```
#include <avr/io.h>
#include <util/delay.h>

// Pin definitions
#define Motor_Forward PD0
#define Motor_Backward PD1
#define Motor_Left PD2
#define Motor_Right PD3
#define Ultrasonic_Trig PD4
#define Ultrasonic_Echo PD5
#define Voice_Recognition_Input PD6
#define Pulse_Sensor_Input PC0
#define Temp_Sensor_Input PC1

// Function Prototypes
void setup();
void forward();
void backward();
void left();
void right();
void stop();
float readTemperature();
int readPulse();
int readUltrasonic();
void voiceControl();

int main(void) {
    setup();
    while (1) {
        voiceControl(); // Control wheelchair based on voice input
    }
```

```
}

// Initialize the inputs/outputs
void setup() {
  // Set motor pins as output
  DDRD |= (1<<Motor_Forward) | (1<<Motor_Backward) | (1<<Motor_Left) | (1<<Motor_Right);

  // Set ultrasonic and voice recognition pins
  DDRD |= (1<<Ultrasonic_Trig);
  DDRD &= ~(1<<Ultrasonic_Echo) & ~(1<<Voice_Recognition_Input); // Input for Echo & Voice Rec

  // Set pulse and temperature sensor pins as input
  DDRC &= ~(1<<Pulse_Sensor_Input) & ~(1<<Temp_Sensor_Input);

  // Set initial motor state to stop
  stop();
}

// Control movement based on voice input
void voiceControl() {
  if (PIND & (1<<Voice_Recognition_Input)) {
    // Read voice input (example: HIGH for forward)
    int voiceCommand = PIND & (1<<Voice_Recognition_Input);

    if (voiceCommand == 1) {
      forward();
    } else if (voiceCommand == 2) {
      backward();
    } else if (voiceCommand == 3) {
      left();
    } else if (voiceCommand == 4) {
```

```
            right();
        } else {
            stop();
        }
    }
}

// Motor functions
void forward() {
    PORTD |= (1<<Motor_Forward);
    PORTD &= ~((1<<Motor_Backward) | (1<<Motor_Left) | (1<<Motor_Right));
}

void backward() {
    PORTD |= (1<<Motor_Backward);
    PORTD &= ~((1<<Motor_Forward) | (1<<Motor_Left) | (1<<Motor_Right));
}

void left() {
    PORTD |= (1<<Motor_Left);
    PORTD &= ~((1<<Motor_Forward) | (1<<Motor_Backward) | (1<<Motor_Right));
}

void right() {
    PORTD |= (1<<Motor_Right);
    PORTD &= ~((1<<Motor_Forward) | (1<<Motor_Backward) | (1<<Motor_Left));
}

void stop() {
    PORTD &= ~((1<<Motor_Forward) | (1<<Motor_Backward) | (1<<Motor_Left) |
(1<<Motor_Right));
```

```
}

// Read temperature using LM35 sensor
float readTemperature() {
  ADMUX = (1<<REFS0) | (Temp_Sensor_Input); // AVcc with external cap at AREF pin
  ADCSRA = (1<<ADEN) | (1<<ADSC); // Start conversion
  while (ADCSRA & (1<<ADSC)); // Wait for conversion to finish
  int adcValue = ADC;
  return (adcValue * 5.0 / 1024.0) * 100; // Convert ADC value to temperature in °C
}

// Read pulse sensor data
int readPulse() {
  ADMUX = (1<<REFS0) | (Pulse_Sensor_Input);
  ADCSRA = (1<<ADEN) | (1<<ADSC);
  while (ADCSRA & (1<<ADSC));
  return ADC; // Return pulse sensor value
}

// Read distance from ultrasonic sensor
int readUltrasonic() {
  PORTD |= (1<<Ultrasonic_Trig); // Trigger high
  _delay_us(10);
  PORTD &= ~(1<<Ultrasonic_Trig); // Trigger low
  uint16_t pulse_width = 0;
  while (!(PIND & (1<<Ultrasonic_Echo))); // Wait for echo
  while (PIND & (1<<Ultrasonic_Echo)) {
    pulse_width++;
    _delay_us(1);
  }
  return pulse_width / 58; // Convert pulse width to distance in cm
```

}