```
#include "mbed.h"

Ticker toggle_BP_ticker; // Déclaration du ticker
DigitalIn  mon_bouton(D7); // Déclaration du bouton poussoir
void toggle_BP(void); // Déclaration de la fonction d'interruption du ticker


// intialisation des pins des deux moteurs

DigitalOut pin_bobine_1_1(D2); // intialisation des pins du moteur 1 (droite) D2 à D5
DigitalOut pin_bobine_1_2(D3);
DigitalOut pin_bobine_2_1(D4);
DigitalOut pin_bobine_2_2(D5);
DigitalOut pin_bobine_3_1(D6); // intialisation des pins du moteur 2(gauche) D6 à D9
DigitalOut pin_bobine_3_2(D7);
DigitalOut pin_bobine_4_1(D8);
DigitalOut pin_bobine_4_2(D9);

void reglage_pins1(int pin1 , int pin2, int pin3, int pin4) {

pin_bobine_1_1 = pin1;
pin_bobine_1_2 = pin2;
pin_bobine_2_1 = pin3;
pin_bonbine2_2 = pin4;

}

void reglage_pins2(int pin5, int pin6, int pin7, int pin8 ) {

pin_bobine_3_1 = pin5;
pin_bobine_3_2 = pin6;
pin_bobine_4_1 = pin7;
pin_bobine_4_2 = pin8;

}
// Fonction qui fait tourner le moteur 1
void prochainStep1(int numero)
{
if (numero == 0) {
reglage_pins1(1, 0, 1, 0); // Position angulaire du moteur1 de 90° dans le sens positif
}
if (numero == 1) {
reglage_pins1(0, 1, 1, 0); // Position angulaire du moteur1 de 180° dans le sens positif
}
if (numero == 2) {
reglage_pins1(0, 1, 0, 1); // Position angulaire du moteur1 de 270° dans le sens positif
}
```

```
if (numero == 3) {
reglage_pins1(1, 0, 0, 1); // Position angulaire du moteur1 de 360° dans le sens positif
}
}
// Fonction qui fait tourner le moteur 1
void prochainStep2(int numero)
{
if (numero == 0) {
reglage_pins2(1, 0, 1, 0); // Position angulaire du moteur2 de 90° dans le sens positif
}
if (numero == 1) {
reglage_pins2(0, 1, 1, 0); // Position angulaire du moteur2 de 180° dans le sens positif
}
if (numero == 2) {
reglage_pins2(0, 1, 0, 1); // Position angulaire du moteur2 de 270° dans le sens positif
}
if (numero == 3) {
reglage_pins2(1, 0, 0, 1); // Position angulaire du moteur2 de 360° dans le sens positif
}
}
// Fonction avancer le robot en avant
void marche_avant(float attente, int nombre_de_pas) {
    for (int i=0; i <= nombre_de_pas; i++) {
          prochainStep1(i % 4);
          prochainStep2(i % 4);
          wait(attente); }
}
// Fonction avancer le robot en arrière
void marche_arriere(float attente, int nombre_de_pas) {
    for (int i=0; i <= nombre_de_pas; i++) {
          prochainStep1(3 - (i % 4));
          prochainStep2(3 - (i % 4));
          wait(attente); }
}
// Fonction tourner le robot à droite
void rotation_de_droite(float attente, int nombre_de_pas) {
    for (int i=0; i <= nombre_de_pas; i++) {
          prochainStep1(i % 4); /* On tourne le moteur 1 dans le sens positif et on fixe
le deuxième moteur */
          prochainStep2(0);
          wait(attente); }
}
// Fonction tourner le robot à gauche
void rotation_de_gauche(float attente, int nombre_de_pas) {
    for (int i=0; i <= nombre_de_pas; i++) {
```

```
            prochainStep1(3 - (i % 4)); /* On tourne le moteur 1 dans le sens négatif et
on fixe le deuxième moteur */
            prochainStep2(0);
            wait(attente); }
}
// Fonction arrêter le robot
void arrêt_du_robot(float attente) {
    nombre_de_pas=0 ; // Arrêter le moteur revient à le faire avancer avec un pas nul
    for (int i=0; i <= nombre_de_pas; i++) {
            prochainStep1(i % 4);
            prochainStep2(i % 4); }
wait(attente);
}

// Test du programme en avançant, reculant et tournant le robot

int main() {

/* Initialisation du ticker attaché à la fonction (toggle_led) et l'intervalle de temps
(en s) pour sortir de la boucle après un certain temps */

toggle_BP_ticker.attach(&toggle_BP, 1);
int a = mon_bouton.read();

while(1) {
// 1000 pas en marche avant, rotation de 20 cm/s
marche_avant(0.157, 1000); /* Une vitesse angulaire de chaque roue autour de son axe de
10rad/s dans le sens avant (1.6 tours/s ou 0.157s pour chaque quart de tour) pour chaque
moteur */
wait(10);

marche_arriere(0.157, 1000); /* Une vitesse angulaire de chaque roue autour de son axe
de 10rad/s dans le sens arrière (1.6 tours/s ou 0.157s pour chaque quart de tour) pour
chaque moteur */
wait(10);

rotation_de_droite(0.157, 1000); /* Une vitesse angulaire du robot autour de lui même de
0.334rad/s dans le sens des aiguilles pour une vitesse de 20cm/s du moteur 1 (sens
positif) */
wait(10);

rotation_de_gauche(0.157, 1000); /* Une vitesse angulaire du robot autour de lui même de
0.334rad/s dans le sens trigonométrique pour une vitesse de 20cm/s du moteur 1 (sens
négatif) */
wait(10);
}
void toggle_BP() { // Fonction d'interruption du ticker
a = !a; }
```