Write an 8051 C program to implement a simple password-based door lock. The password is entered0 through 4 switches connected to P1.0 to P1.3, and if the correct password (1010) is entered, it opens the door connected to P2.0. Setting 0 to P2.0 opens the door. The door should close after a 5-second delay generated using for loops.

**#include <reg51.h>**

**// Define LED for door lock and switch states**

**sbit DOOR\_LOCK = P2^0; // Lock connected to P2.0**

**sbit sw1 = P1^0; // Buttons connected to P1 pins**

**sbit sw2 = P1^1;**

**sbit sw3 = P1^2;**

**sbit sw4 = P1^3;**

**// Correct password: 1010**

**unsigned char password[4] = {1, 0, 1, 0};**

**void delay(unsigned int count) {**

**int i;**

**for(i = 0; i < count; i++);**

**for(i = 0; i < 100; i++);**

**}**

**unsigned char check\_password() {**

**if(sw1 == password[0] && sw2 == password[1] && sw3 == password[2] && sw4 == password[3]) {**

**return 1; // Password is correct**

**}**

**return 0; // Incorrect password**

**}**

**void main() {**

**DOOR\_LOCK = 1; // Initially, door is locked**

**while(1) {**

**if (check\_password()) {**

**DOOR\_LOCK = 0; // Unlock the door**

**delay(5000);**

**DOOR\_LOCK = 1; // Lock the door again after a delay**

**}**

**}**

**}**

Write an 8051 C program to control a fan based on the temperature in an industrial environment. Assume the temperature data is available in port 1. If the temperature value exceeds 25, the fan should be turned on, otherwise it should be turned off. A delay time of 2 minutes is given for stabilization in either case.

**#include <reg51.h>**

**// Assume that temperature data comes from a sensor connected to P1**

**sbit FAN = P2^0; // FAN connected to P2.0**

**unsigned char temperature;**

**void delay() {**

**int i,j,k;**

**for(i = 0; i < 60; i++){ // for 60 seconds**

**for(i = 0; i < 1000; i++){ // for 1 second**

**for(k = 0; k < 100; i++); // for 1 ms**

**}**

**}**

**}**

**void main() {**

**while(1) {**

**temperature = P1; // Read temperature from sensor connected to Port 1**

**if(temperature > 25) {**

**FAN = 1; // Turn ON the fan if temperature > 25**

**} else {**

**FAN = 0; // Turn OFF the fan if temperature <= 25**

**}**

**delay(); // Delay for stabilization**

**}**

**}**

**Receive data from a port and send acknowledgement**

**#include <reg51.h> // Include 8051 register definitions**

**void main() {**

**unsigned char received\_data;**

**// Configure P1 as input and P2 as output**

**P1 = 0xFF; // Configure Port 1 as input (high-impedance state)**

**while (1) {**

**received\_data = P1; // Read data from Port 1**

**if (received\_data == 0x55) { // Check if specific data (e.g., 0x55) is received**

**P2 = 0xFF; // Turn on all LEDs connected to Port 2 (output high)**

**} else {**

**P2 = 0x00; // Turn off all LEDs**

**}**

**}**

**}**

**Receiving Serialized Data and Reconstructing a Byte**

This program receives serialized data bit-by-bit from **Pin P3.2**, reconstructs the byte, and outputs it to **Port 2**.

#include <reg51.h> // Include 8051 register definitions

sbit serial\_in = P3^2; // Define the pin for serial input

void delay() {

unsigned int i;

for (i = 0; i < 100; i++); // Simple delay

}

unsigned char receive\_serial() {

unsigned char i, data\_byte = 0;

for (i = 0; i < 8; i++) { // Receive 8 bits

data\_byte <<= 1; // Shift left to make room for the next bit

if (serial\_in) { // Check if the input bit is high

data\_byte |= 0x01; // Set the LSB of the byte

}

delay(); // Short delay between receiving bits

}

return data\_byte; // Return the reconstructed byte

}

void main() {

unsigned char received\_data;

while (1) {

received\_data = receive\_serial(); // Receive serialized data

P2 = received\_data; // Output the reconstructed byte to Port 2

}

}

**Write a C program to send out the value 44h serially one bit at a time via P1.0. The LSB should go out first.**