ASSIGNMENT-04(BRANCHING INSTRUCTIONS)

1. Put a random number in R3 and increment it until it equals E1h.

```
CSEG AT 0
MOV R3,#34H
REPEAT:
INC R3
CJNE R3,#0E1H,REPEAT
END
```

1. Put a random number in address 20h and increment it until it equals a random number put in R5.

```
CSEG AT 0
MOV 20H,#34H
MOV R5,#0FFH
MOV A,R5
REPEAT:
INC 20H
CJNE A,20H,REPEAT
MOV R5,A
END
```

2. Detect both the OV flag & CY flag being set in 8051.If set put 1 in R7, else put 0 in R7.

```
CSEG AT 0
MOV A,#0FFH
MOV R0,#0FFH
ADD A,R0
JC LABEL
MOV R7,#00H
SJMP LAST
LABEL:
MOV C,PSW.2
MOV R7,#01H
```

```
LAST:
CLR A
END
```

3. Count the number of 1s in any number in register B and put the count in R5.

```
CSEG AT 0
MOV B,#0FFH
MOV A,B
MOV R0,#8
REPEAT:
RRC A
JNC LABEL
INC R5
LABEL:DJNZ R0,REPEAT
CLR A
END
```

4. Transfer the data in internal RAM locations 10h to 20h to internal RAM locations s30h to 40h.

```
CSEG AT 0
MOV R0,#10H
MOV A,#01H
REPEAT:
MOV @R0,A
INC R0
INC A
CJNE R0,#20H,REPEAT
MOV R0,#10H
MOV R1,#30H
LABEL:
MOV A,@R0
MOV @R1,A
INC R0
INC R1
CJNE R0,#20H,LABEL
CLR A
END
```

5. Write a program to copy a block of 10 bytes of data from RAM locations starting at 35H to RAM locations starting at 60H.

```
CSEG AT 0
MOV R1,#35H
MOV A,#01H
REPEAT:
MOV @R0,A
INC R0
INC A
CJNE R0,#3EH,REPEAT
MOV R0,#35H
MOV R1,#60H
LABEL:
MOV A,@R0
MOV @R1,A
INC R0
INC R1
CJNE R0,#3EH,LABEL
END
```

6. Assuming that in ROM space at 250H contains ‘Vector’, write a program to transfer the bytes into RAM locations starting at 40H.

```
CSEG AT 0250H

ST:DB 'VECTOR'

CSEG AT 0

MOV DPTR,#ST

MOV R0,#40H

MOV R1,#6

REPEAT:

CLR A

MOVC A,@A+DPTR

MOV @R0,A

INC R0

INC DPTR

DJNZ R1,REPEAT

END
```

7. Let the assembler locate (initialize) the string ‘Welcome’ in ROM space. Write an ALP to bring in the string into the RAM space.

```
MY_SEG SEGMENT CODE

RSEG MY_SEG

ST:DB  'WELCOME'

CSEG AT 0

START:

MOV DPTR,#ST

MOV R0,#40H

MOV R1,#7

REPEAT:

CLR A

MOVC A,@A+DPTR

MOV @R0,A

INC DPTR

INC R0

DJNZ R1,REPEAT

END
```

9. Write a program to add the following numbers and save the result in R2, R3. The data is stored in on-chip ROM.
MYDATA:	DB	53, 94, 56, 92, 74, 65, 43, 23, 83

```
CSEG AT 0050H

MYDATA:DB 53,94,56,92,74,65,43,23,83

CSEG AT 0

MOV DPTR,#MYDATA

MOV R3,#00H
MOV R0,#9

REPEAT:

CLR A

MOVC A,@A+DPTR

ADD A,R3

JNC LABEL

INC R2

LABEL:

MOV R3,A

INC DPTR

DJNZ R0,REPEAT

END
```

10. Write a sub-routine that adds to 8-bit numbers and stores the result in r6(MSB) and r7(LSB) and call it.

```
CSEG AT 0

MAIN:

ACALL ADD_8BIT

MOV R7,A

MOV 20H.0,C

MOV R6,20H

CSEG AT 0030H

ADD_8BIT:

MOV A,#0FFH

MOV R0,#0FFH

ADD A,R0

RET

END
```

10. Write a sub-routine to create a delay of about 1 ms and call it.

```
CSEG AT 0
```

```
MAIN:
ACALL DELAY_1ms
MOV A,#55H
CSEG AT 0005H
DELAY_1ms:
MOV R0,#250
DJNZ R0,$
MOV R0,#247
DJNZ R0,$
RET
END
```

12.Write a sub-routine to create any approximate delay within of 1 ms up to 100ms.

```
CSEG AT 0
MAIN:
ACALL DELAY_100ms
MOV A,#44H
CSEG AT 0050H
DELAY_1ms:
MOV R0,#250
DJNZ R0,$
MOV R0,#247
DJNZ R0,$
RET
DELAY_100ms:
MOV R1,#100
REPEAT:
ACALL DELAY_1ms
DJNZ R1,REPEAT
RET
```

END