Assignment 5

Write Program in KEIL Embedded C:

1. Write a C program for the 8051 to print Fibonacci series. Take input from port P1 and display output on port P4 with delay of 100 milliseconds. (Create manual delay using loop).

```
⇨ #include <reg51.h>
⇨
⇨ void delay(unsigned int time) {
⇨ unsigned int i, j;
⇨ for(i = 0; i < time; i++)
⇨ for(j = 0; j < 1275; j++);
⇨ }
⇨
⇨ void main() {
⇨ unsigned char a = 0, b = 1, c, n, i;
⇨
⇨ P1 = 0xFF; // Set port P1 as input
⇨ P4 = 0x00; // Initialize port P4
⇨
⇨ while(1) {
⇨ n = P1; // Take input from port P1
⇨
⇨ for(i = 0; i < n; i++) {
⇨ if(i <= 1)
⇨ c = i;
⇨ else {
⇨ c = a + b;
⇨ a = b;
⇨ b = c;
⇨ }
⇨ P4 = c; // Display output on port P4
⇨ delay(100); // Delay of 100 milliseconds
⇨ }
⇨ }
⇨ }
```

2. Write a program in C for 8051 to find the square root of a number and show the output on port.

```
⇨ #include <reg51.h>
⇨ #include <math.h>
⇨
⇨ void main() {
⇨ unsigned int number, result;
⇨
⇨ P1 = 0xFF; // Set port P1 as input
⇨ P4 = 0x00; // Initialize port P4
⇨
⇨ while(1) {
⇨ number = P1; // Take input from port P1
⇨ result = sqrt(number); // Calculate square root
⇨
⇨ P4 = result; // Display result on port P4
⇨ }
⇨ }
```

3. Write a C program for the 8051 to sort a list of n numbers. Sorted list should be displayed Port P0, P1, and P2 & P3 periodically with the delay of 150 milliseconds. (Hint: 1st number on P0, 2nd number on P1, 3rd number on P2, 4th number on P3, 5th number P1 and so on).

```
⇨ #include <reg51.h>
⇨
⇨ void delay(unsigned int time) {
⇨ unsigned int i, j;
⇨ for(i = 0; i < time; i++)
⇨ for(j = 0; j < 1275; j++);
⇨ }
⇨
⇨ void bubbleSort(unsigned char arr[], unsigned char n) {
⇨ unsigned char i, j, temp;
⇨ for(i = 0; i < n - 1; i++) {
⇨ for(j = 0; j < n - i - 1; j++) {
⇨ if(arr[j] > arr[j + 1]) {
⇨ temp = arr[j];
⇨ arr[j] = arr[j + 1];
⇨ arr[j + 1] = temp;
⇨ }
⇨ }
⇨ }
⇨ }
⇨
⇨ void main() {
⇨ unsigned char numbers[] = {10, 5, 8, 2, 3}; // Example list of numbers
⇨ unsigned char n = sizeof(numbers) / sizeof(numbers[0]);
⇨ unsigned char i = 0;
⇨
⇨ P0 = 0x00; // Initialize port P0
⇨ P1 = 0x00; // Initialize port P1
⇨ P2 = 0x00; // Initialize port P2
⇨ P3 = 0x00; // Initialize port P3
⇨
⇨ while(1) {
⇨ bubbleSort(numbers, n); // Sort the numbers
⇨
⇨ // Display sorted list on ports P0, P1, P2, and P3 periodically
⇨ for(i = 0; i < n; i++) {
⇨ if(i % 4 == 0)
⇨ P0 = numbers[i];
⇨ else if(i % 4 == 1)
⇨ P1 = numbers[i];
⇨ else if(i % 4 == 2)
⇨ P2 = numbers[i];
⇨ else if(i % 4 == 3)
⇨ P3 = numbers[i];
⇨
⇨ delay(150); // Delay of 150 milliseconds
⇨ }
⇨ }
⇨ }
```

To be done using EdSim51 simulator in 8051:

1. Write an assembly language program to sort the string “embedded systems” in ascending order.

```
⇨ ORG 0H
⇨
⇨ MOV DPTR, #String ; Point to the string
⇨ CALL SortString ; Call the sorting subroutine
⇨ SJMP $ ; Infinite loop
⇨
⇨ SortString:
⇨ MOV R0, #0 ; Outer loop counter
⇨ OuterLoop:
⇨ MOV R1, R0 ; Inner loop counter
⇨ INC R1 ; Point to next character
⇨ InnerLoop:
⇨ MOV A, @DPTR ; Load character at current position
⇨ CJNE A, #' ', SkipCheck ; If not space, skip to next character
⇨ INC DPTR ; Move to next character
⇨ SJMP Continue ; Continue with next iteration
⇨ SkipCheck:
⇨ MOV A, @DPTR ; Load character at current position
⇨ INC DPTR ; Move to next character
⇨ CJNE A, #' ', CheckNext ; If not space, skip to next check
⇨ SJMP Continue ; Continue with next iteration
⇨ CheckNext:
⇨ MOV R2, A ; Store character in R2
⇨ MOV A, @DPTR ; Load next character
⇨ CJNE A, #' ', Compare ; If not space, compare characters
⇨ SJMP Continue ; Continue with next iteration
⇨ Compare:
⇨ CJNE R2, A, NotSwap ; If characters are in correct order, skip swap
⇨ MOVX A, @DPTR ; Load character to be swapped
⇨ DEC DPTR ; Move back to original position
⇨ XCH A, @DPTR ; Swap characters
⇨ INC DPTR ; Move to next character
⇨ NotSwap:
⇨ INC DPTR ; Move to next character
⇨ DJNZ R1, InnerLoop ; Decrement inner loop counter and repeat if not zero
⇨ Continue:
⇨ INC R0 ; Increment outer loop counter
⇨ CJNE R0, #14, OuterLoop ; Continue sorting until end of string
⇨ RET
⇨
⇨ String: DB "embedded systems$"
```

2. You are required to count number of times “d” occurred in the string of “embedded systems” and display it at memory location using indirect addressing mode.

```
⇨ ORG 0H
⇨
⇨ MOV DPTR, #String ; Point to the string
⇨ MOV R1, #0 ; Initialize counter
⇨ MOV R2, #'d' ; Search character
⇨
⇨ CountLoop:
⇨ MOV A, @DPTR ; Load character
⇨ CJNE A, #'$', EndCount ; If end of string, exit loop
⇨ CJNE A, R2, NotMatch ; If not 'd', skip increment
⇨ INC R1 ; Increment counter
⇨ NotMatch:
⇨ INC DPTR ; Move to next character
⇨ SJMP CountLoop ; Repeat loop
⇨ EndCount:
⇨ MOV @R0, R1 ; Store the count at memory location pointed by R0
```

- ⇨ **SJMP $; Infinite loop**
- ⇨
- ⇨ **String: DB "embedded systems$"**