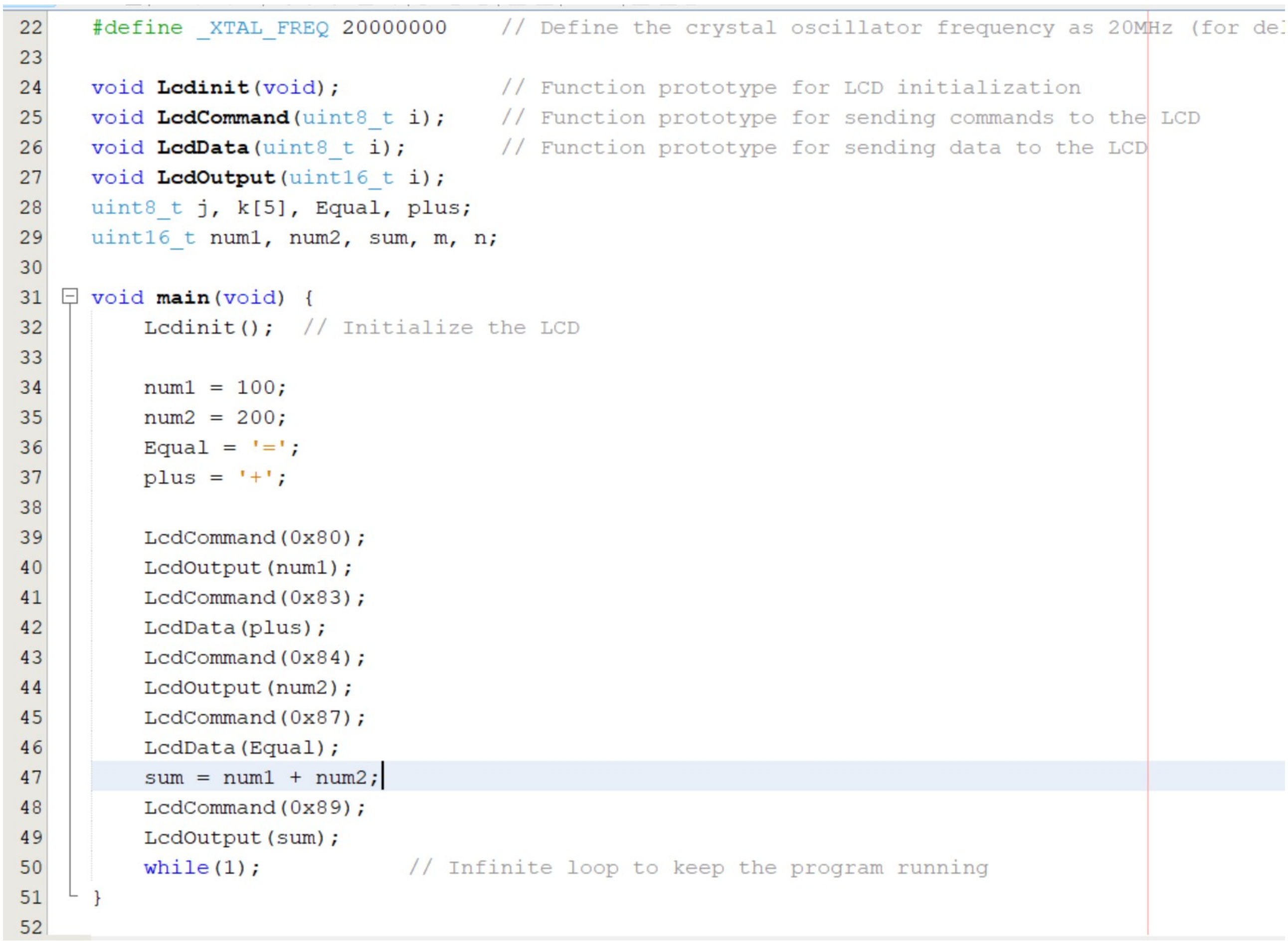
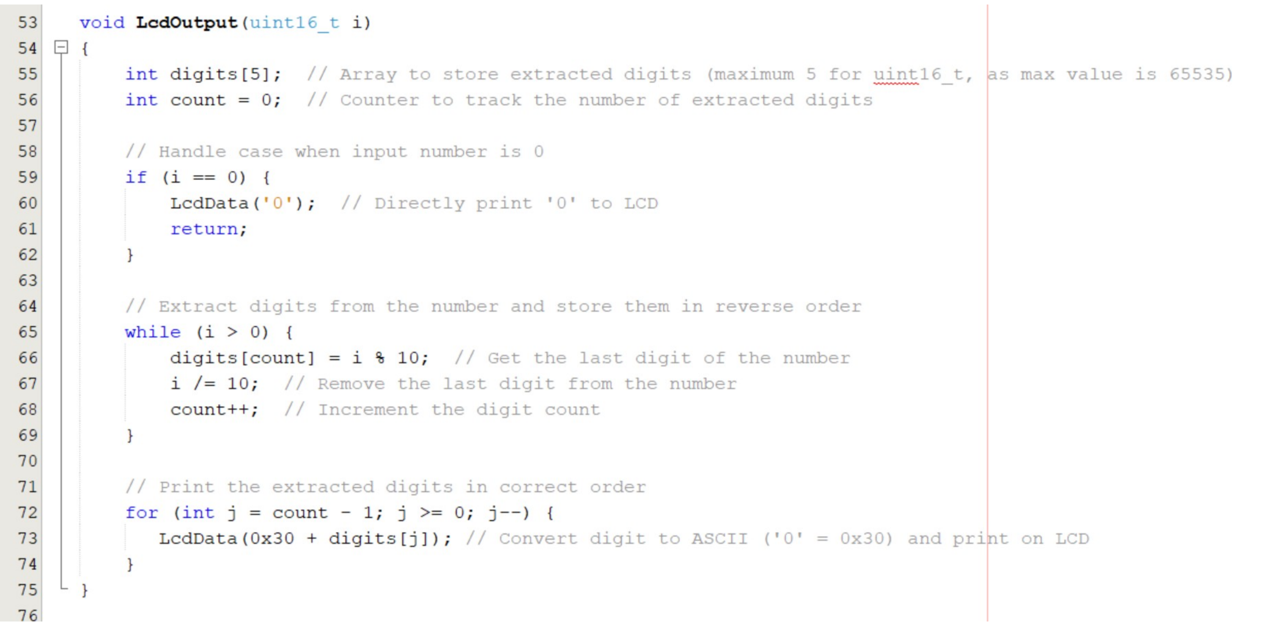
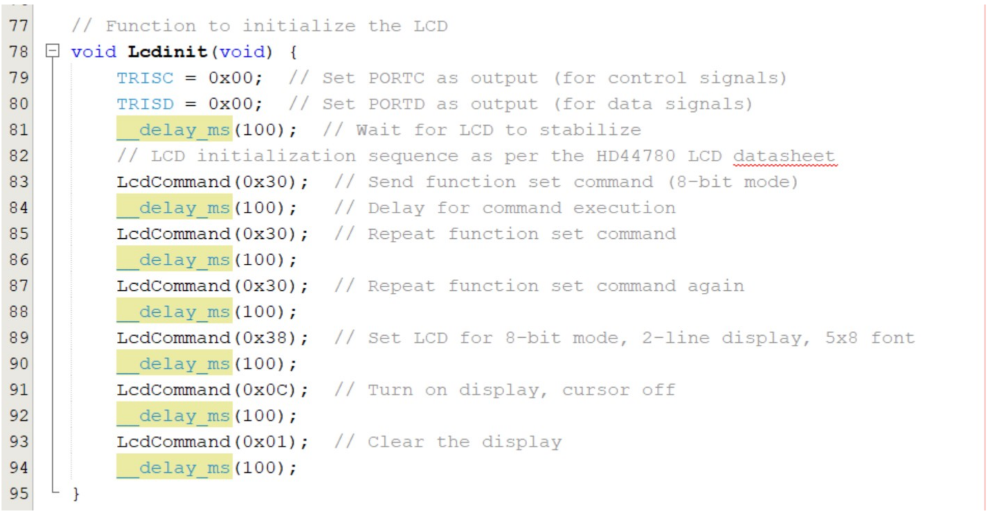
Class 4 Class Program 1: Print Sum of Two Number On LCD



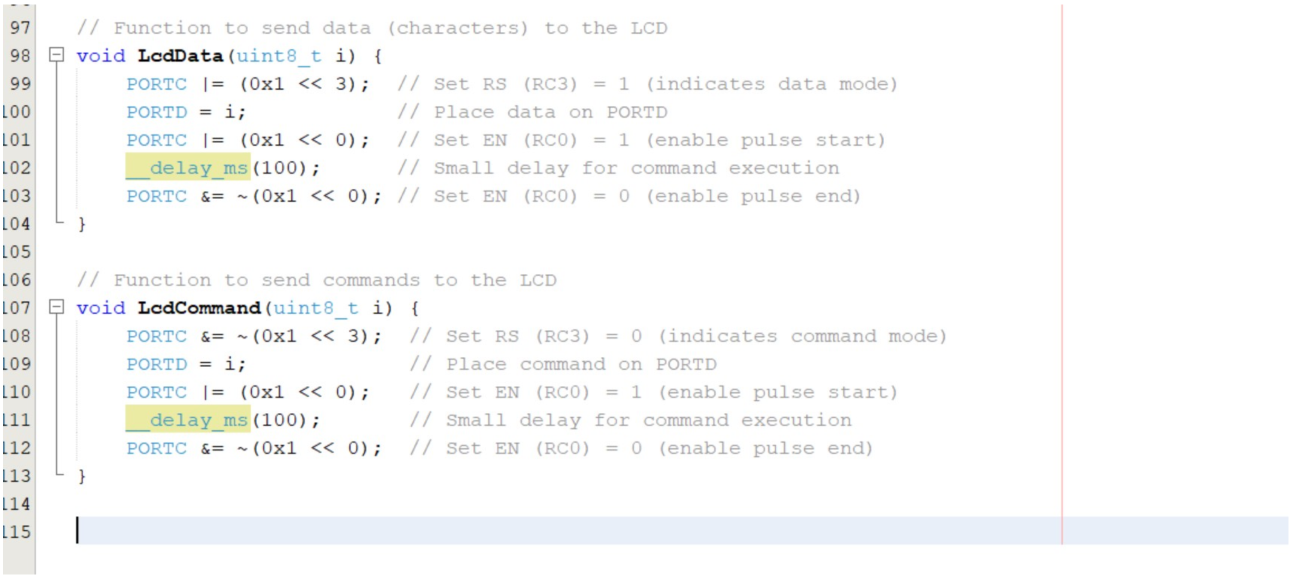
```c
#define _XTAL_FREQ 20000000    // Define the crystal oscillator frequency as 20MHz (for de

void Lcdinit(void);            // Function prototype for LCD initialization
void LcdCommand(uint8_t i);    // Function prototype for sending commands to the LCD
void LcdData(uint8_t i);       // Function prototype for sending data to the LCD
void LcdOutput(uint16_t i);
uint8_t j, k[5], Equal, plus;
uint16_t num1, num2, sum, m, n;

void main(void) {
    Lcdinit();  // Initialize the LCD

    num1 = 100;
    num2 = 200;
    Equal = '=';
    plus = '+';

    LcdCommand(0x80);
    LcdOutput(num1);
    LcdCommand(0x83);
    LcdData(plus);
    LcdCommand(0x84);
    LcdOutput(num2);
    LcdCommand(0x87);
    LcdData(Equal);
    sum = num1 + num2;
    LcdCommand(0x89);
    LcdOutput(sum);
    while(1);           // Infinite loop to keep the program running
}

```



```c
void LcdOutput(uint16_t i)
{
    int digits[5];  // Array to store extracted digits (maximum 5 for uint16_t, as max value is 65535)
    int count = 0;  // Counter to track the number of extracted digits

    // Handle case when input number is 0
    if (i == 0) {
        LcdData('0');  // Directly print '0' to LCD
        return;
    }

    // Extract digits from the number and store them in reverse order
    while (i > 0) {
        digits[count] = i % 10;  // Get the last digit of the number
        i /= 10;  // Remove the last digit from the number
        count++;  // Increment the digit count
    }

    // Print the extracted digits in correct order
    for (int j = count - 1; j >= 0; j--) {
       LcdData(0x30 + digits[j]); // Convert digit to ASCII ('0' = 0x30) and print on LCD
    }
}

```



```
// Function to initialize the LCD
void Lcdinit(void) {
    TRISC = 0x00;  // Set PORTC as output (for control signals)
    TRISD = 0x00;  // Set PORTD as output (for data signals)
    __delay_ms(100);  // Wait for LCD to stabilize
    // LCD initialization sequence as per the HD44780 LCD datasheet
    LcdCommand(0x30);  // Send function set command (8-bit mode)
    __delay_ms(100);   // Delay for command execution
    LcdCommand(0x30);  // Repeat function set command
    __delay_ms(100);
    LcdCommand(0x30);  // Repeat function set command again
    __delay_ms(100);
    LcdCommand(0x38);  // Set LCD for 8-bit mode, 2-line display, 5x8 font
    __delay_ms(100);
    LcdCommand(0x0C);  // Turn on display, cursor off
    __delay_ms(100);
    LcdCommand(0x01);  // Clear the display
    __delay_ms(100);
}
```



```
// Function to send data (characters) to the LCD
void LcdData(uint8_t i) {
    PORTC |= (0x1 << 3);  // Set RS (RC3) = 1 (indicates data mode)
    PORTD = i;            // Place data on PORTD
    PORTC |= (0x1 << 0);  // Set EN (RC0) = 1 (enable pulse start)
    __delay_ms(100);      // Small delay for command execution
    PORTC &= ~(0x1 << 0); // Set EN (RC0) = 0 (enable pulse end)
}

// Function to send commands to the LCD
void LcdCommand(uint8_t i) {
    PORTC &= ~(0x1 << 3);  // Set RS (RC3) = 0 (indicates command mode)
    PORTD = i;             // Place command on PORTD
    PORTC |= (0x1 << 0);   // Set EN (RC0) = 1 (enable pulse start)
    __delay_ms(100);       // Small delay for command execution
    PORTC &= ~(0x1 << 0);  // Set EN (RC0) = 0 (enable pulse end)
}

```