# CODES

**OPERATIONS**

```
org 0000h
//add
mov a,#25h
mov b,#12h
add a,b
mov 10h,a
//sub
mov a,#25h
subb a,b
mov 11h,a
//mul
mov a , #25h
mul ab
mov 12h,a
mov 13h,b
//div
mov a,#25h
mov b,#12h
div ab
mov 14h,a
mov 15h,b
//logical operators
//AND
mov a,#25h
mov b,#12h
anl a,b
mov 10h,a
//OR
mov a,#25h
mov b,#12h
orl a,b
mov 17h,a
//XOR
mov a,#25h
mov b,#12h
xrl a,b
mov 18h,a
//complement
cpl a,
mov 19h,a
//clear
clr a
mov 20h, a
halt:sjmp halt
end
```

**To add following data and then use the simulator to examine the EY flag**

```
org 0000h
mov a,#92h
mov r0,#23h
add a,r0
```

```
jnc l1
inc r7
l1:mov r1,#66h
add a,r1
jnc l2
inc r7
l2:mov r2,#87h
add a,r2
jnc l3
inc r7
l3:mov r3,#073h
add a,r3
jnc l4
jnc r7
l4:
here:sjmp here
end
```

**TO LOAD VALUE INTO R0 R4 THEN PUSH INTO STACK THEN CLEAR AND POP THEM BACK**

```
org 0000h
mov r0,#25h
mov r1,#35h
mov r2,#45h
mov r3,#55h
mov r4,#65h
push 0
push 1
push 2
push 3
push 4
mov r0,#00h
mov r1,#00h
mov r2,#00h
mov r3,#00h
mov r4,#00h
pop 0
pop 1
pop 2
pop 3
pop 4
end
```

**CALCULATE CUBE OF A NUMBER**

```
org 0000h
mov a,20h
mov b,a
mul ab
mov r0,b
mov b,20h
mul ab
mov 23h,a
mov r1,b
mov a,r0
mov b,20h
```

```
mul ab
mov 21h,b
mov b,r1
add a,b
mov 22h,a
mov a,21h
addc a,#0h
mov 21h,a
stop:sjmp stop
end
```

**FACTORIAL**

```
org 0000h
mov r1,#05h
mov r7,#01h
lcall fact
mov r7,a
fact:
mov a,r7
cjnc r1,#00,up
sjmp up1
up:
mov b,r1
mul ab
djnz r1,up
up1:
ret
end
```

**LARGEST NUMBER IN THE SERIES**

```
ORG 0000H
MOV R0,#50H
MOV A,@R0
MOV R2,A
DEC R2
INC R0
MOV B,@R0
INC RO
BACK:MOV A,@R0
CJNE A,B,LOOP
JMP LOOP1
LOOP: JC LOOP1
MOV B,A
LOOP1: INC R0
DJNZ R2,BACK
NEXT: MOV 60H,B
END
```

**SMALLEST NUMBER**

```
ORG 0000H
MOV R0,#50H
MOV A,@R0
MOV R2,A
DEC R2
INC R0
```

```
MOV B,@R0
INC RO
BACK:MOV A,@R0
CJNE A,B,LOOP
JMP LOOP1
LOOP: JNC LOOP1
MOV B,A
LOOP1: INC R0
DJNZ R2,BACK
NEXT: MOV 60H,B
END
```

**VIT UNIVERSITY**

```
ORG 0000H
MOV DPTR,#MYDATA
MOV R0,#40H
MOV R2,#13H
BAC:CLR A
MOVC A,@A+DPTR
MOV @R0,A
INC DPTR
INC R0
DJNZ R2,BACK
HERE:SJMP HERE
ORG 200H
MY DATA:DB "VIT UNIVERSITY"
END
```

**TRANSFERRING STRING ADDRESS STARTING FROM 200H TO RAM LOCATIONS STARTING AT 40H AND THEN MOVE TO 60H**

```
ORG 0000H
MOV DPTR,#MYDATA
MOV R0,#40H
MOV R2,#11H
MOV R1,#60H
MOV R3,#11H
BAC:CLR A
MOVC A,@A+DPTR
MOV @R0,A
INC DPTR
INC R0
DJNZ R2,BACK
MOV R0,#40H
AGAIN:MOV A,@R0
MOV @R1,A
INC R0
INC R1
DJNZ R3,AGAIN
HERE:SJMP HERE
ORG 200H
MY DATA:DB "VIT UNIVERSITY"
END
```

**CONVERSION OF DATA- TO GET A BYTE OF HEX DATA FROM P1, CONVERT IT TO DECIMAL AND THEN TO ASCII. PLACE THE ASCII IN RAM LOCATIONS STARTING AT 40H**

```
ORG 0000H
MOV P1,#0FBH
MOV R0, #40H
MOV A, P1
LOOP: MOV B, #10
DIV AB
XCH A,B
ADD A, #30H
MOV @R0, A
XCH A,B
INC R0
JNZ LOOP
END
```

**TRANSFER CONTENT OF P1 TO P2 USING ACCUMULATOR**

```
mov a, #0ffh ;a=ff hex
mov p1,a ; make p1 and i/p port
back: mov a, p1
mov p2, a
sjmp back
end
```

**GENERATE WAVE FORM HAVING XTAL=11.0592Hz**

```
ORG 000H
MOV TMOD,#10H
AGAIN:MOV TL1,#1AH
MOV TH1,#0FFH
SETB TR1
BACK: JNB TF1,BACK
CLR TR1
CPL P1.5
CLR TF1
SJMP AGAIN
END
```

**TOGGLE ALL BITS OF P1 USING 200 ms WITH CRYSTAL FREQUENCY OF 11.0592Hz**

```
MOV A,#55H
AGAIN: MOV P0,A
MOV P1,A
MOV P2,A
MOV P3,A
ACALL DELAY
CPL A
SJMP AGAIN
DELAY:MOV R5,#2
HERE1:MOV R4,#180
HERE2:MOV R3,#255
HERE3:DJNZ R3,HERE3
DJNZ R4, HERE2
DJNZ R5, HERE1
RET
```

**TO GET 40% AND 70% DUTY CYCLE USING INTERRUPTS**

```
ORG 0000H
BACK:
        SETB P0.1
```

```
        SETB P1.1
        LCALL DELAY
        LCALL DELAY
        LCALL DELAY
        LCALL DELAY
        CPL P0.1
        LCALL DELAY
        LCALL DELAY
        LCALL DELAY
        CPL P1.1
        LCALL DELAY
        LCALL DELAY
        LCALL DELAY
        SJMP BACK
        DELAY:
        MOV R0,#01H
        MOV R1,#01H
        A1:MOV R1,A2
        DJNZ R0,A1
        RET
        END
```

**CREATE A PROGRAM FOR 50 ms on & 50 ms off**

```
ORG 0000H
        LJMP MAIN
ORG 000BH
        DJNZ R0,TEN
        MOV R0,#36
        JB P1.1,ONN
        DJNZ R2,TEN
        MOV R2,#0
        SET B P1.1
        CPL P0.1
        SJMP TEN
        ONN: DJNZ R1,TEN
        MOV R1,#5
        CLR P1.1
        CLR P0.1
        TEN:RET I
ORG 0030H
        MAIN:
        MOV R0,#36
        MOV R1,#5
        MOV R2,#0
        MOV TMOD,#2H
        MOV TH0,#-255
        MOV IE,#82H
        SETB P1.1
        SET B TR0
BACK:SJMP BACK
END
```