

Lab 2: Requirement Description

- **Introduction to addressing mode**
 - Link: [lab2 : Addressing Mode - YouTube](#)
 - Link: [Lab02 Addressing Mode - HackMD](#)
- **Lab requirements:**
 - **Basic (70%)**
 - **Description:**

Please store the value 0x03 at Data Memory address [0x100], and store the value 0x08 at Data Memory address [0x101]. After that, please use at least one type of indirect addressing register to ensure that the values at Data Memory addresses [0x102] through [0x108] are all computed as the result of adding or subtracting the values from the preceding two addresses.

When the address contains even values, such as [0x102] or [0x104], the value at the current address should be the sum of the values from the previous two addresses. Conversely, when the address contains odd values, like [0x103] or [0x105], the value at the current address should be computed as the result of subtracting the values from the previous two addresses.
 - **Example:**
 1. The value at Data Memory address [0x102] is calculated as the sum of the values at Data Memory addresses [0x100] and [0x101], i.e., $[0x102] = [0x100] + [0x101]$.
 2. The value at Data Memory address [0x103] is determined by subtracting the value at Data Memory address [0x101] from the value at Data Memory address [0x102], i.e., $[0x103] = [0x102] - [0x101]$.
 3. This procedure should be continued up to Data Memory address [0x108].
 - **Standard of grading:**
 1. The values at Data Memory addresses [0x100] to [0x108] should be as follows: 0x03, 0x08, 0x0B, 0x03, 0x0E, 0x0B, 0x19, 0x0E, and 0x27 respectively.
 2. Use at least one type of indirect addressing register.
 - **Advanced(30%):**
 - **Description:**

Please store the value 0x05 at Data Memory address [0x000], and store the value 0x02 at Data Memory address [0x018]. After that, please use at least one type of FSR to ensure the value at Data Memory address [0x001] represents the sum of the values at Data Memory addresses [0x000] and [0x018]. Furthermore, subtract the value of [0x017] from the values of [0x000] and [0x018]. Proceed to Data Memory address [0x002], where the value should be the sum of the values at Data Memory addresses [0x001] and [0x017]. In a similar fashion, subtract the value of [0x016] from the values of [0x001] and [0x017], and so on.

- **Standard of grading:**
 1. The values at Data Memory addresses [0x000] to [0x008] should be 0x05, 0x07, 0x0A, 0x0E, 0x14, 0x1C, 0x28, 0x38, and 0x50 respectively. The values at Data Memory addresses [0x010] to [0x018] should be 0x20, 0x18, 0x10, 0x0C, 0x08, 0x06, 0x04, 0x03, and 0x02 respectively. (Please provide a screenshot from MPLAB).
 2. Use at least one type of indirect addressing register.
- **Bonus (20%):**
 - **Description:**
Store the values 0xB5, 0xF8, 0x64, 0x7F, 0xA8, and 0x15 at Data Memory addresses [0x100] to [0x105], respectively. Implement cocktail sort using at least one type of indirect addressing register to arrange the above six values in ascending order. After sorting, store the resulting values sequentially at locations [0x110] to [0x115].
 - **Standard of grading:**
 1. The values at addresses [0x110] to [0x115] should be 0x15, 0x64, 0x7F, 0xA8, 0xB5, and 0xF8 respectively.
 2. Use at least one type of indirect addressing register.
 3. Use cocktail sort

Source: cocktail sort

```
void cocktail_sort(int arr[], int len) {
    int i, left = 0, right = len - 1;
    int temp;
    while (left < right) {
        for (i = left; i < right; i++)
            if (arr[i] > arr[i + 1]) {
                temp = arr[i];
                arr[i] = arr[i + 1];
                arr[i + 1] = temp;
            }
        right--;
        for (i = right; i > left; i--)
            if (arr[i - 1] > arr[i]) {
                temp = arr[i];
                arr[i] = arr[i - 1];
                arr[i - 1] = temp;
            }
        left++;
    }
}
```