```
/* PROJET DE SCIENCES DE L'INGÉNIEUR : SIMULATEUR DE VOL

  2016 - 2017  */



/* Ce programme sera chargé dans la maquette */


/* J'inclus la librairie "Servo" qui me permet de contrôler facilement des
servomoteurs */

#include <Servo.h>

/* J'initialise les "instances" associées à mes servomoteurs : une liste de
quatre éléments */

Servo servo[4];

/* J'initialise ensuite les variables générales */

int servoPins[4] = {5 , 6 , 9 , 11}; // les broches reliées aux servomoteurs
int knobPins[4] = {0 , 1 , 2 , 3}; // les broches reliées aux joysticks
int knobVals[4] = {0 , 0 , 0 , 0}; // la liste contenant les valeurs lues par les
joysticks
int led[4] = {2 , 3 , 4 , 7};
int minCentreMax[4][3] = {
  {}, // Romain
  {}, // Maxence
  {10 , 76 , 160}, // Florian (profondeur) /* NB : cette matrice est en « travaux
  » et n'est pas utilisée */
  {15 , 94 , 140}, // Florian (direction)
};
int pos[4] = {90 , 90 , 90 , 90}; // position du servomoteur
int dir[4]; // variable logique déterminant la rotation du servomoteur : 1 si
horaire, 0 si trigonométrique, 2 si nulle
int temps[4]; // variable qui stockera le temps entre deux changements de position

/* Ces variables vont servir à l'exécution du programme (incrément, tampon, etc)
*/

int i;
int y;
int z;

/* La fonction setup() ne s'exécute qu'une fois, à la mise sous tension de
l'Arduino */

void setup() {

  /* J'associe chacune des instances à un port de ma liste servoPins (en suivant
  l'ordre du montage réel) */
  for (i = 0 ; i < 4 ; i++) {
    servo[i].attach(servoPins[i]);
    servo[i].write(minCentreMax[i][1]);
  }
  Serial.begin(9600);
```

```
}

void loop() {
  for (i = 0 ; i < 4 ; i++) {
    knobVals[i] = analogRead(knobPins[i]); // je lis la valeur de tension en sortie du joystick et la stocke dans une variable
    Serial.println(knobVals[i]); // sortie de la valeur sur le port série pour test
    dir[i] = determineDir(knobVals[i]); // puis je détermine dans quel sens le servomoteur doit tourner
    temps[i] = determineTime(knobVals[i]);
  }

/* Puis je fais tourner le servomoteur en fonction de la valeur de la variable dir, et j'allume la LED associée au sens : led[0] si

    trigonométrique, led[1] si horaire */
  for (i = 0 ; i < 4 ; i++) {
    moveServo(dir[i], temps[i], servo[i]);
  }
}

void moveServo(int direction, int delai, Servo a) {

switch (direction) {

  borner(pos[i], minCentreMax[i][0], minCentreMax[i][2]);

  case 0: // si dir = 0
    a.write(pos[i]);
    pos[i]--; // on diminue la position, ce qui fait tourner le servomoteur dans le sens trigonométrique
    digitalWrite(led[i], HIGH);
    delay(delai);
    break;

  case 1: // si dir = 1
    a.write(pos[i]);
    pos[i]++; // on augmente la position, ce qui fait tourner le servomoteur dans le sens horaire
    digitalWrite(led[i], HIGH);
    delay(delai);
    break;

  case 2: // si dir = 2
    a.write(pos[i]); // on ne change pas le position, donc le servomoteur reste fixe
    digitalWrite(led[i], LOW); // et la LED restent éteintes
    break;

  }

}

/* Cette fonction me permet de déterminer la rotation du servomoteur par rapport à la valeur lue en sortie du joystick */

int determineDir(int a) {
```

```
int determineDir(int a) {

  if (a < 450) {
    return 0;

  } else if (a > 630) {
    return 1;

  } else {
    return 2;

  }
}

/* Cette fonction me permet de borner une valeur afin qu'elle ne sorte pas d'un
intervalle */

int borner(int variable, int minimum, int maximum) {

  if (variable <= minimum) {

    variable = minimum;
    return variable;

  }

  else if (variable >= maximum) {

    variable = maximum;
    return variable;

  } else return variable;

}
```