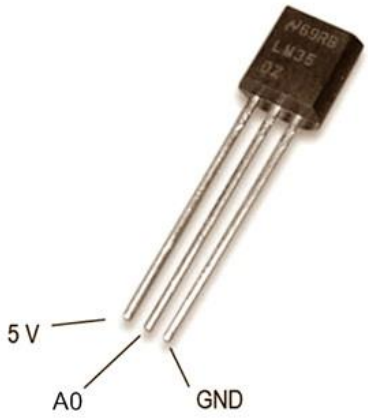
هنا سيتم تخزين القيمة المرسلة من الحاسوب في المتغير X ، لاحظ ان المتغير x هو من النوع String لأنه أي حرف او رقم يرسله الجهاز الاخر سيتم استقباله على شكل متغير نصي.

# Project 7 حساس درجة الحرارة

Project 7

## المكونات

1- حساس حرارة LM35
2- اردوينو UNO
3- Breadboard
4- اسلاك توصيل

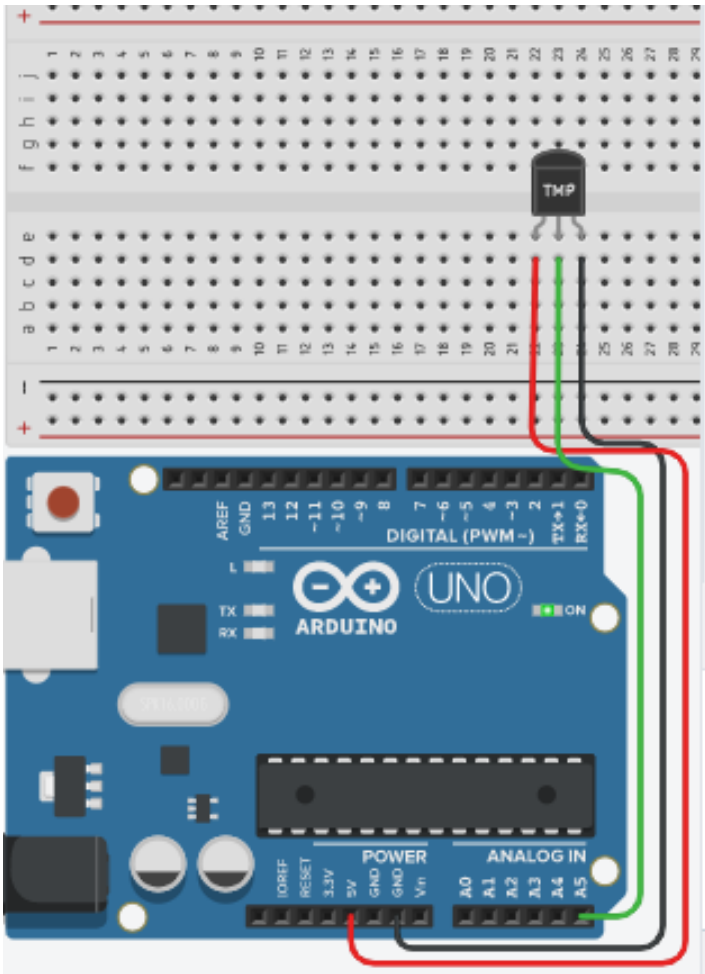



## شرح الربط

**اولا:** ايصال الطرف الموجب من جهة اليسار كما موضح في الصورة مع 5 فولت للاردوينو

**ثانيا:** ايصال الطرف السالف من جهة اليمين كما موضح في الصورة مع **GND** الخاص بالاردوينو

**ثالثا :** ايصال الرف الذي في الوسط الخاص بإستقبال القراءة التماثلية من الحساس بالمنفذ **A5** او اي منفذ من المنافذ التماثلية من **A0** ال **A5**




```
1  void setup()
2  {
3      Serial.begin(9600);
4
5  }
6
7  void loop()
8  {
9    double temp = analogRead(A5);
10   temp = temp * 0.48828125;
11   Serial.println(temp);
12   delay(1000);
13 }
```

Serial Monitor

```
46.88
85.94
93.75
50.78
42.97
42.97
```

## شرح الكود

```
1  void setup()
2  {
3      Serial.begin(9600);
4
5  }
6
7  void loop()
8  {
9    double temp = analogRead(A5);
10   temp = temp * 0.48828125;
11   Serial.println(temp);
12   delay(1000);
13 }
```

1

2

3

Serial Monitor

```
42.97
42.97
42.97
42.97
42.97
42.97
```

4

**قبل البدء بشرح الكود انصحكم بالرجوع الى صفحة 31 و 32 من اجل مراجعة عمل الدوال**

**1-** الدالة ;SERIAL.BEGIN(9600) تستخدم هذه الدالة لفتح منفذ الاتصال مع الحاسوب حيث لكل اتصال رقم منفذ خاص به الرقم 9600 يمثل منفذ الاتصال بالحاسوب عبر ال **serial monitor**

**2-** تم تعريف متغير يقبل الفواصل العشرية من نوع **double** هذا المتغير يقرأ القيم التماثلية التي يرسلها الحساس الى بطاقة الاردوينو

**3-** يتم ضرب ناتج القراءة بهذا الرقم **0.48828125** وذلك للتحويل من الفولتيه الى درجة الحرارة حسب منحني ربط الفولتية بدرجة الحرارة اما الامر التالي **Serial.println(temp);** لطباعة نتائج القراءة كما موضح في الصندوق رقم 4 ثم الانتظار ثانية واحده لطباعة القراءه التاليه

4- هنا نتيجة قراءة الحساس

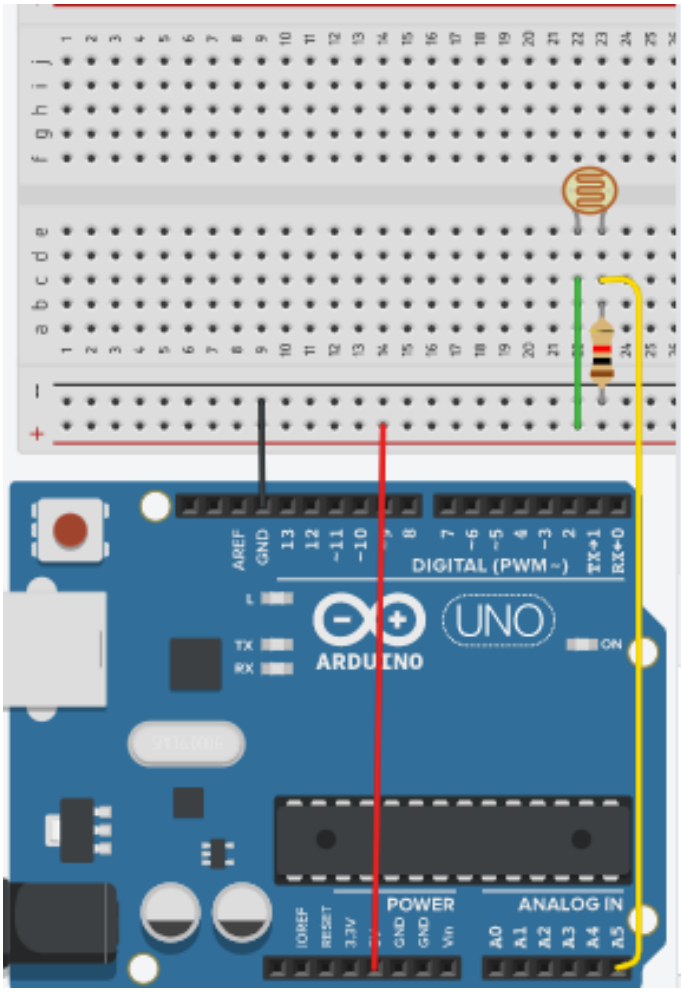
**لمشاهدة التجربة و تشغيلها او التعديل عليها اضغط هنا او على الصورة**

# Project 8

## المقاومة الضوئية لتحسس شدة الضوء LDR

### المكونات

1- مقاومة ضوئية LDR
2- اردوينو UNO
3- Breadboard
4- اسلاك توصيل




```
1  void setup()
2  {
3    Serial.begin(9600);
4  }
5
6  void loop()
7  {
8    int light= analogRead(A5);
9    Serial.print("light: ");
10   Serial.println(light);
11   delay(1000);
12 }
```

Serial Monitor

## شرح الكود

```
1  void setup()
2  {
3    Serial.begin(9600);
4  }
5
6  void loop()
7  {
8    int light= analogRead(A5);
9    Serial.print("light: ");
10   Serial.println(light);
11   delay(1000);
12 }
```

1 2 3 4

Serial Monitor

```
light: 224
light: 193
light: 193
light: 193
```

5

**قبل البدء بشرح الكود انصحكم بالرجوع الى صفحة 31 و 32 من اجل مراجعة عمل الدوال**

**-1** الدالة ;SERIAL.BEGIN(9600) تستخدم هذه الدالة لفتح منفذ الاتصال مع الحاسوب حيث لكل اتصال رقم منفذ خاص به الرقم 9600 يمثل منفذ الاتصال بالحاسوب عبر ال **serial monitor**

**-2** تم تعريف متغير من نوع **integer** هذا المتغير يقرأ القيم التماثلية التي ترسلها المقاومة الضوئية الى بطاقة الأردوينو

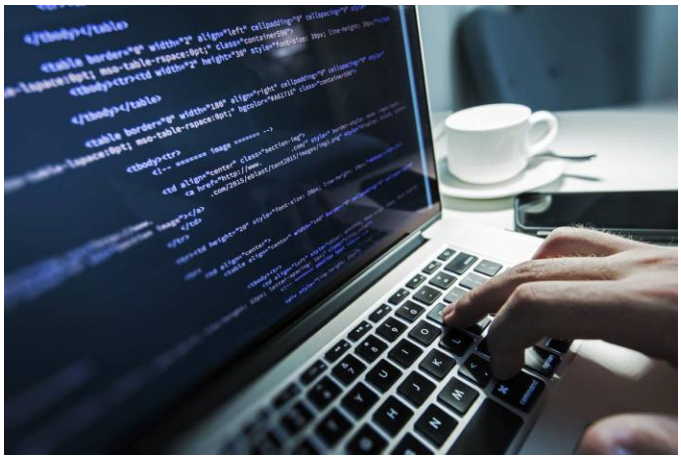
**-3** الدالة **Serial.printin** يمكننا من خلال هذه الدالة طباعة عبارة او كلمة توضيحية

كما نستخدمها لطباعة القراءات الخاصة بالحساس او المستشعر كما موضح في رقم 5

-4 ينتظر فترة 1 ثانية من اجل طباعة القراءة التالية

-5 **Serial Monitor** المراقب التسلسلي وهي النافذة التي من خلالها نتعرف على قراءة الحساس وحسب الامر الذي نعطيه كما فيشرح نقطة 3

**لمشاهدة التجربة و تشغيلها او التعديل عليها اضغط هنا او على الصورة**



تم بحمد الله الانتهاء من اعداد الجزء الاول من كتاب اردوينو بالعربي

انتظرونا في الجزء الثاني لبرمجة الأردوينو والمتحكمات الدقيقة بصورة متقدمة