Lab de Programmation et construction Arduino

GNG1503 – Génie de la Conception

Objectif

Utiliser la plate-forme Arduino IDE Software pour écrire des commandes logiques de contrôle simples. Ce code sera utilisé pour contrôler une variété d'entrées et de sorties connectées à un microcontrôleur Arduino Uno. Ce laboratoire exposera également les élèves à des codes de contrôle simples et à des ressources en ligne disponibles pour le logiciel Arduino.

Contexte

En conjonction avec les différents modèles de microcontrôleurs, la société Arduino fournit un environnement de développement intégré (IDE) Source Ouverte Arduino. Cet environnement de programmation est écrit en Java et a été conçu pour introduire la programmation à un public qui est moins familier avec le codage. Les programmes écrits dans cet environnement sont appelés «croquis» et le langage utilisé simple avec un support pour les langages de programmation C et C ++.

Un croquis Arduino de base se compose d'au moins deux fonctions: une fonction 'setup' et une fonction 'loop'. La fonction « setup » ne s'exécute qu'une fois au début de l'exécution du programme. Il exécute toutes les actions qui sont initialement nécessaires pour exécuter le reste du programme, comme l'initialisation de tous les composants externes et le réglage de la fréquence de communication entre la carte Arduino et le PC. La fonction « loop »e agit en tant que pilote de programmes et s'exécute dans une boucle continue (c'est-à-dire "boucle pour toujours"). Cette boucle spécifie l'ordre des opérations que le microcontrôleur exécutera sur une base continue.

Les cartes Arduino, ainsi que de nombreux composants externes compatibles Arduino, contiennent du matériel pour la communication série. Cette forme de communication peut envoyer des données à une vitesse élevée, mais transmet des bits de données en série (c'est-à-dire l'un après l'autre, un bit unique à la fois). Cela permet aux composants du système (ou d'un PC) de communiquer des séquences de données plus grandes et plus complexes avec beaucoup plus de facilité, avec moins de contraintes sur les broches physiques d'entrée ou de sortie.

En raison de la nature d'open-source d'Arduino, les sociétés tierces, ainsi que les utilisateurs finaux sont libres de prendre le logiciel et de le personnaliser selon leurs besoins individuels. Cela signifie que pour presque n'importe quelle application, les croquis sont disponibles pour téléchargement en ligne. Les entreprises fabriquant des périphériques externes (comme un écran de moteur ou des capteurs) à utiliser avec les cartes Arduino ont souvent des croquis d'exemple et des bibliothèques de croquis disponibles pour une utilisation spécifique pour leurs produits. Ces ressources peuvent être utilisées pour créer un croquis personnalisé qui peut utiliser plusieurs périphériques à la fois, en combinant des aspects de nombreuses esquisses d'exemples différents. Avant d'écrire votre propre code, comme nous le ferons ici, il en vaut la peine de regarder en ligne pour déterminer si quelque chose de similaire est disponible en ligne.

Téléchargements

Ce laboratoire utilise des bibliothèques IDE Arduino qui ne viennent pas avec l'environnement IDE Arduino. Pour que ces programmes fonctionnent correctement, ces

bibliothèques doivent être téléchargées et insérées dans le dossier des bibliothèques dans les fichiers système. Vous trouverez une liste des bibliothèques requises dans la section de dépannage. Celles-ci devraient déjà être installées sur les ordinateurs de laboratoire, mais peuvent être installées si un étudiant utilise son ordinateur personnel ou si les ordinateurs de laboratoire ont été corrompus ou modifiés (par exemple, par d'autres élèves).

L'Arduino IDE peut être téléchargé ici: https://www.arduino.cc/en/Main/Software

Les bibliothèques utilisées dans ce laboratoire sont:

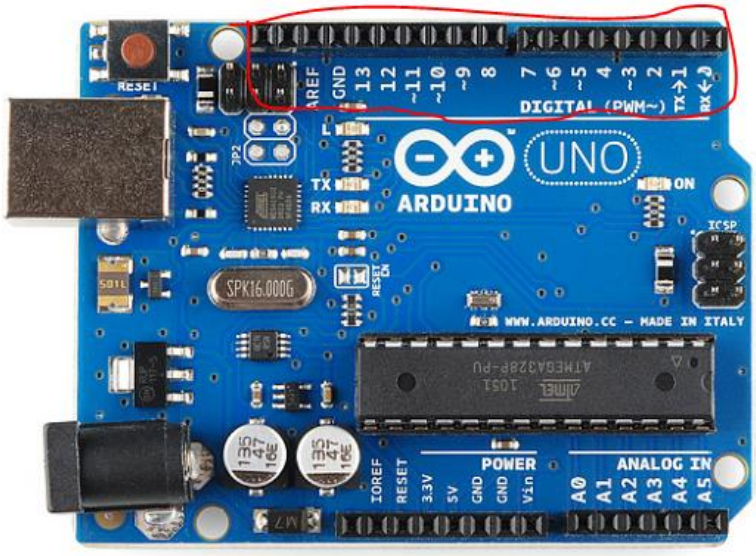
- Dallas Temperature: https://github.com/milesburton/Arduino-Temperature-Control-Library

Pour installer les bibliothèques requises, décompressez les dossiers téléchargés dans un seul dossier, puis déplacez ce dossier dans le dossier des bibliothèques Arduino dans le dossier des fichiers système. Redémarrez l'IDE Arduino pour que les bibliothèques apparaissent. Pour la plupart des utilisateurs de Windows, la voie par défaut est Program Files (x86)>Arduino>libraries mais cela peut changer avec différents systèmes d'exploitation, différentes versions de Windows ou si l'IDE Arduino a été installé à un emplacement différent sur l'ordinateur. Adafruit.com a un tutoriel utile sur les installations de bibliothèques qui peuvent être trouvés ici: https://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use

Carte Arduino

- L'Arduino détecte l'environnement en recevant des entrées provenant de différents types de capteurs et peut également affecter son environnement avec des sorties qui commandent des lumières, des moteurs, des réchauffeurs ou d'autres actionneurs. Ces signaux d'entrée et de sortie sont disponibles sur un en-tête de broche numérique (voir schéma ci-dessous).

    - **Broche Numérique**:



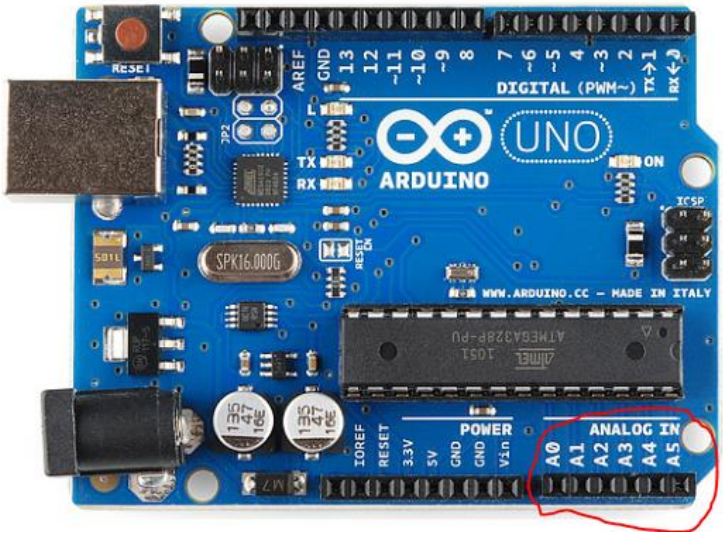
- Ces broches peuvent être configurées comme entrées digitales et utilisées pour envoyer des signaux dans le processeur Arduino (par exemple pour lire les niveaux de tension logiques détectés). Les signaux digitales sont normalement

binaires et n'ont que deux valeurs de tension distinctes (0V ou «faible» ou 5V ou «haut»).

- Les broches peuvent également être configurées pour servir de SORTIE pour alimenter les tensions logiques digitales «élevées» ou «faibles» du processeur Arduino (par exemple, pour allumer ou éteindre les éléments).
- **Pin13 (LED).** Il y a un voyant intégré préréglée à la broche digitale 13. Lorsque la valeur de la broche est élevée par le processeur, le voyant de la carte est allumé, lorsque la broche est BASSE, elle est éteinte. Cela peut être utilisé comme indicateur d'état lorsque les programmes sont en cours d'exécution.

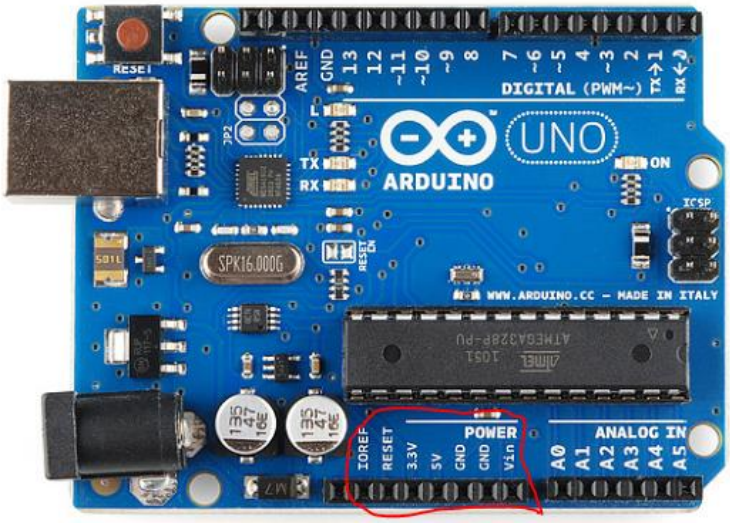
Il existe de nombreux types de signaux réels qui ne sont pas binaires digitales (c'est-à-dire qu'ils ont plus de niveaux de tension que juste «élevé» ou «faible»). Ces types de signaux peuvent avoir une gamme continue de valeurs de tension et sont appelés signaux analogiques. Ces signaux doivent être convertis en représentations digitales qui peuvent être utilisées à l'intérieur de la partie d'exécution logicielle du processeur Arduino (qui utilise uniquement des signaux binaires numériques). Les entrées de signaux analogiques peuvent être converties en numérique via les têtes de broche analogiques (voir schéma ci-dessous). La conversion analogique-digitales (A / D ou "A à D") se fait à l'intérieur du processeur avec des circuits spécialisés.

- **Broches Analogiques:**



Pour alimenter les circuits électriques simples, l'Arduino fournit également quelques valeurs de sortie de tension numériques communes (par exemple, 3,3 V et 5 V) ainsi qu'une tension 0 V ou de référence (terre ou GND). Ces broches sont situées à côté des têtes des broches analogiques (voir schéma ci-dessous). La broche 'Vin' est une tension de baisse de diode éloignée de la tension d'entrée fournie pour alimenter la carte Arduino et générer les autres tensions.

- **Broches de Tension:**



Logiciel Arduino

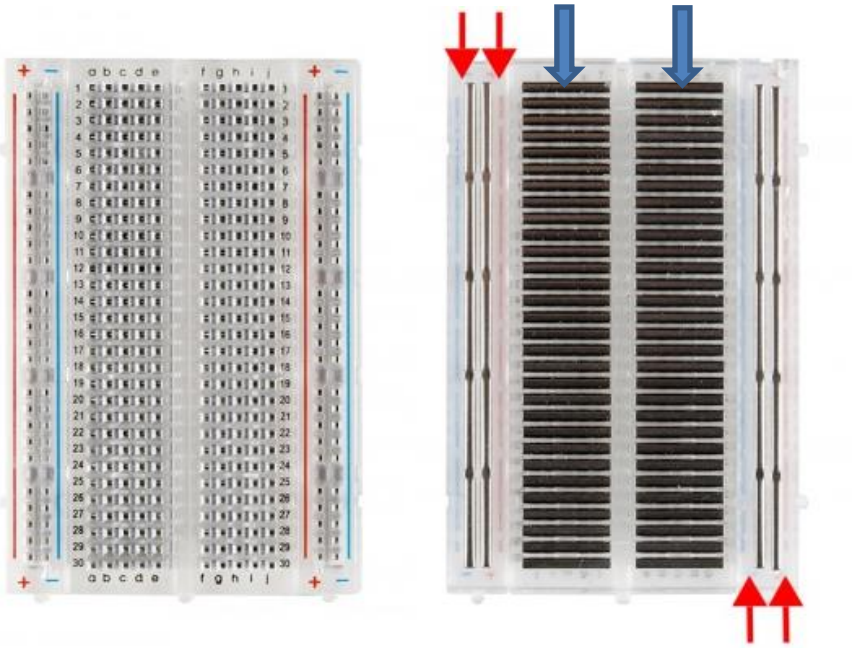
Vous pouvez dire à votre Arduino quoi faire en écrivant du code dans le langage de programmation Arduino et en utilisant l'environnement de développement Arduino. Certains faits importants pour le laboratoire sont donnés ci-dessous et sont destinés à aider les nouveaux programmeurs (ce qui est probablement moins utile pour les programmeurs Arduino plus expérimentés). Si vous avez besoin de plus d'aide, demandez à l'assistant d'enseignement de clarifier ou d'utiliser l'internet pour essayer de trouver les réponses à vos questions:

- **Barres de commentaires:** "/* */" et "//": Elles vous permettent d'écrire des commentaires dans votre code. Tout ce que vous écrivez entre "/* */" ou après "//" sera un commentaire et n'affectera pas le code.
- **Point-virgule** ";": Ce caractère termine une instruction de programme et permet au compilateur de connaître 'la fin de la ligne / instruction courante'. Un compilateur est un programme informatique qui transforme le code source écrit dans un langage de programmation en un autre langage informatique, généralement celui qui peut être exécuté plus facilement sur un processeur réel.
- **Fonction "setup":** La fonction de configuration ne s'exécute qu'une seule fois et exécute toutes les actions requises initialement pour exécuter le reste du programme, comme initialiser les composants périphériques et régler la fréquence de communication entre la carte Arduino et le PC. Https://www.arduino.cc/fr/Reference/Setup
- **Fonction "loop":** la fonction boucle sert de pilote de programme; Il s'exécute en boucle continue et spécifie l'ordre d'opérations que le microcontrôleur exécutera. Https://www.arduino.cc/fr/Reference/Loop. L'exécution commence en haut, parcourt le contenu de la boucle et recommence à s'exécuter à partir du haut. Cette procédure est répétée pour toujours.
- **Parenthèses** "{}" Démarre et termine une fonction ou est utilisé pour regrouper un ensemble de déclarations différentes, ce qui effectivement les rends comme une seule instruction séquentielle longue, exécutée sous la forme d'un «globe».
- **pinMode ():** configure la broche spécifiée pour se comporter soit comme INPUT soit comme OUTPUT:
    - Syntaxe: pinMode (broche, mode). Pour plus d'informations, consultez https://www.arduino.cc/fr/Reference/PinMode

- **digitalWrite ():** Écrit une valeur HIGH ou LOW sur une broche digitale.
    - Si la broche a été configurée comme OUTPUT avec pinMode (), sa tension sera réglée sur la valeur correspondante: 5V (ou 3.3V sur les cartes 3,3V) pour HIGH, 0V (terre) pour LOW.
    - Syntaxe: digitalWrite (pin, value) Voir https://www.arduino.cc/fr/Reference/DigitalWrite pour plus d'informations.
- **Delay (Retard):** Interrompt le programme pour la quantité de temps (en millisecondes) spécifiée comme paramètre.
    - Syntaxe: delay (ms) https://www.arduino.cc/fr/Reference/Delay
- **Variable**: une variable est un emplacement de stockage qui est apparié avec un nom symbolique associé (un identifiant), qui contient une certaine quantité connue ou inconnue d'information appelée la valeur de cette variable.
  Dans ce laboratoire nous allons utiliser deux types de variables de valeurs de type: "int" et "float".
    - Int: Permet de stocker des nombres entiers (c'est-à-dire comptant des nombres entiers). Https://www.arduino.cc/fr/Reference/Int
    - Syntax: int variable = value;
    - Float: stocker les nombres à virgule flottante, un nombre qui a un point décimal et un exposant. Il en résulte que ces types de variables peuvent avoir une portée beaucoup plus grande que ce qui est possible pour les entiers, mais ils auront des erreurs d'ajustement car ils sont mis en œuvre avec une précision finie (c'est-à-dire qu'un certain nombre de chiffres décimaux peut être utilisé). Voir https://www.arduino.cc/fr/Reference/Float pour plus d'informations
    - Syntaxe: float variable = valeur;
- **Serial.begin()**: Cela démarre le matériel de l'interface de communication série. Vous aurez besoin de cela, si vous voulez imprimer quoi que ce soit à l'aide du moniteur en série.
    - Pour plus d'informations, voir https://www.arduino.cc/fr/Serial/Begin
    - Syntaxe: Serial.begin (9600) définit l'interface série à exécuter à un débit de 9600 bits série / s
- **Serial.print():** Imprime les données sur le port série en tant que texte lisible par l'utilisateur
    - Syntaxe: Serial.print(valeur)
    - Voir https://www.arduino.cc/fr/Serial/Print pour plus d'informations.
- **Déclaration "if":** Voir https://www.arduino.cc/fr/Reference/If : Cela vous permet d'exécuter conditionnellement les instructions suivantes, en fonction d'une condition testée.

Plaque de prototypage

Une plaque de prototypage (breadboard en anglais) est utilisée pour prototyper un circuit temporaire. Tu peux construire, tester et analyser un circuit sans connections permanentes. Il est fait de bandes terminales et de bandes de distribution. Les bandes terminales sont utilisées pour tenir n'importe quel nombre de composantes en place et créer des connections électriques dans une rangée horizontale. Les bandes de distribution sont les longues bandes verticales qui sont utilisées pour faciliter les connections à la tension (+) et la terre (-) en les plaçant dans une colonne. Pour du contexte général sur le fonctionnement des plaques de prototypage, regarde ce vidéo du YouTube : https://www.youtube.com/watch?v=gPXlPs53Wfs ou https://www.youtube.com/watch?v=6WReFkfrUIk (anglais).



Aperçu des Appareils et Équipement

L'équipement qui va être utilise dans ce lab inclus:

- 1 x Microcontrôleur Arduino Uno R3
- 1 x Cable USB 2.0 Type A Plug à USB 2.0 Type B Plug
- 1 x Ordinateur de Lab ou Personnel (avec Windows, Macintosh, ou Linux OS)
- Logiciel Arduino IDE
- Bibliothèques correspondantes Arduino IDE (expliqué plus bas)
- 1 x Module de relais à 2 canaux 5V
- 1 x Capteur de température DS18B20
- 1 x Résistance 4.7K ohm
- 1 x Moitié plaque de prototypage
- 1 x Prise murale 12v
- 4 x Cable male-femelle
- 7 x Cable male-male
- 1 x Coussin chauffant 5x10cm
- 1 x Ventilateur 12v
- 1 x Bloc d'aluminium

Préparation Pré-Lab

Avant d'arriver au laboratoire, les élèves doivent passer en revue le manuel du laboratoire et se familiariser avec les procédures et les procédures du laboratoire. Les étudiants peuvent utiliser leur propre ordinateur pour ce laboratoire s'ils le souhaitent (le logiciel est gratuit), à condition qu'ils aient téléchargé et installé l'IDE Arduino, ainsi que les bibliothèques requises décrites dans la section téléchargements. La révision de la syntaxe IDE Arduino, ainsi que l'essai d'écrire certains des programmes à l'avance est également encouragée. Une liste utile de commandes utilisées dans l'IDE Arduino peut être trouvée ici: https://www.arduino.cc/en/Reference/HomePage.

Tous les ordinateurs de laboratoire ont déjà installé le logiciel.

Questions Pré-Lab

Quel est le langage de programmation utilise avec Arduino?

Le langage de programmation utilisé est C et C++

Quels sont les 2 parties principales dans un programme Arduino?

Les deux parties principales dans un programme arduino sont la fonction setup() et la fonction loop().

Quelles sont les 3 sections principales sur la plaque Arduino Uno?

Les 3 sections principales sur la plaque arduino sont : la broche numérique, les broches analogiques et les broches de tension.

Quelle est la différence entre les bandes terminales et les bandes de distribution sur une plaque de prototypage?

Les bandes terminales sont utilisées pour tenir n'importe quel nombre de composantes en place et créer des connections électriques dans une rangée horizontale.

Les bandes de distribution sont les longues bandes verticale qui sont utilisées pour faciliter les connections à la tension (+) et la terre (-) en les plaçant dans une colonne.

Quelle est la signifiance de la polarité électrique?

Etat d'une structure possédant deux pôles .

Partie A - Programme de clignotement des DEL

La prochaine partie de ce laboratoire passe en revue la méthode de base pour connecter et programmer un microcontrôleur Arduino. Il utilise un schéma d'échantillonnage inclus dans l'une des bibliothèques par défaut de l'IDE d'Arduino et est principalement utilisé pour vérifier que le microcontrôleur et la connexion fonctionnent correctement et donne une idée de base de la façon d'exécuter un programme Arduino très simple.

1. Connectez la carte Arduino à un port USB ouvert de l'ordinateur à l'aide du câble USB. Les DELs de la carte Arduino doivent s'allumer, car la connexion USB est alimentée.
2. Lancez le logiciel Arduino IDE et choisissez 'Arduino / Genuino Uno' dans le menu déroulant Tools>Board. Dans l'onglet "Tools", sélectionnez "Port" et choisissez le port série auquel l'Arduino est connecté. Si le port série correct n'est pas apparent, débranchez l'Arduino, et voyez quel port disparaît ``. Rebranchez la carte et sélectionnez ce port. N'oubliez pas que si vous ne sélectionnez pas le port série correct, vous ne pourrez pas afficher de sortie imprimée.
3. Ouvrez Fichier> Exemples> 01.Basics> Blink. Cela ouvrira un nouveau croquis qui ressemble à celle ci-dessous.

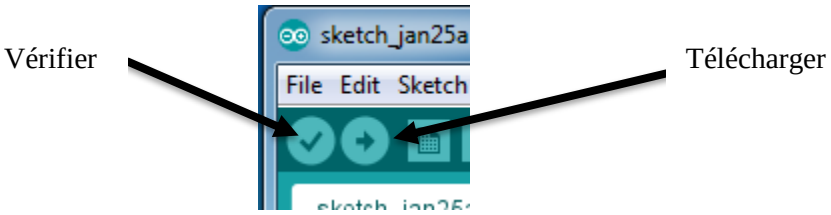
```
  modified 2 Sep 2016
  by Arturo Guadalupi

  modified 8 Sep 2016
  by Colby Newman
*/


// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

4. Maintenant vérifiez et téléchargez le croquis sur la carte Arduino. Pour vérifier, cliquez sur la coche en haut à gauche. L'IDE Arduino essayera de compiler l'esquisse (sans télécharger à la carte), et avertira de toute erreur de syntaxe dans la programmation. Une fois cette opération terminée, appuyez sur le bouton de téléchargement (flèche à côté de la coche). Le croquis sera recompilé et chargée au tableau.



5. Une fois le téléchargement terminé, le programme s'exécutera automatiquement sur le microcontrôleur. Dans ce cas, la DEL marquée «L» sur l'Arduino doit clignoter lentement.

Quelle est la vitesse du clignotement (c'est-à-dire le taux de clignotement et son cycle)?

1 clignotement toutes les 2 secondes

Essayez de modifier la valeur de votre délai. Changez-le à 5000 et à 300, vérifiez et téléchargez le code. Qu'est-il arrivé au «L» sur la carte Arduino dans les deux cas?

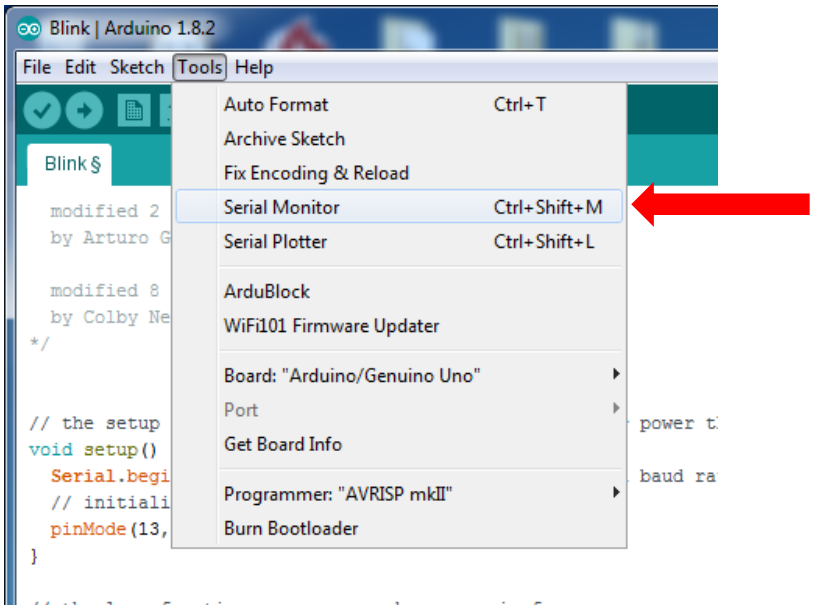
On remarques que le L clignote plus rapidement à 300 que à 5000.

6. Changez la variable LED_BUILTIN à 13, ils représentent la même broche sur l'Arduino. Imprimez la valeur de cette variable dans le moniteur série en ajoutant Serial.begin(9600); et Serial.print(LED_BUILTIN); comme dans la photo suivante.
7. Télécharge ton nouveau code.

```
// the setup function runs once when you press reset or power the board
void setup() {
  Serial.begin(9600); //start a serial connection for a baud rate of 9600
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
  Serial.print(LED_BUILTIN);        //print the value of LED_BUILTIN
}
```

8. Pour ouvrir le moniteur série, sélectionnez Tools> Serial Monitor.



9. Remplacez Serial.print(LED_BUILTIN) par Serial.println(LED_BUILTIN) puis par Serial.println ("LED_BUILTIN"). Vérifiez et téléchargez votre code. Que se passe-t-il dans ces deux cas?

Dans le premier cas, on remarquait que les 13 s'ecrivaient les uns a la suite des autres.

Dans le second cas, les 13 s'ecrivent les uns en dessous des autres.

Par la suite on remarque que la valeur 13 à été changée en "LED_BUILTIN"

Partie B – Lecture du capteur de température

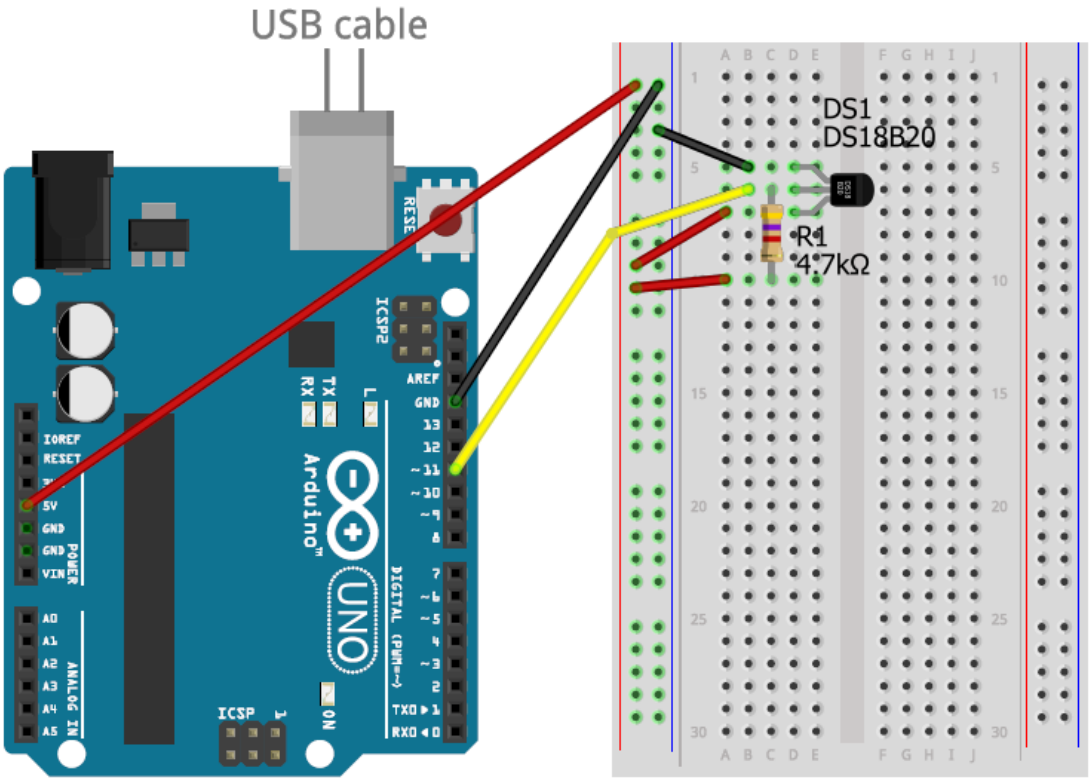
Cette partie du lab utilise un circuit simple sur une plaque de prototypage pour faire la connexion entre les composants nécessaires du lab (voir diagramme ci-dessous).

Prendre note que le plus organisé qu'est un circuit, le plus facile que c'est à le déboguer, si/quand il y a des problèmes et le moins probable que des câbles se détachent, etc. En général, tu veux que tes câbles soient rentrés complètement dans la plaque pour que tu puisses voir toutes les connexions que tu fais et pour que les personnes qui t'aident peuvent aussi les voir! Utiliser des câbles de couleur (rouge pour représenter +5V et noir pour la terre par exemple) est une pratique commune pour aider à déboguer. Tandis que le circuit va ***fonctionner*** avec *n'importe quel* couleur de câble, suive ces conventions va rendre ton circuit 'meilleur', comme des améliorations qu'on a en usabilité, en incorporant des commentaires dans les programmes d'ordinateur.

Ce programme va lire la température de l'environnement à l'aide d'un capteur de température DS18B20. La température sera imprimée sur le moniteur série de l'IDE Arduino. Le capteur de température a une plage de fonctionnement de température entre -55 ° C et + 125 ° C et est précis à ± 0,5 ° C dans la plage de -10 ° C à + 85 ° C.

1. Ouvrez un croquis exemple dans le Arduino IDE (File>Examples>MAX31850 DallasTemp>Simple). Assurez-vous que le bon microcontrôleur et port de communication sont toujours sélectionnés en regardant la partie inférieure droite de la fenêtre. Tu peux suivre le sens des instructions en lisant les commentaires (mots gris) dans le code.

2. Connecter le capteur de température DS18B20 à l'Arduino en utilisant le schéma de câblage indiqué dans la figure ci-dessous.



*Note: le capteur de température ci-dessus est le symbole mais la figure suivante est son apparence réelle

Le capteur de température doit être installée solidement à l'intérieur du bloc d'aluminium.

3. Assure-toi que les bibliothèques ‘OneWire’ et ‘DallasTemperature’ sont en orange en haut du croquis, qui indique qu’elles sont reconnues et installées correctement. Le lien pour télécharger les bibliothèques et comment les installer peut être trouvé en haut de ce manuel.

```
Simple §
#include <OneWire.h>
#include <DallasTemperature.h>
```

4. Change la valeur de la variable ONE_WIRE_BUS à 11 puisque c'est là où notre capteur est branché à l'Arduino.

```
#include <DallasTemperature.h>

// Data wire is plugged into port 11 on the Arduino
#define ONE_WIRE_BUS 11
```

5. À la fin de la boucle, ajoute une ligne pour imprimer au moniteur de série pour indiquer les unités de la valeur et un délai pour mesurer la température chaque 0.5 secondes, comme dans la figure suivante.

```
  Serial.print("Temperature for the device 1 (index 0) is: ");
  Serial.println(sensors.getTempCByIndex(0));
  Serial.println("C");
  delay(500); // wait 500 miliseconds
}
```

6. Branche la plaque Arduino à l'ordinateur en utilisant un câble USB et télécharge le code à l'Arduino. Ouvre le moniteur de série (Tools>Serial Monitor) pour voir si tu lis la température correctement.
7. Débranche le câble USB quand tu as terminé.

Ne débranchez pas le capteur de température

Partie C – Régulateur de température

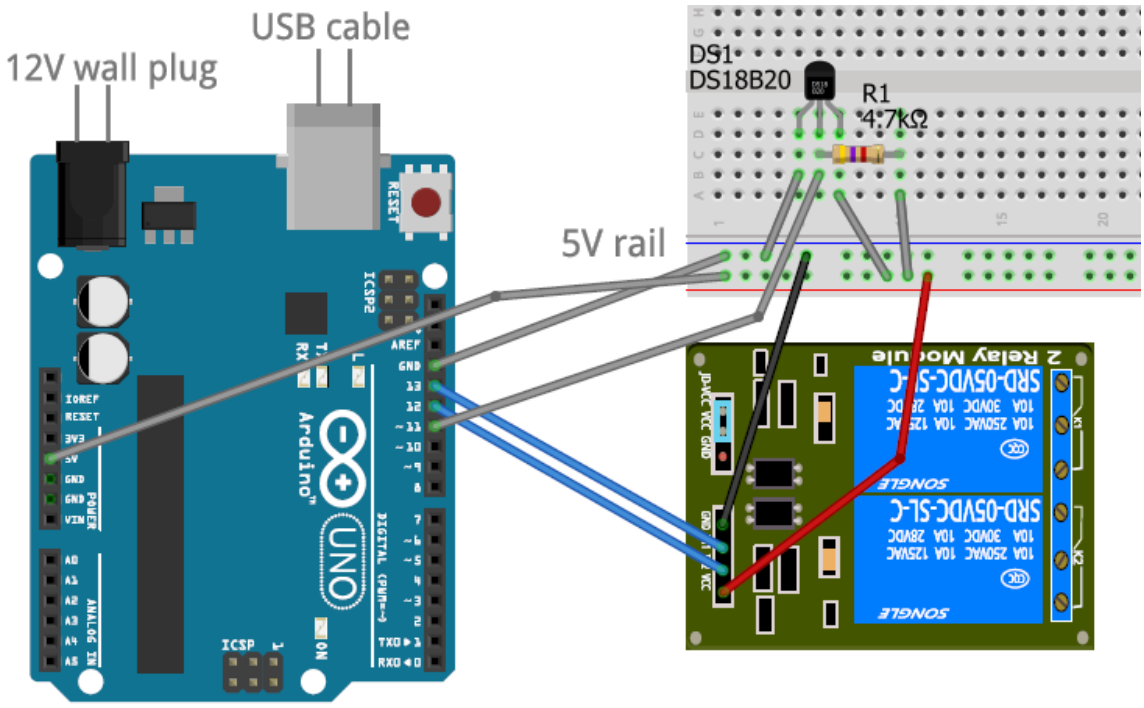
Dans cette partie du laboratoire, vous devrez créer un programme pour contrôler la température d'un morceau de matériau (aluminium) en utilisant un coussin chauffant (chaufferette) et un ventilateur (refroidisseur). Votre code doit chauffer et refroidir le matériau jusqu'à ce que la température atteigne la valeur désirée. **Vous devriez utiliser le code de capteur de température de la partie précédente** pour lire la température afin que vous puissiez voir comment la température de votre matériel change en temps réel. La première valeur que vous devrez atteindre est 35ºC. Vous devez enregistrer le temps nécessaire pour que le bloc d'aluminium atteigne la valeur désirée de la température et représenter un graphique de la température par rapport au temps. La deuxième valeur que vous devrez atteindre est 25ºC (vous êtes obligé de refroidir le bloc de 35ºC à 25ºC). Vous devez également enregistrer le temps qu'il faut pour atteindre cette température et tracer un autre graphique de cette température par rapport au temps.

À la fin du laboratoire, vous devriez pouvoir tracer deux graphiques à l'aide de Microsoft Excel.

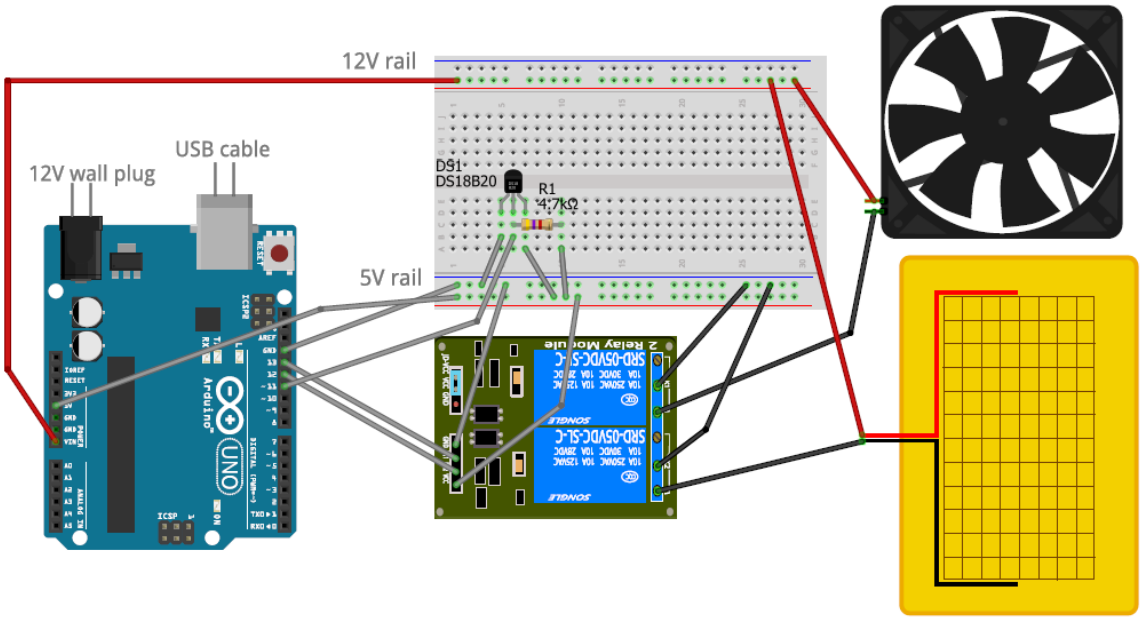
Pour créer le reste du circuit tu vas ajouter un ventilateur, un coussin chauffant et un relais à ton circuit existant de capteur. Le relais agit comme un interrupteur et est contrôlé par l'Arduino. Tu va connecter le ventilateur et le coussin chauffant à des canaux séparés du relais pour les contrôler séparément.

Commence par connecter le relais à l'Arduino. Les câbles gris dans la figure suivante représentent celles qui étaient déjà présentes du circuit précédent du capteur de température, tu peux les ignorer.

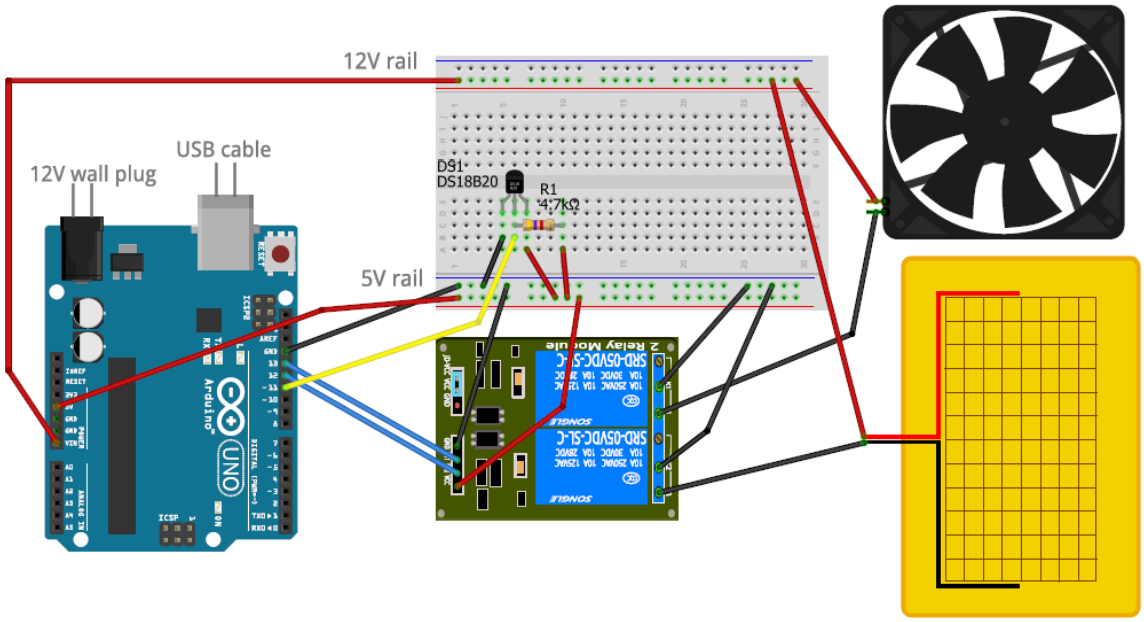
- Relais GND à bande GND
- Relais VCC à bande 5V
- Relais IN1 à Arduino 13
- Relais IN2 à Arduino 12



Ensuite attache le ventilateur et le coussin chauffant à un canal sur le relais, en choisissant les bornes du milieu et de gauche. Ceci est pour que le circuit est normalement ouvert(NO). Le relais se déclenche à bas-niveau, ce qui veut dire qu'un signal bas de l'Arduino ferme le circuit et cause le ventilateur ou coussin de s'allumer. Ces composantes peuvent fonctionner avec 5V mais vont être beaucoup plus efficace à 12V, alors installe une bande de distribution de 12V sur la plaque qui est connectée à Arduino Vin (prend de la puissance de la prise murale 12V).



Voici le circuit sans câbles gris :



1. Pour cette section, ouvre le modèle pour le code du Campus Virtuel et remplace-les *** dans le code avec des valeurs appropriés, basé sur ce que tu as appris dans les sections précédentes.
    a. Initialise toutes les variables que tu vas avoir besoin.
    b. Définie les broche OUTPUT que tu vas utiliser pour la chaufferette et le refroidissant.
    c. Écris une déclaration conditionnelle qui va contrôler la température.

Quelle est la différence entre une déclaration “if” et une déclaration “if … else”?

“if” va réaliser une action si une condition est vraie

“if … else” va permettre de passer a la seconde condition si la precedente est fausse (= sinon)

d. Contrôle le ventilateur et le cousin chauffant avec des commandes digitales de l’Arduino, tu vas remarquer que le relais va être activé (circuit fermé) avec un signal LOW (bas).

e. Enregistre le temps jusqu’à ce que le matériel atteint sa température désirée (écrits le programme pour le faire!).

Explique comment tu vas mesurer le temps avec ton code?

On utilise la fonction "delay" afin de pouvoir mesurer le temps avec le code.

f. Imprime dans le moniteur de série le changement de température et le temps quand la valeur a été enregistré (i.e. tu as besoin d’être capable de mesure le temps, alors assure-toi d’avoir répondu à la question ci-dessus).

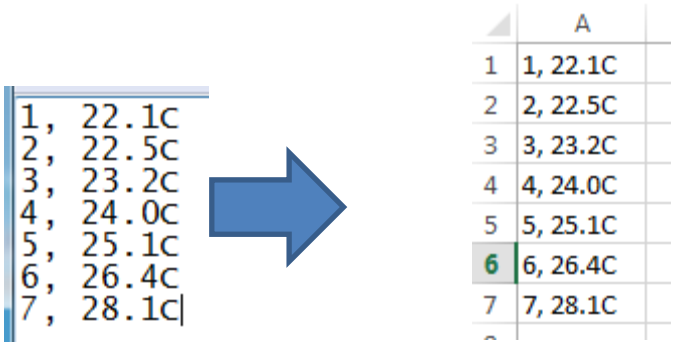
Indice: N’oublie pas que tu peux simplifier le travail en important des données dans Excel en utilisant des caractères spéciaux dans ta sortie formaté!

2. Toutes composantes ont besoin d’être mise à la terre et contrôlés/alimentés par l’Arduino. Si tu n’es pas certain et ne veux pas briser des composantes électroniques, demande à l’assistant d’enseignement d’aider à vérifier ton circuit avant de le brancher!
3. Branche l’Arduino à l’ordinateur en utilisant un câble USB et télécharge le code à l’Arduino. Connecte la prise baril de l’alimentation 12V à l’Arduino et branche-le dans le mur.

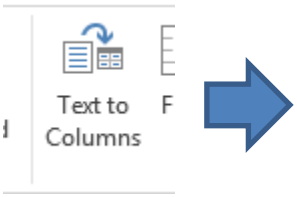
Représentation graphique de données brutes

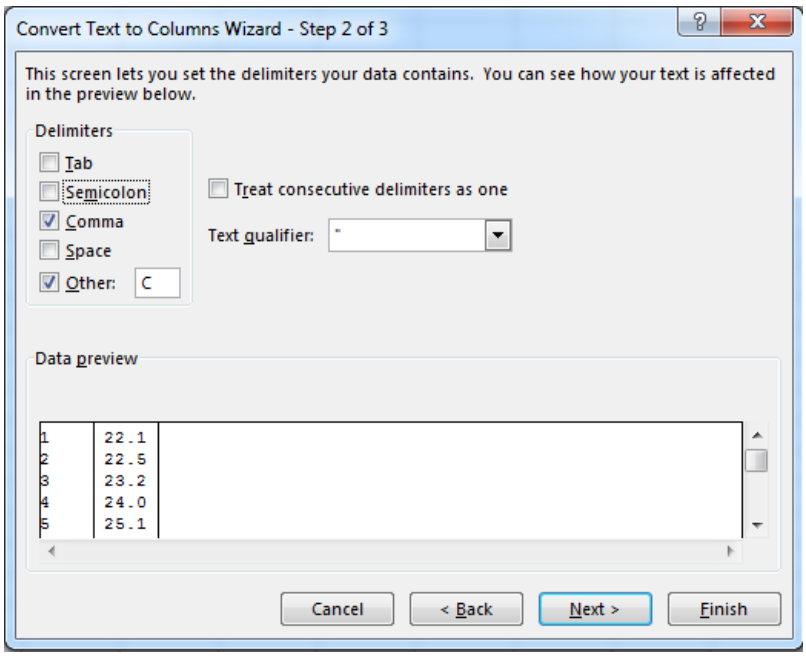
À partir du moniteur série Arduino, vous voulez obtenir des données qui peuvent facilement être placées dans un graphique et manipulées. Puisque le graphe tracera les différentes coordonnées de la température par rapport au temps que vous voulez quelque chose qui ressemble à (temps, température). Si vous obtenez les données suivantes à partir du moniteur série, copiez-les dans une feuille de calcul Excel. Mieux encore, vous pouvez utiliser une instruction println avec un format spécifique qui simplifie le processus d'importation dans Excel. Ensuite, il vous suffit de capturer ou de couper et de coller ***tous*** vos résultats dans un fichier texte qui sera facile à importer (voir le laboratoire précédent sur la façon d'importer des fichiers texte dans Excel).

Conseil: Dans votre instruction println, utilisez les caractères de délimitation appropriés (comme les virgules).

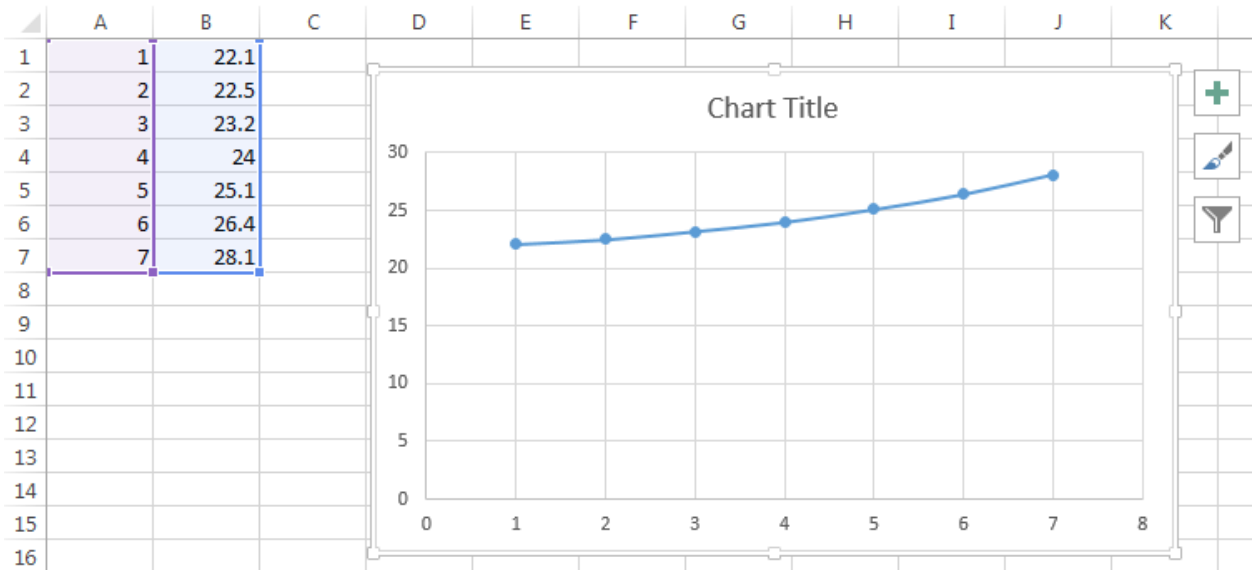


Vous pouvez maintenant diviser les données en différentes colonnes à l'aide de l'outil Texte vers colonnes. Il se trouve dans l'onglet DONNEES. Sélectionnez tout d'abord les cellules que vous souhaitez diviser et cliquez sur le bouton. Ensuite, choisissez Délimité (puisque nous avons des virgules entre nos valeurs) et Suivant>. Dans la fenêtre suivante, sélectionnez 'Comma' et tapez C pour 'Autre'. Il vous montrera un aperçu de vos nouvelles colonnes et vous pouvez maintenant cliquer sur Terminer.





Sélectionnez les nouvelles colonnes à nouveau avec vos données pour faire un graphique. Cliquez sur le type de graphique souhaité dans l'onglet INSERT. Cela représentera tous vos points. Vous pouvez maintenant changer les titres de diagramme / axe et d'autres options avec le '+' à côté du graphique quand vous cliquez dessus.


A
B
C
D
E
F
G
H
I
J
K
1 1 22.1
2 2 22.5
3 3 23.2
4 4 24
5 5 25.1
6 6 26.4
7 7 28.1
8
9
10
11
12
13
14
15
16
Chart Title
30
25
20
15
10
5
0
0
1
2
3
4
5
6
7
8