Hendrikse Jérémy

Compte rendu 2

Séance laborieuse tournée autour des matrices :

Toujours dans l'objectif de faire des matrices, j'ai voulu utilisé std::vector qui permet de faire des tableau à dimensions variables mais cela n'existe pas pour Arduino.

J'ai ensuite voulu utiliser les arrays qui sont de dimension fixe, mais en voulant avoir une représentation visuelle dans le moniteur série à l'aide d'une fonction display je me suis rendu compte qu'il faut absolument que la dimension du tableau soit une valeur fixe connue. La solution de variables globale fonctionne mais j'ai besoin de fixer des valeurs de maxX et maxY plus tard dans le programme. Les pointeurs non plus ne fonctionne pas.

```
String display(int tab[][], int maxX, int maxY){
  return "le tab sous forme de String"
}
```

declaration of 'tab' as multidimensional array must have bounds for all dimensions except the first

```
String display(int tab[][maxY], int maxX, int maxY){
  return "le tab sous forme de String";
}
```

'maxY' was not declared in this scope

J'ai donc voulu faire du C++ sur VScode un IDE séparé connaissant la librairie std puis de transférer le code vers Arduino, mais je n'ai pas réussi à faire fonctionner C++ sur mon ordinateur rapidement j'ai donc abandonné cette solution. C'était un choix car on s'est rendu compte plus tard dans la séance que cette solution ne fonctionnerait pas. (on = Logan Robin Romain et moi car je suis allé demander plusieurs fois durant la séance leur avis pour trouver des solutions et des astuces (car ils connaissent bien C++)).

Finalement j'ai opté pour faire des classes sur Arduino car on n'a pas besoin de passer en paramètre le tableau multidimensionnel dans les méthodes de la class. J'ai donc appris à faire des classes en C++ et commencer à écrire ma classe Matrice qui a comme instances maxX : nombre de colonnes de la matrice, maxY : nombre de ligne de la matrice, mat : tableau d'entiers à deux dimensions. Après avoir finalement compris et réussi ma class j'ai fait une première compilation qui a mené à une nouvelle erreur de dimension. Au moment de la compilation de la class, la dimension du tableau doit être connu et une valeur fixée et ne peut pas être une variable même de type « const ».

```
#ifndef Matrice_h
#define Matrice_h
#include "Arduino.h"
class Matrice {
  //parametre
  private:
    static const int maxX;
    static const int maxY;
    int mat[][];
```

declaration of 'mat' as multidimensional array must have bounds for all dimensions except the first

```
#ifndef Matrice_h
#define Matrice_h
#include "Arduino.h"
class Matrice {
  //parametre
  private:
    static const int maxX;
    static const int maxY;
    int mat[][maxY];
```

array bound is not an integer constant before ']' token

Après avoir cherché de multiples façons d'écrire la class on est abouti à la conclusion qu'il fallait obligatoirement fixer la taille de la matrice et qu'elle serait arbitrairement choisi très grande avant la compilation. Et que si dans l'exécution du programme la matrice était trop petite, il faudrait combiner plusieurs matrices de même taille et avoir une matrice de matrice. Ou avoir une class

strictement identique à la première à une seule exception, la taille de la matrice qui serait aussi arbitrairement choisi plus grand à la première avant la compilation.

Une solution alternative existe, ce sont les template (idée de Romain) mais qui sont d'une complexité extrême et donc clairement pas accessible pour un projet de 27h et étudiant débutant en C++.

```
template <int N, int M>
class Matrix {
  int mat[N][M];
};

void setup() {
  Serial.begin(9600);

  Matrix<5, 6> m;
}
```

NB : cette matrice va servir à mettre toutes les données récoltées par le LIDAR et créer une carte 2D de la pièce, elle est donc essentielle à ce projet.

NB 2 : Arduino n'est vraiment pas adapté pour les matrices et peu de documentation existe sur ce sujet.