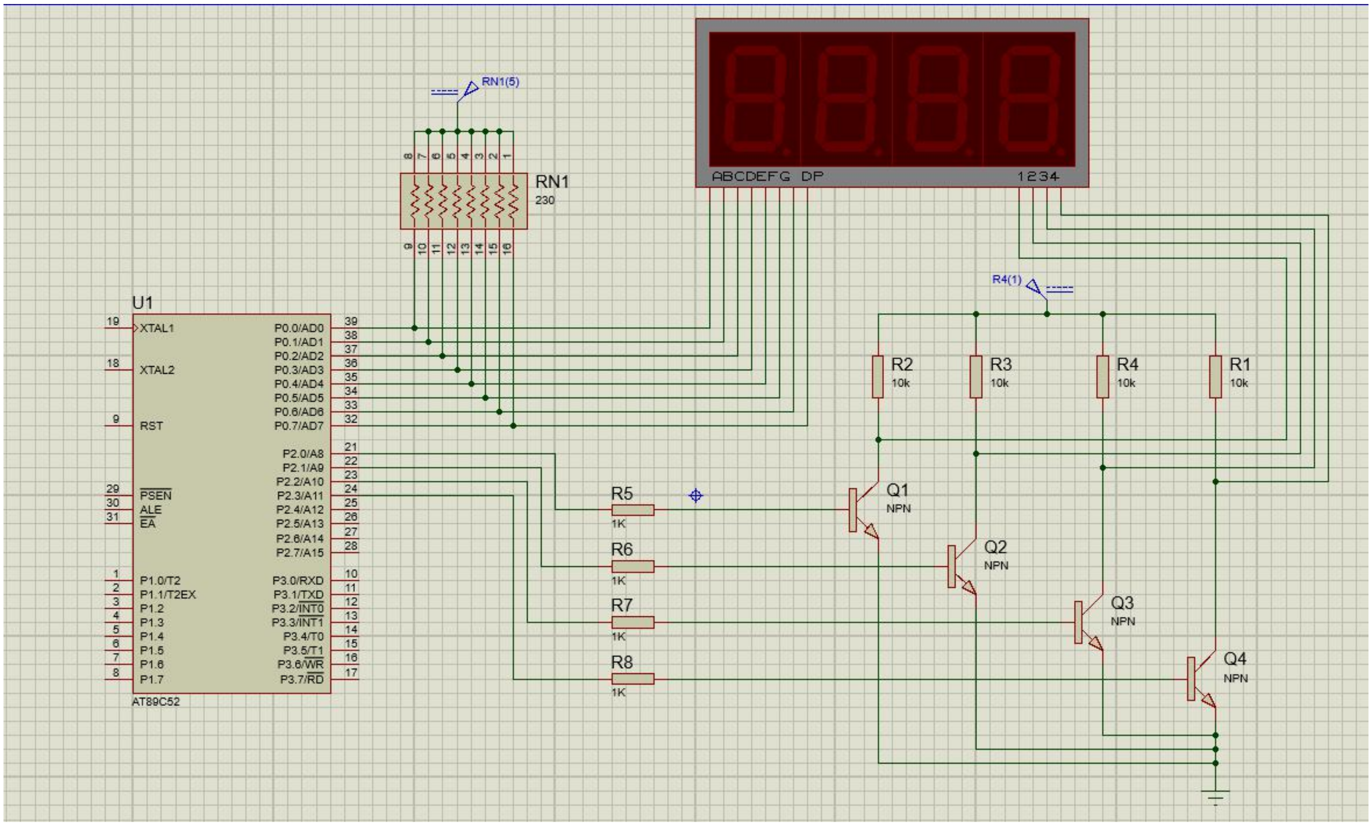


Fig: Circuit Diagram Interfacing 7 Segment Display with AT89C52 Micro-controller

1. Write a code to design a single digit decimal counter that counts up from 0 to 9 and back to 0. This process should repeat indefinitely.

<u>Program Code</u>:

```
ORG 00H

MOV 40H, #3FH

MOV 41H, #06H

MOV 42H, #5BH

MOV 43H, #4FH

MOV 44H, #66H

MOV 45H, #6DH
```

```
        MOV 46H, #7DH

        MOV 47H, #07H

        MOV 48H, #7FH

        MOV 49H, #6FH

        MOV P2, #01H

REPEAT:

        MOV R0, #40H

        MOV R7, #0AH

INCREASE:

        MOV P0, @R0

        INC R0

        ACALL DELAY

        DJNZ R7, INCREASE

        MOV R7, #08H

        DEC R0

DECREASE:

        DEC R0

        MOV P0, @R0

        ACALL DELAY

        DJNZ R7, DECREASE

        AJMP REPEAT

DELAY:

        MOV R3, #7

LOOP3:

        MOV R4, #255
```

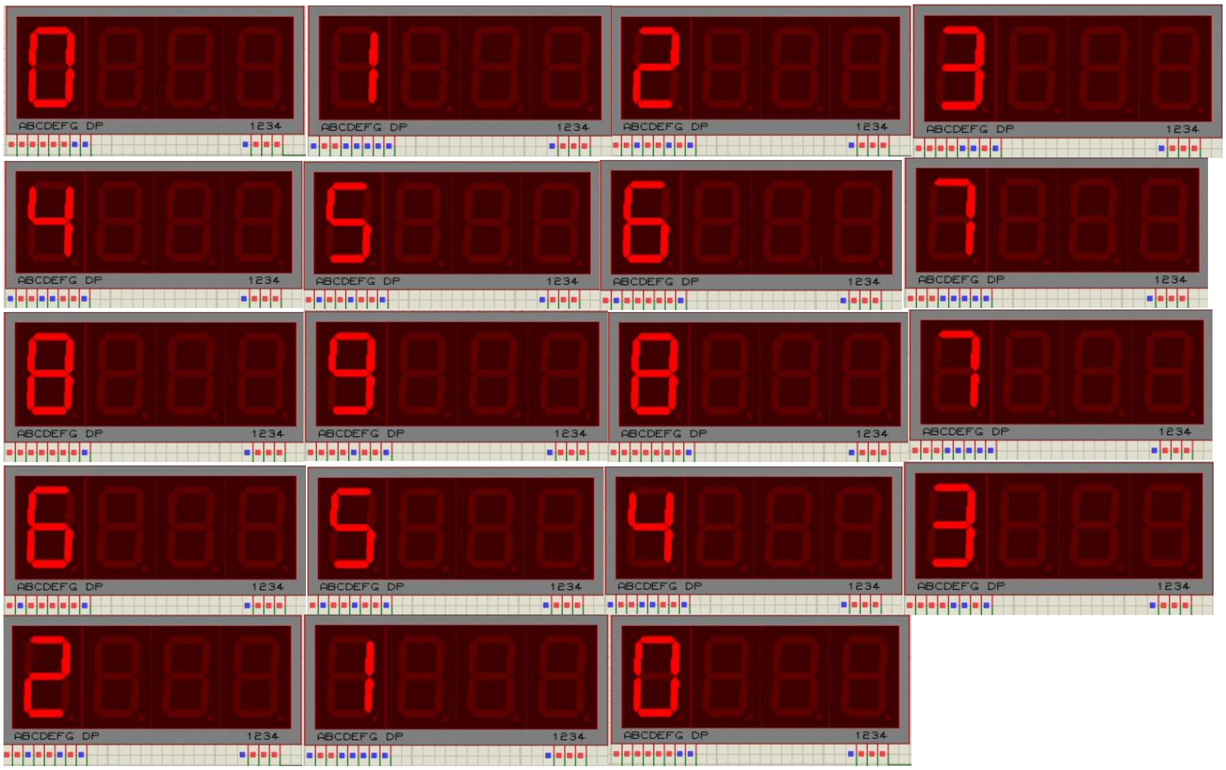
```
LOOP2:

        MOV R5, #255

LOOP1:

        DJNZ R5, LOOP1

        DJNZ R4, LOOP2

        DJNZ R3, LOOP3

        RET

        END
```

C Program Code:

```
#include <reg51.h>

unsigned char led_pattern[10] = {

        0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d,

        0x7d, 0x07, 0x7f, 0x6f};

void delay(int time)

{

        unsigned int i,j;

        for (i=0;i<time;i++)

        for (j=0;j<125;j++);

}

void display(int i)

{

        P0 = led_pattern[i];

        delay(1000);

}

void main(void)
```

```
{
        unsigned int i;
        P2 = 0x01;
        while(1)
        {
                for(i=0; i<10; i++)
                        display(i);
                for(i=8; i>0; i--)
                        display(i);
        }
}
```

Results:



2. Write a code to design a double digit decimal counter that counts up from 00 to 20 and back to 00 indefinitely.

Program Code:

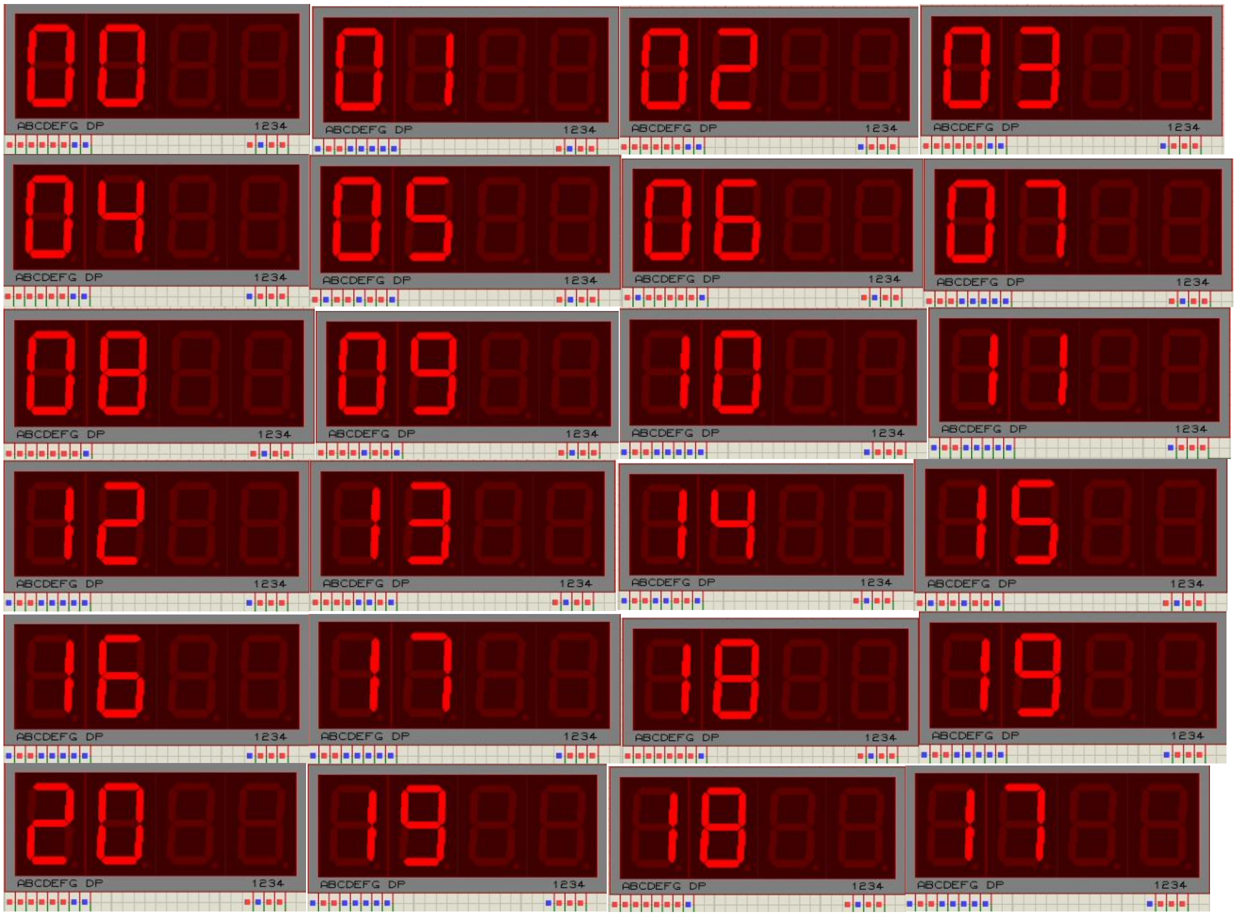
```
        ORG 00H
        MOV 40H, #3FH
        MOV 41H, #06H
        MOV 42H, #5BH
        MOV 43H, #4FH
        MOV 44H, #66H
        MOV 45H, #6DH
        MOV 46H, #7DH
        MOV 47H, #07H
        MOV 48H, #7FH
        MOV 49H, #6FH
        MOV 4AH, #3FH
        MOV 50H, 40H
        MOV 51H, 41H
        MOV 52H, 42H
AGAIN:  MOV R1, #50H
        MOV R6, #02H
LOOP2:  MOV R0, #40H
        MOV R5, #0AH
LOOP1:  MOV R7, #255
MAIN:   MOV A,@R1
        MOV P2, #01H
        MOV P0, A
        ACALL DELAY
        MOV A, @R0
        MOV P2, #02H
        MOV P0, A
        ACALL DELAY
        DJNZ R7, MAIN
        INC R0
        DJNZ R5, LOOP1
        INC R1
        DJNZ R6, LOOP2
        MOV R7, #255
LOP:    MOV A, @R1
        MOV P2, #01H
        MOV P0, A
        ACALL DELAY
        MOV A, @R0
        MOV P2, #02H
        MOV P0, A
        ACALL DELAY
        DJNZ R7, LOP
        DEC R1
        MOV R6, #02H
LOOP22: MOV R0, #49H
        MOV R5, #0AH
```

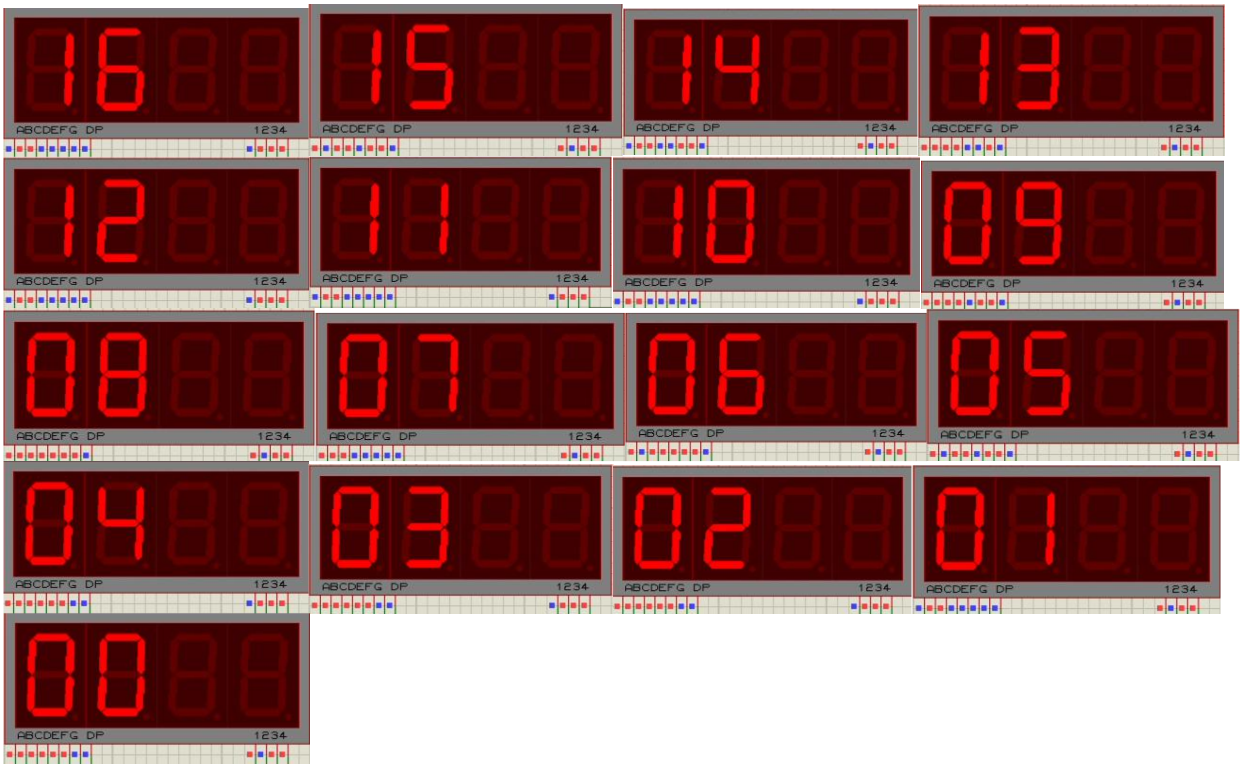
```
LOOP11: MOV R7, #255

MAIN_D: MOV A, @R1

        MOV P2, #01H

        MOV P0, A

        ACALL DELAY

        MOV A, @R0

        MOV P2, #02H

        MOV P0, A

        ACALL DELAY

        DJNZ R7, MAIN_D

        DEC R0

        DJNZ R5, LOOP11

        DEC R1

        DJNZ R6, LOOP22

        AJMP AGAIN

DELAY:

        MOV R3, #02H

DEL1:   MOV R2, #0FAH

DEL2:   DJNZ R2, DEL2

        DJNZ R3, DEL1

        RET

        END
```

C Program Code:

```
#include <reg51.h>

unsigned char led_pattern[10] = {
       0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d,
       0x7d, 0x07, 0x7f, 0x6f};
void delay(int time)
{
       unsigned int i,j;
       for (i=0;i<time;i++)
       for (j=0;j<125;j++);
}
void display(unsigned int i)
{
       unsigned int j, led1, led2;
       led1 = i / 10;
       led2 = i % 10;
       for(j=0; j<10; j++)
       {
              P2 = 0x1;
              P0 = led_pattern[led1];
              delay(40);
              P2 = 0x2;
              P0 = led_pattern[led2];
              delay(40);
       }
}
void main(void)
{
       unsigned int i;

       while(1)

       {

              for(i=0; i<20; i++)

                      display(i);

              for(i=20; i>0; i--)

                      display(i);

       }

}
```

Results:





3.Write a code to display the first (N) numbers of the Fibonacci sequence, where the number (N) must be stored in a memory location and can be any integer from 1 to 10. The sequence should repeat indefinitely.

<u>Program Code</u>:

```
ORG 00H
        MOV P2, #00H
        MOV DPTR, #LABEL1
        MOV R0, #50H
        MOV R7, #8
        MOV A, R7
        MOV R6, A

; FIRST TWO TERMS OF FIBONACCI SEQUENCE
        MOV R1, #00H
        MOV R2, #01H
        MOV A, R1
        MOV @R0, A
        INC R0
        DEC R6
        MOV A, R2
        MOV @R0, A
        INC R0
        DEC R6
; CALCULATION OF FIBONACCI TERMS
        AGAIN: MOV A, R1
        ADD A, R2
        MOV @R0, A
        INC R0
        MOV B, R2
        MOV R1, B
        MOV R2, A
        DJNZ R6, AGAIN
; HEX TO DEC CONVERTER
        MOV R0, #50H
        MOV A, R7
        MOV R6, A
        AGN2: MOV A, @R0
        MOV R4, #00H
        MOV B, #0AH
        DIV AB
        MOV R2, A
        SUBB A, #0AH
        JC SKIP
        MOV A, R2
        MOV R3, B
        MOV B, #0AH
        DIV AB
        MOV R4, A
```

```
       MOV A, B
       MOV B, R3
SKIP: MOV A, R2
       SWAP A
       ADD A, B
       MOV B, R4
       MOV @R0, A
       INC R0
       DJNZ R6, AGN2
; DISPLAY
       REPEAT: MOV R0, #50H
       MOV A, R7
       MOV R4, A
LOOP1: MOV R6, #255
MAIN:  MOV A, @R0
       MOV B, A
       ANL A, #0FH
       MOV P2, #02H
       ACALL DISPLAY
       MOV P0, A
       ACALL DELAY
       MOV A, B
       ANL A, #0F0H
       SWAP A
       MOV P2, #01H
       ACALL DISPLAY
       MOV P0, A
       ACALL DELAY
       DJNZ R6, MAIN
       INC R0
       DJNZ R4, LOOP1
       AJMP REPEAT
```

```
DELAY: MOV R3, #02H
DEL1: MOV R2, #0FAH
DEL2: DJNZ R2, DEL2
        DJNZ R3, DEL1
        RET
; CHOOSE REQUIRED PATTERN
DISPLAY: MOVC A, @A + DPTR
        RET
; LED PATTERNS FOR NUMBERS 0-9
LABEL1: DB 3FH
        DB 06H
        DB 5BH
        DB 4FH
        DB 66H
        DB 6DH
        DB 7DH
        DB 07H
        DB 7FH
        DB 6FH
        END
```

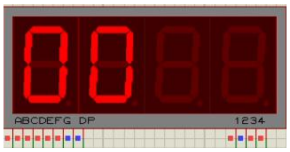
C Program Code:

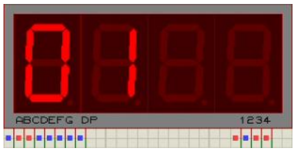
```
#include <reg51.h>
#define N 10
unsigned char led_pattern[10] = {
        0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d,
        0x7d, 0x07, 0x7f, 0x6f};
void delay(int time)
{
        unsigned int i,j;
```

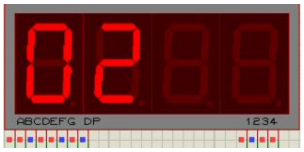
```
    for (i=0;i<time;i++)
    for (j=0;j<125;j++);
}
void display(unsigned int i)
{
    unsigned int j, led1, led2;
    led1 = i / 10;
    led2 = i % 10;
    for(j=0; j<10; j++)
    {
        P2 = 0x1;
        P0 = led_pattern[led1];
        delay(40);
        P2 = 0x2;
        P0 = led_pattern[led2];
        delay(40);
    }
}
void main(void)
{
    unsigned int i, fibo_seq[N]={0, 1};
    for(i=2; i<N; i++)
    fibo_seq[i] = fibo_seq[i-1] +
    fibo_seq[i-2];
    while(1)
```

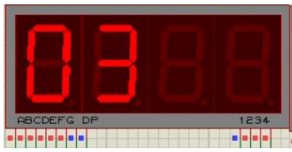
```
        for(i=0; i<N; i++)
                display(fibo_seq[i]);
}
```

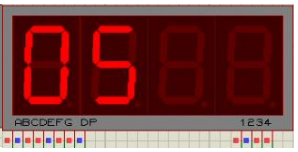
Results:

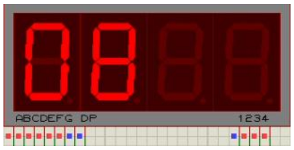


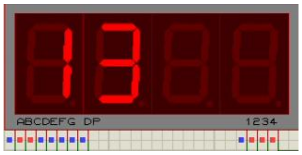












4.Write a code to generate the multiplication table of a number (N) stored in a memory location which can be any integer from 1 to 10. Repeat the sequence indefinitely.

Program Code:

```
ORG 00H
        MOV R7, #8
        MOV P2, #00H
        MOV DPTR, #LABEL1
        MOV B, R7
        MOV R0, #5AH
        MOV R6, #10
AGN:    MOV B, R6
        MOV A, R7
        MUL AB
        MOV @R0, A
        DEC R0
        DJNZ R6, AGN
; HEX TO DEC CONVERTER
        MOV R0, #51H
        MOV R6, #10
```

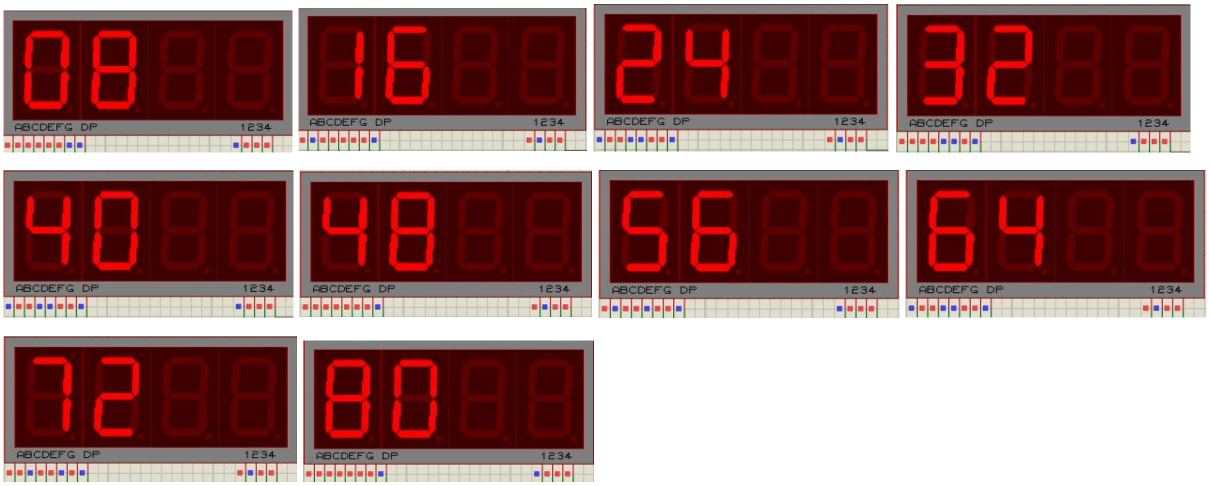
```
AGN2:   MOV A, @R0
        MOV R4, #00H
        MOV B, #0AH
        DIV AB
        MOV R2, A
        SUBB A, #0AH
        JC SKIP
        MOV A, R2
        MOV R3, B
        MOV B, #0AH
        DIV AB
        MOV R4, A
        MOV A, B
        MOV B, R3
        MOV R2, A
SKIP:   MOV A, R2
        SWAP A
        ADD A, B
        MOV B, R4
        MOV @R0, A
        INC R0
        DJNZ R6, AGN2
; DISPLAY
REPEAT: MOV R0, #51H
        MOV R4, #10
LOOP1: MOV R7, #255
MAIN:   MOV A, @R0
        MOV B, A
        ANL A, #0FH
        MOV P2, #02H
        ACALL DISPLAY
        MOV P0, A
        ACALL DELAY
        MOV A, B
        ANL A, #0F0H
        SWAP A
        MOV P2, #01H
        ACALL DISPLAY
        MOV P0, A
        ACALL DELAY
        DJNZ R7, MAIN
        INC R0
        DJNZ R4, LOOP1
        AJMP REPEAT
DELAY: MOV R3, #02H
DEL1: MOV R2, #0FAH
DEL2: DJNZ R2,  DEL2
        DJNZ R3, DEL1
        RET
; CHOOSE REQUIRED PATTERN
DISPLAY: MOVC A,@A+DPTR
        RET
; LED PATTERNS FOR NUMBERS 0-9
LABEL1: DB 3FH
        DB 06H
        DB 5BH
        DB 4FH
        DB 66H
        DB 6DH
        DB 7DH
        DB 07H
        DB 7FH
        DB 6FH
```

```
		END
```

C Program Code:

```
#include <reg51.h>

#define N 6

unsigned char led_pattern[10] = {

	0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d,

	0x7d, 0x07, 0x7f, 0x6f};

void delay(int time)

{

	unsigned int i,j;

	for (i=0;i<time;i++)

	for (j=0;j<125;j++);

}

void display(unsigned int i)

{

	unsigned int j;

	for(j=0; j<15; j++)

	{

		P2 = 0x1;

		P0 = led_pattern[i / 10];

		delay(40);

		P2 = 0x2;

		P0 = led_pattern[i % 10];

		delay(40);

	}
}
void main(void)
{
        unsigned int i;
        while(1)
        for(i=1; i<=10; i++)
                display(N*i);
}
```

Results:



5.Write a code to display the roll numbers of your lab group members one by one in static format. Each student roll number should be of four characters. Display of student roll numbers should repeat indefinitely.

Program Code:

```
ORG 00H
; C090
        MOV 40H, #39H
        MOV 41H, #3FH
        MOV 42H, #6FH
        MOV 43H, #3FH
```

```
; C091
       MOV 44H, #39H
       MOV 45H, #3FH
       MOV 46H, #6FH
       MOV 47H, #06H
; C092
       MOV 48H, #39H
       MOV 49H, #3FH
       MOV 4AH, #6FH
       MOV 4BH, #5BH
; C093
       MOV 4CH, #39H
       MOV 4DH, #3FH
       MOV 4EH, #6FH
       MOV 4FH, #4FH
; DISPLAY
REPEAT: MOV R0, #40H
       MOV R4, #4
LOOP1: MOV R7, #255
MAIN:  MOV A, @R0
       SETB P2.0
       MOV P0, A
       ACALL DELAY
       CLR P2.0
       INC R0
       MOV A, @R0
       SETB P2.1
       MOV P0, A
       ACALL DELAY
       CLR P2.1
       INC R0
       MOV A, @R0
```

```
          SETB P2.2
          MOV P0, A
          ACALL DELAY
          CLR P2.2
          INC R0
          MOV A, @R0
          SETB P2.3
          MOV P0, A
          ACALL DELAY
          CLR P2.3
          DEC R0
          DEC R0
          DEC R0
          DJNZ R7, MAIN
          INC R0
          INC R0
          INC R0
          INC R0
          DJNZ R4, LOOP1
          AJMP REPEAT
DELAY: MOV R3, #02H
DEL1:     MOV R2, #0FAH
DEL2:     DJNZ R2, DEL2
          DJNZ R3, DEL1
          RET
          END
```

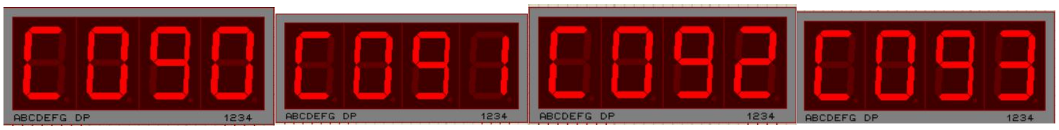
<u>C Program Code</u>:

```
#include <reg51.h>
unsigned char led_pattern[10] = {
        0x3f, 0x06, 0x5b, 0x4f, 0x66, 0x6d,
        0x7d, 0x07, 0x7f, 0x6f};
```

```
    unsigned char dept_init = 0x39;
void delay(int time)
{
    unsigned int i,j;
    for (i=0; i<time; i++)
    for (j=0; j<125; j++);
}
void display(unsigned int i)
{
    unsigned int j, led2, led3, led4;
    led2 = i / 100;
    led3 = (i - led2 * 100) / 10;
    led4 = i - led2 * 100 - led3 * 10;
    for(j=0; j<20; j++)
    {
        P2 = 0x1;
        P0 = dept_init;
        delay(10);
        P2 = 0x2;
        P0 = led_pattern[led2];
        delay(10);
        P2 = 0x4;
        P0 = led_pattern[led3];
        delay(10);
        P2 = 0x8;
```

```
                P0 = led_pattern[led4];

                delay(10);

        }

}

void main(void)

{

        unsigned int i;

        unsigned int roll_no[4] = {090,

        091, 092, 093};

        while(1)

        for(i=0; i<4; i++)

                display(roll_no[i]);

}
```

Results:



6.Write a code to display the roll numbers of your lab group members in scrolling format, separated by using decimal point. Roll numbers should be scrolled towards the left and is repeated indefinitely.

Program Code:

```
ORG 00H
; C090
        MOV 40H, #39H
        MOV 41H, #3FH
        MOV 42H, #6FH
        MOV 43H, #0BFH
```

```
; C091
        MOV 44H, #39H
        MOV 45H, #3FH
        MOV 46H, #6FH
        MOV 47H, #86H
; C092
        MOV 48H, #39H
        MOV 49H, #3FH
        MOV 4AH, #6FH
        MOV 4BH, #0DBH
; C093
        MOV 4CH, #39H
        MOV 4DH, #3FH
        MOV 4EH, #6FH
        MOV 4FH, #0CFH
; C09
        MOV 50H, #39H
        MOV 51H, #3FH
        MOV 52H, #6FH
; DISPLAY
REPEAT: MOV R0, #40H
        MOV R4, #10H
LOOP1: MOV R7, #255
MAIN: MOV A, @R0
        SETB P2.0
        MOV P0, A
        ACALL DELAY
        CLR P2.0
        INC R0
        MOV A, @R0
        SETB P2.1
        MOV P0, A
        ACALL DELAY
        CLR P2.1
        INC R0
        MOV A, @R0
        SETB P2.2
        MOV P0, A
        ACALL DELAY
        CLR P2.2
        INC R0
        MOV A, @R0
        SETB P2.3
        MOV P0, A
        ACALL DELAY
        CLR P2.3
        DEC R0
        DEC R0
        DEC R0
        DJNZ R7, MAIN
        INC R0
        DJNZ R4, LOOP1
        AJMP REPEAT
DELAY: MOV R3, #02H
DEL1: MOV R2, #0FAH
DEL2: DJNZ R2, DEL2
        DJNZ R3, DEL1
        RET
        END
```

C Program Code:

```
#include <reg51.h>
unsigned char scroll_pattern[] = {
        0x39, 0x3f, 0x6f, 0xbf, 0x39, 0x3f,
        0x6f, 0x86, 0x39, 0x3f, 0x6f, 0xdb,
        0x39, 0x3f, 0x6f, 0xcf, 0x39, 0x3f,
        0x6f};
void delay(int time)
{
        unsigned int i,j;
        for (i=0; i<time; i++)
        for (j=0; j<125; j++);
}
void display(unsigned int i)
{
        unsigned int j;
        for(j=0; j<20; j++)
        {
                P2 = 0x1;
                P0 = scroll_pattern[i-4];
                delay(10);
                P2 = 0x2;
                P0 = scroll_pattern[i-3];
                delay(10);
                P2 = 0x4;
                P0 = scroll_pattern[i-2];
                delay(10);
                P2 = 0x8;
                P0 = scroll_pattern[i-1];
                delay(10);}
}
void main(void)
{unsigned int i;
        while(1)
        for(i=4; i<20; i++)
                display(i);
}
```