```
#include <EEPROM.h> //Including EEPROM library

#include <Zumo32U4.h>

#include "TurnSensor.h"




Zumo32U4ButtonA buttonA;

Zumo32U4ButtonB buttonB;

Zumo32U4ButtonC buttonC;

Zumo32U4LCD lcd;

Zumo32U4Motors motors;

Zumo32U4LineSensors linesensor;

Zumo32U4Buzzer buzzer;

L3G gyro;

Zumo32U4ProximitySensors proxSensors;


//Song that the zumo will play under the sensor show.

const char fugue[] PROGMEM =

  "! O5 L16 agafaea dac+adaea fa<aa<bac#a dac#adaea f"

  "O6 dcd<b-d<ad<g d<f+d<gd<ad<b- d<dd<ed<f+d<g d<f+d<gd<ad"

  "L8 MS <b-d<b-d MLe-<ge-<g MSc<ac<a ML d<fd<f O5 MS b-gb-g"

  "ML >c#e>c#e MS afaf ML gc#gc# MS fdfd ML e<b-e<b-"

  "O6 L16ragafaea dac#adaea fa<aa<bac#a dac#adaea faeadaca"

  "<b-acadg<b-g egdgcg<b-g <ag<b-gcf<af dfcf<b-f<af"

  "<gf<af<b-e<ge c#e<b-e<ae<ge <fe<ge<ad<fd"

  "O5 e>ee>ef>df>d b->c#b->c#a>df>d e>ee>ef>df>d"

  "e>d>c#>db>d>c#b >c#agaegfe f O6 dc#dfdc#<b c#4";




//Creating int value for sensvaluer
```

```
unsigned int linesensorValues[5];


int account_balance = EEPROM.read(0);
const int money_deposit = 30; // Fixed amount of money to deposit (when e.g pushing button)
//Creating global variables
const double P = 0.3;
const double D = 8;
double lastE = 0;
const unsigned char maxSpeed = 200;
int tapeNum = 0;
bool linePID = false;
bool lineSTD = false;
int stepNumConeDrive = 0;
int coneNum = 0;
bool runGyro = false;
int lastDir = 0;
unsigned long followTime = 10000;
unsigned long startTime;
bool followMe = true;
String myString = "Follow";


// --- Helper functions
int32_t getAngle() {
  // turnAngle is a variable defined in TurnSensor.cpp
  // This fancy math converts the number into degrees turned since the
  // last sensor reset.
  return (((int32_t)turnAngle >> 16) * 360) >> 16;
}
```

```
int32_t angle = getAngle();



int menu = 1;

void setup() {
  //Iniates the LCD and linesensors
  lcd.init();
  linesensor.initFiveSensors();
  proxSensors.initThreeSensors();
  //Sends you to the start of the mnu
  updateMenu();
}

void loop() {
  //Button A sends you "downwards" on the menu
  if (buttonA.getSingleDebouncedPress()){
    menu++;
    updateMenu();
    delay(100);
  }
  //Button C sends you "upwards" on the manu
  if (buttonC.getSingleDebouncedPress()){
    menu--;
    updateMenu();
    delay(100);
  }
  //Button B sends you one step depper in the menu
  if (buttonB.getSingleDebouncedPress()){
    executeAction();
    updateMenu();
    delay(100);
  }
}

//This is a function that updates what you see on the LCD
//If you go "out" of the menu area, you are sent back into the last place you were
//The menu has 3 main menus, Autonom, Line and Account. In each of these you have the undermenu
//In the under menu you choose what "action" the Zumo are going to execute
void updateMenu() {
  switch (menu) {
    case 0:
      menu = 1;
      break;
    case 1:
      lcd.clear();
      lcd.print(">Autonom");
      lcd.gotoXY(0, 1);
      lcd.print("Line");
      break;
    case 2:
      lcd.clear();
      lcd.print(">Line");
      lcd.gotoXY(0, 1);
      lcd.print("Account");
      break;
    case 3:
      lcd.clear();
      lcd.print(">Account");
      lcd.gotoXY(0, 1);
      lcd.print(" ");
      break;
    case 4:
      menu = 3;
      break;
    case 9:
      menu = 10;
      break;
    case 10:
      lcd.clear();
      lcd.print(">Straight");
      lcd.gotoXY(0,1);
      lcd.print("Square");
      break;
    case 11:
      lcd.clear();
      lcd.print(">Square");
      lcd.gotoXY(0,1);
      lcd.print("Circle");
      break;
    case 12:
      lcd.clear();
      lcd.print(">Circle");
      lcd.gotoXY(0,1);
      lcd.print("Cone");
      break;
    case 13:
      lcd.clear();
      lcd.print(">Cone");
      lcd.gotoXY(0,1);
      lcd.print("Show");
      break;
    case 14:
      lcd.clear();
      lcd.print(">Show");
      lcd.gotoXY(0,1);
      lcd.print("Back");
      break;
    case 15:
      lcd.clear();
      lcd.print(">Back");
      break;
    case 16:
      menu = 15;
      break;
    case 19:
      menu = 20;
      break;
    case 20:
      lcd.clear();
      lcd.print(">Standard");
      lcd.gotoXY(0,1);
      lcd.print("PID");
      break;
    case 21:
      lcd.clear();
      lcd.print(">PID");
      lcd.gotoXY(0,1);
      lcd.print("Calib");
      break;
    case 22:
      lcd.clear();
      lcd.print(">Calib");
      lcd.gotoXY(0,1);
      lcd.print("Back");
      break;
    case 23:
      lcd.clear();
      lcd.print(">Back");
      lcd.gotoXY(0,1);
      lcd.print(" ");
      break;
    case 24:
      menu = 23;
      break;
    case 29:
      menu = 30;
      break;
    case 30:
      lcd.clear();
      lcd.print(">View acc");
      lcd.gotoXY(0,1);
      lcd.print("++Cash");
      break;
    case 31:
          lcd.clear();
          lcd.print(">++Cash");
          lcd.gotoXY(0,1);
          lcd.print("Back");
          break;
        case 32:
          lcd.clear();
          lcd.print(">Back");
          lcd.gotoXY(0,1);
          lcd.print(" ");
          break;
        case 33:
          menu = 32;
      }
    }

    //This function executes the menu that you have chosen
    void executeAction() {
      switch (menu) {
        case 1:
          action1();
          break;
        case 2:
          action2();
          break;
        case 3:
          action3();
          break;
        case 10:
```

```
  action10();
  break;
case 11:
  action11();
  break;
case 12:
  action12();
  break;
case 13:
  action13();
  break;
case 14:
  action14();
  break;
case 15:
  action15();
  break;
case 20:
  action20();
  break;
case 21:
  action21();
  break;
case 22:
  action22();
  break;
case 23:
  action23();
  break;
```

```
    case 30:
      action30();
      break;
    case 31:
      action31();
      break;
    case 32:
      action32();
      break;



  }
}
//This function sends you to the undermenu of "Autonom"
void action1() {
  lcd.clear();
  menu = 10;
  }
//This function sends you to the undermenu of "Line"
void action2() {
  lcd.clear();
  menu = 20;
}
//This function send you to the undermenu of "Account"
void action3() {
  lcd.clear();
  menu = 30;
}
```

```
//Function for driving straight forward, stop, turn 180 degree and drive back

//This function cost 10 "units", if you dont have enough in your account the LCD will let you know and you have to fill up your account first.

void action10(){

 if ( account_balance >= 10){

   account_balance -= 10;

   EEPROM.write(0, account_balance);

   lcd.clear();

   lcd.print("Driving");

   turnSensorSetup();

   delay(500);

   turnSensorReset();

   motors.setSpeeds(200, 200); // drives forward for 2 second

   delay(2000);

   motors.setSpeeds(0, 0); // stops car 100 ms to give motors a break

   delay(50);

   motors.setSpeeds(150, -150);// turning the car 180 degrees

   turnSensorUpdate();

   angle = getAngle();

   while ( angle >= -180 ){

     turnSensorUpdate();

     angle = getAngle();

     if ( angle <= -180) break;

   }

   motors.setSpeeds(0, 0); //stops car for 100 ms

   delay(100);

   motors.setSpeeds(200, 200); // drives back to start

   delay(2000);
```

```
    motors.setSpeeds(0, 0); // stops car
  }
  else{
    lcd.clear();
    lcd.print("To low");
    lcd.gotoXY(0,1);
    lcd.print("balance");
    delay(2000);
  }
}


//Function for the Zumo to drive in a square, returning where it started
//This function cost 10 "units", if you dont have enough in your account the LCD will let you know and
you have to fill up your account first.


void action11(){
  if ( account_balance >= 10){
    account_balance -= 10;
    EEPROM.write(0, account_balance);
    lcd.clear();
    lcd.print("Turning");
    turnSensorSetup();
    delay(500);
    turnSensorReset();
    for(int i=0;i<4;i++){
      motors.setSpeeds(200, 200); // drives forward
      delay(500);
      motors.setSpeeds(0, 0); // stops car
      delay(50);
      turnSensorReset();
      motors.setSpeeds(150, -150);// turning the car 180 degrees
      turnSensorUpdate();
      angle = getAngle();
      while ( angle >= -90 ){
        turnSensorUpdate();
        angle = getAngle();
        if ( angle <= -90) break;
     }
      motors.setSpeeds(0,0);
      delay(500);
    }
  }
  else{
    //Insufficent funds, fill upp account
    lcd.clear();
    lcd.print("To low");
    lcd.gotoXY(0,1);
    lcd.print("balance");
    delay(2000);
  }
}
```

//Function for the Zumo to drive in a circle

//This function cost 10 "units", if you dont have enough in your account the LCD will let you know and you have to fill up your account first.

```
void action12(){
  if ( account_balance >= 10){
    account_balance -= 10;

    EEPROM.write(0, account_balance);

    lcd.clear();

    lcd.print("Circling");

    turnSensorSetup();

    delay(500);

    turnSensorReset();

    motors.setSpeeds(80, 200); //Start turning

    delay(50);

    turnSensorUpdate();

    angle = getAngle();

    while ( angle != 0){

      turnSensorUpdate();

      angle = getAngle();

    }

    motors.setSpeeds(0, 0); // stops car entirely

  }

  else{

    lcd.clear();

    lcd.print("To low");

    lcd.gotoXY(0,1);

    lcd.print("balance");

    delay(2000);

  }

}
```

//Function for drive for cone driving

//This function cost 10 "units", if you dont have enough in your account the LCD will let you know and you have to fill up your account first.

```
void action13(){
 if ( account_balance >= 10){
   account_balance -= 10;
   EEPROM.write(0, account_balance);
   lcd.gotoXY(0,0);
   lcd.print("Gyro");
   lcd.gotoXY(0,1);
   lcd.print("Calib");
   turnSensorSetup();
   delay(500);
   turnSensorReset();
   lcd.clear();
   lcd.gotoXY(0,0);
   lcd.print("Press B");
   lcd.gotoXY(0,1);
   lcd.print("to start");
   buttonB.waitForPress();
   runGyro = true;
   stepNumConeDrive = 0;
   while ( runGyro ){
     turnSensorUpdate(); //Updates sensor
     angle = getAngle(); //Gets the angle value
     coneDrive();       //Calling the actual cone driving function
     // Update the display
     lcd.gotoXY(0, 0);
     lcd.print(angle);
     lcd.print(" ");
   }
 }
```

```
  else{
    //Insufficent funds
    lcd.clear();
    lcd.print("To low");
    lcd.gotoXY(0,1);
    lcd.print("balance");
    delay(2000);
  }
}
//Cone drive function for driving through 4 cones and ending straight
void coneDrive (){
  //Starts by turning to the left to 55 degrees
  while ( stepNumConeDrive == 0){
    motors.setSpeeds(-100, 100);
    turnSensorUpdate();
    angle = getAngle();
    lcd.gotoXY(0, 0);
    lcd.print(angle);
    lcd.print(" ");
    if (angle >= 55){
      motors.setSpeeds(0,0);
      delay(20);
      stepNumConeDrive = 1;
      break;
    }
  }
//If it has driven past 3 cones it will know to stop when it has reached 0 degree
  if (coneNum >= 3){
    motors.setSpeeds(100,200);
    if ( angle >= 0){
      motors.setSpeeds(0,0);
      stepNumConeDrive = 0;
      coneNum = 0;
      runGyro = false;
    }
  }
//Turning to the right
  else if (stepNumConeDrive == 1){
    motors.setSpeeds(200, 100);
    if ( angle <= -55){
      stepNumConeDrive = 2;
      coneNum += 1;

    }
  }
//Turning to the left
  else if (stepNumConeDrive == 2){
    motors.setSpeeds(100,200);
    if (angle >= 55){
      stepNumConeDrive = 1;
    }
  }
}

//Function for our sensorshow. This segment of code will first follow you.
//After 10 seconds the Zumo will stop and start turning towards you, while standing at the same spot.
//The zumo will stop the show when the song is finished.
void action14(){
```

```
if ( account_balance >= 10){
 account_balance -= 10;
 buzzer.playFrequency(440, 200, 15);
 delay(1000);//Gives the user a second before the Zumo drives
 buzzer.playNote(NOTE_A(4), 2000, 15);
 delay(200);
 buzzer.stopPlaying();
 delay(1000);
 while(buzzer.isPlaying());
 buzzer.playFromProgramSpace(fugue); //Plays sound program from
 startTime = millis();
 while(buzzer.isPlaying()){ //Sensorshow is on while the song is playing
  //Reads proxsensors
  proxSensors.read();
  int cent_left = proxSensors.countsFrontWithLeftLeds(); //Stores cent left prox sensor
  int cent_right = proxSensors.countsFrontWithRightLeds(); //Stores cent right prox sensor
 //Every 10 second the Zumo changes mode from following to turning
  if ( (millis()-startTime) >= followTime){
   if( followMe){
   followMe = false;
   myString = "Turn";
   ledRed(1);
   ledGreen(0);
   }
   else{
    followMe = true;
    myString = "Follow";
    ledRed(0);
    ledGreen(1);
    }
    startTime = millis();   //Resets the timer since last change
  }
  //Prints what "mode" the Zumo is in and the values from the sensors
  lcd.clear();
  lcd.gotoXY(0,0);
  lcd.print(myString);
  lcd.gotoXY(0,1);
  lcd.print(" ");
  lcd.print(cent_left);
  lcd.print(" ");
  lcd.print(cent_right);
  lcd.print(" ");
  //Calls on function for following if in follow mode
  if ( followMe) follower(cent_left, cent_right);
  //Calls on function for turning if not in follow mode
  else if( !followMe ) turner(cent_left, cent_right);
  delay(100);


}
//Stops the Zumo and indicates that its going back to the menu
motors.setSpeeds(0,0);
ledRed(0);
ledGreen(0);
lcd.clear();
lcd.gotoXY(0,0);
lcd.print("Back to");
lcd.gotoXY(0,1);
```

```
    lcd.print("meny");
    delay(1000);
  }

  //Insufficent funds
  else{
    lcd.clear();
    lcd.print("To low");
    lcd.gotoXY(0,1);
    lcd.print("balance");
    delay(2000);
  }
}
//Function for following
void follower(int myCentLeft, int myCentRight){
  //Turning left and indicating last direction is left
  if ( myCentLeft > myCentRight){
    motors.setSpeeds(50, 150);
    lastDir = 1;
  }
  //Turning right and indicating last direction is right
  else if (myCentRight > myCentLeft){
    motors.setSpeeds(150, 50);
    lastDir = 2;
  }
  //Driving straight and indicating last directiong is straight
  else if ( myCentRight == myCentLeft){
    motors.setSpeeds(100, 100);
    lastDir = 0;
```

```
 }
}

void turner(int myCentLeft, int myCentRight){
 //Rotating left and indicating last direction is left
 if ( myCentLeft > myCentRight ){
  //If the Zumo was rotating to the right it stops first before continueing rotating to the left
  if ( lastDir == 2){
     motors.setSpeeds(0,0);
     delay(20);
  }
  motors.setSpeeds(-150, 150);
  lastDir = 1; //Indicating rotating to left
 }
 //Rotating right and indicating last direction is right
 else if ( myCentRight > myCentLeft){
  //If the Zumo was rotating to the left it stops first before continueing rotating to the right
  if ( lastDir == 1){
    motors.setSpeeds(0,0);
     delay(20);
  }
  motors.setSpeeds(150, -150);
  lastDir = 2; //Indicating rotating to right
 }

 else{
  motors.setSpeeds(0,0);
  lastDir = 0; //Indicating stop
 }
```

```
}


//This function sends you back to the main menu
void action15(){
  lcd.clear();
  menu = 1;
}


//Function for the Zumo to drive "linesensor" without PD - regulator.
//The zumo follows the line until the "B" button on the car is pressed
void action20(){
  if ( account_balance >= 10){
    account_balance -= 10;
    EEPROM.write(0, account_balance);
    lcd.clear();
    lineSTD = true;
    tapeNum = 0;
    delay(1000);
    while ( lineSTD ){
      int position = linesensor.readLine(linesensorValues);//Reads the value of the linesensors
      bool myTape = false;
      //Breaks out of the "while" loop and sends you back to the menu
      if ( buttonB.isPressed()){
        lineSTD = false;
        break;
      }
       //When all of the linesensor reads a higher value than 800, the zumo is now either on a dead end,
theres missing tape or the zumo has been lifted up.
```

```
      if ( linesensorValues[0] >= 800 && linesensorValues[1] >= 800 && linesensorValues[2] >= 800 &&
linesensorValues[3] >=800 && linesensorValues[4] >=800 ){

        motors.setSpeeds(0,0);  //Stops the car

        delay(500);             //A small delay so you can press the "B" button to stop the linefollower

        myTape = true;

        if ( buttonB.isPressed()){

          lineSTD = false;

          break;

        }

      }

      //Prints linesensors value on lcd

      lcd.gotoXY(0,0);

      lcd.print(position);

      //Using function to choose motorpower

      direct(position, myTape, tapeNum);

      delay(50);

    }

    //Stops the car and goes back to main manu

    motors.setSpeeds(0,0);

    lcd.clear();

    lcd.gotoXY(0,0);

    lcd.print("Back to");

    lcd.gotoXY(0,1);

    lcd.print("Meny");

    delay(1000);

    menu = 1;

  }

  else{

    //To low balance in account.
    lcd.clear();
    lcd.print("To low");
    lcd.gotoXY(0,1);
    lcd.print("balance");
    delay(2000);
  }
}

//Function that decides what motorpower the linefollower function without PD - regulator has
void direct(int x, bool myTape, int Num ){
  //Missing tape, the zumo drives straight for 2 seconds with reduced speed
  if ( myTape && Num == 0 ){
    motors.setSpeeds(110,100);
    delay(2000);
    tapeNum = 1; // Indicating that it has come back on track
  }
  //Dead end, the zumo drives back into the track
  else if ( myTape && Num == 1 ){
    motors.setSpeeds(0,0);// Stops the zumo
    delay(200);
    motors.setSpeeds(100, -100);//Turns the zumo 180 degree
    delay(1450);
    motors.setSpeeds(100,100);//Drive back to the turn
    delay(1500);
    motors.setSpeeds(0,0);//Stops the zumo
    delay(20);
    motors.setSpeeds(100, -100);//Turns 90 degree
    delay(800);
    motors.setSpeeds(0,0);//Stops the zumo
    delay(20);
    motors.setSpeeds(100,100);//Continues on the track
    tapeNum = 0;  // Indicating that has come back on the track
  }
  //The zumo has different motorpowers depending of what the linesensor reads.
  //If it reads under 2000, it will give more motorpower to the right wheel
  //If it reads over 2000 it will read more motorpower to the left wheel
  else if( x < 1500){
    motors.setSpeeds(0,175);
  }
  else if( x < 1600){
    motors.setSpeeds(20,175);
  }
  else if( x < 1700){
    motors.setSpeeds(40,175);
  }
  else if( x < 1800){
    motors.setSpeeds(50,175);
  }
  else if( x > 2500){
    motors.setSpeeds(175, 0);
  }
  else if( x > 2400){
    motors.setSpeeds(175, 20);
  }
  else if( x > 2300){
    motors.setSpeeds(175, 40);
  }
  else if( x > 2200){
    motors.setSpeeds(175, 50);
  }
  //Straight forward
  else{
    motors.setSpeeds(150,150);
  }

}
//Function for the Zumo to drive line follower with PD - regulator
//This function cost 10 "units", if you dont have enough in your account the LCD will let you know and
you have to fill up your account first.
void action21(){
  if ( account_balance >= 10){
    account_balance -= 10;
    EEPROM.write(0, account_balance);
    lcd.clear();
    linePID = true;
    tapeNum = 0;
    delay(1000);
    while ( linePID ){
      //Goes out of the function if the "B" button is pressed
      if ( buttonB.isPressed()){
        linePID = false;
        break;
      }
      //Reads linesensor value and "error"
      int position = linesensor.readLine(linesensorValues);
      int e = position - 2000;  // calculates the error
      //Calculating speed difference & setting speed
      int u = P * e + D *(e-lastE);
      lastE = e;              //Saves the last error
      int leftSpeed = (int)maxSpeed + u;
      int rightSpeed = (int)maxSpeed - u;
      //Contraining our motors between 0 and maxspeed
      leftSpeed = constrain(leftSpeed, 0, (int)maxSpeed);
      rightSpeed = constrain(rightSpeed, 0, (int)maxSpeed);
       //When all of the linesensor reads a higher value than 800, the zumo is now either on a dead end,
theres missing tape or the zumo has been lifted up.
      if ( linesensorValues[0] >= 800 && linesensorValues[1] >= 800 && linesensorValues[2] >= 800 &&
linesensorValues[3] >=800 && linesensorValues[4] >=800){
        motors.setSpeeds(0,0); // stops the zumo
        delay(500);           // a small delay to let you press the "B" button to get out of the function
        if ( buttonB.isPressed()){
        linePID = false;
        break;
      }
      //Calling the function for either driving over missing tape or dead end.
        blackTape();
       }
      //Motor output
      motors.setSpeeds(leftSpeed, rightSpeed);
      //Prints linesensors on lcd
      lcd.print(position);
      lcd.gotoXY(0,0);
  }
  //Stops the Zumo and sends it back to main menu
    motors.setSpeeds(0,0);
    lcd.clear();
```

```
    lcd.gotoXY(0,0);
    lcd.print("Back to");
    lcd.gotoXY(0,1);
    lcd.print("Meny");
    delay(1000);
    menu = 1;
  }
  else{
    //To low balance in account.
    lcd.clear();
    lcd.print("To low");
    lcd.gotoXY(0,1);
    lcd.print("balance");
    delay(2000);
  }
}

void blackTape(){
  //Missing tape, it will now drive straight with reduced speed
  if( tapeNum == 0  ){
    motors.setSpeeds(110,100);
    delay(2000);
    tapeNum = 1; //Indicating that it has come over missing tape
  }
  //Dead end, the zumo drives back into the track
  else if ( tapeNum == 1){
    motors.setSpeeds(0,0);// Stops the zumo
    delay(200);
    motors.setSpeeds(100, -100);//Turns the zumo 180 degree
    delay(1550);
    motors.setSpeeds(0,0);//Stop the zumo
    delay(20);
    motors.setSpeeds(100,100);//Drive back to the turn
    delay(1500);
    motors.setSpeeds(0,0);//Stops the zumo
    delay(20);
    motors.setSpeeds(100, -100);//Turns 90 degree
    delay(900);
    motors.setSpeeds(0,0);//Stops the zumo
    delay(20);
    motors.setSpeeds(100,100);//Continues on the track
    tapeNum = 0; // Indicating that has come back on the track
  }
}


//Function for calibrating the line sensors
//This function cost 10 "units", if you dont have enough in your account the LCD will let you know and you have to fill up your account first.
void action22(){
  if ( account_balance >= 10){
    account_balance -= 10;
    EEPROM.write(0, account_balance);
    lcd.clear();
    lcd.print("Cali -");
    lcd.gotoXY(0,1);
    lcd.print("brating");
    delay(1000);
    int i = 0;
    while(i < 100){
     linesensor.calibrate();
     motors.setSpeeds(150, -150);
     i++;
    }
    motors.setSpeeds(0,0); // Stops the Zumo
    buzzer.play(">g32>>c32"); //Buzzer for letting you know its done calibrating
    lcd.clear();
  }
  else{
    //To low balance in account.
    lcd.clear();
    lcd.print("To low");
    lcd.gotoXY(0,1);
    lcd.print("balance");
    delay(2000);
  }
}
//Function for sending you back to main menu
void action23(){
  lcd.clear();
  menu = 1;
}
//Function for reading the Account Balance
void action30(){
  lcd.clear();
  lcd.print("Balance:");
  lcd.gotoXY(0,1);
  lcd.print(EEPROM.read(0));
  delay(2000);
}
//Function for filling up your Account Balance
void action31(){
  lcd.clear();
    //Max balance is 255 and will let you know if its full
    if ( (account_balance + money_deposit) <= 255){
      account_balance += money_deposit;
      EEPROM.write(0, account_balance);
      lcd.clear();
      lcd.gotoXY(0,0);
      lcd.print("Balance:");
      lcd.gotoXY(0,1);
      lcd.print(EEPROM.read(0));
      delay(1000);
  }
    else {
      account_balance = 255;
      EEPROM.write(0, account_balance);
      lcd.clear();
      lcd.print("Account");
      lcd.gotoXY(0,1);
      lcd.print("is full!");
      delay(2000);
  }
}
//Function for sending you back to main menu
void action32(){
  lcd.clear();
  menu = 1;
}
```