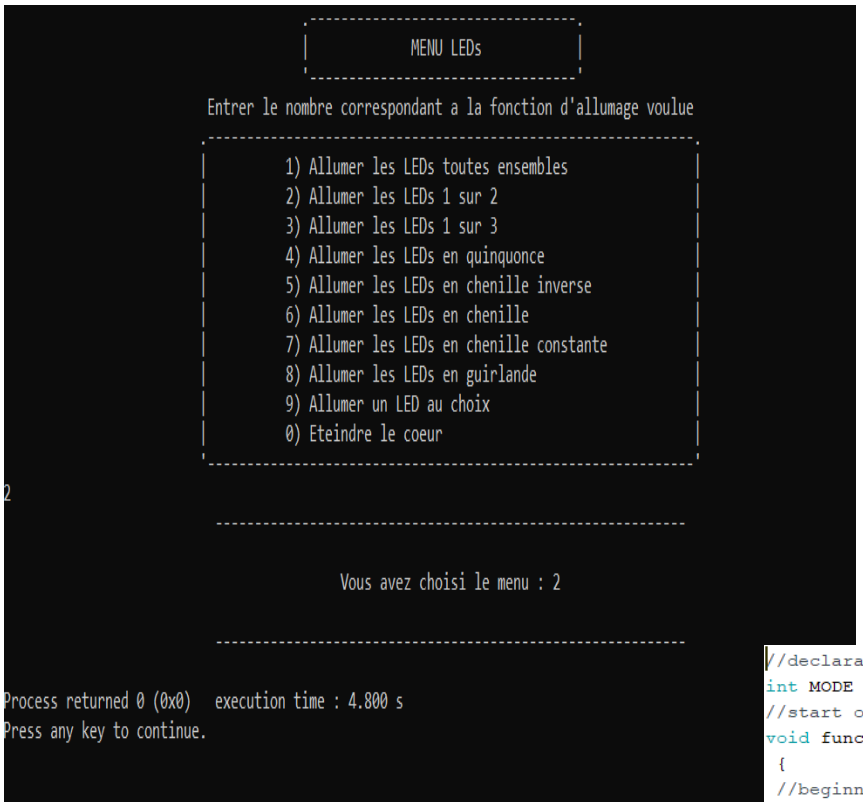
RAPPORT TECHNIQUE DU CODE ARDUINO ET DU CODE C

A la suite du montage électrique, nous avons commencé par coder le fichier Arduino, afin d'avoir différents modes d'allumage.
Pour cela, nous avons différents fichiers :

- Un fichier « cœur.c » contenant les librairies « cœur.h » et « param.h » ainsi que la déclaration de notre constante qui gère les LEDS. Dans ce fichier, nous retrouvons également un « void set up », dans lequel se trouve notre !!! . Et pour finir, un « void loop » avec l'appel de notre fonction « function() ». Nous retrouverons cette fonction dans notre fichier « param.h » généré par notre générateur de code en C
  (II/fichier 1).
- Un fichier « cœur.h » contenant uniquement le prototype de la fonction appelée dans le fichier « cœur.c » : void function(); .
- Un fichier « param.h » qui lui est généré par le fichier « generationCode.c » du code en C. Cela signifie que ce fichier est modifié à chaque fois que le code en C est compilé et que l'utilisateur choisit un mode.
  C'est-à-dire que si l'utilisateur choisit en premier le « Mode_2 », le programme va générer dans le fichier « param.h » le code de la fonction du « Mode_2 ».




```
//declaration of an integer MODE variable
int MODE = 2;
//start of the function
void function()
 {
 //beginning of a loop for
//condition of the loop : initialize variable i to 2,
 for(int i = 2; i < 12; i+=2)
 {
//sets led i on
 digitalWrite(i, HIGH);
 //waits 60 milliseconds
 delay(60);
//sets led i off
 digitalWrite(i, LOW);
 //waits 60 milliseconds
 delay(60);
 }
}//end of a loop for
```

Après le code Arduino, nous avons réalisé le code en C. Ici aussi, nous avons plusieurs fichiers :

- Un fichier « main.c » dans lequel nous commençons par inclure les différentes bibliothèques qui vont permettre de pouvoir passer de fichier en fichier. De plus, il contient la fonction principale « int main() » dans laquelle est appelé la fonction « display_menu() » du « menu.c ».
- Dans le fichier « menu.h », nous retrouvons uniquement le prototype de la fonction « display_menu() ».
- Ensuite le fichier « menu.c » contient toute la fonction « display_menu() ». C'est-à-dire qu'il y a l'affichage de tous les différents modes d'allumage des LEDs grâce au « printf(« ») ».
  De plus, nous avons une boucle « if » si l'utilisateur choisit l'allumage du « Mode_9 ». En effet, cette boucle permet à l'utilisateur quelle LED, il veut allumer.
  Ensuite, nous avons un « switch/case » où dans chaque case, nous avons l'appel de la fonction du mode qu'il a choisi. C'est-à-dire que si l'utilisateur choisit le « Mode_4 », le « case 4 » va appeler le code de cette fonction.
- Le fichier « generationCode.h » comporte tous les prototypes des fonctions déclarées dans « generationCode.c ». Dans les prototypes, il ne faut pas oublier d'y mettre les paramètres s'il y en a.
- Pour finir, on retrouve dans le « generationCode.c », toutes les fonctions pour les différents modes. Pour chacune de ces fonctions, nous devons générer un fichier « param.h » qui va pouvoir écrire directement le code dans le fichier de l'Arduino. Ensuite, pour pouvoir écrire dans le fichier, il faut utiliser un « fprintf ».

Dans chacun des modes, nous avons alors un code différent. Cependant, nous retrouvons pour chaque fonction, la déclaration d'une variable « MODE » permettant d'identifier le bon mode, ainsi que le début de la fonction « void function() ».
Ensuite nous retrouvons des programmes différents :

- Mode 1 : 2 boucles « for » qui permettent de faire clignoter mes LEDs dans un temps déterminé.
- Mode 2 : 1 boucle « for » qui permet de faire clignoter les LEDs une par une mais une LED sur 2.
- Mode 3 : 1 boucle « for » sur le même principe que le Mode 2 mais une LED sur 3.
- Mode 4 : 4 boucles « for » qui permettent de faire clignoter toutes les LEDs pairs en alternées avec les LEDs impairs.
- Mode 5 : 1 boucle « for » qui permet de faire clignoter les LEDs successivement en partant vers la gauche.
- Mode 6 : même principe que le Mode 5 mais en partant vers la droite.
- Mode 7 : même principe que le Mode 6 mais les LEDs ne s'éteignent pas à chaque fois qu'elles passent à la suivante.
- Mode 8 : il n'y a pas de boucle, nous avons mis des « digitalWrite() » pour que les LEDs s'allument et s'éteignent à un temps déterminé. L'algorithme réalisé permet de réaliser un allumage ressemblant à une « guirlande ».
- Mode 9 : le code permet à l'utilisateur de choisir la LED qu'il veut faire clignoter. Pour cela, nous avons intégrer une nouvelle variable LED.
- Mode 10 : nous avons intégrer une variable LED initialisée à une valeur RANDOM, c'est-à-dire que les LEDs vont s'allumer et s'éteindre aléatoirement.