Main Task 1

* **How do you edit the code provided to make the other two LEDS blink?**

To make the other two LEDs (LED2 and LED3) blink in addition to LED1, you need to modify the code so that it controls all three LEDs in sequence, not just LED1.

Here’s how you would do that **in words**:

1. **Define the other LEDs in the code** – Add two more DigitalOut objects for LED2 and LED3, just like LED1.
2. **Turn each LED on and off in order** – Instead of toggling only LED1 inside the loop, you write separate commands to:
   * Turn LED1 on,
   * Wait for 500 milliseconds,
   * Turn LED1 off and wait again,
   * Then do the same for LED2,
   * And repeat the same for LED3.
3. **Use a delay between each LED** – This ensures that each LED turns on one at a time with a delay in between, creating the "blinking in sequence" effect.
4. **Repeat the sequence in a loop** – Put all of the above steps inside the while (true) loop so it keeps repeating.

This approach ensures all three LEDs blink one after the other, each staying on for 500 milliseconds, then off for 500 milliseconds before the next LED starts.

* **Write a function that makes each LED blink in sequence: LED1 blinks first, then LED2, and finally LED3. Each LED should stay on for 500 ms, then off for 500 ms before the next LED blinks. How would you implement this function using delay mechanisms?**

**Steps to Implement the Function:**

1. **Define the LEDs**:  
   First, create DigitalOut variables for LED1, LED2, and LED3. This allows the program to control them individually.
2. **Create the Blink Function**:  
   Write a function (e.g., blink\_sequence()) that will contain the blinking logic for the three LEDs.
3. **Inside the Function – Blink Each LED One by One**:
   * Turn **LED1** on using led1 = 1.
   * Use a delay of **500 milliseconds** (ThisThread::sleep\_for(500ms)) to keep it on.
   * Turn **LED1** off using led1 = 0.
   * Delay again for **500 milliseconds** before moving to the next LED.
   * Repeat the same steps for **LED2** and then **LED3**.
4. **Use ThisThread::sleep\_for() for Delays**:  
   This is Mbed OS's built-in function for pausing the program. It's used to create the **500 ms on/off timing** between LED transitions.
5. **Call the Function in a Loop**:  
   In the main() function, place a while (true) loop that keeps calling blink\_sequence() so that the LEDs continue blinking in sequence forever.

* **Modify your code from Question 2 to make all three LEDs blink simultaneously. Each LED should turn on for 300 ms and then off for 300 ms in sync. What changes are required in the timing and control structure to achieve this?**

Changes Required in the Timing and Control Structure:

1. Turn on all LEDs at the same time:  
   In your code, instead of blinking each LED one after the other, you set all three LEDs to 1 (on) at the same moment.
2. Use a single delay for ON time:  
   After turning them all on, use ThisThread::sleep\_for(300ms) to keep them on for 300 milliseconds.
3. Turn off all LEDs at the same time:  
   Set all three LEDs to 0 (off) together.
4. Use a single delay for OFF time:  
   Again, use ThisThread::sleep\_for(300ms) to keep them off for 300 milliseconds.
5. Repeat the pattern in a loop:  
   Place the ON/OFF pattern inside a while (true) loop so it repeats continuously.

* **Write code to create a pattern where: • LED1 turns on for 200 ms, then off. • LED2 turns on after LED1 turns off, staying on for 400 ms. • LED3 turns on after LED2 turns off, staying on for 600 ms. After LED3 turns off, the sequence should repeat indefinitely. Explain how you would manage each LED’s timing for this staggered effect.**

How to Manage the LED Timing for the Staggered Effect:

1. Turn on LED1 first and keep it on for 200 milliseconds:
   * Set led1 = 1.
   * Use ThisThread::sleep\_for(200ms).
   * Then set led1 = 0.
2. After LED1 turns off, turn on LED2 for 400 milliseconds:
   * Set led2 = 1.
   * Use ThisThread::sleep\_for(400ms).
   * Then set led2 = 0.
3. After LED2 turns off, turn on LED3 for 600 milliseconds:
   * Set led3 = 1.
   * Use ThisThread::sleep\_for(600ms).
   * Then set led3 = 0.
4. Repeat the entire pattern:
   * Place all the steps inside a while (true) loop to repeat the sequence continuously.

Timing Control Explanation:

* The timing is managed sequentially, meaning the program waits for each LED’s ON time to finish before starting the next.
* There are no overlapping delays, so only one LED is on at a time.
* ThisThread::sleep\_for() is used to pause the program exactly for how long each LED should stay on.

This method ensures a smooth, predictable, and non-overlapping LED blink pattern where the sequence goes: LED1 → LED2 → LED3 → repeat.

* **Create a function where all three LEDs blink together for five times (200 ms on, 200 ms off for each blink). After five blinks, LED1 stays on continuously, while LED2 and LED3 remain off. What additional logic would you add to handle this two-phase behaviour?**

**Phase 1: Blink All LEDs Together 5 Times**

1. Use a for loop that runs **5 times**.
2. In each loop iteration:
   * Turn **all three LEDs on** (led1 = 1; led2 = 1; led3 = 1).
   * Wait for **200 ms** using ThisThread::sleep\_for(200ms).
   * Turn **all three LEDs off**.
   * Wait again for **200 ms**.
3. This makes all three LEDs blink **in sync**, 5 times.

**Phase 2: Set Final LED States**

1. After the loop finishes, set the final state of the LEDs:
   * led1 = 1 → LED1 stays **on**.
   * led2 = 0 and led3 = 0 → LED2 and LED3 stay **off**.
2. **Do not use a loop** for this part—just set the states once and leave them.

**Additional Logic to Handle Two-Phase Behavior:**

* Use a **counter or loop** to manage the 5 blinks.
* Use **sequential logic**: execute the blinking phase first, then follow with the final LED states.
* Avoid placing everything inside an infinite loop, since the behavior should stop after the 5 blinks and hold the final state.