Class 2:

Task: Blink LED Using Push Button

```
/**
 ******************************************************************************
 * @file           : main.c
 * @author         : Sagar More
 * @brief          : Class 2 - Task
 * @board          : STM32F446RE
 *
 *
 *  ------------------------------------------
 *  |      Button         |       LED         |
 *  ------------------------------------------
 *  |    ON Press         |  PD12  |  PD13    |
 *  ------------------------------------------
 *  |       PC4           |  ON    |   OFF    |
 *  ------------------------------------------
 *  |       PC5           |  OFF   |  ON      |
 *  ------------------------------------------
 *  |       PC6           |  ON    |  ON      |
 *  ------------------------------------------
 *
 ******************************************************************************
 */
```

```
/*
 Register Memory Mapping
 ----------------------------------------
 RCC Base Address: 0x40023800
 RCC_AHB1ENR Offset: 0x30 → RCC_AHB1ENR: 0x40023830

 GPIOD Base Address (LED): 0x40020C00
 GPIOD_MODER Offset: 0x00 → GPIOD_MODER: 0x40020C00
 GPIOD_ODR   Offset: 0x14 → GPIOD_ODR:   0x40020C14

 GPIOC Base Address (Switch): 0x40020800
 GPIOC_MODER Offset: 0x00 → GPIOC_MODER: 0x40020800
 GPIOC_IDR   Offset: 0x10 → GPIOC_IDR:   0x40020810


*/

#include <stdint.h>
#include <stdbool.h>

/* ------------------------------------------------------------------
 * Memory-Mapped Registers
 * ------------------------------------------------------------------*/
#define RCC_AHB1ENR   (*(volatile uint32_t *)0x40023830)  // Clock enable register

#define GPIOD_MODER   (*(volatile uint32_t *)0x40020C00)  // GPIOD mode register
#define GPIOD_ODR     (*(volatile uint32_t *)0x40020C14)  //  GPIOD output register

#define GPIOC_MODER   (*(volatile uint32_t *)0x40020800)  // GPIOC mode register
#define GPIOC_IDR     (*(volatile uint32_t *)0x40020810)  // GPIOC input data register

```

```c
/* ---------------------------------------------------------------------
 * Main Function
 * ---------------------------------------------------------------------*/
int main(void)
{
    /* -------------------------------
     *  Enable Clock for GPIOC & GPIOD
     * -------------------------------
     * RCC_AHB1ENR: Bit 2 -> GPIOCEN, Bit 3 -> GPIODEN
     * 0x5 = (1 << 2) | (1 << 3)
     */
    RCC_AHB1ENR |= (1 << 2) | (1 << 3);

    /* -------------------------------
     * Configure GPIOD PIN12 and PIN13 as Output
     * -------------------------------
     * MODER[24:25] = 01 (PIN12 Output Mode)
     * MODER[26:27] = 01 (PIN13 Output Mode)
     */

    GPIOD_MODER |=  (1U << 24);  // Set MODER10 = 1
    GPIOD_MODER &= ~(1U << 25);  // Set MODER11 = 0

    GPIOD_MODER |=  (1U << 26);  // Set MODER10 = 1
    GPIOD_MODER &= ~(1U << 27);  // Set MODER11 = 0

    /* -------------------------------
     * Configure GPIOC PIN4, PIN5 and PIN6 as Input
     * -------------------------------
     * MODER[8:9] = 00 (Input Mode)
     * MODER[10:11] = 00 (Input Mode)
     * MODER[12:13] = 00 (Input Mode)
     */

    // GPIOC Pin4, pin5 and pin6 to input mode
    GPIOC_MODER &= ~((3U << 8) | (3U << 10) | (3U << 12));

    // Flag to track LED state
    volatile bool pc4LedFlag = false;
    volatile bool pc5LedFlag = false;
    volatile bool pc6LedFlag = false;
```

```
    while (1)
    {
        /* Check for the PC4 Button*/
        pc4LedFlag = ((GPIOC_IDR & (1U << 4)) == 0);

        /* Check for the PC5 Button*/
        pc5LedFlag = ((GPIOC_IDR & (1U << 5)) == 0);

        /* Check for the PC6 Button*/
        pc6LedFlag = ((GPIOC_IDR & (1U << 6)) == 0);

        /* -------------------------------
         * Control LED based on Flag
         * -------------------------------*/
        if (pc4LedFlag){
            GPIOD_ODR |=  (1U << 12);
            GPIOD_ODR &= ~(1U << 13);
        }
        else if (pc5LedFlag){
            GPIOD_ODR &= ~(1U << 12);
            GPIOD_ODR |=  (1U << 13);
        }else if(pc6LedFlag){
            GPIOD_ODR |=  (1U << 12);
            GPIOD_ODR |=  (1U << 13);
        }else{
            GPIOD_ODR &=  ~(1U << 12);
            GPIOD_ODR &=  ~(1U << 13);
        }
    }
}

```