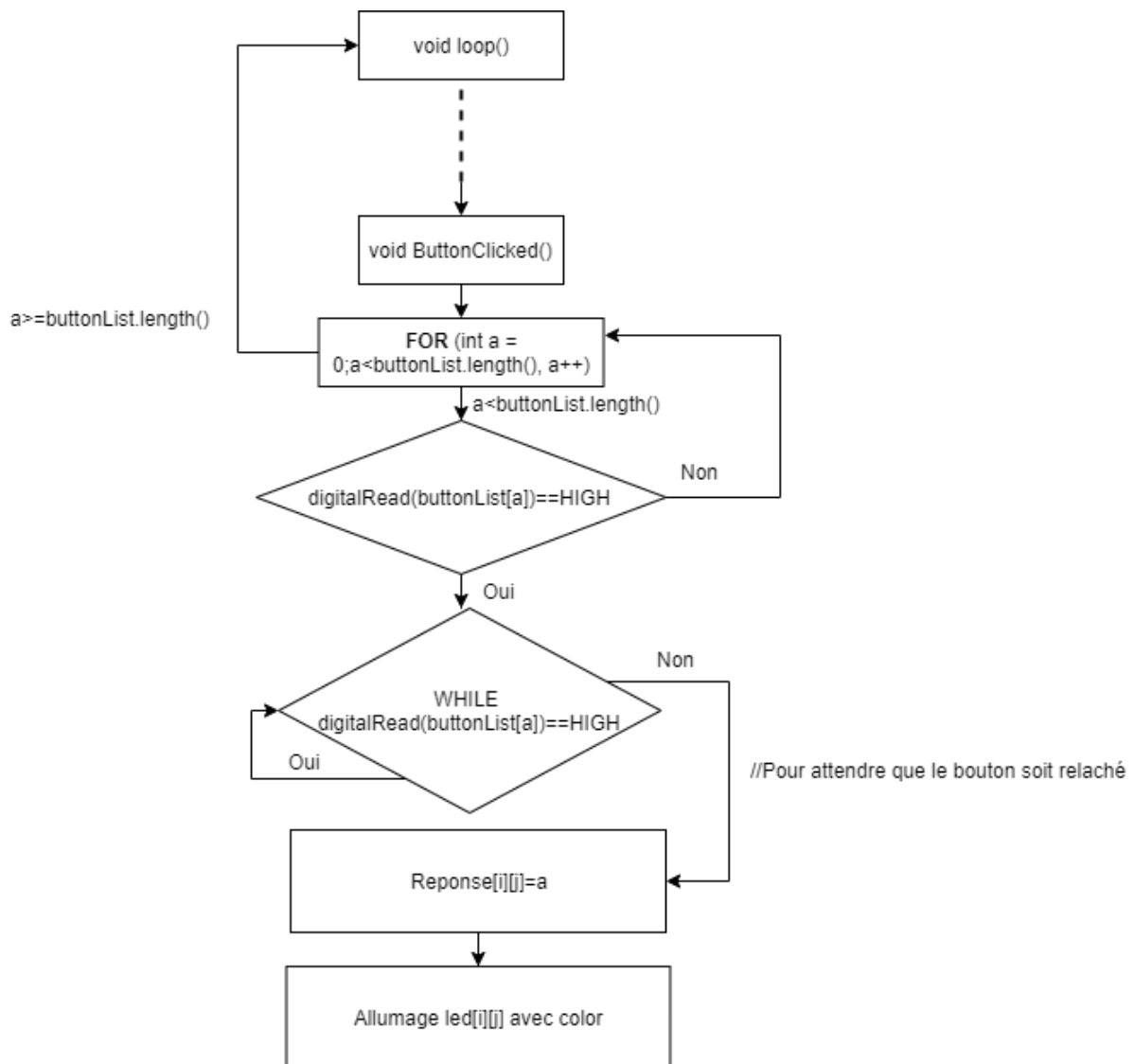


Objectif de la séance :

- Comprendre le fonctionnement des sous-programmes sur arduino
- Ecrire la fonction qui permet d'allumer une LED en fonction du bouton cliqué puis le tester en commun avec manon qui de son côté s'occupe de l'allumage des LEDS.

Premièrement, durant la semaine j'ai réalisé l'algorithme que je vais utiliser pour le code du bouton. Je le mets ci-dessous. C'est mon point de départ de cette séance.




Ensuite, j'ai regardé cette vidéo : <https://www.youtube.com/watch?v=2pxYEwaMtaI> pour comprendre le fonctionnement des fichiers ".h", ".c" avec les ".ino" c'est grâce à eux que la lecture de nos programmes sera optimale.

Je créerai mon premier fichier "functions.h" et "functions.c" avec notepad++ plus tard.

Pour la phase de test, j'utilise tout d'abord une loop classique, qui teste la conceptualisation que j'avais eue. En effet, le but est d'envoyer la réponse uniquement lorsqu'un bouton est relâché.

Voici le premier résultat, qui fonctionne avec une led verte !



```
main
const int b_vert=2;
int nbButton=1;
int buttonList[1]=(b_vert);
int n;
int i;
int response[12][4];

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  n=0;
  i=0;
  pinMode(b_vert,INPUT);
  pinMode(3,OUTPUT);
  digitalWrite(3,HIGH);
}

void loop() {
  // put your main code here, to run repeatedly:
  for (int a=0;a<nbButton;a++){
    if (digitalRead(buttonList[a])==LOW){ //Si on appuye sur le bouton
      digitalWrite(3,LOW); //allume la LED
      while (digitalRead(buttonList[a])==LOW){ //Tant que le bouton est pressé
        response[n][i]=a; // on ajoute "a" à la réponse de la ligne n et colonne i
        Serial.println(response[n][i]); //je l'affiche pour voir quand est-ce que l'information est envoyé.
      }
    }
  }
}
```

Enregistrement terminé

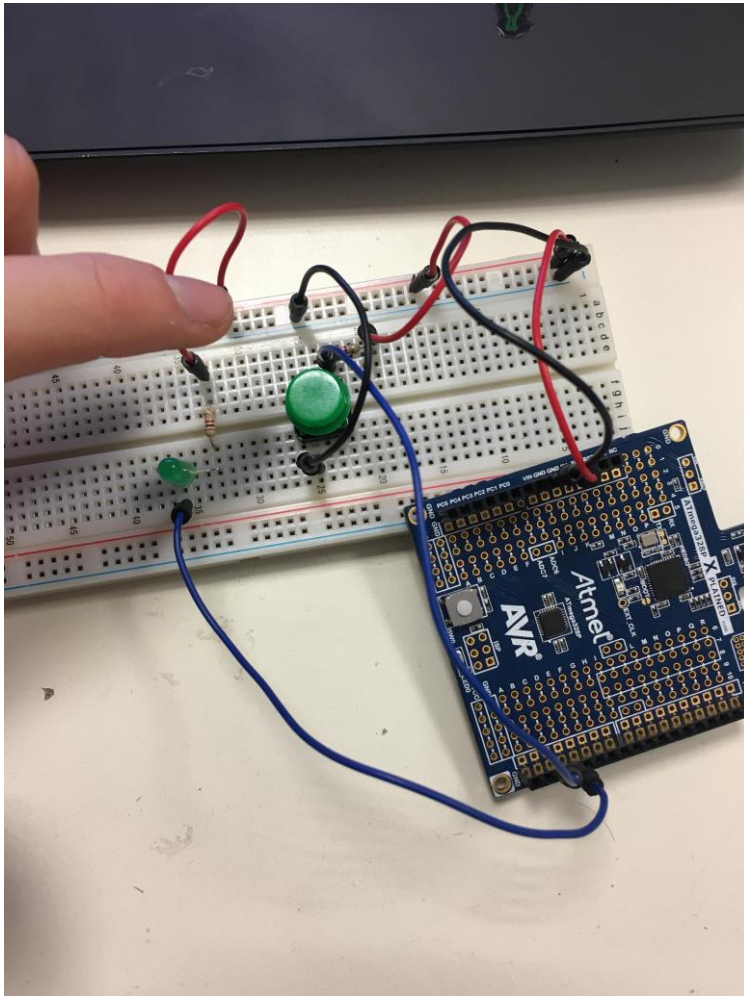
Le croquis utilise 2042 octets (6%) de l'espace de stockage de programmes. Le maximum est de 30720 octets.
Les variables globales utilisent 192 octets (9%) de mémoire dynamique, ce qui laisse 1856 octets pour les variables locales. Le maximum est de 2048 octets.

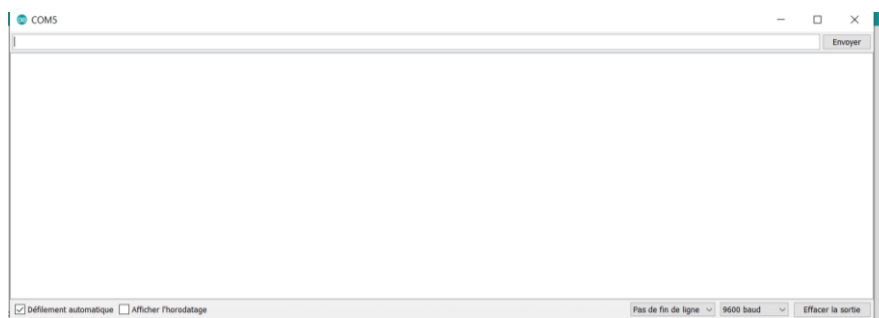
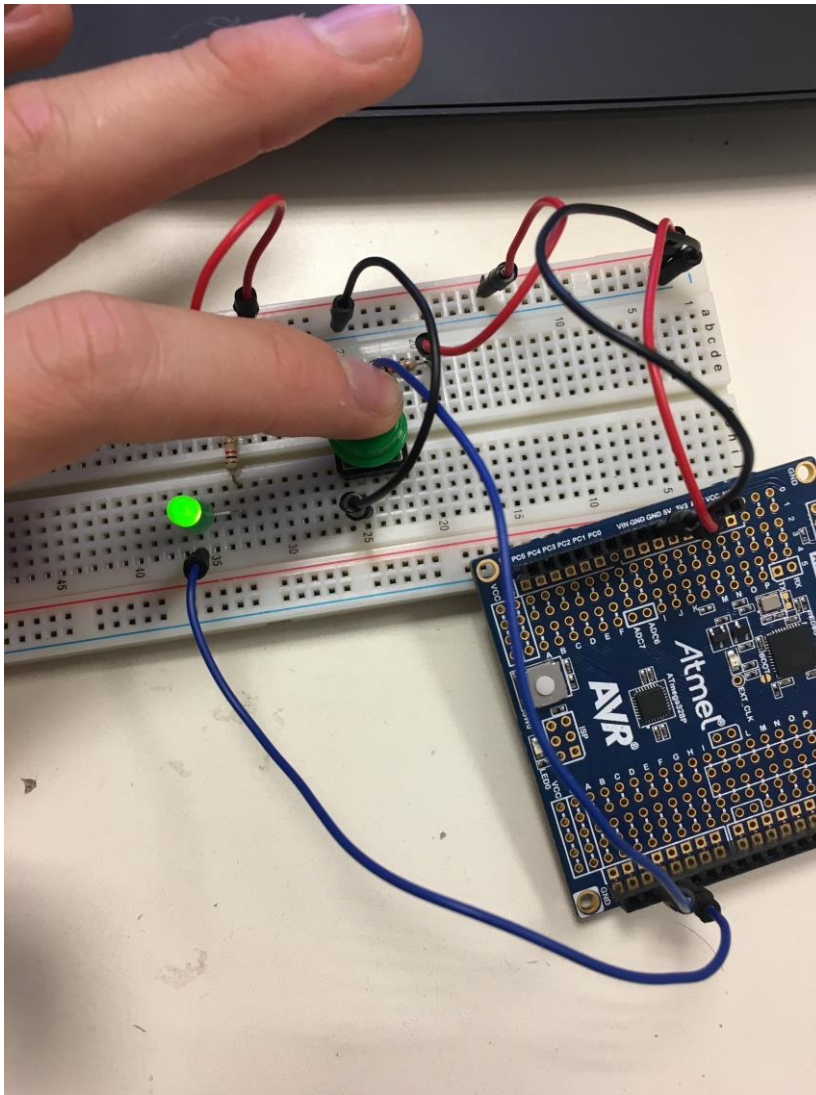
28 Arduino Nano sur COM5

(J'ai déjà défini des variables utiles dans le futur du projet mais inutiles en dimension 1)

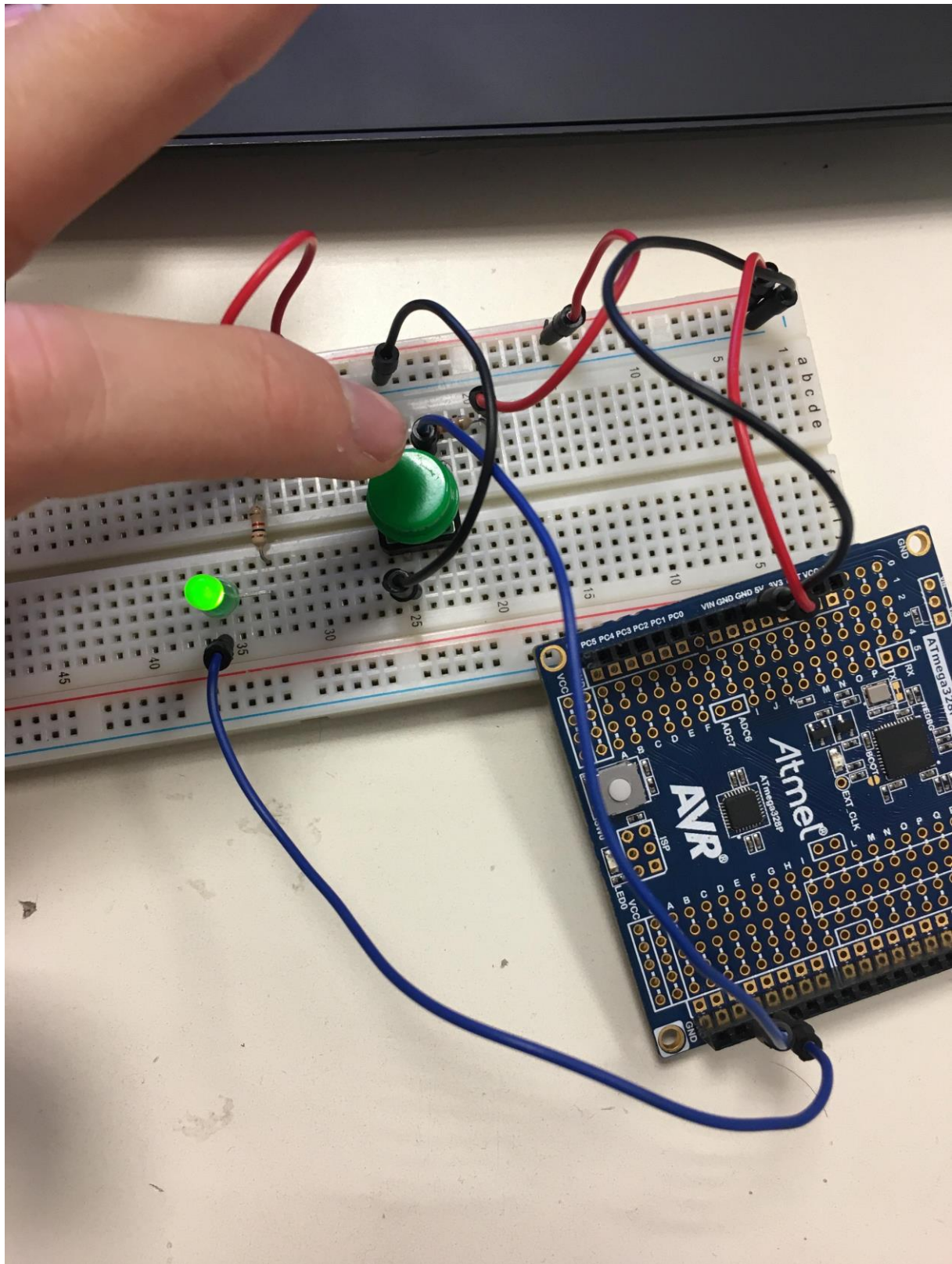
Voici le rendu :

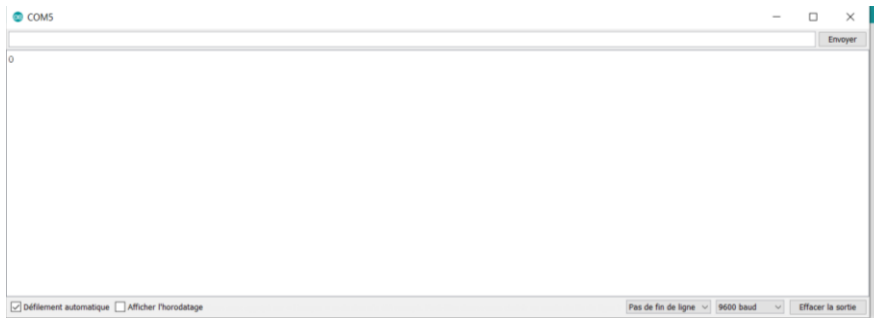
Avant/pendant la pression :





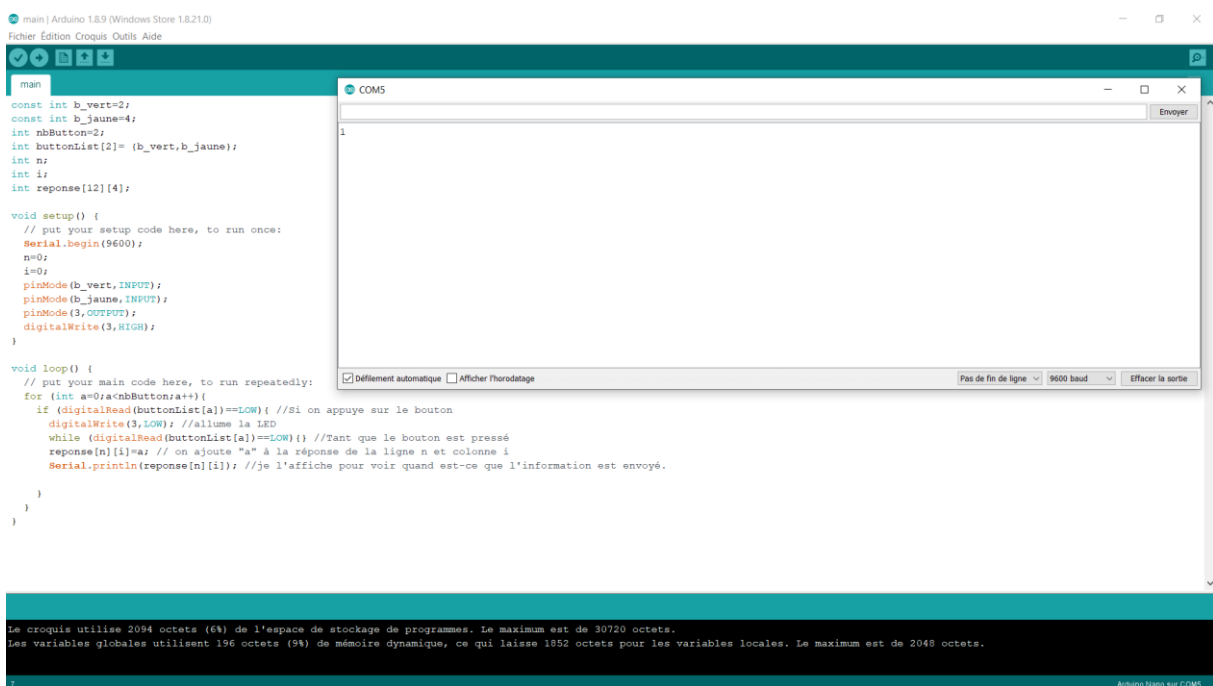
Après la pression :





Avec les LEDS :

En rajoutant un bouton jaune, on obtient 1 quand on clique dessus.



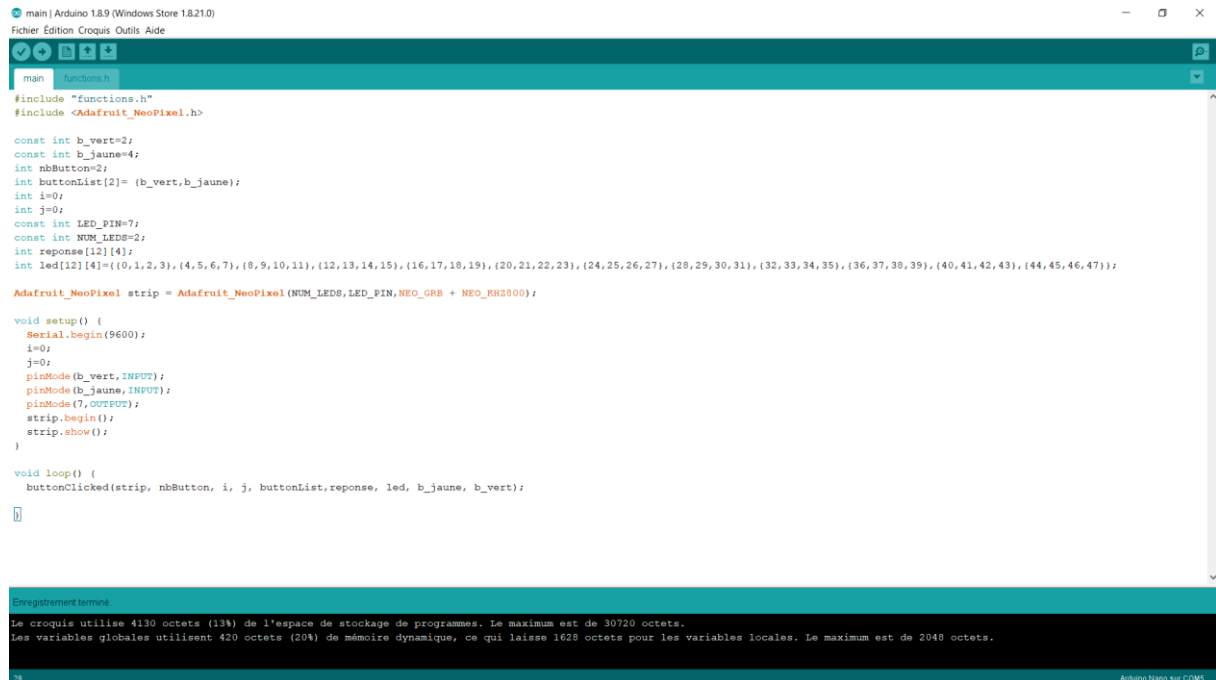
Ensuite en raccord avec Manon, on décide d'une liste à 2 dimensions où seront stockées les LEDS. Lorsqu'on allume la première led, on allume la led[0][0] etc.

<http://www.locoduino.org/spip.php?article227> Ce site m'a permis de savoir comment fonctionne précisément les listes.

J'ai finalement décidé de prendre un fichier ".h" situé dans le même dossier que notre main, et j'ai écrit dedans deux fonctions, qui permettent l'une de détecter la pression d'un bouton, qui

appelle à son tour une fonction permettant d'allumer les leds de la couleur souhaité. J'ai finis ceci à la maison, il faudra que je réessaye dans la semaine pour voir le rendu.

main



```

main | Arduino 1.8.9 (Windows Store 1.8.21.0)
Fichier Edition Croquis Outils Aide

main functions.h
#include "functions.h"
#include <Adafruit_NeoPixel.h>

const int b_vert=2;
const int b_jaune=4;
int nbButton=2;
int buttonList[2]= {b_vert,b_jaune};
int i=0;
int j=0;
const int LED_PIN=7;
const int NUM_LEDS=2;
int reponse[12][4];
int led[12][4]={{0,1,2,3},{4,5,6,7},{8,9,10,11},{12,13,14,15},{16,17,18,19},{20,21,22,23},{24,25,26,27},{28,29,30,31},{32,33,34,35},{36,37,38,39},{40,41,42,43},{44,45,46,47}};

Adafruit_NeoPixel strip = Adafruit_NeoPixel(NUM_LEDS,LED_PIN,NEO_GRB + NEO_KHZ800);

void setup() {
  Serial.begin(9600);
  i=0;
  j=0;
  pinMode(b_vert,INPUT);
  pinMode(b_jaune,INPUT);
  pinMode(7,OUTPUT);
  strip.begin();
  strip.show();
}

void loop() {
  buttonClicked(strip, nbButton, i, j, buttonList,reponse, led, b_jaune, b_vert);
}

```

Enregistrement terminé

Le croquis utilise 4130 octets (13%) de l'espace de stockage de programmes. Le maximum est de 30720 octets.
Les variables globales utilisent 420 octets (20%) de mémoire dynamique, ce qui laisse 1628 octets pour les variables locales. Le maximum est de 2048 octets.

28 Arduino Nano sur COM5

functions.h



```

main - functions.h | Arduino 1.8.9 (Windows Store 1.8.21.0)
Fichier Edition Croquis Outils Aide

main functions.h
#include <Adafruit_NeoPixel.h>

void allumerLed(Adafruit_NeoPixel strip, int a, int i, int j, int led[12][4],int buttonList[2], int b_jaune, int b_vert){
  if (buttonList[a]==b_vert){ //si on appuie sur le bouton vert
    strip.setPixelColor(led[i][j],0,255,0); //la led jème led de la ième ligne s'allume en vert
    strip.show();
  }
  else if (buttonList[a]==b_jaune){ //idem
    strip.setPixelColor(led[i][j],255,125,0);
    strip.show();
  }
}

void buttonClicked(Adafruit_NeoPixel strip, int nbButton, int i, int j, int buttonList[2], int reponse[12][4], int led[12][4],int b_jaune, int b_vert){
  for (int a=0;a<nbButton;a++){ //on parcourt la liste des boutons
    if (digitalRead(buttonList[a])==HIGH){ //dès qu'une valeur est à high
      Serial.println(buttonList[a]); //on l'affiche pour vérifier que tout marche
      allumerLed(strip, a, i, j, led, buttonList, b_jaune, b_vert); //on allume la led
      while (digitalRead(buttonList[a])==HIGH){ //tant qu'on appuie sur le bouton rien ne se passe et le programme est en "pause"
        reponse[i][j]=a; //dès qu'on relâche le bouton, le programme reprend et on ajoute la réponse à la liste réponse.
        Serial.println(reponse[i][j]);
      }
    }
  }
}

```

Enregistrement terminé

Le croquis utilise 4130 octets (13%) de l'espace de stockage de programmes. Le maximum est de 30720 octets.
Les variables globales utilisent 420 octets (20%) de mémoire dynamique, ce qui laisse 1628 octets pour les variables locales. Le maximum est de 2048 octets.

28 Arduino Nano sur COM5

Il faudra que j'essaye pendant les vacances avec toutes les LEDS de Manon.

C'est un bon début de programme qui viendra s'ajouter au code final ! Celui-ci marche pour les LEDS RGB.