```
#include <Wire.h>
#include "rgb_lcd.h"
#include "HX711.h"

HX711 loadcell;
rgb_lcd lcd;
int n = 0;
volatile byte state = HIGH;
int run_pump = 0;

const int LOADCELL_DOUT_PIN = 7; // plug DOUT pin on pin 7
const int LOADCELL_SCK_PIN = 8; // plug SCK pin on pin 8
const int dirPin = 10;
const int stepPin = 11;

long LOADCELL_DIVIDER = 1680; // adjust the loadcell divider by calibrating with a known weight
float amount = 0;
int isBig;
float reading;
int enable_pin = 5;

void setup() {

  Serial.begin(9600);
  pinMode(2, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);
  pinMode(13, OUTPUT);
  pinMode(stepPin, OUTPUT);
  pinMode(dirPin, OUTPUT);
  pinMode(enable_pin, OUTPUT);

  attachInterrupt(digitalPinToInterrupt(2), change, LOW);
  attachInterrupt(digitalPinToInterrupt(3), choose, LOW);


  lcd.begin(16, 2);
  lcd.clear();
  // Set the spinning direction CW/CCW:
  digitalWrite(dirPin, LOW);
  scale_setup();

}

void loop() {
  float value = loadcell.get_value() / LOADCELL_DIVIDER;
  Serial.println(value);
  if (value > 222 && value < 232){
    isBig = 1;
    run_lcd();
  }
  else if (value > 258 && value < 268){
    isBig = 0;
    run_lcd();
  }
  else {
    run_pump = 0;
    lcd.clear();
    lcd.print("Place the glass ");
    lcd.setCursor(0,1);
    lcd.print("on the platform");
  }
  delay(1000);
}

void run_lcd(){
  if (run_pump == 0) {
    // Start screen
    if (n == 0) {
      lcd.clear();
      lcd.print("Press 'select' ");
      lcd.setCursor(0,1);
      lcd.print("to start the machine!");
    }

    else {
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("Number of drink: ");
      lcd.print(n);
      delay(500);    // removes the flashing of the screen during activity
    }

  }

  else {
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Is prepared!");
    lcd.print(n);
    pump_liquid(amount);
    delay(5000);
  }
}

void(*resetFunc) (void) = 0; // can be called after the drink is finished

void change() {
  static unsigned long last_interrupt_time = 0;
  unsigned long interrupt_time = millis();
  // if interruptions come faster than 200ms, assume it's a bounce and ignore
  if (interrupt_time - last_interrupt_time > 100) {
    n++;
    if (n > 4) { // number of drink options
      n=1;
    }
    last_interrupt_time = interrupt_time;
  }
}

void choose() { // allows the loop to access the function for pumping
```

```
    run_pump = 1;
    if (isBig == 1) {
      amount = 460;
    }
    else if (isBig == 0) {
      amount = 470;
    }
}

void pump_liquid(float amount) { // runs the pump until the desired amount of liquid weight has been reached
  Serial.print(amount);
  int accelerate = 2000;
  for (int i = 0; i<= 350; i++){ // accelerates the pump
    digitalWrite(enable_pin, HIGH);
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(accelerate);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(accelerate);
    accelerate = accelerate - 4;
  }

  reading = loadcell.get_value() / LOADCELL_DIVIDER;
  Serial.print(reading);
  int k = 0;
  while (reading < amount && reading > 1){
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(600);
    digitalWrite(stepPin, LOW);
    delayMicroseconds(600);

    if (k > 100){
      reading = loadcell.get_value() / LOADCELL_DIVIDER;
      k = 0;
    }
    k++;
  }
  digitalWrite(enable_pin, LOW);

  terminateProcess();

}
void terminateProcess() {
  while (reading > 1) {
  lcd.clear();
  lcd.print("The drink is ready!");
  lcd.setCursor(0,1);
  lcd.print("You can take your drink.");
  reading = loadcell.get_value() / LOADCELL_DIVIDER;
  delay(500);
  }
  run_pump = 0;
  resetFunc();

}
```

```
void scale_setup() {
  // run this function in the setup function of the full code
  loadcell.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN); // starts measuring the weight
  // loadcell.set_scale(LOADCELL_DIVIDER); // sets the calibration factor for the scale object
  loadcell.tare();
}
```