

Un cœur qui bat au rythme de vos rêves !

Groupe 5 :

VANMANSART Timothé

GUYOT Erwan

LEGROUX Pauline

HOANG Kevin

Projet A1- FONDAMENTAUX SCIENTIFIQUES

A1 2018/2019



**Première partie : AVOIR UNE VUE D’ENSEMBLE DU PROJET**

1. ***Dessinez l’architecture du projet – comment avez-vous compris le projet ?***

Le projet se scinde en 4 modules :

Le module 1 consiste à créer un montage qui allume une LED en lien avec un phototransistor qui capte les infrarouges en fonction de l’afflux sanguin. On doit ensuite créer un code Arduino qui affiche dans un .csv, le temps depuis le démarrage de l’Arduino et le pouls à cet instant.

Le module 2 consiste à créer un montage de LED représentant un cœur dont le but est d’afficher les battements du cœur selon différents mode d’éclairage / de clignotement. Il faut créer un fichier de code en c où le cœur doit battre au rythme des informations stockées dans le fichier .csv.

Le module 3 correspond à la création du fichier .csv et la récupération des données du premier module.

Le module 4 est intégralement en langage C, on doit charger en mémoire les informations du fichier csv générées par le module précédent et traité les données pour afficher certaines informations à partir de données brutes. On utilisera donc des algorithmes de tri et de recherche pour accéder à certaines valeurs.

**Deuxième partie : ANALYSEZ LES STRUCTURES DE DONNEES DU PROJET**

1. ***Représentation graphique de toutes les structures nécessaires, organisation des fichiers de code Arduino et du code C et dépendances entre les fichiers.***

Module 1 : Arduino

Main.c

Cardio.h

Int pouls\_alea() ;

Int timer() ;

Void RenvoiCSV(… ,…) ;

Module2 : Cœur de LEDs (C non Arduino et C Arduino)

Main.c

Menu.h

Int choiceParm() ;

Int validation() ;

codeGeneration.h

void all() ;

void alternance2() ;

void alternance3() ;

void unique() ;

void chenille1() ;

void chenille2() ;

void chenille3() ;

création

Code C Arduino

Param.h

#define MODE…

#define PARAMETRE …

Module 4 : Dans le main.c est intégré le menu.h avec tous les prototypes de fonctions. Ces prototypes retrouvent les fonctions dans le menu.c ou est intégré le donnes.h et ainsi de suite.

Main.c

Menu.c

Menu.h

Int menulecture() ;

donnees.h

Int ft\_lecturecriture() ;

Int ft\_lectureinverse () ;

Int ft\_lecturerecherche();

In ft\_lectureligne() ;

Int ft\_lectureminmax() ;

Int ft\_lecturemoyenne() ;

Donnees.c

action.h

void swap(int \*nb1, int \*nb2);

int ft\_max(int tableau[], int taille);

int ft\_min(int tableau[], int taille);

void tri\_selection(int tableau[], int tableau2[], int taille);

void rechdicho(int tableau[], int ligne);

void ft\_mintableau(int pouls[], int ligne);

void ft\_maxtableau(int pouls[], int ligne);

void ft\_moyenne(int tab[]);

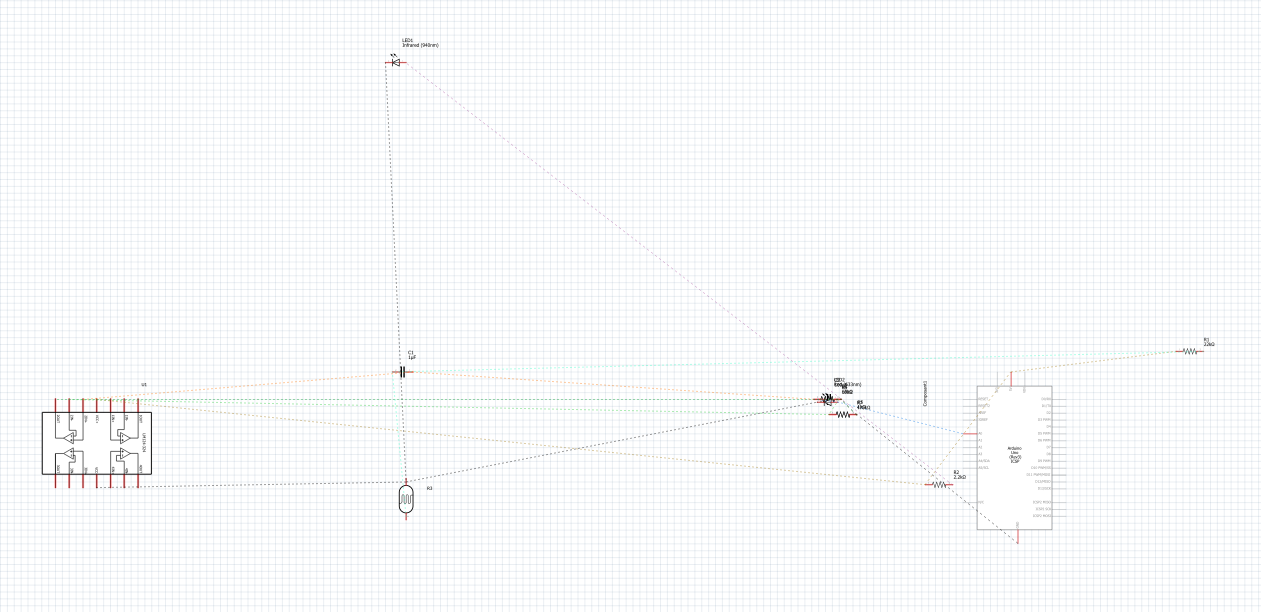
int rechdichocase(int tab[], int ligne, int nb1) ;

Action.c

**Troisième partie : MODULARISATION & WORKFLOW DE FONCTIONS & SCHEMAS ELECTONIQUES**

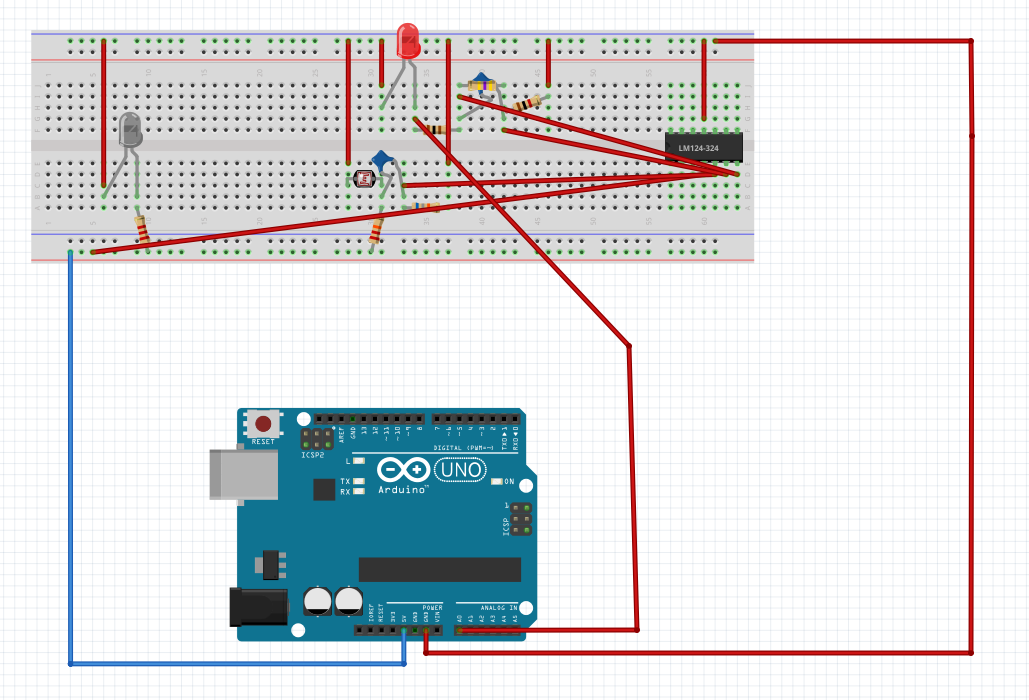
1. ***Schémas électroniques avec les composants sur Fritzing (vue platine et vue schématique des modules cardio et cœur de LEDs). Comme cette partie comporte une évaluation séparée du projet, vous pouvez faire un document à part entière.***

**Module 1 : capteur infrarouge**

Ce montage sert à réaliser une mesure du pouls d’une personne grâce à une LED infrarouge (LED en gris sur le schéma mais bleutée en vrai) et un phototransistor (au milieu de la breadboard, mais ressemblant à une LED noire en vrai).

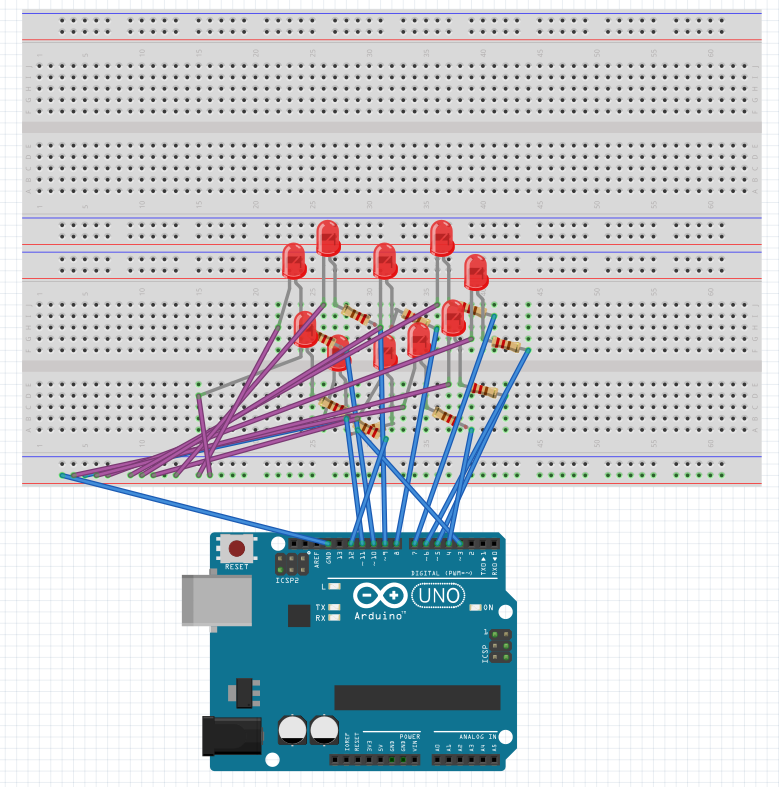
Vue platine montage 1

Pour récupérer le pouls, on place le doigt dans une pince dans lesquelles il y a des trous au-dessus et en dessous pour placer la LED infrarouge et le phototransistor. On place la LED infrarouge et le phototransistor de part et d’autre de notre doigt : à chaque pulsation du cœur, l’afflux sanguin augmente dans le doigt. Cette variation d’afflux sanguin change la luminosité de la LED infrarouge perçu par le phototransistor (moins d’infrarouges traversent le doigt), le photo transistor capte cette modification et fait s’allumer une LED rouge normale en rythme avec le battement du cœur. La LED rouge est constamment allumée et s’éteint lors d’un afflux sanguin (d’un battement de cœur). Le nombre de battements en fonction du temps est une donnée récupérée par la carte Arduino à travers le port analogique A0.

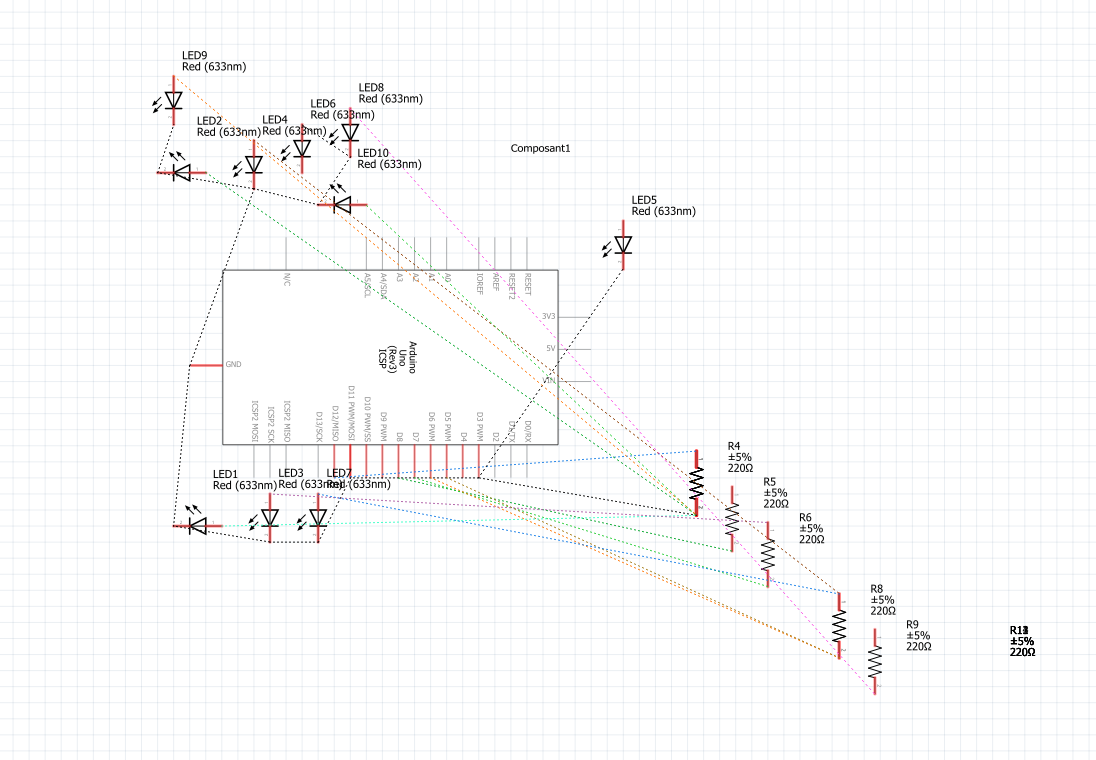


Vue schématique montage 1

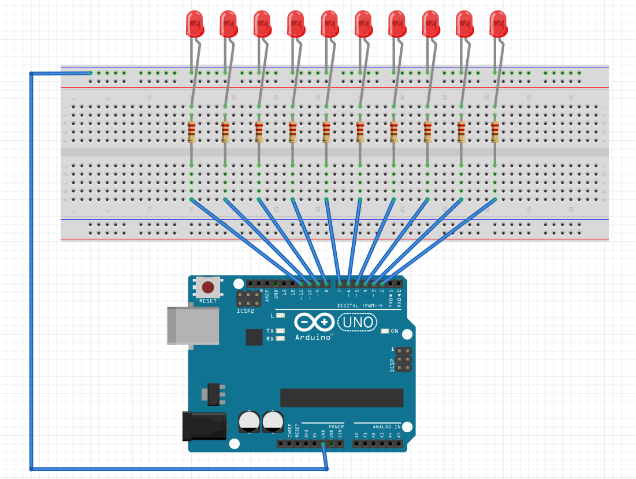
**Module 2 : Cœur de LEDs**



Vue schématique montage2 : cœur de LEDs



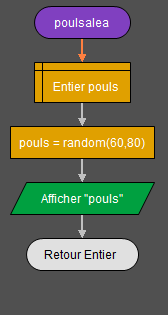
Vue platine montage2 : cœur de LEDs

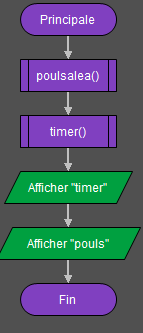


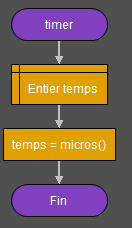
Vue schématique montage2 : ligne de LEDs

On a représenté le cœur sous forme de ligne pour une plus grande visibilité et une meilleure compréhension du montage demandé sur fritzing et réel. Cependant les deux montages sont exactement les mêmes, en effet ils ont les mêmes composants, seul la place des LEDs et des résistances est différente. On utilise ici les ports 2 à 11 : chaque port est relié à une résistance puis à la borne positive de la LED. La borne négative de la LED, elle, est reliée au GND. On ne branche qu’une seule LED par port pour pouvoir allumer et éteindre chaque port indépendamment et ainsi pouvoir contrôler une LED à la fois. Dans le code Arduino nous indiquons quel port nous souhaitons allumer ainsi nous pouvons faire des effets différents pour les battements du cœur.

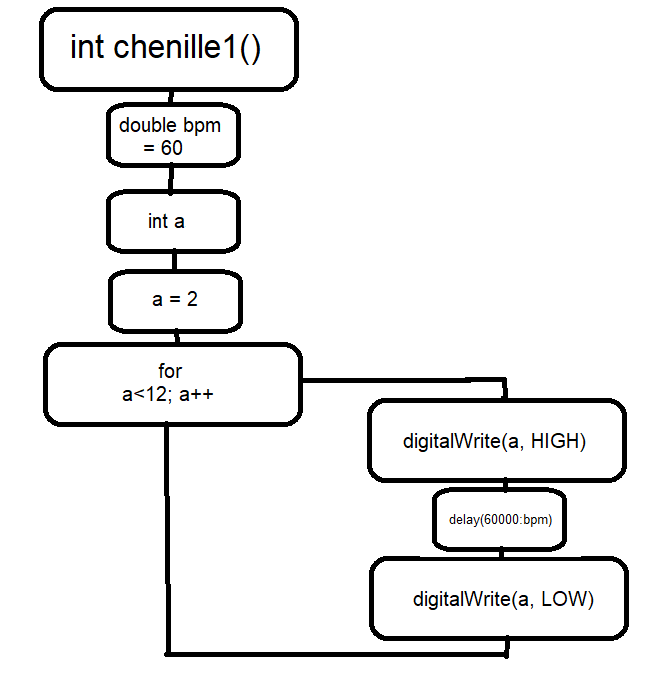
1. ***Description algorithmique chaque partie du projet (module cardio, module cœur de LEDs (inclus la génération automatique du paramétrage à partir d’un programme en C), module Preprocessing/acquisition des données, module lecture et traitement de données en C).***

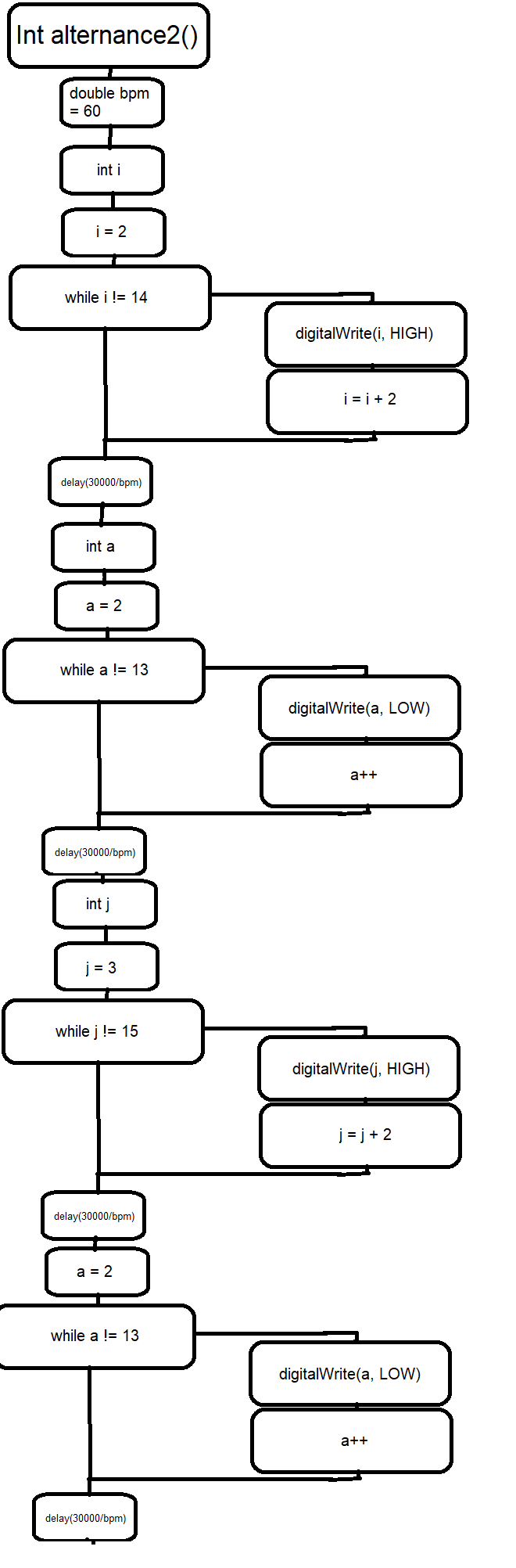
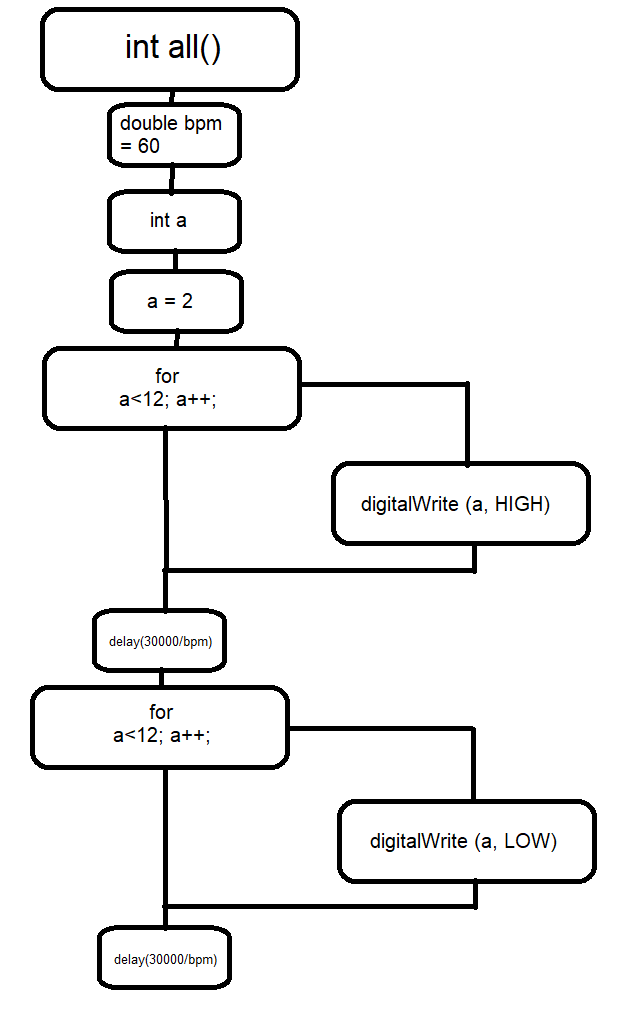


Module1 :



Ce module est composé d’un main et de deux fonctions. Dans la fonction timer, on créé le temps et la fonction poulsalea créé un pouls compris entre 60 et 80. Ces deux fonctions sont ensuite appelé dans le main puis on affiche ces valeurs.

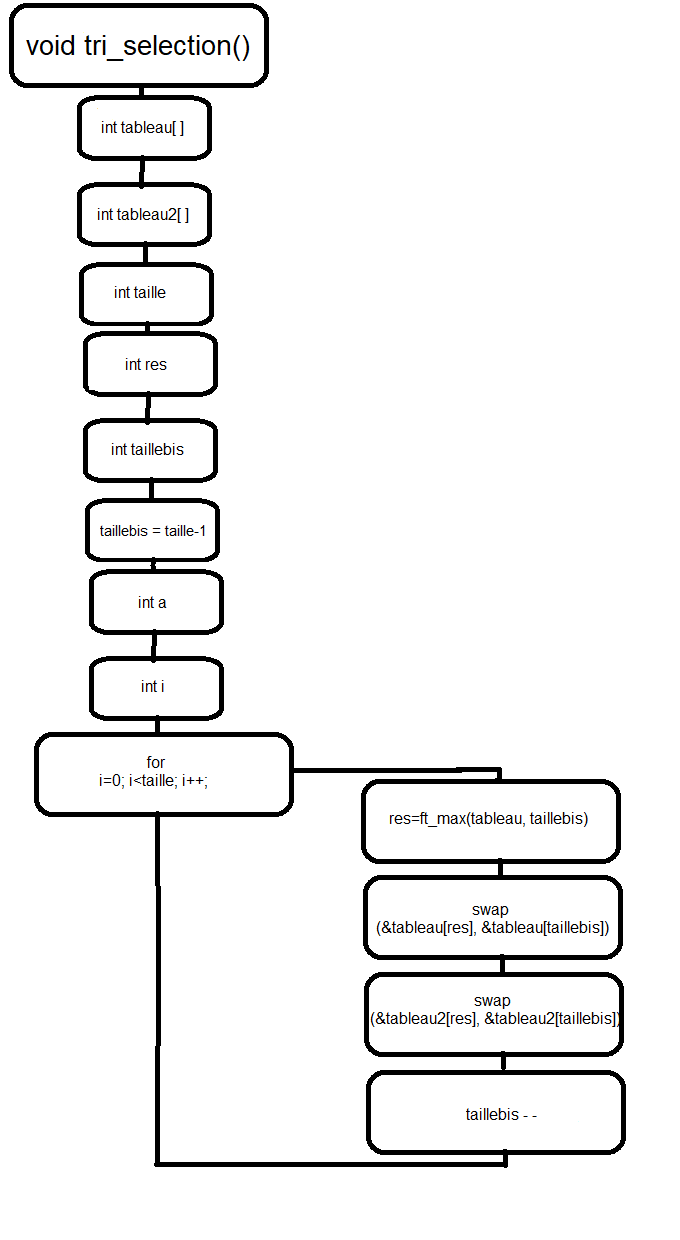
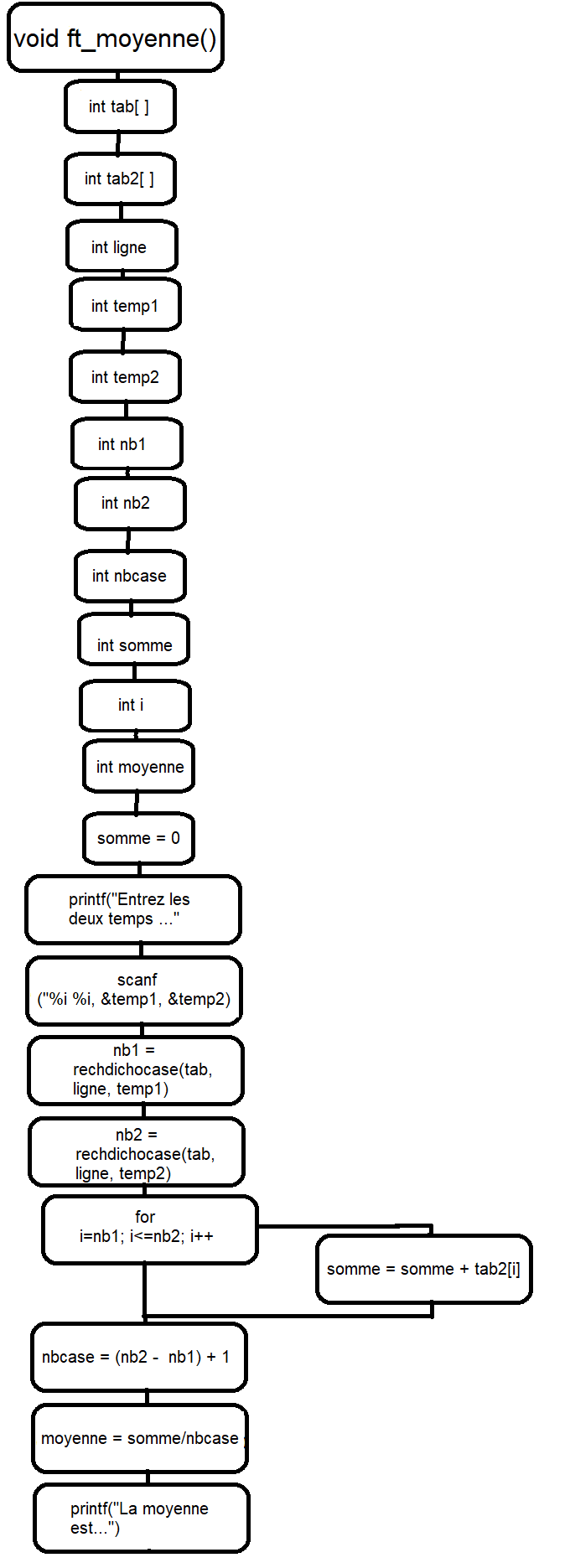
Module2 :



Dans le module2, nous utilisons différentes fonctions qui correspondent aux différents modes d’allumage des LEDs. On commence toujours par la LED 2 car le premier port que nous utilisons est le port2.Dans la fonction chenille1(), comme représenté au-dessus, on initialise des variables puis dans une boucle « for » on allume la première LED puis après un délai et on finit par l’éteindre. La boucle « for » permet de passer à la LED suivante à chaque tour de boucle. Pour la chenille rapide le délai est plus petit. Pour la chenille fondue on allume toute les LEDs une par une puis on les éteint une par une.

Pour la fonction « all », on initialise des variables puis on les allume toutes avec une boucles « for » et on les éteint toutes avec une autres boucles « for ». Pour la fonction unique on choisit la LED que l’on veut allumer et on procède de la même façon.

Pour les fonctions d’alternance, comme alternance2 représentée ici, on initialise des variables puis on en allume une sur deux puis on les éteint toutes et on décale d’une LED et on recommence. Pour alternance 3, on allume qu’une LED sur trois avec le même principe.

Module4 :

Le module 4 est composé de 7 fichiers : main.c, menu.c, menu.h, donnes.c, donees.h, actions.c et actions.h. Un fichier CSV est relié au donnees.c pour pouvoir récupérer les données qu’il contient. Le main.c est un fichier qui contient le menu.h et appelle une fonction qui affiche le menu et toutes les actions réalisables sur le CSV. Le fichier action.c contient toutes les fonctions qui sont appelées dans le menu.c pour réaliser les actions comme le tri sélection.

Dans cette fonction on initialise des variables puis on entre dans une boucle « for » il cherche la valeur maximum grâce à une fonction puis la met tout à droite avec un swap et diminue la taille pour la boucle suivante. Ainsi toutes les valeurs sont dans l’ordre.

Pour la fonction de moyenne, on initialise des variables puis on demande à l’utilisateur entre quelles valeurs de temps il souhaite la moyenne. Après avoir récupérer ces valeurs on additionne toutes les valeurs de pouls entre les valeurs de temps grâce à un « for ». Ensuite on divise par le nombre de case pour obtenir la moyenne et on l’affiche.

1. ***Prototypes de l'ensemble des fonctions du projet (faites-le le plus « graphiquement » possible)***

Module 2:

Module1 :

Cardio.h

Int pouls\_alea() ;

Int timer() ;

Void RenvoiCSV(… ,…) ;

Menu.h

Int choiceParm() ;

Int validation() ;

Module 4 :

codeGeneration.h

void all() ;

void alternance2();

void alternance3( ;

void unique();

void chenille1() ;

void chenille2() ;

void chenille3() ;

Menu.h

Int menulecture() ;

donnees.h

Int ft\_lectureecriture() ;

Int ft\_lecture inverse () ;

Int ft\_lecturerecherche();

In ft\_lectureligne() ;

Int ft\_lectureminmax() ;

Int ft\_lecturemoyenne() ;

action.h

void swap(int \*nb1, int \*nb2);

int ft\_max(int tableau[], int taille);

int ft\_min(int tableau[], int taille);

void tri\_selection(int tableau[], int tableau2[], int taille);

void rechdicho(int tableau[], int ligne);

void ft\_mintableau(int pouls[], int ligne);

void ft\_maxtableau(int pouls[], int ligne);

void ft\_moyenne(int tab[]);

**Quatrième partie : REPARTISSEZ-VOUS LES TACHES**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom : LEGROUX | | | Rôle principal : chef de projet | | | | | | |
| 11/11 | **12/11** | **13/11** | | **14/11** | **15/11** | **16/11** | **17/11** | **18/11** | 19/11 |
| Installer les logiciels nécessaires et prises de connaissances du sujet | Module 1 complété, Module 3 complété ; Début du module 2 | Montage du cœur  Programmation de la génération du code | | Début des livrables de code pour les fonctions de recherches et règlement du problème du csv | Livrables  Code moyenne | Livrables  Préparation soutenance |  |  | Soutenance |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom : GUYOT | | | Rôle principal : Arduino | | | | | | |
| 11/11 | **12/11** | **13/11** | | **14/11** | **15/11** | **16/11** | **17/11** | **18/11** | 19/11 |
| Installer les logiciels nécessaires et prises de connaissances du sujet | Module 1 complété, Module 3 complété ; Début du module 2 | Programmation des différents modes de clignotement du cœur | | Programmation du code du module 4 | Code module 4 | Livrables  Préparation soutenance |  |  | Soutenance |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom : VANMANSART | | | Rôle principal : Montage électronique puis code | | | | | | |
| 11/11 | **12/11** | **13/11** | | **14/11** | **15/11** | **16/11** | **17/11** | **18/11** | 19/11 |
| Installer les logiciels nécessaires et prises de connaissances du sujet | Module 1 complété, Module 3 complété ; Début du module 2 | Montage du cœur  Programmation de la génération du code | | Début des livrables de code pour les fonctions de recherches et règlement du problème du csv | Livrables  Code moyenne | Livrables  Préparation soutenance |  |  | Soutenance |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nom : HOANG | | | Rôle principal : module4 | | | | | | |
| 11/11 | **12/11** | **13/11** | | **14/11** | **15/11** | **16/11** | **17/11** | **18/11** | 19/11 |
| Installer les logiciels nécessaires et prises de connaissances du sujet | Module 1 complété, Module 3 complété ; Début du module 2 | Recherches sur le code du module 1 pour simuler un battement | | Programmation du code du module 4 | Code module 4 | Livrables  Préparation soutenance |  |  | Soutenance |