**INDEX**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Program** | **Page No.** |
| 1 | Increment a number five time using INR |  |
| 2 | Decrement a number five times using DCR |  |
| 3 | Add two numbers with the sum whose sum will not exceed the processor range (ignore the carry). |  |
| 4 | Subtract two numbers whose result will not exceed the processor range. |  |
| 5 | Add two numbers affecting or not affecting carry flag. |  |
| 6 | Subtract two numbers affecting carry and overflow flag. |  |
| 7 | Check if a number is equal to 10 or not. |  |
| 8 | Check if a number is less than, equal to or greater than 10. |  |
| 9 | Check if two numbers are equal or not. |  |
| 10 | Find the greatest of three numbers. |  |
| 11 | Search an element in an array of ten numbers. |  |
| 12 | Find Sum of ten elements in an array |  |
| 13 | Search a number from an array of ’n’ elements. |  |
| 14 | You are given an array of 25/ (odd number of) elements with every number repeated even number of times except a single number. Find the number which has occurred odd number of times. |  |

**1. Increment a number five time using INR**

data SEGMENT

n1 DB 03h

data ENDS

code SEGMENT

assume CS:code, DS:data

start:

MOV AX,data

MOV DS,AX

MOV CX,05h

MOV AL,n1

l1:

INC AL

LOOP l1

ADD AL,30h

MOV DL,AL

MOV AH,02h

INT 21h

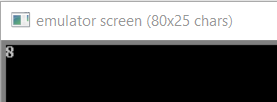
MOV AH,4ch

INT 21h

code ENDS

END start

**OUTPUT**



**2. Decrement a number five times using DCR**

data SEGMENT

n1 DB 08h

data ENDS

code SEGMENT

assume CS:code, DS:data

start:

MOV AX,data

MOV DS,AX

MOV CX,05h

MOV AL,n1

l1:

DEC AL

LOOP l1

ADD AL,30h

MOV DL,AL

MOV AH,02h

INT 21h

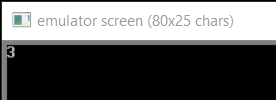
MOV AH,4ch

INT 21h

code ENDS

END start

**OUTPUT**



**3. Add two numbers with the sum whose sum will not exceed the processor range (ignore the carry).**

data SEGMENT

n1 DB 05h

n2 DB 02h

data ENDS

code SEGMENT

assume CS:code, DS:data

start:

MOV AX,data

MOV DS,AX

MOV AL,n1

MOV BL,n2

ADD AL,BL

ADD AL,30h

MOV DL,AL

MOV AH,02h

INT 21h

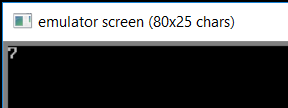
MOV AH,4ch

INT 21h

code ENDS

END start

**OUTPUT**



**4. Subtract two numbers whose result will be positive and within the range of the processor.**

data SEGMENT

n1 DB 07h

n2 DB 02h

data ENDS

code SEGMENT

assume CS:code, DS:data

start:

MOV AX,data

MOV DS,AX

MOV AL,n1

MOV BL,n2

SUB AL,BL

ADD AL,30h

MOV DL,AL

MOV AH,02h

INT 21h

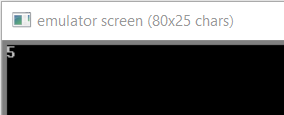
MOV AH,4ch

INT 21h

code ENDS

END start

**OUTPUT**



**5. Add two numbers affecting or not affecting carry flag.**

data SEGMENT

n1 DB 81h

n2 DB 80h

cfMask EQU 01h

data ENDS

code SEGMENT

assume CS:code, DS:data

start:

MOV AX,data

MOV DS,AX

MOV AL,n1

MOV BL,n2

ADD AL,BL

LAHF ; Loads contents of flag register into ah

AND AH, cfMask ; Check if CF is set or not by anding it with 1

MOV DL, AH

ADD DL, 30h

MOV AH, 02h

INT 21h

MOV DL, AL

ADD DL, 30h

MOV AH, 02h

INT 21h

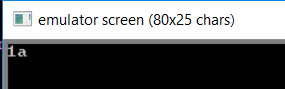
MOV AH, 04ch

INT 21h

code ENDS

END start

**OUTPUT**



**6. Subtract two numbers affecting carry and overflow flag.**

data SEGMENT

a DB 0efh

b DB 01fh

caff DB 'Carry flag =1 i.e. affected.$'

cnotaff DB 'Carry flag not affected. $'

oaff DB 'overflow flag =1 i.e. affected.$'

onotaff DB 'overflow not affected.$'

data ENDS

code SEGMENT

assume CS:code,DS:data

start:

MOV AX,data

MOV DS,AX

MOV AL,a

MOV BL,b

SUB AL,BL

JO oa

JNO ona

oa:

MOV AH,09h

LEA DX,oaff

INT 21h

JC ca

JNC cna

ona:

MOV AH,09h

LEA DX,onotaff

INT 21h

JC ca

JNC cna

ca:

MOV AH,09h

LEA DX,cnotaff

INT 21h

cna:

MOV AH,09h

LEA DX,cnotaff

INT 21h

endit:

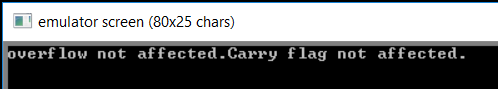
MOV AH,4ch

INT 21h

code ENDS

END start

**OUTPUT**



**7. Check if a number is equal to 10 or not.**

data SEGMENT

msg DB 'input value is not equal to 10 $'

msg2 DB 'input value is equal to 10 $'

n1 DB 08h

data ENDS

code SEGMENT

assume CS:code, DS:data

start:

MOV AX,data

MOV DS,AX

MOV AL,n1

if:

CMP AL, 'A'

JE else

MOV AH, 9

LEA DX, msg

INT 21h

JMP endif

else:

MOV AH, 9

LEA DX, msg2

INT 21h

endif:

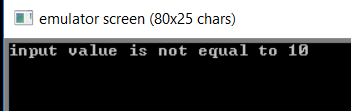
MOV AH,4ch

INT 21h

code ENDS

END start

**OUTPUT**



**8. Check if a number is less than, equal to or greater than 10.**

data SEGMENT

msg DB 'input value is less than 10 $'

msg2 DB 'input value is greater than 10 $'

msg3 DB 'input value is equal to 10 $'

n1 DB 0Ch

data ENDS

code SEGMENT

assume CS:code, DS:data

start:

MOV AX,data

MOV DS,AX

MOV AL,n1

CMP AL, 0Ah

JE EQUAL

JMP NOTEQUAL

EQUAL:

MOV AH, 9

LEA DX, msg3

INT 21h

JMP END:

NOTEQUAL:

JC LESSTHAN

MOV AH, 9

LEA DX, msg2

INT 21h

JMP END

LESSTHAN:

MOV AH, 9

LEA DX, msg

INT 21h

END:

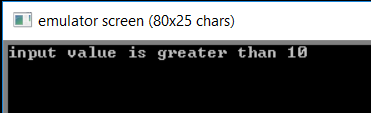
MOV AH,4ch

INT 21h

code ENDS

END start

**OUTPUT**



**9. Check if two numbers are equal or not.**

data SEGMENT

msg DB 'numbers are not equal $'

msg2 DB 'numbers are equal $'

n1 DB 07h

n2 DB 07h

data ENDS

code SEGMENT

assume CS:code, DS:data

start:

MOV AX,data

MOV DS,AX

MOV BL,n1

MOV CL,n2

if:

CMP BL,CL

JE else

MOV AH, 9

LEA DX, msg

INT 21h

JMP endif

else:

MOV AH, 9

LEA DX, msg2

INT 21h

endif:

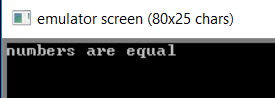
MOV AH,4ch

INT 21h

code ENDS

END start

**OUTPUT**



**10. Find the greatest of three numbers.**

DATA SEGMENT

NUM1 DB 05h

NUM2 DB 02h

NUM3 DB 07h

LRGT DB ?

data ENDS

CODE SEGMENT

ASSUME DS:DATA, CS:CODE

START:

MOV AX,DATA

MOV DS,AX

MOV AL,NUM1

MOV LRGT,AL

CMP AL,NUM2

JGE SKIP1

MOV AL,NUM2

MOV LRGT,AL

SKIP1:

MOV AL,LRGT

CMP AL,NUM3

JGE SKIP2

MOV AL,NUM3

MOV LRGT,AL

SKIP2:

MOV AL, LRGT

ADD AL,30h

MOV DL,AL

MOV AH,02h

INT 21h

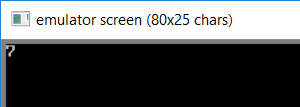
MOV AH,4CH

INT 21H

code ENDS

END START

**OUTPUT**



**11. Search an element in an array of ten numbers.**

DATA SEGMENT

MSG1 DB 10,13,'CHARACTER FOUND :) $'

MSG2 DB 10,13,'CHARACTER NOT FOUND :($'

MSG3 DB 10,13,'ENTER 10 NUMBERS : $'

MSG4 DB 10,13,'ENTER THE NUMBER TO BE SEARCHED : $'

NEW DB 10,13,'$'

INST DB 10 DUP(0)

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE,DS:DATA

START:

MOV AX,DATA

MOV DS,AX

LEA DX,MSG3

MOV AH,09H

INT 21H

MOV BX,00

UP:

MOV AH,01H

INT 21H

CMP AL,0DH

JE DOWN

MOV [INST+BX],AL

INC BX

JMP UP

DOWN:LEA DX,NEW

MOV AH,09H

INT 21H

LEA DX,MSG4

MOV AH,09H

INT 21H

MOV AH,01H

INT 21H

MOV CX,BX

MOV DI,0

UP1:

CMP AL,[INST+DI]

JE DOWN1

INC DI

LOOP UP1

LEA DX,MSG2

MOV AH,09H

INT 21H

JMP FINISH

DOWN1:

LEA DX,MSG1

MOV AH,09H

INT 21H

FINISH:

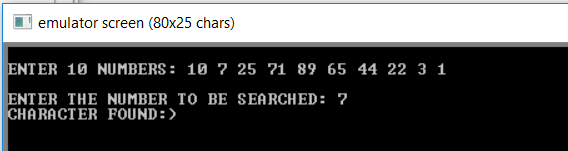
INT 3

CODE ENDS

END START

END

**OUTPUT**



**12. Find Sum of ten elements in an array**

DATA SEGMENT

ARR DB 5,3,7,1,9,2,6,8,4,10

LEN DW $-ARR

SUM DW ?

DATA ENDS

CODE SEGMENT

ASSUME DS:DATA, CS:CODE

START:

MOV AX,DATA

MOV DS,AX

LEA SI,ARR

MOV AX,0

MOV CX,LEN

REPEAT:

MOV BL,ARR[SI]

MOV BH,0

ADD AX,BX

INC SI

LOOP REPEAT

MOV SUM, AX

MOV AX,SUM

MOV AH,0

MOV DL,10

DIV DL

ADD AX,3030h

MOV DH, AH

MOV DL, AL

MOV AH,02h

INT 21h

MOV DL,DH

INT 21h

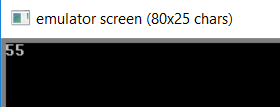
MOV AX,4c00H

INT 21H

CODE ENDS

END START

**OUTPUT**



**13. Search a number from an array of ’n’ elements.**

DATA SEGMENT

MSG1 DB 10,13,'CHARACTER FOUND :) $'

MSG2 DB 10,13,'CHARACTER NOT FOUND :($'

MSG3 DB 10,13,'ENTER THE STRING : $'

MSG4 DB 10,13,'ENTER THE CHARACTER  TO BE SEARCHED : $'

NEW DB 10,13,'$'

INST DB 10 DUP(0)

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE,DS:DATA

START:

MOV AX,DATA

MOV DS,AX

LEA DX,MSG3

MOV AH,09H

INT 21H

MOV BX,00

UP:

MOV AH,01H

INT 21H

CMP AL,0DH

JE DOWN

MOV [INST+BX],AL

INC BX

JMP UP

DOWN:LEA DX,NEW

MOV AH,09H

INT 21H

LEA DX,MSG4

MOV AH,09H

INT 21H

MOV AH,01H

INT 21H

MOV CX,BX

MOV DI,0

UP1:

CMP AL,[INST+DI]

JE DOWN1

INC DI

LOOP UP1

LEA DX,MSG2

MOV AH,09H

INT 21H

JMP FINISH

DOWN1:

LEA DX,MSG1

MOV AH,09H

INT 21H

FINISH:

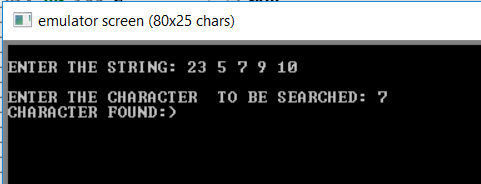
INT 3

CODE ENDS

END START

END

**OUTPUT**



**14. You are given an array of 25/ (odd number of) elements with every number repeated even number of times except a single number. Find the number which has occurred odd number of times.**

data SEGMENT

ARR DB 1,1,1,1,1,1,2,2,2,2,2,2,3,3,3,3,3,3,4,4,4,4,4,4,5

data ENDS

code SEGMENT

ASSUME CS:code,DS:data

START:

MOV AX,data

MOV DS,AX

LEA SI,ARR

MOV AL,0

MOV BL,0

MOV CL,25

XORING:

MOV BL,ARR[SI]

XOR AL,BL

INC SI

LOOP XORING

MOV AL,BL

ADD AL,30h

MOV DL,AL

MOV AH,02h

INT 21h

MOV AH,4ch

INT 21h

code ENDS

END START

**OUTPUT**

