```
#include <LiquidCrystal.h>

/*** INITIALISIERUNG ***/
LiquidCrystal lcd(8, 9, 2, 3, 4, 5);  // set up LCD
const int buttons[] = {33, 35, 37};
const int led       = 31;
const int disp      = 51;

/*** GETRAENKELISTE ***/
int DRINKS = 9;
char *drink_name[] = { "Leitungs Wasser", "Wasser", "Schweppes", "Limo", "Bier", "Cola"
int  drink_price[] = { 50, 100, 450, 150, 250, 300, 300, 200, 900 }; //Preise in Cent
int  drink_storage[] = { 5, 4, 2, 1, 1, 1, 1, 1, 1 };

/*** HAUPMENUE NAVIGATION ***/
int selected_item_in_menu = 0; //Getraenke ID (enspricht Array Position)
int menu_cursor_position  = 0; //0 oder 1, Zeilenauswahl des Displays

/*** Pay menu stuff ***/
int toPay = 0;
long errorLed = 0;
long errorDisp = 0;

/*** counter for last input to reset to default ***/
long idleCheck = 0;

/*** TASTER ENTPRELLEN ***/
unsigned long last_select_time[3] = {0};
int pushed[3] = {0};

/*** STATEMASHINE IN DER LOOP ***/
int active_menu = 0;

/*** CUSTOM CHARS ***/
//Custom Char generiert auf: http://fusion94.org/lcdchargen/

// Arrow sign
byte custom_char_1[8] = {
  0b00000,
  0b00000,
  0b01000,
  0b01100,
  0b01110,
  0b01100,
  0b01000,
  0b00000
};

// Euro sign
byte custom_char_2[8] = {
  0b01110,
         0b10001,
         0b11100,
         0b10000,
         0b11100,
```

```
        0b10001,
        0b01110,
        0b00000
};
/*******************************************************/
/*** FUNKTIONEN DEFINIEREN                           */
/*******************************************************/

/*
   Gibt Wert zurueck, der in gegebenen Grenzen liegt
*/
int limit (int value, int min, int max) {
  if (value < min)
    return min;
  else if (value > max)
    return max;

  return value;
}

/*
    Hauptmenue auf dem Display anzeigen, abhaengig von Cursorposition
    mittels [selected_item_in_menu - menu_cursor_position (+ 1)] wird dafuer gesorgt,
    dass das Display sich nur "verschiebt", wenn Cursor nicht mehr bewegt werden kann
*/
void print_menu() {
  Serial.println("print menu start");
  char buffer[100]; //buffer fuer sprintf

  lcd.createChar(1, custom_char_2); //definiere ein chustomChar (Anzeigen mit Serial.wr

  lcd.clear();

  lcd.setCursor(2, 0); //Setze Curser auf Anfangsposition des oberen Namens
  lcd.print(drink_name[selected_item_in_menu - menu_cursor_position]);
  lcd.setCursor(11, 0);  //Setze Curser auf Anfangsposition des oberen Preises
  if (drink_storage[selected_item_in_menu - menu_cursor_position] > 0){
    sprintf(buffer, "%d,%02d",
            drink_price[selected_item_in_menu - menu_cursor_position] / 100,
            drink_price[selected_item_in_menu - menu_cursor_position] % 100);
    lcd.print(buffer);
    lcd.write(byte(1));
  } else {
    lcd.setCursor(13, 0);
    lcd.print("---");
  }

  lcd.setCursor(2, 1);  //Setze Curser auf Anfangsposition des unteren Namens
  lcd.print(drink_name[selected_item_in_menu - menu_cursor_position + 1]);
  lcd.setCursor(11, 1);  //Setze Curser auf Anfangsposition des unteren Preises
  if (drink_storage[selected_item_in_menu - menu_cursor_position + 1] > 0){
    sprintf(buffer, "%d,%02d",
            drink_price[selected_item_in_menu - menu_cursor_position + 1] / 100,
            drink_price[selected_item_in_menu - menu_cursor_position + 1] % 100);
    lcd.print(buffer);
```

```
    lcd.write(byte(1));
  } else {
    lcd.setCursor(13, 1);
    lcd.print("---");
  }



  //setze Auswahldreieck
  lcd.setCursor(0, menu_cursor_position);
  lcd.write(byte(0)); //die ID des CustomChars (0) muss erst als byte geparsed werden
}



/*
    print menu auf dem Display anzeigen, I decided to print no negative numbers here
    makes sense in my opinion and it does not imply that i give any change :)
*/
void print_pay_menu() {
  Serial.println("print pay menu start");
  char buffer[100]; //buffer fuer sprintf

  lcd.clear();

  lcd.setCursor(2, 0); //Setze Curser auf Anfangsposition des oberen Namens
  lcd.print(drink_name[selected_item_in_menu]);
  sprintf(buffer, "%d,%02d",
          drink_price[selected_item_in_menu] / 100,
          drink_price[selected_item_in_menu] % 100);
  lcd.setCursor(11, 0);  //Setze Curser auf Anfangsposition des oberen Preises
  lcd.print(buffer);
  lcd.write(byte(1));

  lcd.setCursor(2, 1);  //Setze Curser auf Anfangsposition des unteren Strings
  lcd.print("left: ");
  sprintf(buffer, "%d,%02d",
          toPay / 100,
          toPay % 100);

  lcd.setCursor(11, 1);  //Setze Curser auf Anfangsposition des left to pay
  lcd.print(buffer);
  lcd.write(byte(1));
}



/*
   Gibt Button Input entprellt als int zurueck, beachte Invertierung der Logik durch INP
   0: button_up
   1: button_down
   2: button_ok
*/
```

```
int read_input () {
  for (int i = 0; i < 3; i++) {
    if (millis() - last_select_time[i] > 25) {
      if (digitalRead(buttons[i]) == LOW && !pushed[i]) {
        pushed[i] = 1;
        last_select_time[i] = millis();

        // reset idle counter
        Serial.println("idle reset bc input");
        idleCheck = millis();

        return i;
      } else if (digitalRead(buttons[i]) == HIGH && pushed[i]) {
        pushed[i] = 0;
        last_select_time[i] = millis();
      }
    }
  }
  return -1;
}

/*
   Aendert Pay menu Variablen je nach Eingabe ab, wird nur aufgerufen, wenn gerade das
*/
void compute_pay_input(int input) {
  // change toPay to never be below 0
  if (input == 0)
    toPay -= 50;
  else if (input == 1)
    toPay -= 100;
  if (toPay < 0)
    toPay = 0;
}


/*
   Aendert Haupmenue Variablen je nach Eingabe ab, wird nur aufgerufen, wenn gerade das
   Sorgt ausserdem dafuer, dass der Cursor zuerst hoch/runter sprint, bevor das Ganze D
*/
void compute_menu_input(int input) {
  if (input == 0 && selected_item_in_menu != limit(selected_item_in_menu - 1, 0, DRINKS
    menu_cursor_position = 0; //cursor bewegt sich in obere Reihe
    selected_item_in_menu = limit(selected_item_in_menu - 1, 0, DRINKS - 1); //verschiel
  } else if (input == 1 && selected_item_in_menu != limit(selected_item_in_menu + 1, 0,
    menu_cursor_position = 1; //cursor bewegt sich in untere Reihe
    selected_item_in_menu = limit(selected_item_in_menu + 1, 0, DRINKS - 1); //verschiel
  }
  // Implementation of jumps from top to bottom
  else if (input == 0 && selected_item_in_menu == limit(selected_item_in_menu - 1, 0, DI
    menu_cursor_position = 1; //cursor bewegt sich in untere Reihe (letztes element)
    selected_item_in_menu = DRINKS-1; // select last element
  } else if (input == 1 && selected_item_in_menu == limit(selected_item_in_menu + 1, 0,
    menu_cursor_position = 0; //cursor bewegt sich in obere Reihe (erstes element)
    selected_item_in_menu = 0; // select first element
  }
```

```
}

/*
   Gibt Auswahl auf dem Display aus
   Waehrend dieser Anzeige werden jegliche Eingaben ignoriert
*/
void print_selection() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Sie kauften:");
  lcd.setCursor(0, 1);
  lcd.print(drink_name[selected_item_in_menu]);

  delay(3000); //Bestaetigung 3 Sekunden anzeigen + Eingaben ignorieren
}



/*********************************************************/
/*** SETUP + LOOP                                      */
/*********************************************************/

void setup() {
  Serial.begin(9600);

  //IO einrichten
  for (int i = 0; i < 3; i++)
    pinMode(buttons[i], INPUT_PULLUP);
  pinMode(led, OUTPUT);

  //Setze Schaltpin des Transistors als Ausgang und standardmaessig auf HIGH
  pinMode(disp, OUTPUT);
  digitalWrite(disp, HIGH);

  //lcd einrichten
  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.createChar(0, custom_char_1); //definiere ein chustomChar (Anzeigen mit Serial.wr
  lcd.createChar(1, custom_char_2); //definiere ein chustomChar (Anzeigen mit Serial.wr

  // check drink names for validity
  for (int i = 0; i < DRINKS; i++){
    Serial.print(drink_name[i]);
    Serial.print(" has length: ");
    Serial.println(strlen(drink_name[i]));

    // cut off everything after character 7 and replace character 8 with .
    if (strlen(drink_name[i]) > 8){
      drink_name[i][7] ='.';
      drink_name[i][8] = '\0';
    }
  }

  print_menu(); //initiales Anzeigen des Haupmenues
}
```

```
void loop() {

  if (millis() - idleCheck > 10000){
    Serial.println("idle reset bc time");
    // reset everything
    idleCheck = millis();

    //resette alle Parameter
    active_menu = 0; //setze Statusvariable zurueck; koennte auch Wechsel in weitere St
    selected_item_in_menu = 0;
    menu_cursor_position = 0;
    print_menu();

  }

  //lese und entprelle Taster, bei -1 soll nichts passieren, da keine Eingabe
  // resets idle counter if input occured
  int input = read_input();



  // switch error LED off
  if (millis() - errorLed > 150)
    digitalWrite(led, LOW);

  // switch display back on
  if (millis() - errorDisp > 150)
    digitalWrite(disp, HIGH);

  if (active_menu == 0) { //Hauptmenue, Getraenkeauswahl

    if (input == 2) {
      if (drink_storage[selected_item_in_menu] > 0){
        // Article available
        active_menu = 1;
        toPay = drink_price[selected_item_in_menu];
        print_pay_menu();
      } else {
        // Article not available
        digitalWrite(disp, LOW);
        errorDisp = millis();
      }
    } else if (input == 0 || input == 1) {
      //veraendere Auswahl
      compute_menu_input(input);
      print_menu();
    }

  } else if (active_menu == 1) { //Bestaetige Auswahl auf Display

    if (input == 2) {
      if (toPay <= 0){
        // enough paid
        drink_storage[selected_item_in_menu] -= 1;
```

```
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Sie kauften:");
        lcd.setCursor(0, 1);
        lcd.print(drink_name[selected_item_in_menu]);
        // wait for 3s and blink led
        for (int i = 0; i < 3; i++){
          digitalWrite(led, HIGH);
          delay(500);
          digitalWrite(led, LOW);
          delay(500);
        }

        //resette alle Parameter
        active_menu = 0; //setze Statusvariable zurueck; koennte auch Wechsel in weiter
        selected_item_in_menu = 0;
        menu_cursor_position = 0;
        print_menu();

      }
      else{
        // not enough paid
        // TODO bonus 1.5
        digitalWrite(led, HIGH);
        errorLed = millis();
      }
    }
    else if (input == 0 || input == 1) {
      //calculate what is still to pay
      compute_pay_input(input);
      print_pay_menu();
    }
  }
}
```