Assignment - 3

```
#include <xc.h>
#include <stdio.h>
#include <stdlib.h> // for qsort function

// Define the system clock frequency (e.g., 8 MHz)
#define _XTAL_FREQ 8000000  // Set the crystal frequency to 8 MHz

// SWAP FUNCTION
void swap(int *x, int *y) // PASS BY REFERENCE
{
    int temp = *x;
    *x = *y;
    *y = temp;
}

// Example array
int arr[] = {5, 2, 3, 1, 4};
int n = 5; // Number of elements in the array
// Comparator for ascending order (for qsort)
int comp_asc(const void *a, const void *b) {
    return (*(int *)a - *(int *)b);
}

// Comparator for descending order (for qsort)
int comp_desc(const void *a, const void *b) {
    return (*(int *)b - *(int *)a);
}
 int sortMethod = 1;
void main(void) {
    // Configure the I/O pins for PORTB, PORTC, and PORTD
    TRISB = 0x00; // Set PORTB as output
    TRISC = 0x00; // Set PORTC as output
    TRISD = 0x00; // Set PORTD as output

    // Select sorting method and order (1 = Bubble Sort, 2 = qsort)
    // For ascending order, 1 = Bubble Sort, 2 = qsort
    // For descending order, 3 = Bubble Sort, 4 = qsort
     // Change this to switch between methods (1-4)

    switch (sortMethod) {
        case 1: // Bubble Sort in Ascending Order
            for (int i = 0; i < n; i++) {
                for (int j = 0; j < n - 1 - i; j++) {
                    if (arr[j] > arr[j + 1]) {
                        swap(&arr[j], &arr[j + 1]);
```

```
            }
        }
    }
    break;

case 2: // qsort in Ascending Order
    qsort(arr, (size_t)n, sizeof(arr[0]), comp_asc);
    break;

case 3: // Bubble Sort in Descending Order
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n - 1 - i; j++) {
            if (arr[j] < arr[j + 1]) {
                swap(&arr[j], &arr[j + 1]);
            }
        }
    }
    break;

case 4: // qsort in Descending Order
    qsort(arr, (size_t)n, sizeof(arr[0]), comp_desc);
    break;

default:
    // Default case if an invalid sort method is selected
    break;
}
// After sorting, output values to PORTB, PORTC, and PORTD
PORTB = (unsigned char)arr[0]; // First value to PORTB
PORTC = (unsigned char)arr[1]; // Second value to PORTC
PORTD = (unsigned char)arr[2]; // Third value to PORTD }
```