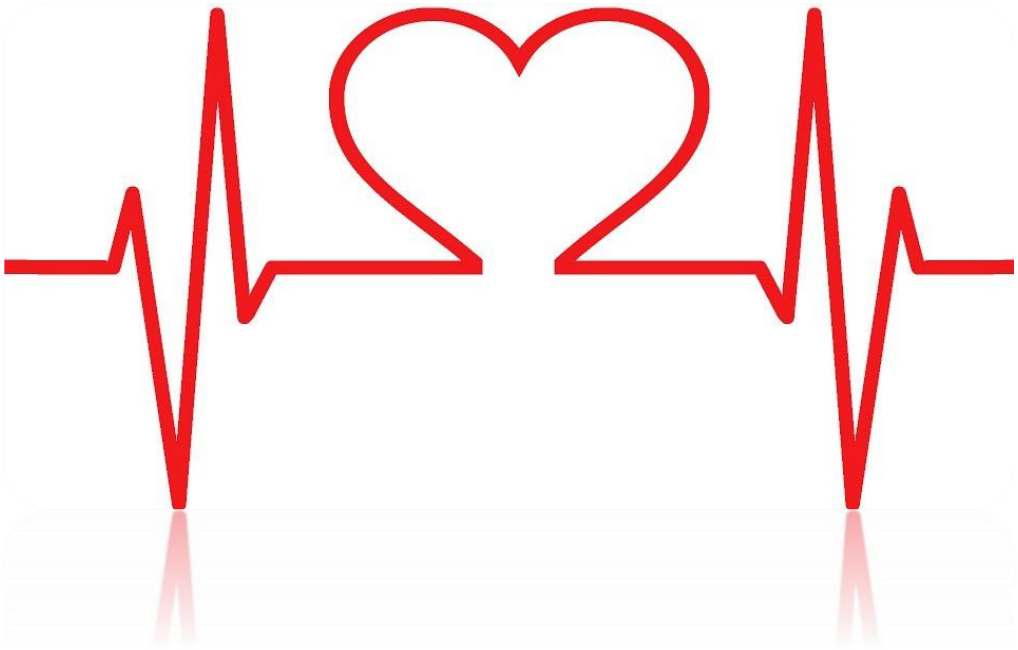
CESI exia
ÉCOLE D'INGÉNIEURS INFORMATIQUE

Projet 01.

HEXART CARE.



Sommaire

Introduction :

Le principal objectif de ce projet a été de réaliser un affichage original du pouls à travers un cœur de LEDS rouges. Nous souhaitons qu'un ensemble de LEDs (formant un cœur) soit allumé.

Plan d'assignation des taches:

Afin d'être le plus efficace possible, chaque personne du groupe a été assignée sur des tâches où elle se sentait à l'aise.

Module01	Montage.	Fadoua.
	Cardio.c	Yacine.
Module02	Precessing.	Yacine.
Module03	Menu1.	Yasmine.
	Générationcode.c/h	Fadoua
	Cœur.c	Yasmine.
Module04	Action.c/h	Yacine.
	Donnée.c/h	
	Menu2.	Yasmine

Architecture du projet :

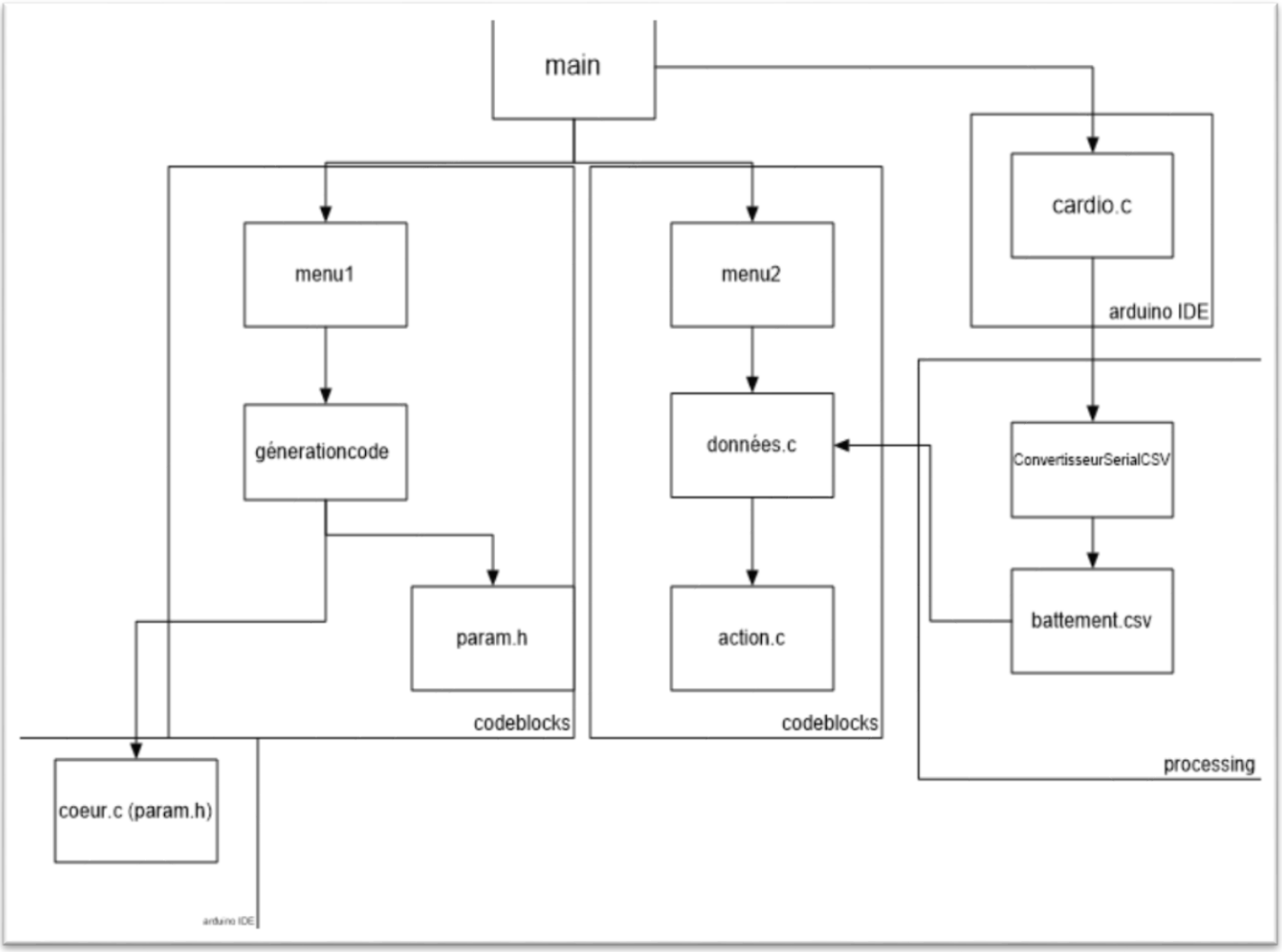
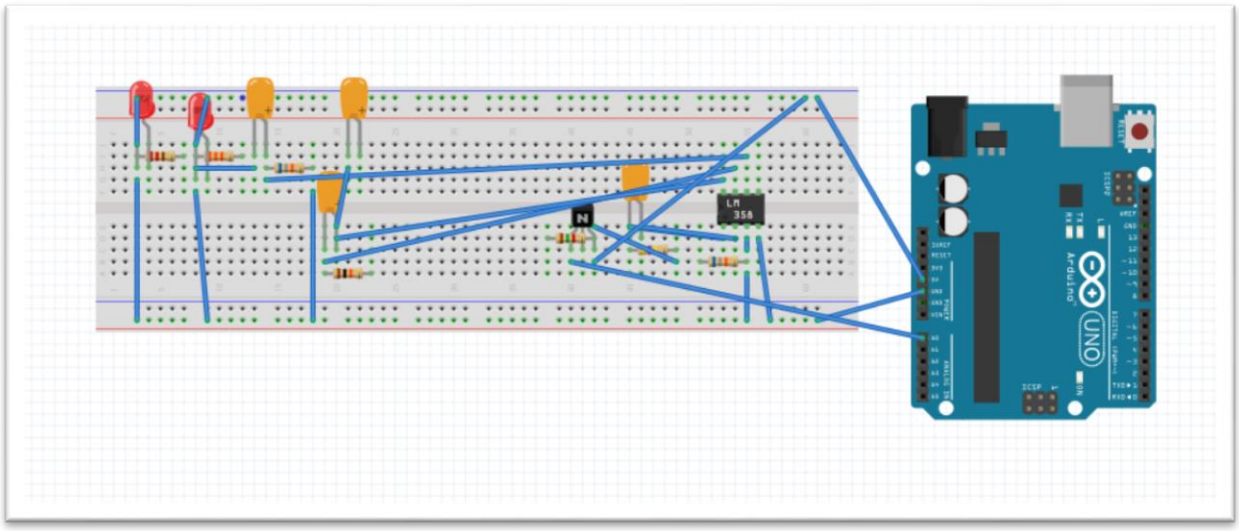


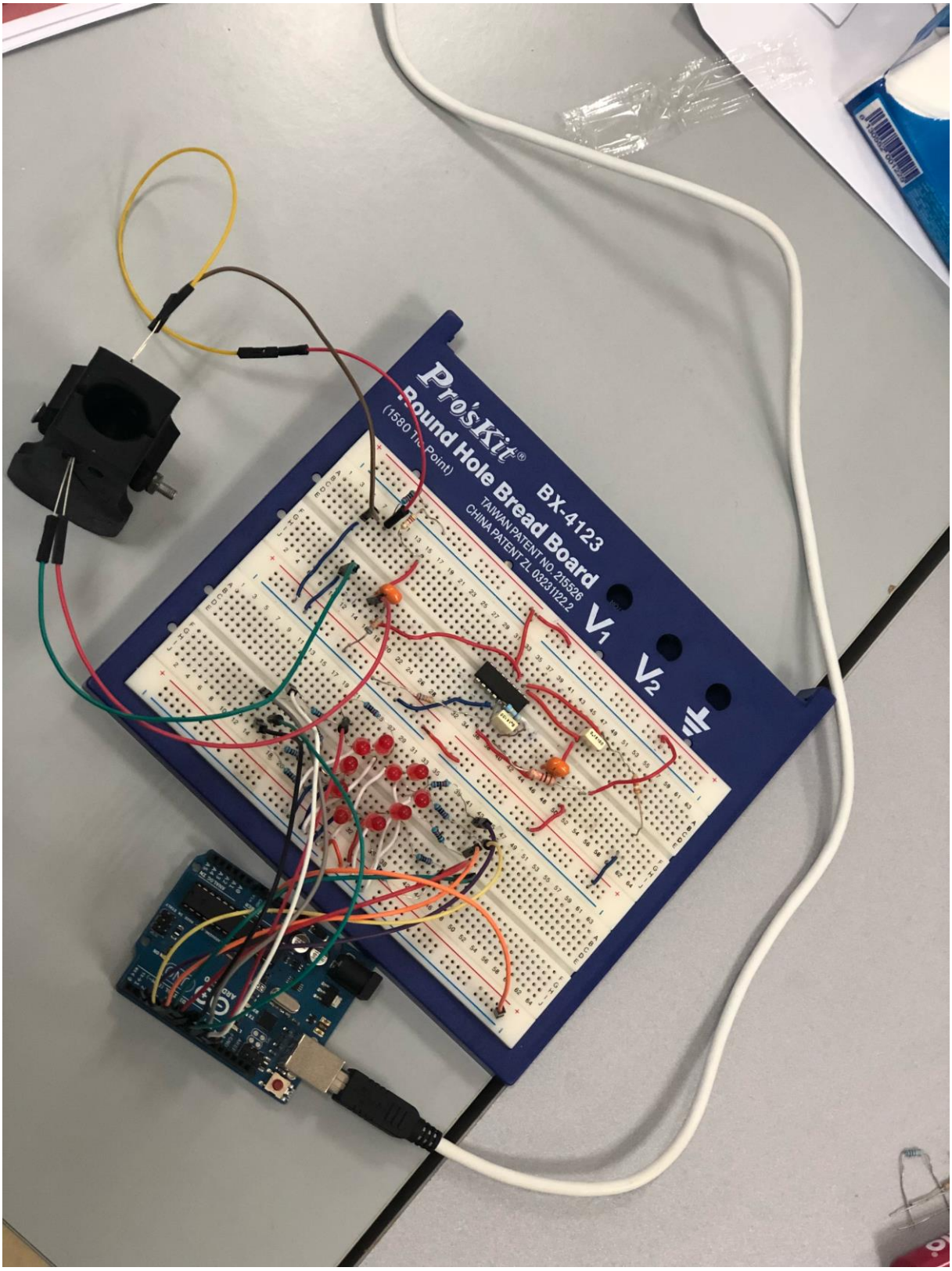
Figure 1 Architecture du projet.

Réalisation des Modules :

Module 01 : Module Cardio.

Nous avons d'abord commencé à réaliser le schéma :





Apres on as coder les function qui vont afficher le pouls

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

void loop()
{
  Serial.println(Pouls_Actuel());
  Serial.println(Delais_Du_Programme());
}
```

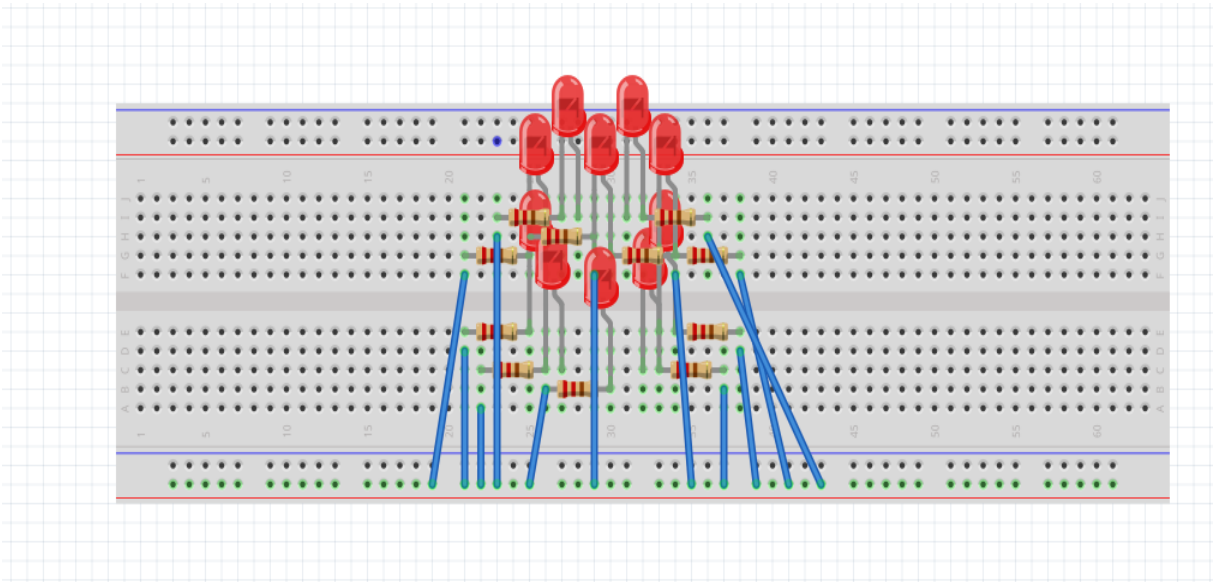
Et voila le code qui contient les fonctions pour calculer le temps et les pouls

Main | Cardio.c | Cardio.h

```
#include "Cardio.h"
#include <Arduino.h>


int Pouls_Actuel()
{
  int valeurPrecedente = 0;
  long tempsPrecedent = 0;
  int valeurActuelle, valeurSeuil;
  long tempsDetection;

  valeurActuelle = analogRead(0);
  valeurSeuil = 650;

  if (valeurActuelle > valeurSeuil)
  {  // on est dans la zone max

        Serial.println(valeurActuelle);


   }

int  Delais_Du_Programme()
{
  int Time = millis();
  Serial.println(Time);
}
```

Module 02 : Module cœur de LEDs.

Dans ce module, on cherche à concrétiser un cœur avec des LEDs, Il est



également nécessaire de créer un menu qui permet de choisir le mode d'affichage désiré.

Voici un schéma fait sur fritzing du cœur de LEDs :

Figure 2 Schéma coeur en LEDS

Les LEDs sont disposées de manière à former un cœur, chaque LED est alimentée de manière séparé par les pins de l'arduino il y'a notamment une résistance entre la LED et le pin de chaque LED afin de ne pas faire surchauffer le composant , on a commencer par le pin numéro deux.

générationcode de param.h

Cette application se découpe en trois parties :

- Main.c
- Menu.c
- Menu.h
- Generationcode.c
- Generationcode.h

La fonction *main* du fichier **Main.c** ne fait qu'appeler la fonction *menu* de **Menu.c**

Menu.c

Ce fichier contient uniquement une fonction *menu*, qui affiche un menu proposant à l'utilisateur de sélectionner l'affichage de son choix. En fonction de sa réponse, une des fonctions de **GenerationCode.c** sera appelée.

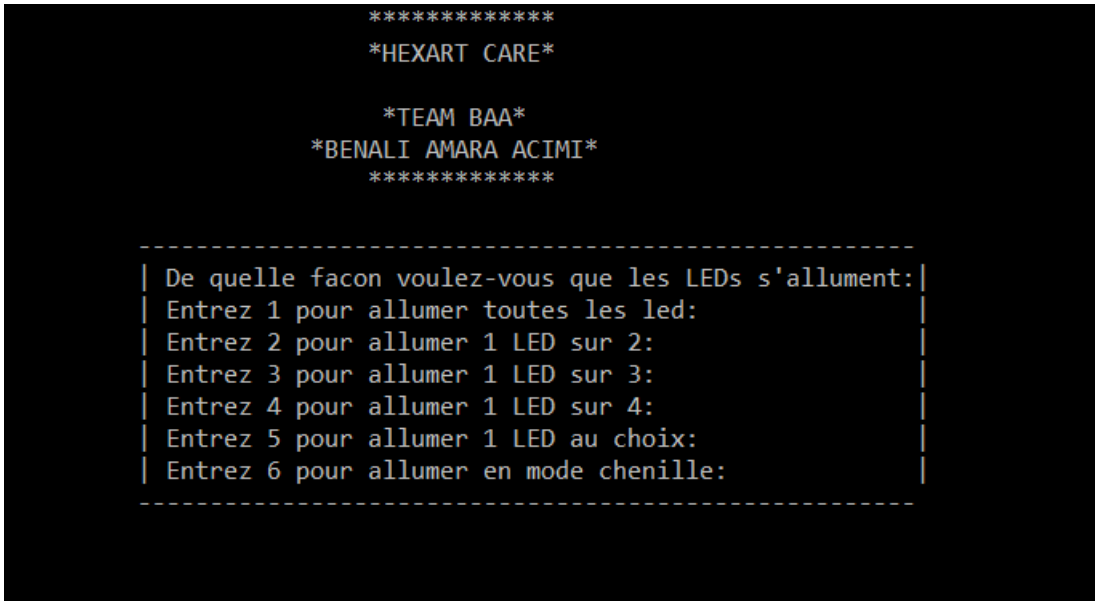


Figure 3 Menu1

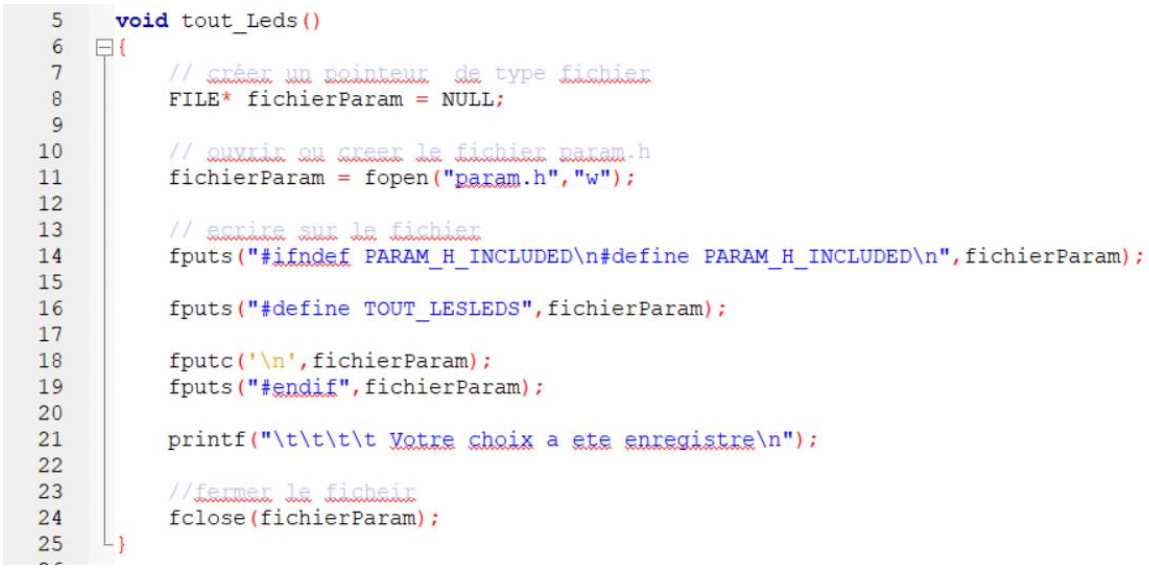
[1]Figure 4 uyuy

Menu.h

ce fichier contient les prototypes des fonctions definies dans le Menu.c

GenerationCode.c

Voici une fonction type, **GenerationCode.c**



```
5   void tout_Leds()
6   {
7       // créer un pointeur  de type fichier
8       FILE* fichierParam = NULL;
9
10      // ouvrir ou creer le fichier param.h
11      fichierParam = fopen("param.h","w");
12
13      // ecrire sur le fichier
14      fputs("#ifndef PARAM_H_INCLUDED\n#define PARAM_H_INCLUDED\n",fichierParam);
15
16      fputs("#define TOUT_LESLEDS",fichierParam);
17
18      fputc('\n',fichierParam);
19      fputs("#endif",fichierParam);
20
21      printf("\t\t\t\t Votre choix a ete enregistre\n");
22
23      //fermer le ficheir
24      fclose(fichierParam);
25  }
```

Figure 5 Exemple géneration de code.

Pour exécuter on fait appel à la bibliothèque **Generationcode.h** (elle contient des prototypes) du coup on l'inclue dans notre generationcode.c

La fonction crée ouvrire un fichier **param.h**, puis écrit dedans un code .

En fonction du mode choisi, la valeur de *mode* (et led si la fonction tout_LEDs a été sélectionnée) va être modifiée par celle du mode désiré.

Enfin, le fichier est refermé et l'application se ferme.

Generationcode.h

Ce fichier est la bibliothèque qui contient les prototypes des fonctions definies dans generationcode.c

Module 03 : Module Processing et Acquisition de données.

Il se constitue d'un fichier ConvertiseureSerial.pde qui son code a était donné dans les ressources ,c'est un programme qui vas recevoir les données envoyées par l'arduino sur le port serial et les convertis et les enregistre sous forme d'un fichier csv

Module04 : Module lecture et traitement de données.

Ici nous avons fait la plus grosse partie du projet qui consiste à coder en C toutes les fonctions qu'on peut choisir comme afficher les donnée, ou les afficher en ordre croissant, decroissant ou le pouls le plus grand ou petit

Le module est divisé comme cela

- Source
  - Main.c
  - Menu2.c :
    - Fonction afficherMenu2 pour afficher les options de menu
  - Donnee.c :
    - Fonction LireFichier pour charger le fichier csv en mémoire
    - Fonction tailleDeFichier pour calculer le nombre des lignes dans le ficher csv
  - Action.c :
    - Fonction afficherCsv pour afficher le tableau
    - Fonction chercher pour trouver le poul correspandant a un temps précis
    - Fonction trouverMin pour trouver le poul le plus petit
    - Fonction trouverMax pour trouver le poul le plus grand
    - Fonction moyenne pour calculer la moyenne des pouls dans un temps donné
    - Fonction afficherLesLignes pour afficher le nombre de lignes en mémoire
- Headers (contion les prototypes des fonctions)
  - Menu2.h
  - Donnee.h
  - Action.h

La strucutre utiliser dans ce module est un tableau de structure

```
4   typedef struct
5   {
6       int time;
7       int pulse;
8   }table;
9
```

Figure 6 Structure de donnée

Main.c /h:

Elle crée un pointeur vers de type structure et appelle la fonction LireFicher()

Donnee.c/h :

Apres l'appel de **lireFicher()** elle va appeller la fonction **tailleDeFicher()** qui vas calculer le nombre des lignes dans fichier **battemment.csv** et retourne la valeur a la fonction **lireFichier()**

Ce dernier va créer un tableau de taille dependant sur la valeur retourner par la fonction **lireFichier()**

Elle charge les données du ficher **battement.csv** et les met dans un tableau de structure pouls

Et retourre cette strucutre a la fonction **main()**

Main.c/h :

la fonction main vas recevoir le tableau de strucuture et appelle la fonction affihcerMenu2 en envoyant le tableau de structure

Menu2.c

La fonction **afficherMenu2()** vas afficher les diffrentes options disponible

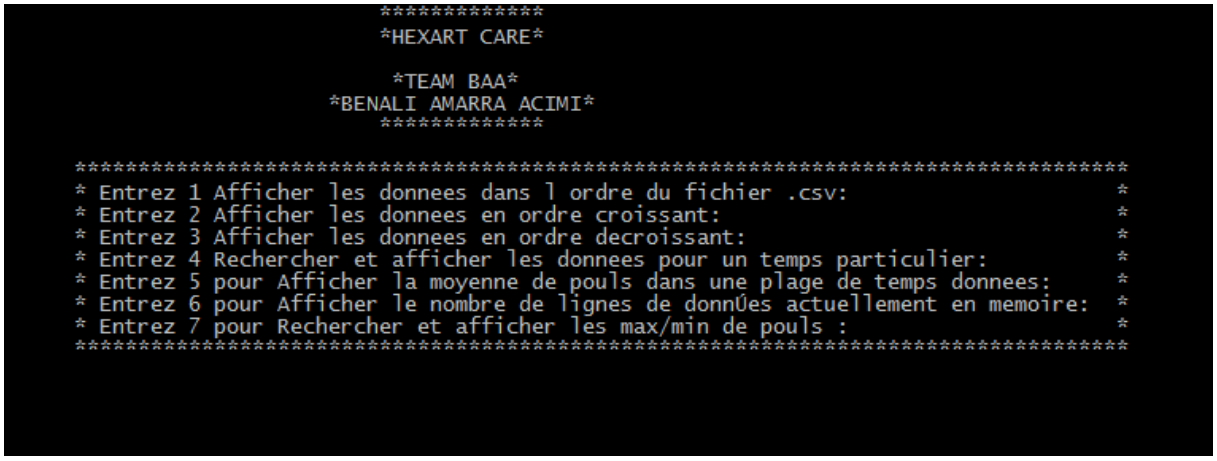


Figure 7 Menu2

L'utilisateur va choisir une option et la fonction **afficherMenu2()** va l'appeler au depent du choix de l'utilisateur en envoyant les paramètres corresependants.

Action.c

Il contient toutes les fonctions qui vont etre appeller par la fonction **afficherMenu2()**

ces fonctions sont :

- afficherCsv()
- afficherLesLignes()
- chercher()
- trouverMin()
- trouverMax()
- moyenne()
- afficherLesLignes()

Conclusion :

Nous avons réussi à obtenir un prototype quasi-fonctionnel, notre montage nous à tout de même permis de tester toutes les parties codes en simulant les battements de cœur par des mouvements au-dessus du phototransistor.