

**Artificial Intelligence and Data Science Department.**

MP / Even Sem 2021-22 / Experiment 8.

YASH SARANG.

47 / D6AD.

EXPERIMENT - 8.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**AIM:** Assembly program to find the GCD/ LCM of two numbers.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**THEORY:**

The program loads two registers with two Numbers and then applies the logic for the GCD of two Numbers. GCD of two numbers is performed by dividing the greater number by the smaller number till the remainder is zero. If it is zero, the divisor is the GCD if not the remainder and the divisor of the previous division are the new set of two numbers. The process is repeated by dividing the greater of the two numbers by the smaller number till the remainder is zero and GCD is found.

Algorithm for GCD of Two Numbers

Step I: Initialize the data segment.

Step II: Load AX and BX registers with the operands.

Step III: Check if the two numbers are equal. If yes go to step X, else go to step IV.

Step IV: Is number 1 > number 2 ? If yes go to step VI else go to step V.

Step V: Exchange the contents of the AX and BX registers, such that AX contains the bigger number.

Step VI Initialize DX register with 00H.

Step VII: Perform the division operation (contents of AX/contents of BX).

Step VIII: Check if there is the remainder. If yes go to step IX, else go to step X.

Step IX: Move the remainder into the AX register and go to step IV.

Step X: Save the contents of BX as GCD.

Step XI: Display the result.

Step XII: Stop.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Program:** Program code for GCD of Two Numbers

.model small

.stack 100

.data

no1 dw 0120

no2 dw 0090

gcd dw 0h

.code

mov ax,@data ; initialize DS

mov ds, ax

mov ax, no1 ; get the first number

mov bx, no2 ; get the second number

again: cmp ax, bx ; check if nos are equal

je endd ; if equal, save the GCD

jb exchg ; if no,

; is AX ; if yes interchange

l2: mov dx, 0

div bx ; check if ax is

; divisible by bx

cmp dx, 0 ;

je endd

mov ax, dx ; mov the remainder

; as no1 data

jmp again

exchg : xchg ax, bx jmp l2

endd : mov gcd, bx

mov ch, 04h ; Count of digits to be

; displayed

mov cl, 04h ; Count to roll by 4 bits

l12: rol bx, cl ; roll bl so that msb

; comes to lsb

mov dl, bl ; load dl with data

; to be displayed

and dl, 0fH ; get only lsb

cmp dl, 09 ; check if digit is 0-9

; or letter A-F

jbe l4

add dl, 07 ; if letter add 37H else

; only add 30H

l4: add dl, 30H

mov ah, 02 ; INT 21H

; (Display character)

int 21H

dec ch ; Decrement Count

jnz l12

mov ah, 4ch

int 21h

end

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**OUTPUT:**

C:\programs>tasm gcd.asm

Turbo Assembler Version 3.0 Copyright (c) 1988, 1991 Borland International

Assembling file: gcd.asm

Error messages: None

Warning messages: None

Passes: 1

Remaining memory: 437k

C:\programs>tlink gcd

Turbo Link Version 3.0 Copyright (c) 1987, 1990 Borland International

C:\programs>gcd

001E

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**