KOD ŹRÓDŁOWY PROGRAMU

```
/* dołączenie używanych bibliotek */
#include <Servo.h>
#include <LiquidCrystal.h>

/* definicje pinów */
#define button 2
#define button2 3
#define serwoPWM 10
#define motorPWM 11
#define in1 12
#define in2 13

#define WARNING A0                  // LED + buzzer
#define echoP A1
#define trigP A2
#define potencjometr A5

Servo serwo;                        // zainicjowanie serwomechanizmu
LiquidCrystal lcd(4, 5, 6, 7, 8, 9);  // zainicjowanie wyświetlacza LCD

/* zmienne */
int i, Vsilnik, Lprod = 0;          // V-prędkość silnika, Lprod- liczba produktów
long dystans;                       // przechowuje odległość
volatile bool  produkcjaSTOP = false; // określa czy zatrzymano produkcje

void setup() {

  /* tryb pracy pinów */
        pinMode(motorPWM, OUTPUT);        // sygnał PWM silnika, sterowanie prędkością
        pinMode(in1, OUTPUT);             // sygnały sterujące kierunkiem obrotów silnika
        pinMode(in2, OUTPUT);
        pinMode(serwoPWM, OUTPUT);        // sygnał PWM serwomechanizmu
        pinMode(trigP, OUTPUT);           // sygnał wyzwalajacy czujnika odległości
        pinMode(echoP, INPUT);            // sygnał odbierany przez czujnik odległosci
        pinMode(WARNING, OUTPUT);         // sygnał uruchamiający diodę LED i buzzer
        pinMode(button, INPUT_PULLUP);    // przycisk obsługujacy przerwanie
        pinMode(button2, INPUT_PULLUP);   // przycisk wznawiający pracę

        lcd.begin(16, 2);                 // rozpoczęcie pracy wyświetlacza
        serwo.attach(serwoPWM);           // przypisanie pinu obsługującego serwomechanizm
        serwo.write(180);                 // ustawienie serwo w pozycji początkowej

        // zdefiniowanie przerwania, wyzwolone zboczem opadającym przycisku- wywołuje funkcję
        'przerwanie'
        attachInterrupt(digitalPinToInterrupt(button), przerwanie, FALLING);
        }

void loop() {
```

```
        if (produkcjaSTOP) {                    // jeżeli zatrzymano produkcje
                zatrzymanie();                  // wywołaj funkcję zatrzymującą działanie układu
  }

        /* wyświetlenie informacji o stanie produkcji */
        lcd.setCursor(0, 0);                    // ustawienie kursora w początkowej pozycji
        lcd.print("V silnika: ");               // wyświetlenie napisu V silnika
        lcd.setCursor(11, 0);                   // ustawienie kursora na znak 11 w 1 wierszu
        lcd.print(map(Vsilnik, 0, 255, 0, 100));  // zmapowanie prędkości na wartosci od 0 do 100%
        lcd.setCursor(15, 0);                   // ustawienie kursora na znak 15 w 1 wierszu
        lcd.print("%");                         // wyświetlenie na koniec znaku "%"
        lcd.setCursor(0, 1);                    // ustawienie kursora w drugim wierszu na 1 znak
        lcd.print("L prod.:");                  // wyświetlenie napisu L.prod:
        lcd.setCursor(9, 1);                    // ustawienie kursora w 2 wierszu na znak 9
        lcd.print(Lprod);                       // wyświetlenie liczby produktów przechowywanej w
                                                                        zmiennej Lprod

        digitalWrite(in1, HIGH);                // ustawienie kierunku obrotów silnika w prawą stronę
        digitalWrite(in2, LOW);

        Vsilnik = map(analogRead(potencjometr), 0, 1023, 0, 255);
                                                                        // zmapowanie wartości
        analogowej odczytanej z potencjometru na zakres, w którym pracuje PWM

        analogWrite(motorPWM, Vsilnik);         // zmiana prędkości silnika

        /* czujnik odległosci */
        digitalWrite(trigP, HIGH);              // wyzwalenie fali ultradźwiękowej w czujniku
        delayMicroseconds(20);                  // przez 20 mikrosekund
        digitalWrite(trigP, LOW);               // odebranie informacji zwrotnej
        dystans = pulseIn(echoP, HIGH) / 58;    // wyznaczenie w cm odległosci zwracanej przez
                                                                        czujnik

        if ( dystans < 5) {                     // jeśli odległość produktu jest mniejsza niż 5 cm
                przesuniecie();                 // wywołanie funkcji przesuwajacej produkt z tasmy
  }

}

/* funkcja obsługująca przeunięcie produktu z tasmy */
void przesuniecie () {

        digitalWrite(in1, LOW);                 // zatrzymanie silnika
        digitalWrite(in2, LOW);

        for (i = 180; i > 0; i--) {

                if (produkcjaSTOP) {
                        zatrzymanie();
    }

                serwo.write(i);                 // wykonanie ruchu
```

```
            delay(20);                  // małe opóźnienie między ruchami
  }

        Lprod++;                    // inkrementacja liczby produktów
        serwo.write(180);           // powrot do pozycji poczatkowej serwa po wykonaniu
                                        calego ruchu
        delay(800);                 // opóźnienie aby serwomechanizm zdążył wrócić do pozycji
                                        bazowej, bez uruchamiania czunika
}

/* funkcja obsługująca zatrzymanie produkcji */
void zatrzymanie () {

        digitalWrite(in1, LOW);         // zatrzymanie silnika
        digitalWrite(in2, LOW);

        digitalWrite(WARNING, HIGH);    // włączenie buzzera i zapalenie diody LED

        /* wyswietlenie informacji o zatrzymaniu produkcji na wyswietlaczu */
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("PRODUKCJA");
        lcd.setCursor(0, 1);
        lcd.print("WSTRZYMANA");

        /* nieskończona pętla */
        while (1) {

            if (digitalRead(button2) == LOW) {      // jeżeli wcisnięto przycisk wznowienia
                                                        produkcji

                digitalWrite(WARNING, LOW);     // wyłącz buzzer i diodę
                lcd.clear();                    // wyczysć wyswietlacz

                produkcjaSTOP = false;          // ustaw wstrzymywanie produkcji na 'false'
                break;                          // wyjdź z pętli
   }
  }
}

/* funkcja wywoływana podczas przerwania */
void przerwanie() {
        produkcjaSTOP = true;           // ustaw wstrzymanie produkcji na 'true'
}
```