Code réalisé

(C Arduino et C)

*Groupe : Simon Arricaut, Charles Carré, Benoît Perriguet et Maxime Garric*

*Vendredi 16 novembre 2018*

Table des matières

[I- Module 1 : Cardio 3](#_Toc530060094)

[a) Contexte : 3](#_Toc530060095)

[b) Réalisation du code : 3](#_Toc530060096)

[II- Module 2 : cœur de LEDs : 4](#_Toc530060097)

[a) Contexte : 4](#_Toc530060098)

[b) Réalisation du code : 5](#_Toc530060099)

[III- Module 3 : Processing et acquisition de données : 11](#_Toc530060100)

[a) Contexte : 11](#_Toc530060101)

[b) Réalisation du code : 11](#_Toc530060102)

[IV- Module 4 : Module lecture et traitement de données : 12](#_Toc530060103)

# Module 1 : Cardio

## Contexte :

Ce premier module est donc de la **programmation Arduino**.

Nous devons réaliser un code nous permettant de détecter, mesurer et calculer le pouls à partir des informations envoyées par le récepteur IR.

Nous devons organiser notre code en deux fichiers :

* **main.c** (qui contient le loop et le setup)
* **cardio.c** (et .h) qui contient les fonctions propres à la récolte des informations et aux calculs pour avoir le pouls.

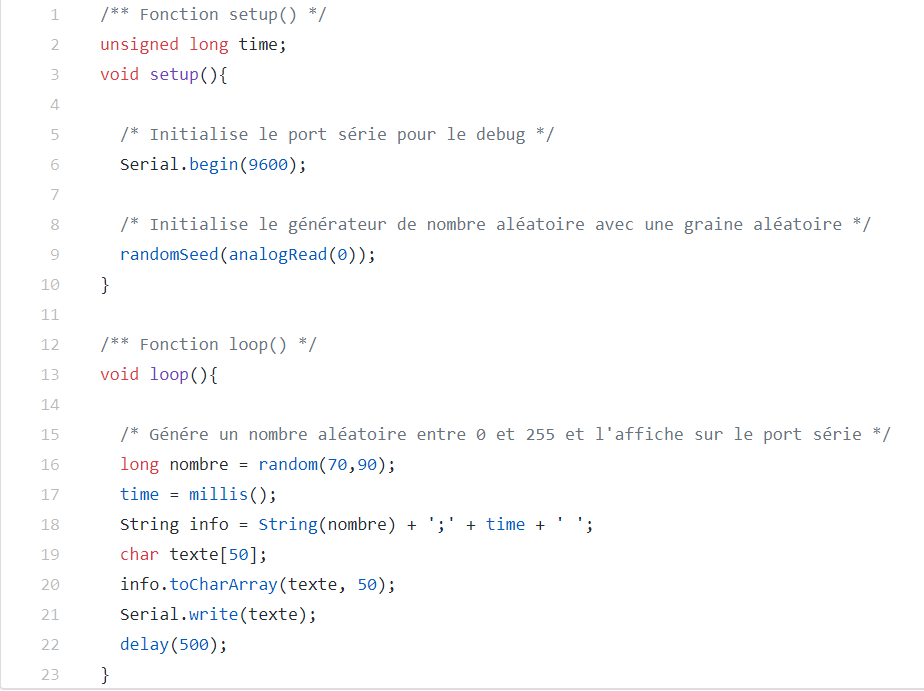
Ce module doit envoyer sur le port série deux valeurs (au format csv avec un « ; ») :

* Le **nombre de millisecondes** depuis le démarrage du programme Arduino
* Le **pouls** calculé à partir des informations récoltées par le récepteur IR.

Attention, en attendant le montage et la mise au point du module Cardio, nous devons envoyer via le port série cette information à intervalles réguliers (avec le même format décrit ci-dessus) générée de façon aléatoire (avec des valeurs cohérentes pour un pouls).

## Réalisation du code :

**Voici le code pour la génération aléatoire :**



# Module 2 : cœur de LEDs :

## Contexte :

La première partie de ce deuxième module consiste à réaliser un module en C Arduino pour **gérer le cœur de LEDs**.

Le code sera donc composé de :

* main.c
* cœur.c(et cœur.h)
* param.h

Le fichier param.h nous permettra de configurer la façon dont seront allumées les LEDs.

--------------------------------------------------------------------------------------------------------------------------------------

La deuxième partie est un programme en langage C. Nous devons **créer le fichier param.h de la deuxième partie à travers un générateur de code en C.**

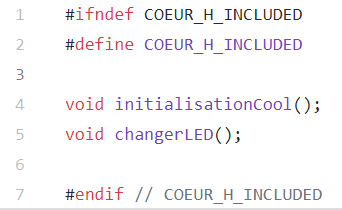
Le code sera composé de :

* **main.c** : contient la fonction main et lance le menu en console.
* **menu.c (et .h)** : contient toutes les fonctions nécessaires pour créer les menus de paramétrage du coeur de LEDs
* **generationCode.c (et .h)**: contient toutes les fonctions nécessaires pour créer le fichier param.h contenant le paramétrage pour le coeur de LED. Ce fichier est indispensable pour la compilation de ce module. Aucune contrainte sur la façon de générer le fichier param.h ni sur le contenu de ce fichier (sachant néanmoins, qu’il doit contenir le paramétrage du coeur de LEDs)

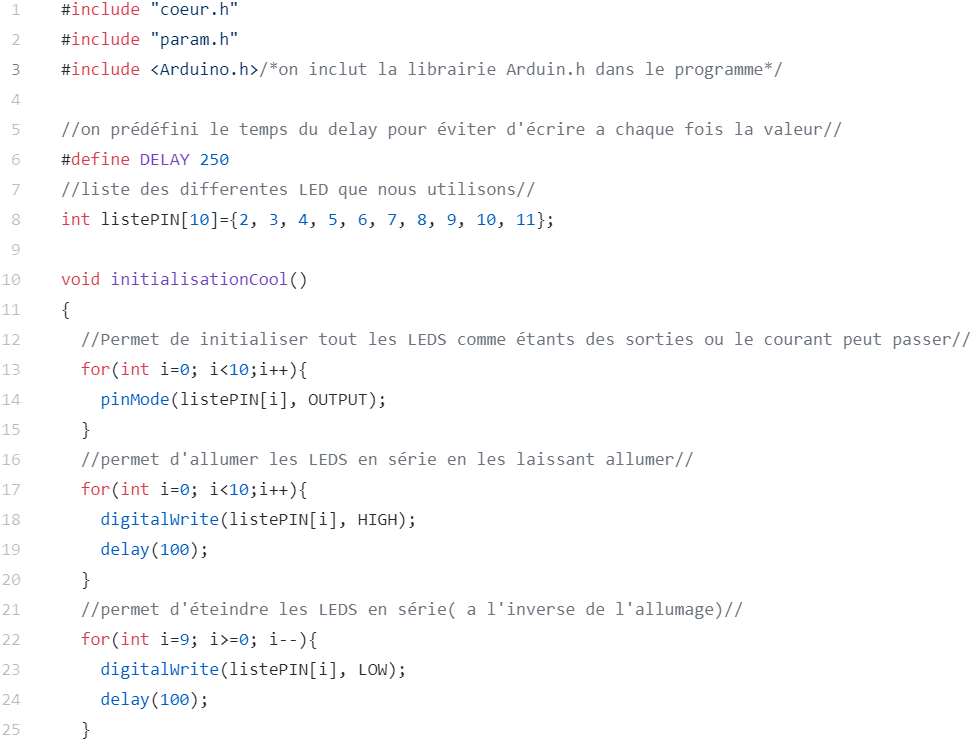
## Réalisation du code :

**Première partie du code :**

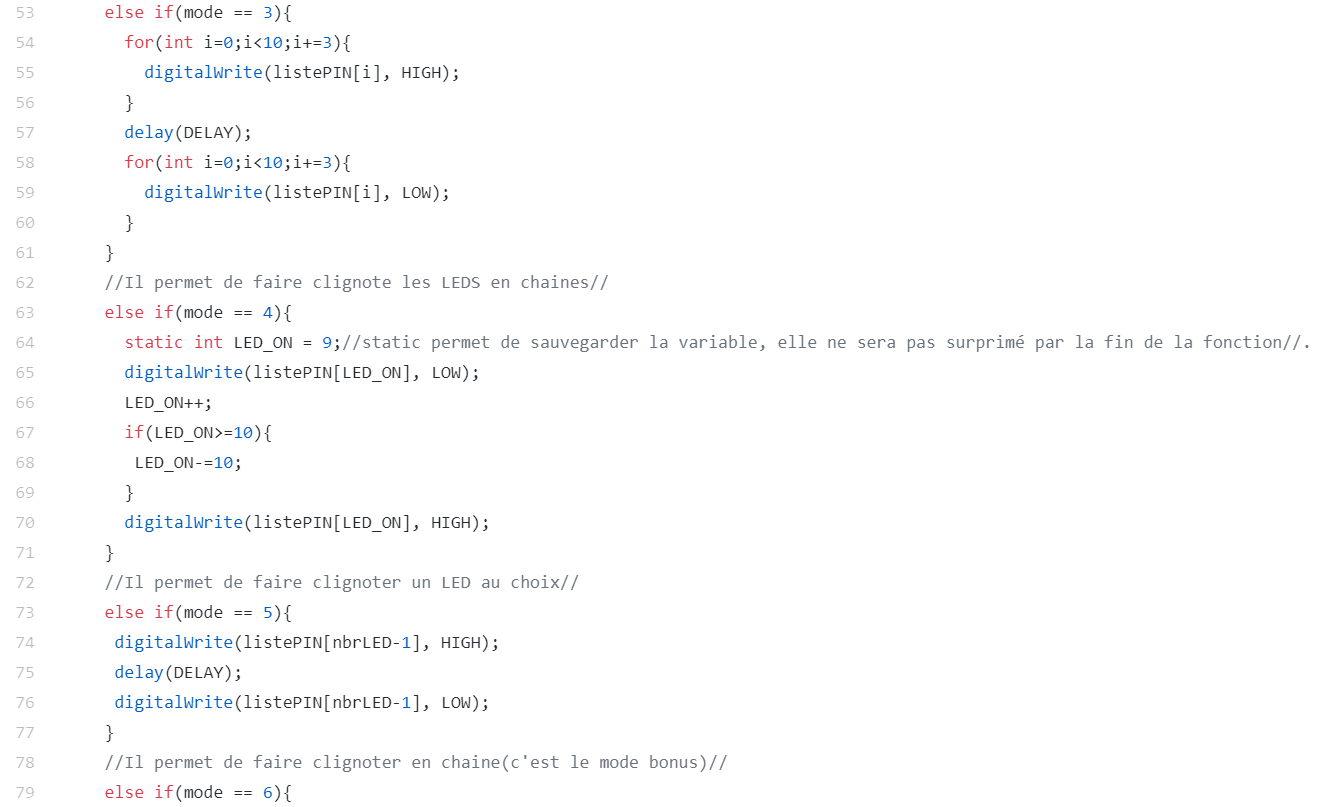
**Voici le fichier cœur.h :**

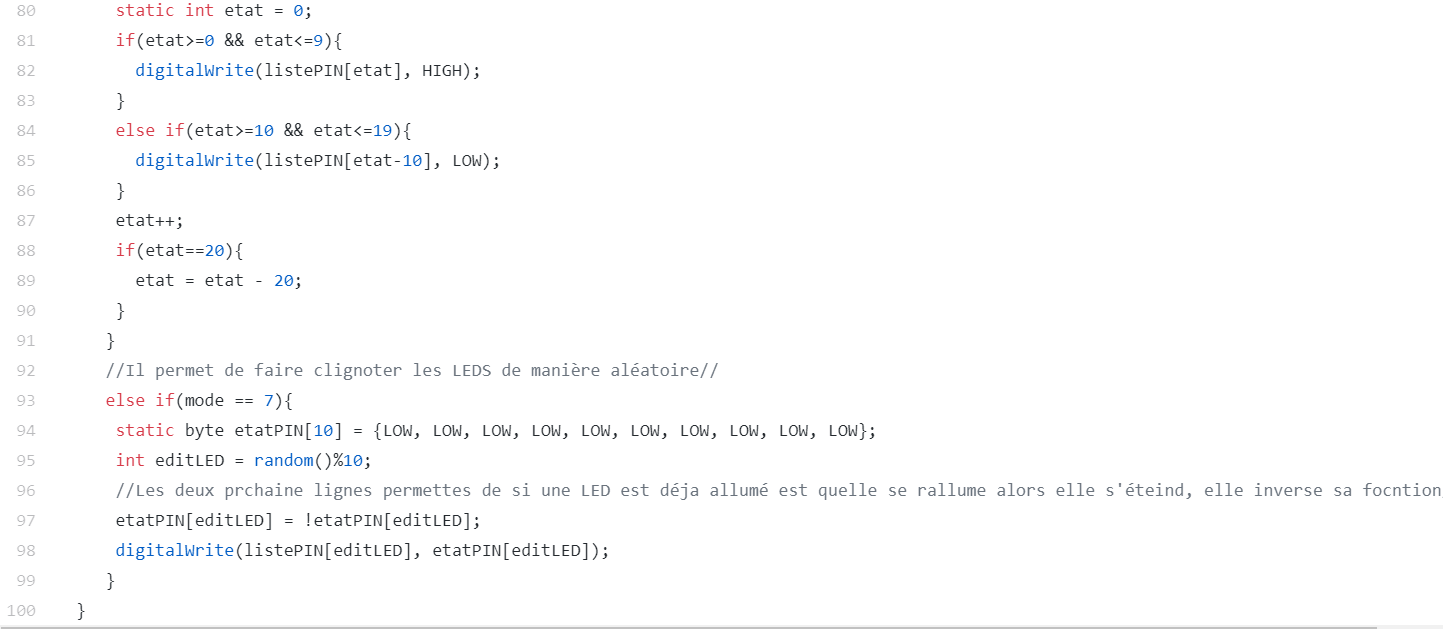


**Voici le fichier cœur.c :**

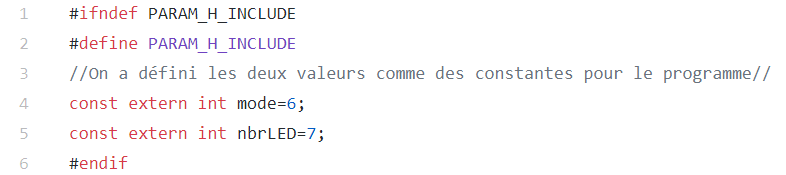




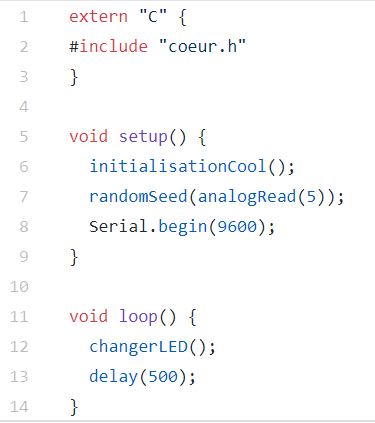




**Voici le fichier param.h :**

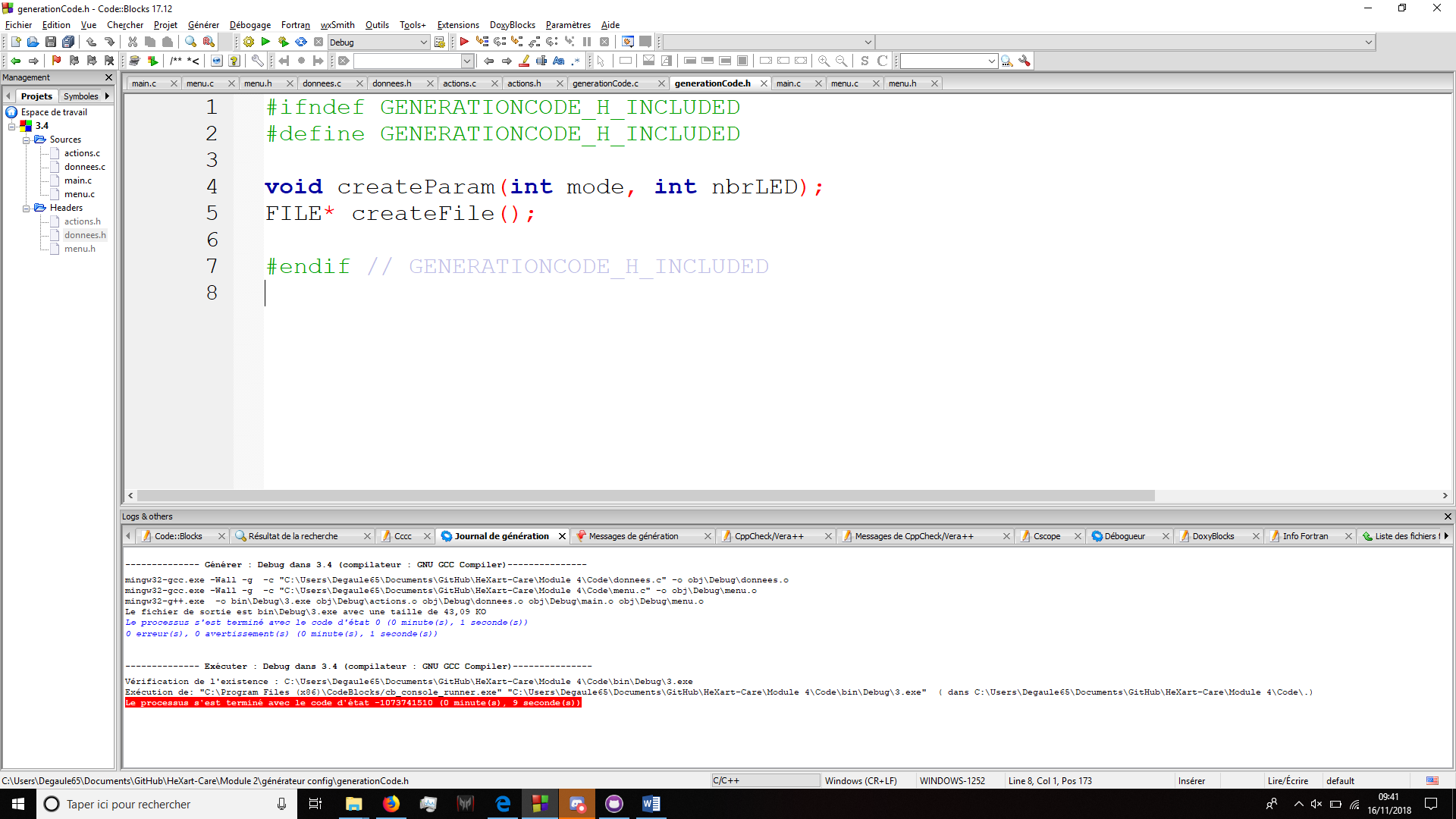


**Voici le fichier Gestion\_LED.ino : ajustement du transfert de données**

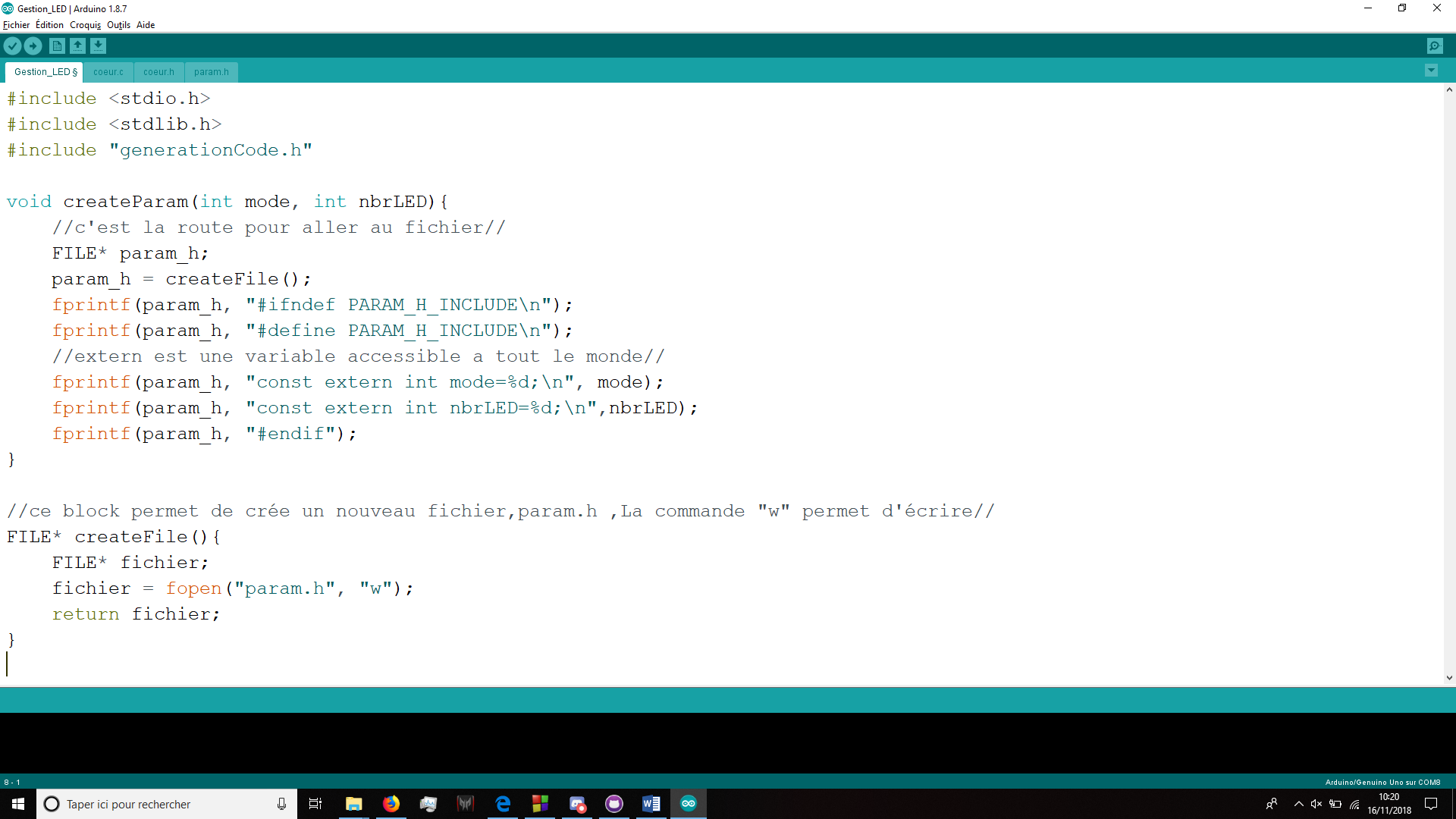


**Deuxième partie du code :**

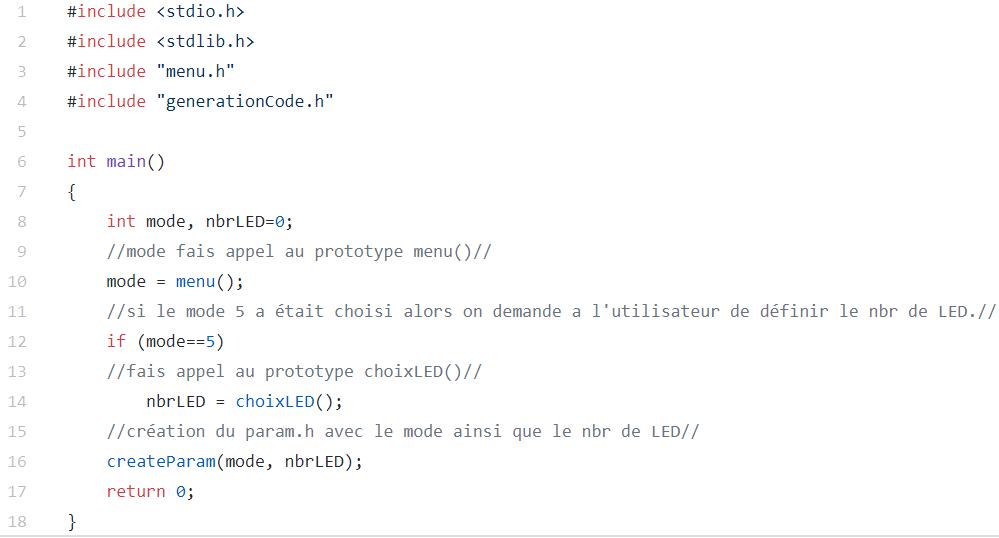
**Voici le fichier generationCode.h :**



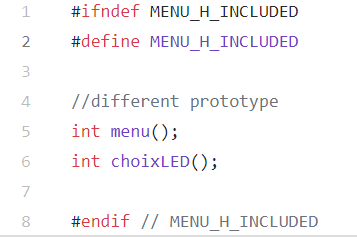
**Voici le fichier generationCode.c :**



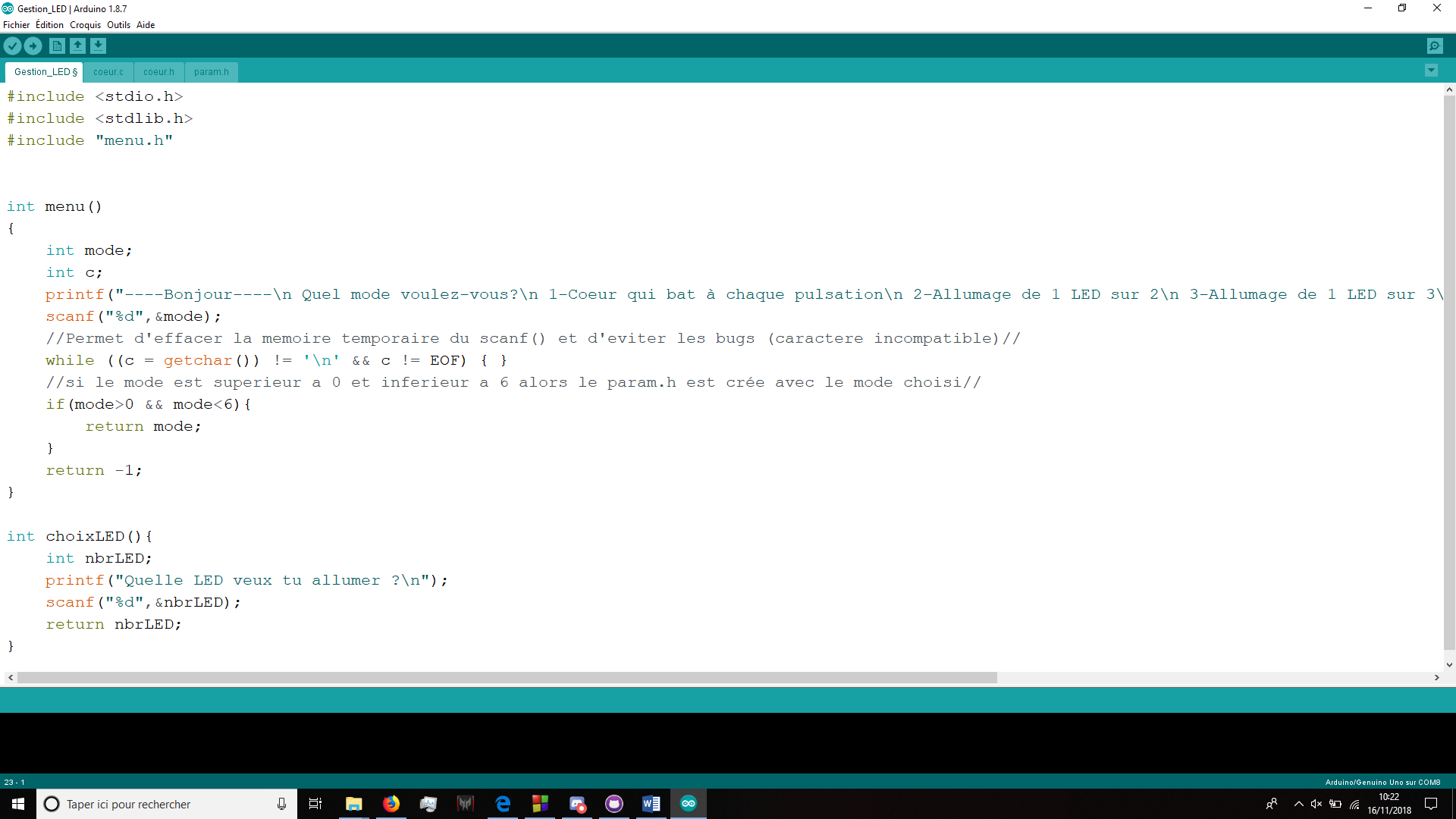
**Voici le fichier main.c :**



**Voici le fichier menu.h :**



**Voici le fichier menu.c :**



# Module 3 : Processing et acquisition de données :

## Contexte :

Nous voulons **récupérer toutes les valeurs des pouls lues sur la sortie série de l’Arduino et les enregistrer dans un fichier csv**.

La visualisation et la récupération des données se basent sur l'environnement Processing 2.

Voici les étapes à suivre :

* Télécharger et installer Processing sur la machine qui fera les relevés http://processing.org
* Ouvrir avec Processing le fichier ConvertisseurSerialCSV.pde
* Modifier la ligne udSerial = new Serial(this, Serial.list()[0], 9600) si nécessaire en mettant le port utilisé.
* Exécuter le programme pour récupérer les informations et générer un programme. Quand le nombre des lignes est suffisant, arrêtez le programme et vérifiez le fichier csv généré.

Nous avons à notre disposition un fichier ConvertisseurSerialCSV.pde. Nous l’avons donc pris afin de pouvoir générer le fichier CSV.

## Réalisation du code :



# Module 4 : Module lecture et traitement de données :

## Contexte :

Ce module sera intégralement réalisé en Langage C (pas C Arduino). La mission de ce module est de **charger en mémoire les informations du fichier csv générées par le module précédent et de traiter les données pour afficher certaines informations à partir de données brutes**.

Le code devra être organisé de la manière suivante :

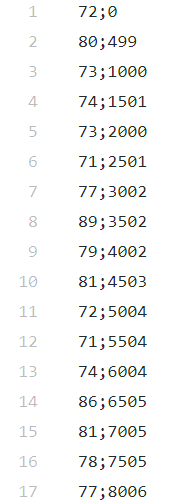
* main. c
* menu.c (et menu.h) : contient les fonctions pour afficher les menus
* donnees.c (et donnees.h): contient toutes les fonctions nécessaires pour la manipulation de la structure de données et la lecture du fichier
* actions.c (et actions.h) : contient toutes les fonctions nécessaires pour le traitement de données (voir ci-après pour plus d’informations sur les actions possibles)

En résumé, le programme démarre, **lit le fichier csv et le charge en mémoire**. Il affiche ensuite un menu avec au moins **les fonctionnalités suivantes** :

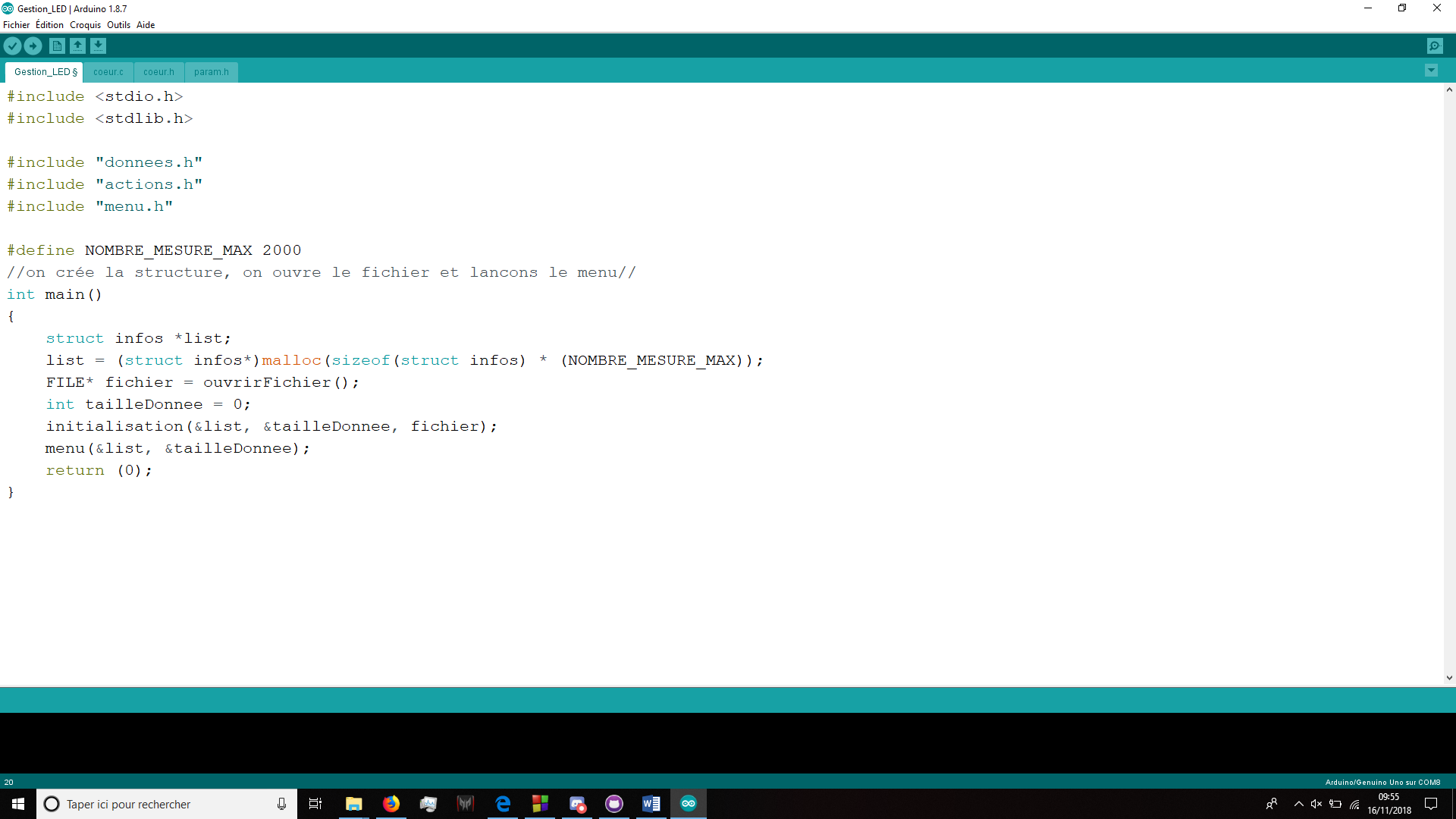
* Afficher les données dans l’ordre du fichier .csv
* Afficher les données en ordre croissant/décroissant (selon le temps, selon le pouls)
* Rechercher et afficher les données pour un temps particulier
* Afficher la moyenne de pouls dans une plage de temps donnée
* Afficher le nombre de lignes de données actuellement en mémoire
* Rechercher et afficher les max/min de pouls (avec le temps associé)
* Autres (facultatif)…
* Quitter l’application

## Réalisation du code :

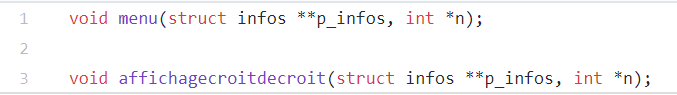
**Voici le fichier Battements.csv :**



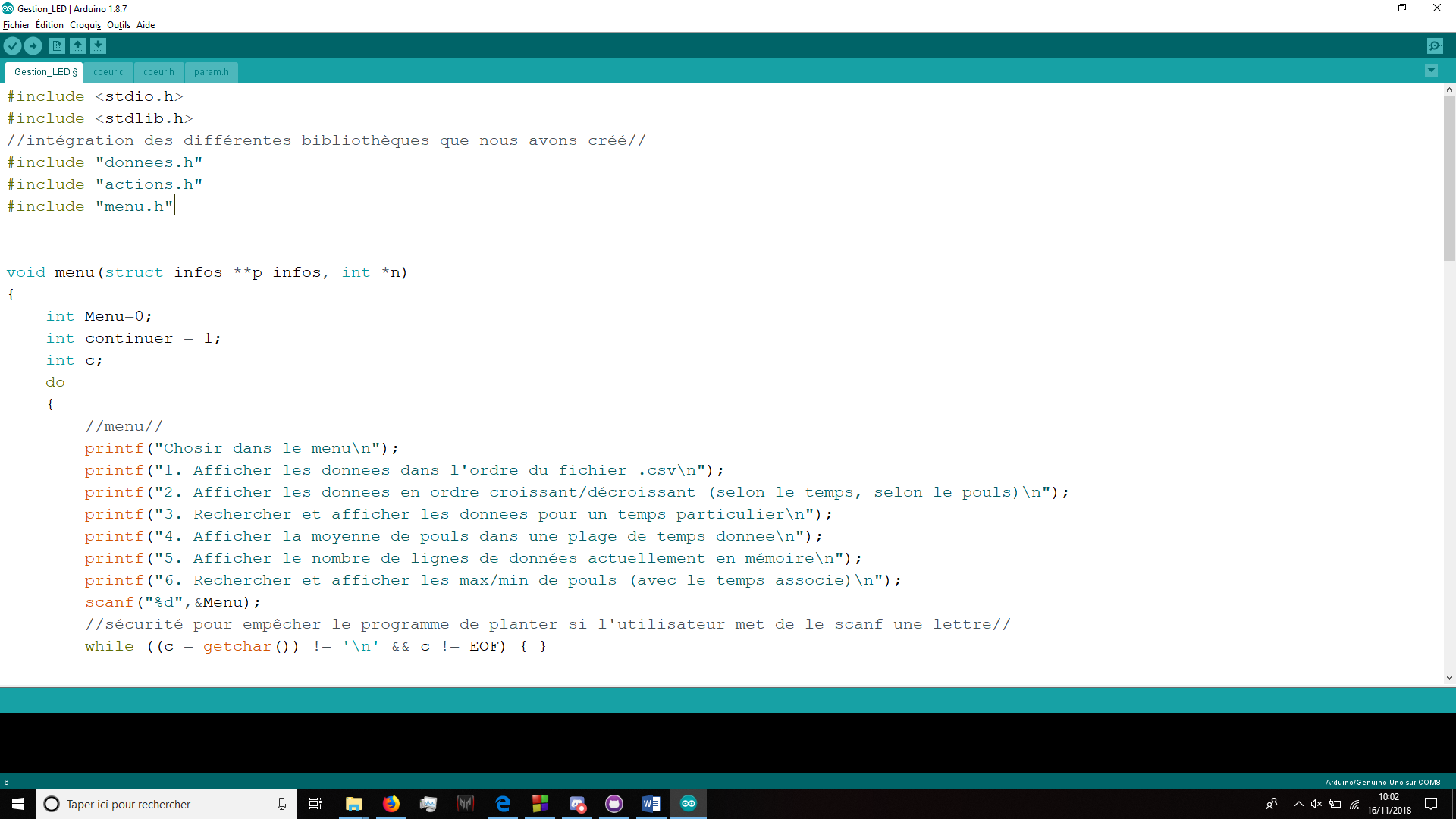
**Voici le fichier main.c :**

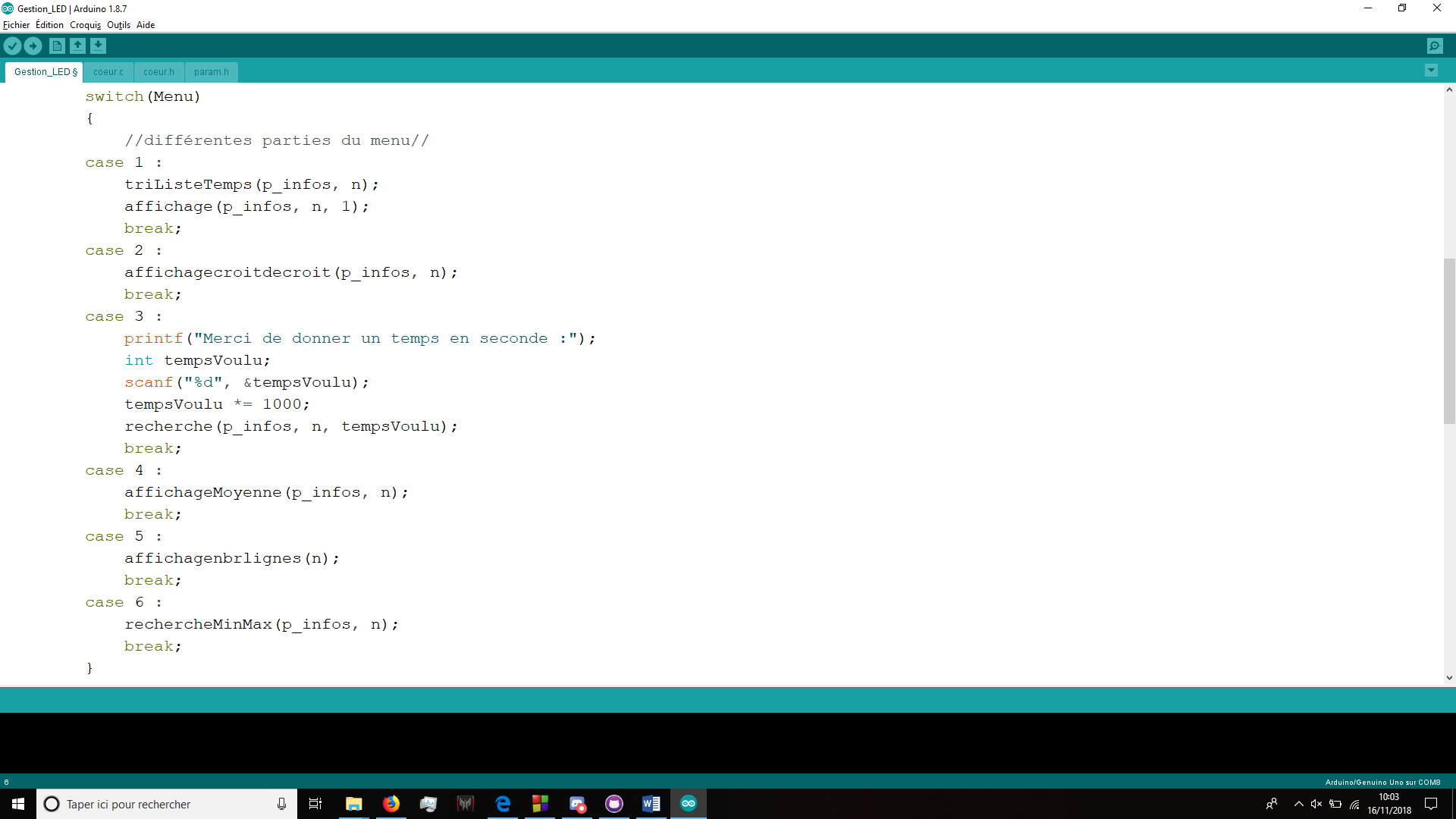


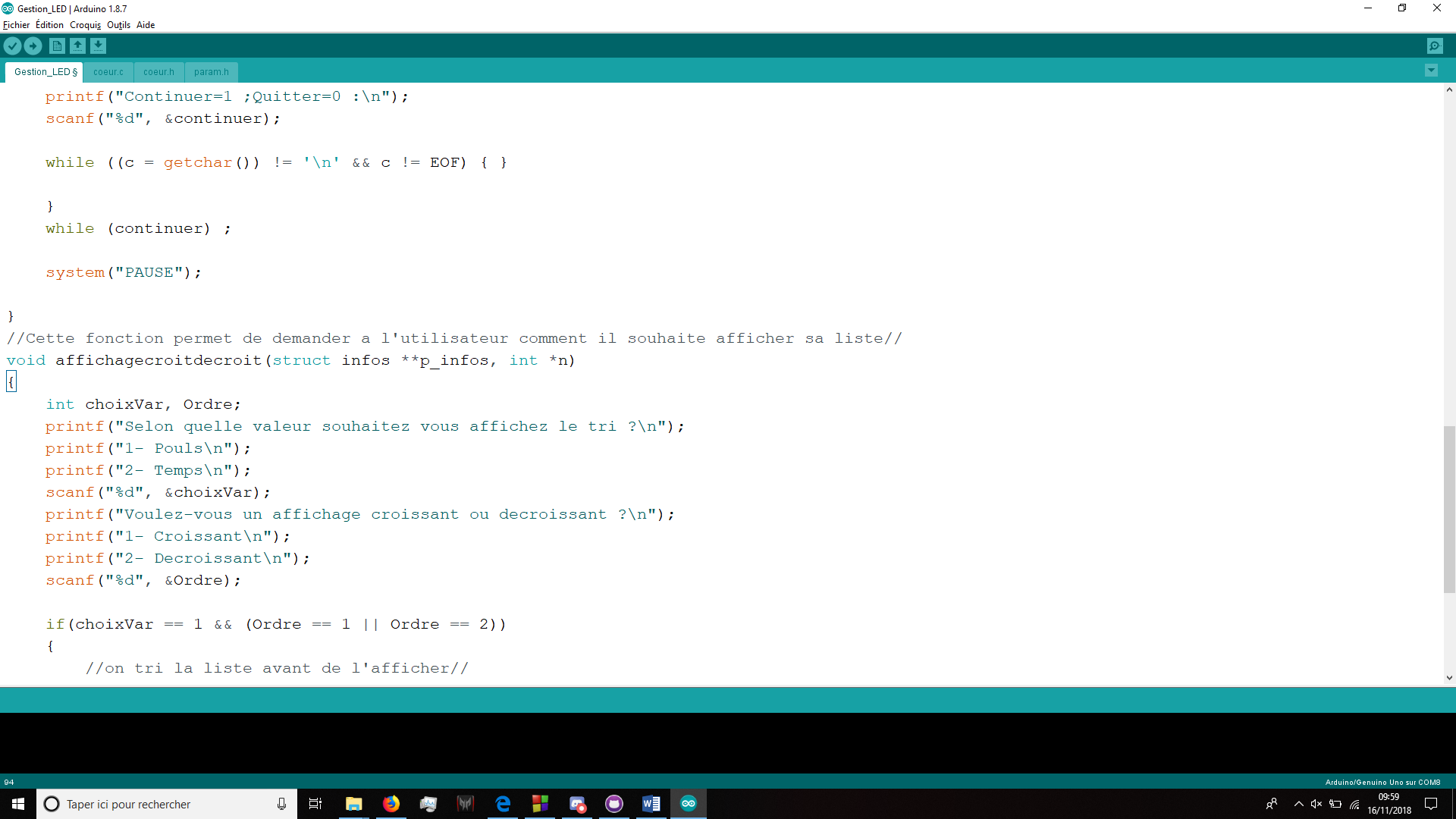
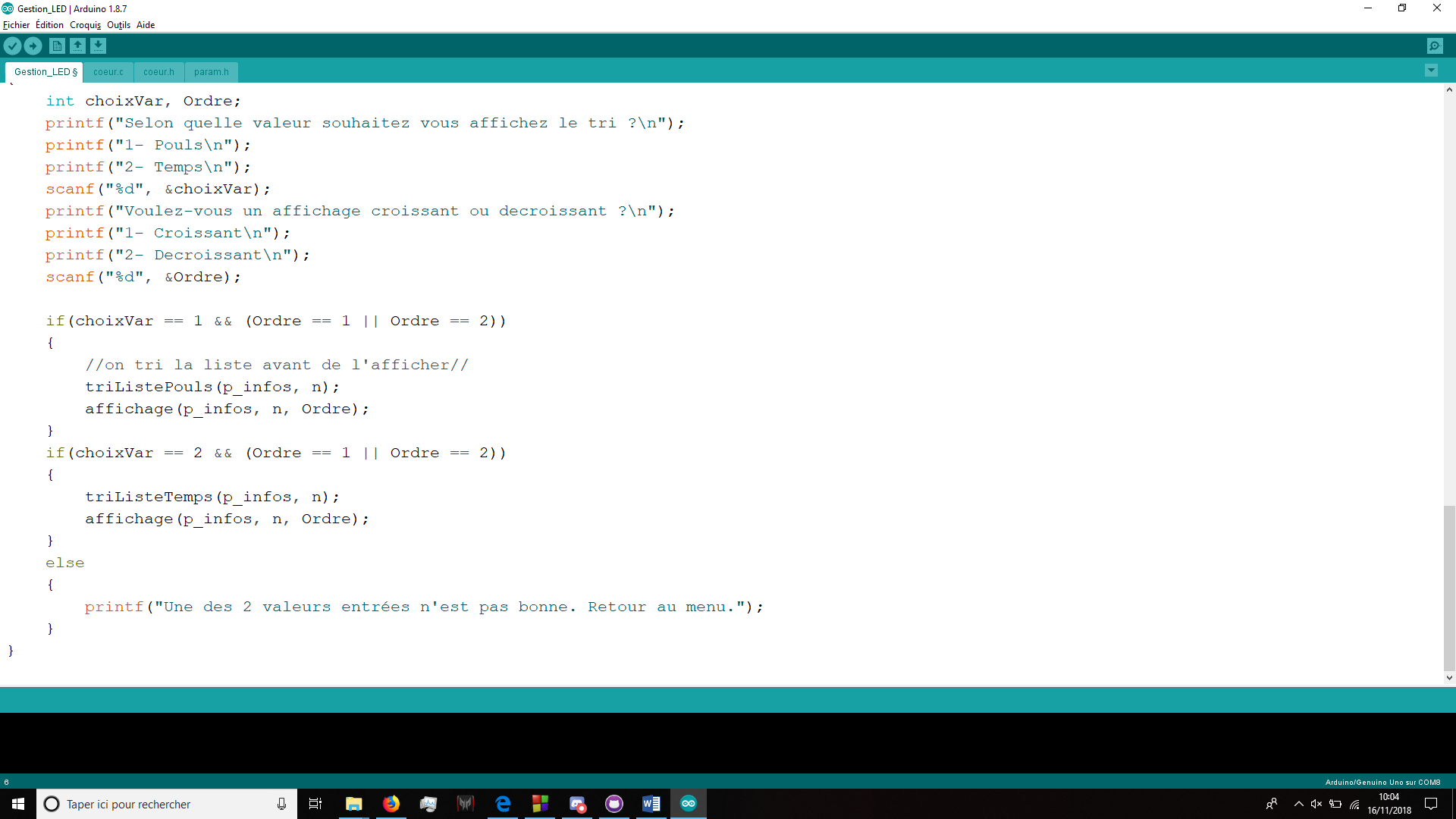
**Voici le fichier menu.h :**



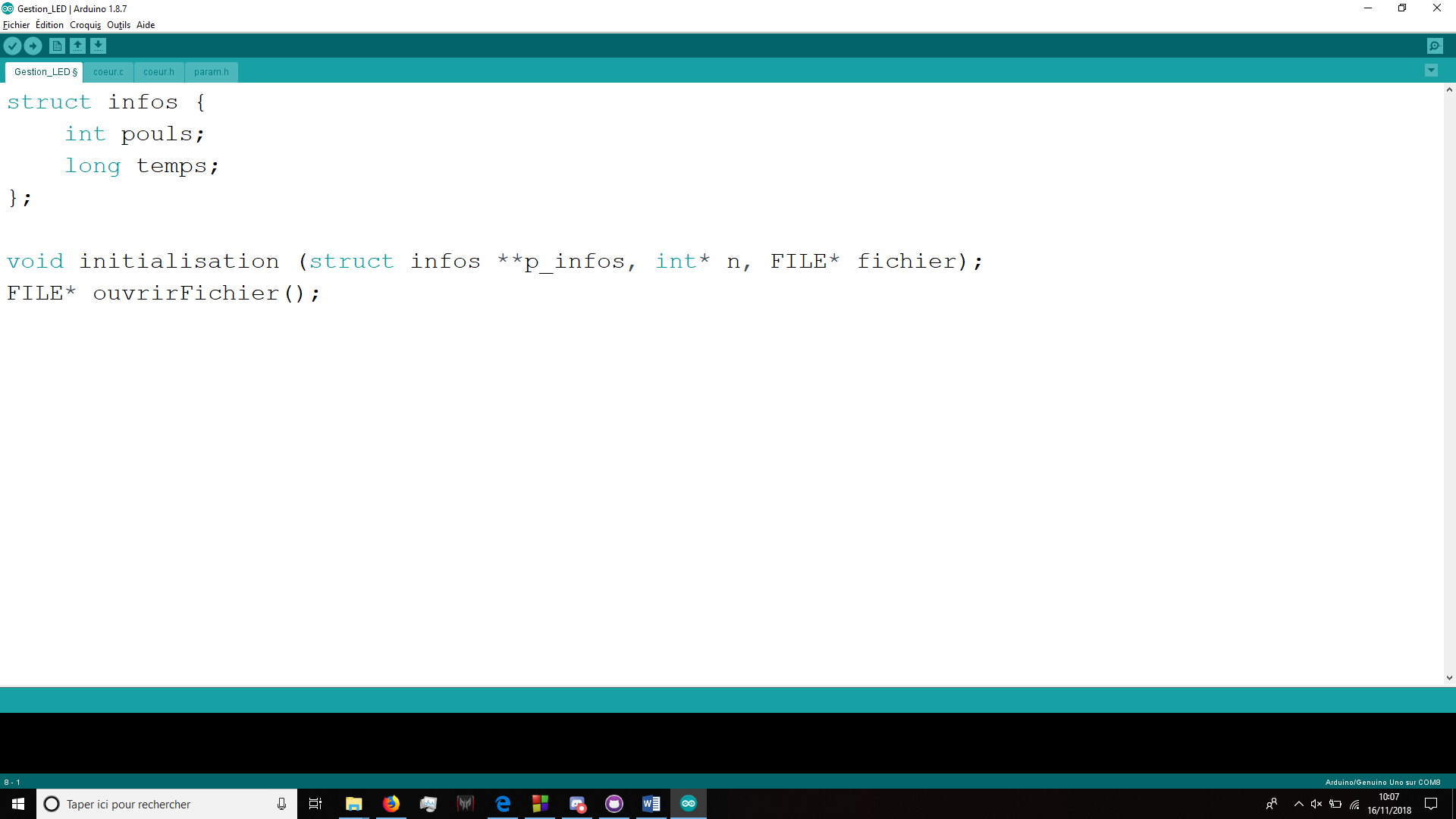
**Voici le fichier menu.c :**



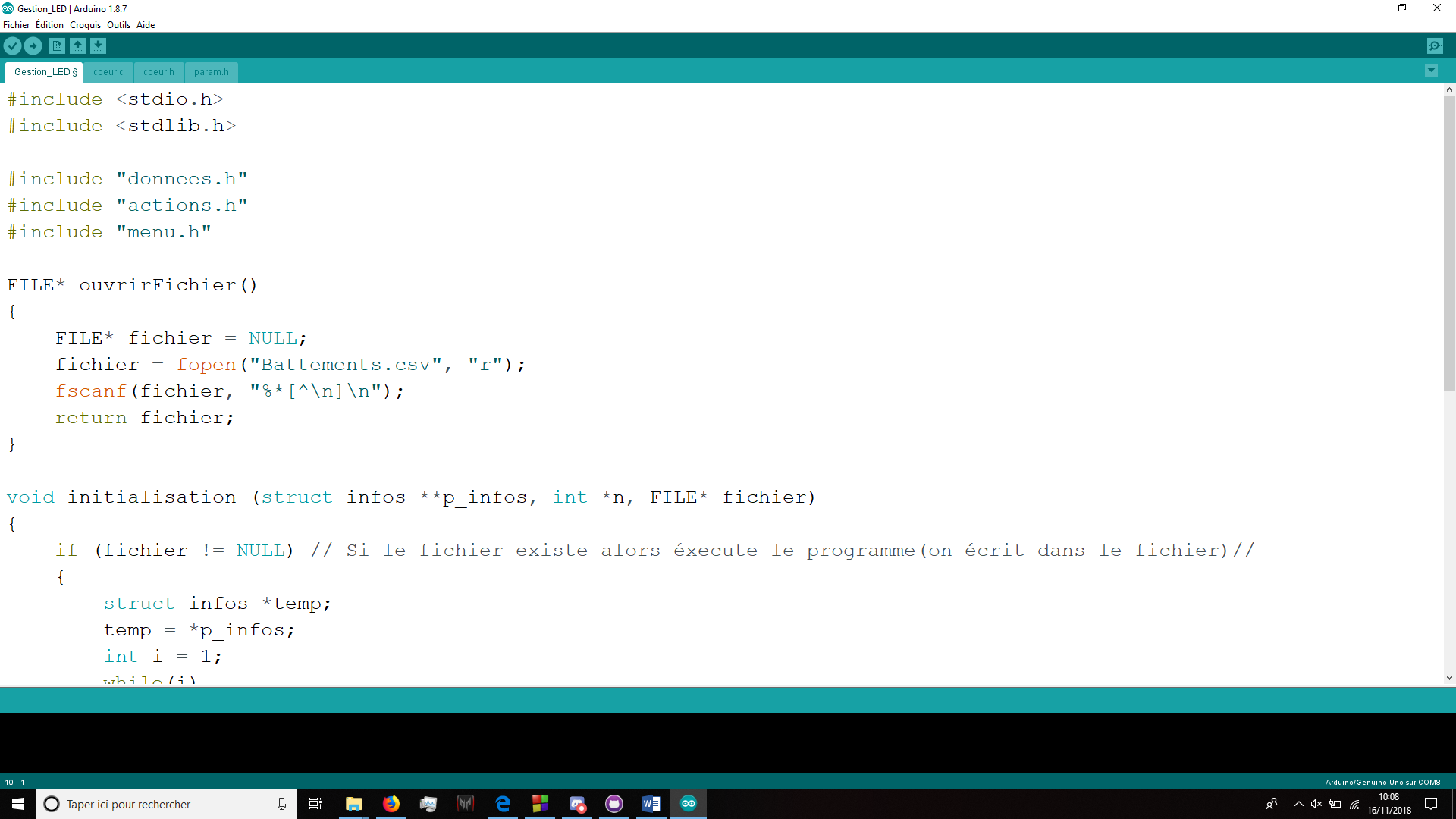


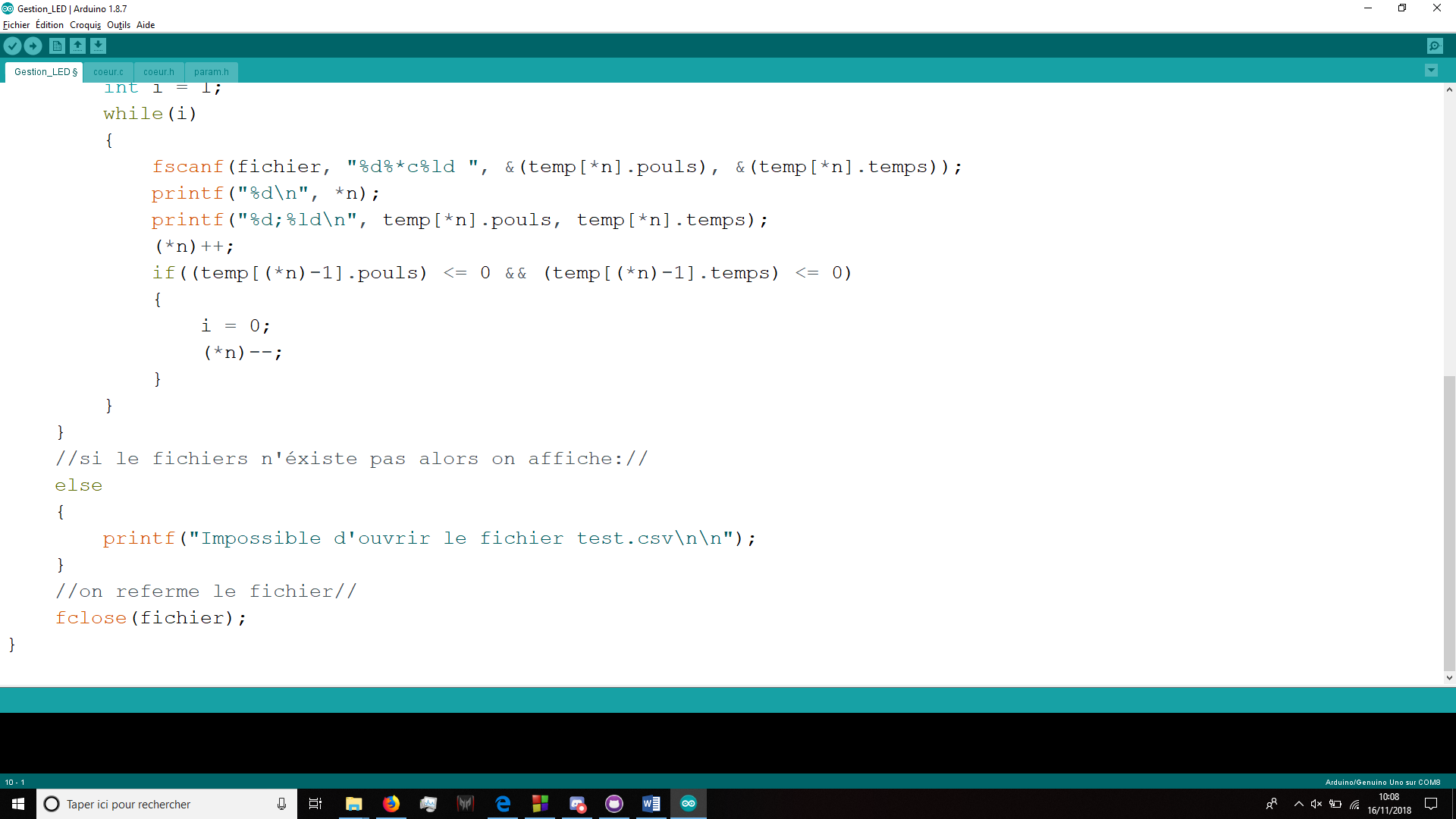


**Voici le fichier donnees.h :**

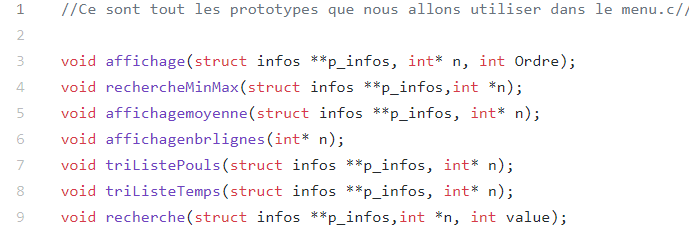


**Voici le fichier donnees.c**





**Voici le fichier action.h :**



**Voici le fichier action.c :**

