V. Mini bargraphe

Réalisation du plan de câblage et du programme d'un mini bargraphe*, constitué de 3 LEDs (qui indiqueront une quantité) et de 2 boutons poussoirs (un noir pour incrémenter et un rouge pour décrémenter).

** Un bargraphe (Bargraph en anglais) est un indicateur visuel de niveau d'un signal quelconque. Par exemple sur l'image de droite, un thermomètre fonctionnant avec des LED, indiquant la température dans une pièce. Ici chaque LED représente 2°C.*

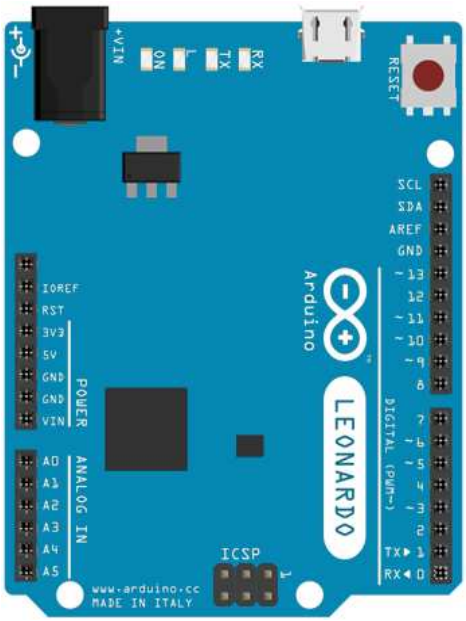
Notre bargraphe partira avec une valeur de 0 : donc aucune LED allumée. Chaque appui sur le bouton poussoir noir allumera une LED supplémentaire. Chaque appui sur le bouton poussoir rouge éteindra une LED. L'ordre d'allumage des LED est :

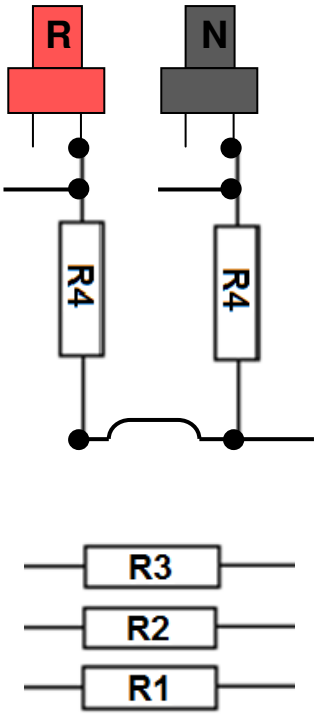
1 : Rouge
2 : Jaune
3 : Vert

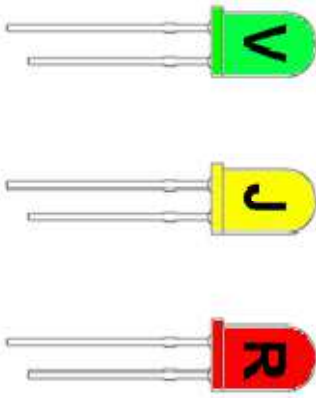
1) Complète le schéma de câblage (utilise des couleurs différentes pour une lecture plus rapide, et trace les traits pour qu'ils ne se coupent pas) avec :

Élément	Broche Arduino
LEDrouge	2
LEDjaune	3
LEDverte	4
BPN	5
BPR	6

R1 = ...Ω
R2 = ...Ω
R3 = ...Ω
R4 = 10kΩ







2) Réalisation du premier programme :

a) Ouvre un nouveau programme et enregistre le dans tes documents sous le nom « **05_Mini_bargraphe** »

Penser à sauvegarder régulièrement et à commenter les lignes de programme

b) Avant le setup () :

Assigner chaque élément à sa broche (s'aider du tableau de la question 1).

Créer les variables :

- « **etatBPN** » de type « **int** » avec comme valeur de départ « **LOW** », qui permettra d'enregistrer l'état du bouton poussoir noir

- « **ancien_etatBPN** » de type « **int** » avec comme valeur de départ « **LOW** », qui permettra d'enregistrer l'ancien état du bouton poussoir noir

- « **etatBPR** » de type « **int** » avec comme valeur de départ « **LOW** », qui permettra d'enregistrer l'état du bouton poussoir rouge

- « **ancien_etatBPR** » de type « **int** » avec comme valeur de départ « **LOW** », qui permettra d'enregistrer l'ancien état du bouton poussoir rouge

- « **nombreLED** » de type « **int** » avec comme valeur de départ « **0** », qui permettra d'enregistrer le nombre de LED allumées

c) Le setup () :

Indiquer pour chaque broche si elle est utilisée en entrée (capteur) ou en sortie (actionneur).

d) Le loop ()

Réaliser un programme qui va :

- Récupérer l'état de « **BPN** » et l'enregistrer dans la variable « **etatBPN** »

- Attendre 0,1 seconde (**100**ms)

- Si la variable « **etatBPN** » est différente de la variable « **ancien_etatBPN** » ET que la variable « **etatBPN** » est à l'état haut : incrémenter la variable « **nombreLED** »

- Mettre la variable « **ancien_etatBPN** » égale à la variable « **etatBPN** »

- Récupérer l'état de « **BPR** » et l'enregistrer dans la variable « **etatBPR** »

- Attendre 0,1 seconde (**100**ms)

- Si la variable « **etatBPR** » est différente de la variable « **ancien_etatBPR** » ET que la variable « **etatBPR** » est à l'état haut : décrémenter la variable « **nombreLED** »

- Mettre la variable « **ancien_etatBPR** » égale à la variable « **etatBPR** »

- Si « **nombreLED** » **< 0** : Mettre « **nombreLED** » **= 0** *(Pour que le nombre de LEDs ne descende pas en dessous de 0)*

- Si « **nombreLED** » **> 3** : Mettre « **nombreLED** » **= 3** *(Pour que le nombre de LEDs ne dépasse pas 3)*

- Appeler la fonction (=sous-programme) « **LED** » avec comme paramètre « **nombreLED** » (nous allons créer cette fonction juste après)

e) Après le loop ()

Réaliser une fonction (sous-programme) sans valeur de retour, nommée « **LED** », avec un paramètre de type « **int** » nommé « **valeur_recue** » qui va:

- Eteindre toutes les LEDs

- Si la « **valeur_recue** » est supérieure ou égale à 1 : allumer la « **LEDrouge** »

- Si la « **valeur_recue** » est supérieure ou égale à 2 : allumer la « **LEDjaune** »

- Si la « **valeur_recue** » est égale à 3 : allumer la « **LEDverte** »

f)  Vérifie le programme. Appelle le professeur quand le programme est valide.