

TABLE OF CONTENTS

1. Practical 1 3
2. Practical 2 15
3. Practical 3 22
4. Practical 4 25
5. Practical 5 35
6. Practical 6 39
7. Practical 7 41

**Question1 Write a program for 32-bit binary Addition ,Subtraction, Division ,and Multiplication . Solution:**

32-Bit Binary Addition:

.model small ;contain two segment data and code

.386 ;instructions for the 80386 processor

.data ; start of data segment

DATA1 dd 00000000H ; initializes memory with 32 bit word

msg db 10,13,"Enter the first no.:: $" ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

msg1 db 10,13,"Enter the second no.:: $" msg2 db 10,13,"The Resultant sum is :: $"

.code ; start of code segment

.startup ;Generates program start-up code

MOV AH,09 ; moving 9 into AH for outputting string MOV DX,OFFSET msg

INT 21H ; calling interrupt to display string whose address present in DX it will check AH value first MOV EBX,0

MOV CX,8 ; setting number of loop

AGAIN: MOV AH,01 ;1ST NO. ENTERED INT 21H

CMP AL,'A' JGE L5

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

JMP L6

L5: SUB AL,37H ; convert the hexadecimal digits into its equivalent ASCII

L6: SHL EBX,4 ; multiply by 16 ADD BL,AL

LOOP AGAIN ; go-to label AGAIN MOV DATA1,EBX

MOV AH,09

MOV DX,OFFSET msg1

INT 21H ; calling interrupt to display string whose address present in DX it will check AH value first MOV EBX,0

MOV CX,8

AGAIN1:MOV AH,01 ;2nd NO. ENTERED INT 21H

CMP AL,'A' JGE L7

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

JMP L8

L7: SUB AL,37H ; convert the hexadecimal digits into its equivalent ASCII

L8: SHL EBX,4 ; multiply by 16 ADD BL,AL

LOOP AGAIN1

ADD EBX,DATA1 ;ADDITION MOV AH,09

MOV DX,OFFSET msg2

INT 21H ; displaying the result MOV CX,8

AGAIN2: ROL EBX,4 rotates the bits within the destination operand to the left MOV DL,BL

AND DL,0FH CMP DL,09

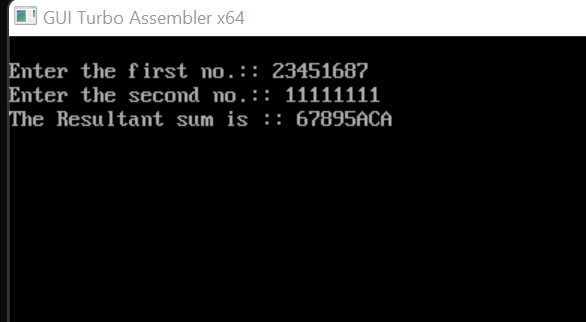
JG L1 ; to o/p given no.

ADD DL,30H JMP PRINT

L1: ADD DL,37H PRINT: MOV AH,02

INT 21H ; Output a character present in DL , as AH value is 2 LOOP AGAIN2 ; go-to label AGAIN2

END



32-Bit Binary Subtration:-

.model small ;contain two segment data and code

.386 ;instructions for the 80386 processor

.data ; start of data segment

DATA1 dd 00000000H ; initializes memory with 32 bit word

msg db 10,13,"Enter the first no.:: $" ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

msg1 db 10,13,"Enter the second no.:: $" msg2 db 10,13,"The Resultant sum is :: $"

.code ; start of code segment

.startup ;Generates program start-up code MOV AH,09

MOV DX,OFFSET msg

INT 21H ; calling interrupt to display string whose address present in DX it will check AH value first MOV EBX,0

MOV CX,8 ; setting number of loop AGAIN: MOV AH,01 ;1ST NO. ENTERED INT 21H

CMP AL,'A' JGE L5

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

JMP L6

L5: SUB AL,37H ; convert the hexadecimal digits into its equivalent ASCII

L6: SHL EBX,4 ; multiply by 16 ADD BL,AL

LOOP AGAIN MOV DATA1,EBX MOV AH,09

MOV DX,OFFSET msg1

INT 21H ; displaying string message MOV EBX,0

MOV CX,8 ; setting number of loop AGAIN1:MOV AH,01 ;2nd NO. ENTERED INT 21H

CMP AL,'A' JGE L7

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

JMP L8

L7: SUB AL,37H ; convert the hexadecimal digits into its equivalent ASCII

L8: SHL EBX,4 ; multiply by 16 ADD BL,AL

LOOP AGAIN1

SUB EBX,DATA1 ;ADDITION MOV AH,09

MOV DX,OFFSET msg2 INT 21H

MOV CX,8

AGAIN2: ROL EBX,4 ; rotates the bits within the destination operand to the left MOV DL,BL

AND DL,0FH CMP DL,09

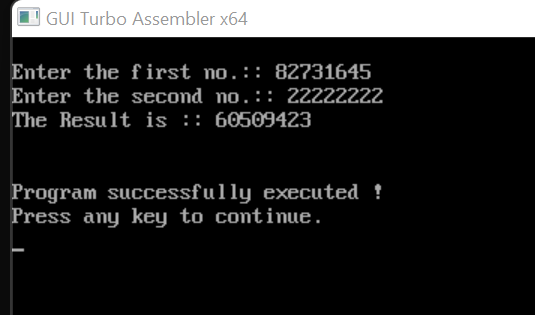
JG L1 ; to o/p given no. ADD DL,30H

JMP PRINT

L1: ADD DL,37H PRINT: MOV AH,02

INT 21H ; Output a character present in DL , as AH value is 2 LOOP AGAIN2

END



32-bit Multiplication:-

.model small ; contain two segment data and code

.386 ;instructions for the 80386 processor

.data ; start of data segment

DATA1 dd 00000000H ; initializes memory with 32 bit word

DATA2 dd 00000000H ; initializes memory with 32 bit word

PROD1 dd ? ; set double word variable

PROD2 dd ? ; set double word variable

msg db 10,13,"Enter the first no.:: $" ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

msg1 db 10,13,"Enter the second no.:: $"

msg2 db 10,13,"The product(in hexadecimal) is :: $"

.code ; start of code segment

.startup ;Generates program start-up code MOV AH,09

MOV DX,OFFSET msg

INT 21H ; calling interrupt to display string whose address present in DX it will check AH value first MOV EBX,0

MOV CX,8 ; setting number of loop

AGAIN: MOV AH,01 ;1ST NO. ENTERED INT 21H

CMP AL,'A' JGE L5

# SUB AL,30H ;convert the hexadecimal digits into its equivalent ASCII

JMP L6

L5: SUB AL,37H ; convert the hexadecimal digits into its equivalent ASCII

L6: SHL EBX,4 ; multiply by 16

ADD BL,AL LOOP AGAIN

MOV DATA1,EBX MOV AH,09

MOV DX,OFFSET msg1

INT 21H ; calling interrupt to display string whose address present in DX it will check AH value first

MOV EBX,0

MOV CX,8 ; setting number of loop

AGAIN1:MOV AH,01 ;2nd NO. ENTERED INT 21H

CMP AL,'A' JGE L7

# SUB AL,30H ;convert the hexadecimal digits into its equivalent ASCII

JMP L8

L7: SUB AL,37H ;convert the hexadecimal digits into its equivalent ASCII

L8: SHL EBX,4 ; multiply by 16 ADD BL,AL

LOOP AGAIN1 ; goto AGAIN1 label

MOV DATA2,EBX MOV EBX,0 MOV EDX,0 MOV EAX,0 MOV EAX,DATA1 MOV EBX,DATA2

MUL EBX

MOV PROD1,EDX MOV PROD2,EAX MOV AH,09

MOV DX,OFFSET msg2

INT 21H ; Output a string terminated by '$’ stored in DX MOV EBX,PROD1

MOV CX,8

AGAIN2: ROL EBX,4 ; rotates the bits within the destination operand to the left

MOV DL,BL

AND DL,0FH ; to o/p the result CMP DL,9

JBE L1

ADD DL,37H MOV AH,02

INT 21H ; Output a character present in DL , as AH value is 2 JMP L2

L1: ADD DL,30H MOV AH,02

INT 21H ; Output a character present in DL , as AH value is 2

L2: LOOP AGAIN2 MOV EBX,PROD2 MOV CX,8

AGAIN3: ROL EBX,4 ; rotates the bits within the destination operand to the left MOV DL,BL

AND DL,0FH ; to o/p the result CMP DL,9

JBE L3

ADD DL,37H MOV AH,02

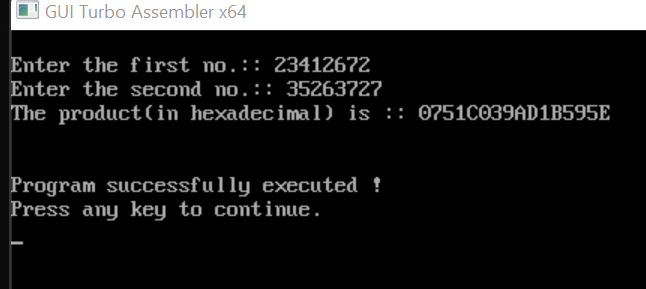
INT 21H ; Output a character present in DL , as AH value is 2 JMP L4

L3: ADD DL,30H MOV AH,02

INT 21H ; Output a character present in DL , as AH value is 2 L4: LOOP AGAIN3

MOV AH,4CH

INT 21H ; causes the current process to terminate END



32-bit division

.model small ; contain two segment data and code

.386 ;instructions for the 80386 processor

.data ; start of data segment

DATA1 dd 00000000H ; initializes memory with 32 bit word DATA2 dd 00000000H ; initializes memory with 32 bit word REM dd ? ; set double word variable

QUO dd ? ; set double word variable

msg db 10,13,"Enter the first no.:: $" ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

msg1 db 10,13,"Enter the second no.:: $" msg2 db 10,13,"The Remainder is :: $" msg3 db 10,13,"The Quotient is :: $"

.code ; start of code segment

.startup ; Generates program start-up code MOV AH,09

MOV DX,OFFSET msg

INT 21H ; calling interrupt to display string whose address present in DX it will check AH value first MOV EBX,0

MOV CX,8 ; setting number of loop

AGAIN: MOV AH,01 ;1ST NO. ENTERED INT 21H

CMP AL,'A' JGE L5 JMP L6

L5: SUB AL,37H ;convert the hexadecimal digits into its equivalent ASCII L6: SUB AL,30H ;convert the hexadecimal digits into its equivalent ASCII SHL EBX,4 ; multiply by 16

ADD BL,AL LOOP AGAIN

MOV DATA1,EBX MOV AH,09

MOV DX,OFFSET msg1

INT 21H ; Output a string terminated by '$’ stored in DX MOV EBX,0

MOV CX,8 ; setting number of loop

AGAIN1:MOV AH,01 ;2nd NO. ENTERED INT 21H

CMP AL,'A' JGE L7

# SUB AL,30H ;convert the hexadecimal digits into its equivalent ASCII

JMP L8

L7: SUB AL,37H ;convert the hexadecimal digits into its equivalent ASCII

L8: SHL EBX,4 ; multiply by 16 ADD BL,AL

LOOP AGAIN1 MOV DATA2,EBX MOV EBX,0 MOV EDX,0 MOV EAX,0 MOV EAX,DATA1 MOV EBX,DATA2 DIV EBX

MOV REM,EDX ;REM=REMAINDER MOV QUO,EAX ;QUO=QUOTIENT MOV AH,09

MOV DX,OFFSET msg2

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 MOV EBX,REM

MOV CX,8

AGAIN2: ROL EBX,4 ; rotates the bits within the destination operand to the left MOV DL,BL

AND DL,0FH ; to o/p the result in rem CMP DL,9

JBE L1

ADD DL,37H MOV AH,02

INT 21H ; Output a character stored in DL, as AH value is 2 JMP L2

L1: ADD DL,30H MOV AH,02

INT 21H ; Output a character stored in DL, as AH value is 2 L2: LOOP AGAIN2

MOV AH,09

MOV DX,OFFSET msg3

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 MOV EBX,QUO

MOV CX,8 ; setting the number of loop

AGAIN3: ROL EBX,4 ; rotates the bits within the destination operand to the left MOV DL,BL

AND DL,0FH ; to o/p the result in quo CMP DL,9

JBE L3

ADD DL,37H MOV AH,02

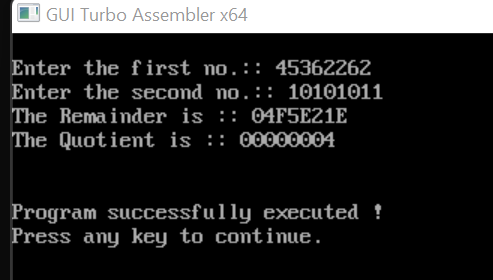
INT 21H ; Output a character present in DL , as AH value is 2 JMP L4

L3: ADD DL,30H MOV AH,02

INT 21H ; Output a character present in DL , as AH value is 2 L4: LOOP AGAIN3

MOV AH,4CH ; causes the current process to terminate INT 21H

END



**Question 2 :-Write a program for 32-Bit BCD Addtion and Subtraction.** Solution:-

32-Bit BCD Addition

.model small ; contain two segment data and code

.386 ;instructions for the 80386 processor

.data ; start of data segment

MESS0 DB 10,13,"ENTER THE FIRST NUMBER:$" ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

MESS1 DB 10,13,"ENTER THE SECOND NUMBER:$" MESS2 DB 10,13,"THE SUM IS:$"

A DD ? ; set double word variable

B DD ? ; set double word variable

C DD ? ; set double word variable

COUNT DB 04h ; reserve byte of memory locations

.code ; start of code segment

.startup ; Generates program start-up code LEA DX,MESS0 ; load the effective address MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV EBX,0 MOV CX,8 AGAIN:

MOV AH,01

INT 21H ; input the number CMP AL,'A'

JGE L5

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

JMP L6

L5: SUB AL,37H ; convert the hexadecimal digits into its equivalent ASCII

L6: SHL EBX,4 ; multiply by 16 ADD BL,AL

LOOP AGAIN MOV A,EBX

LEA DX,MESS1 MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV EBX,0 MOV CX,8 AGAINS:

MOV AH,01

INT 21H ; input the number as AH =01 CMP AL,'A'

JGE L51

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

JMP L61

L51: SUB AL,37H ; convert the hexadecimal digits into its equivalent ASCII

L61: SHL EBX,4 ; multiply by 16 ADD BL,AL

LOOP AGAINS ; Goto label again MOV B,EBX

MOV AX,WORD PTR A ; retrieve only the lowest word (2 bytes) at the "A" address MOV BX,WORD PTR B ; retrieve only the lowest word (2 bytes) at the "B" address

ADD AL,BL

DAA ; represented number in 8-bit packed BCD code MOV BL,AL

ADC AH,BH ; Add with carry BH to AH MOV AL,AH

DAA ; represented number in 8-bit packed BCD code MOV BH,AL

MOV WORD PTR C,BX

MOV AX,WORD PTR A+2 ; retrieve the highest word at the "A" address MOV BX,WORD PTR B+2 ; retrieve the highest word at the "B" address ADC AL,BL ; Add with carry BL to AL

DAA ; represented number in 8-bit packed BCD code MOV BL,AL

ADC AH,BH ; ; Add with carry BH to AH MOV AL,AH

DAA ; represented number in 8-bit packed BCD code MOV BH,AL

MOV WORD PTR C+2,BX

LEA DX,MESS2 MOV AH,09

INT 21H ;Output a string terminated by '$’ stored in DX, as AH =9

MOV BX,WORD PTR C+2 ; replace the highest word at the "C" address MOV DH,2

L1: MOV CH,04H

MOV CL,04H

L2: ROL BX,CL ; rotates the bits within the destination operand to the left MOV DL,BL

AND DL,0FH CMP DL,09 JBE L4

ADD DL,07

L4: ADD DL,30H MOV AH,02

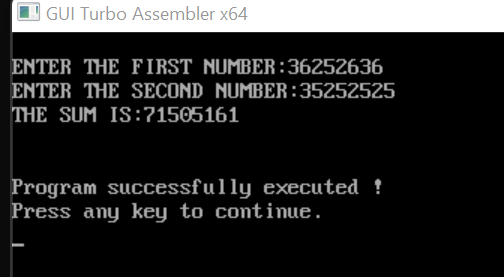
INT 21H ; Output a character present in DL , as AH value is 2 DEC CH

JNZ L2 DEC DH CMP DH,0

MOV BX,WORD PTR C ; replace only the lowest word (2 bytes) at the "C" address JNZ L1

MOV AH,4CH

INT 21H ; causes the current process to terminate END



32-Bit BCD SUBTRATION:-

.model small ; contain two segment data and code

.386 ;instructions for the 80386 processor

.data ; start of data segment

MESS0 DB 10,13,"ENTER THE FIRST NUMBER:$" ; ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

MESS1 DB 10,13,"ENTER THE SECOND NUMBER:$" MESS2 DB 10,13,"THE DIFFERENCE IS:$"

A DD ? ; set double word variable

B DD ? ; set double word variable

C DD ? ; set double word variable

COUNT DB 04h reserve byte of memory locations

.code ; start of code segment

.startup ; Generates program start-up code LEA DX,MESS0 ; load the effective address MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV EBX,0 MOV CX,8 AGAIN:

MOV AH,01

INT 21H ; input the number CMP AL,'A'

JGE L5

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

JMP L6

L5: SUB AL,37H ; convert the hexadecimal digits into its equivalent ASCII

L6: SHL EBX,4 ; multiply by 16

ADD BL,AL

LOOP AGAIN ; goto AGAIN label MOV A,EBX

LEA DX,MESS1 MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV EBX,0 MOV CX,8 AGAINS:

MOV AH,01

INT 21H ; input the number CMP AL,'A'

JGE L51

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

JMP L61

L51: SUB AL,37H ; convert the hexadecimal digits into its equivalent ASCII

L61: SHL EBX,4 ; multiply by 16 ADD BL,AL

LOOP AGAINS ; goto AGAIN label MOV B,EBX

MOV AX,WORD PTR A ; retrieve only the lowest word (2 bytes) at the "A" address MOV BX,WORD PTR B ; retrieve only the lowest word (2 bytes) at the "B" address

SUB AL,BL

DAS ; Adjusts the result of the subtraction to create a packed BCD result MOV BL,AL

SBB AH,BH ; Subtract with borrow BH from AH MOV AL,AH

DAS ; Adjusts the result of the subtraction to create a packed BCD result MOV BH,AL

MOV WORD PTR C,BX ; ; replace only the lowest word (2 bytes) at the "C" address

MOV AX,WORD PTR A+2 ; retrieve the highest word at the "A" address MOV BX,WORD PTR B+2 ; retrieve the highest word at the "B" address SBB AL,BL ;Subtract with borrow BL from AL

DAS ; Adjusts the result of the subtraction to create a packed BCD result MOV BL,AL

SBB AH,BH ; Subtract with borrow BH from AH MOV AL,AH

DAS ; Adjusts the result of the subtraction to create a packed BCD result MOV BH,AL

MOV WORD PTR C+2,BX ; replace the highest word at the "C" address

LEA DX,MESS2 MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV BX,WORD PTR C+2 ; ;replace the highest word at the "C" address MOV DH,2

L1: MOV CH,04H MOV CL,04H

L2: ROL BX,CL ; rotates the bits within the destination operand to the left MOV DL,BL

AND DL,0FH

CMP DL,09 JBE L4 ADD DL,07

L4: ADD DL,30H MOV AH,02

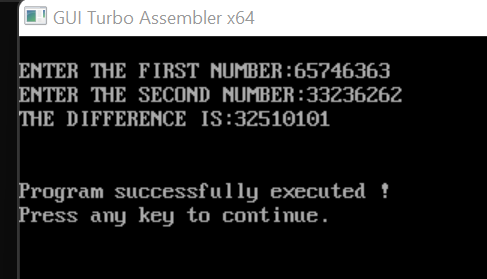
INT 21H ; Output a character present in DL , as AH value is 2 DEC CH

JNZ L2 DEC DH CMP DH,0

MOV BX,WORD PTR C ; retrieve the lowest word at the "C" address JNZ L1

MOV AH,4CH

INT 21H ; causes the current process to terminate END



Question3:- Write a program for Sorting. Solution:-

.model small ; contain two segment data and code

.386 ;instructions for the 80386 processor

.data ; start of data segment

ARRAY DW 20 DUP (?) ; Declaring an array with garbage DATA1 dw 0000H ; initializes memory with word

msg db 10,13,"Enter the size of the array :: $" ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

msg2 db 10,13,"Enter the array :: $" msg3 db 10,13,"The sorted array is :: $"

.code ; start of code segment

.startup ; Generates program start-up code MOV AH,09

MOV DX,OFFSET msg

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 MOV AH,01

INT 21H ; input from user

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

MOV AH,0 MOV CX,AX MOV DATA1,AX MOV AH,09

MOV DX,OFFSET msg2

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 MOV AH,0

MOV SI, 0

MOV BX, OFFSET ARRAY

L1: MOV DL, 0AH ; jump onto next line MOV AH, 02H

INT 21H

MOV DX, SI ; input element of the array MOV AH, 01H

INT 21H

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

MOV SI, DX

MOV [BX + SI], AX ; store at memory location addressed by DS[BX+SI] INC SI

LOOP L1

MOV CX, DATA1

MOV BX, OFFSET ARRAY ; store the offset address of array MOV DI,CX

L2: MOV CX, DATA1 MOV SI, 0

L3: MOV AL, [BX + SI]

CMP AL, [BX + SI + 1] ; compare the value of content in AL and at DS[BX+SI+1] JL L4

XCHG AL,[BX + SI + 1] ; exchange the value of content in AL and at DS[BX+SI+1] MOV [BX + SI],AL

L4: INC SI LOOP L3 DEC DI JNZ L2

MOV CX, DATA1 MOV SI, 1

MOV BX, OFFSET ARRAY MOV AH,09

MOV DX,OFFSET msg3

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

L5: MOV DL, 0AH ; jump onto next line MOV AH, 02H

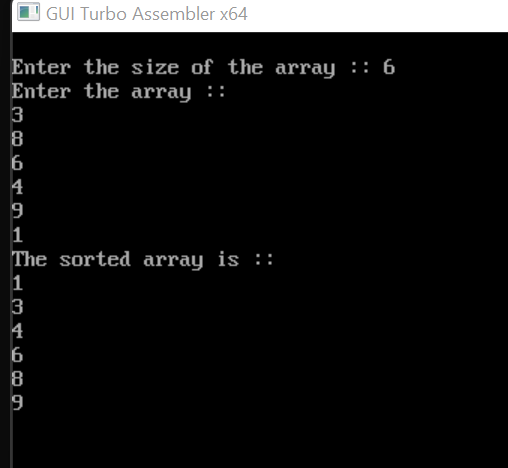
INT 21H

MOV DX, [BX + SI] INC SI

ADD DL, 30H MOV AH, 02

INT 21H ; Output a character present in DL , as AH value is 2 LOOP L5

END



Question4:- Write a program for Linear search and Binary Search. Solution:-

Linear – Search:-

.model small ; contain two segment data and code

.STACK ; tells the assembler to reserve storage

.386 ;instructions for the 80386 processor

.DATA ; start of data segment

ARRAY DB 9 DUP(?) ; Declaring an array with garbage MESS01 DB 13,10,"MAX. NO. OF ELEMENTS IN ARRAY IS 9 $"

MESS02 DB 13,10," $" ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

MESS1 DB 13,10,"ENTER THE NUMBER OF ELEMENTS: $" MESS0 DB 13,10,"ENTER THE NUMBER: $"

MESS2 DB 13,10,"ENTER THE ELEMENT TO BE SEARCHED: $" MESS3 DB 13,10,"VALUE FOUND AT LOCATION - $"

MESS4 DB 13,10,"VALUE NOT FOUND!!!$" ErrMess DB 13,10,"ERROR IN INPUT DIGIT$"

DAT DB ? ; set byte size variable number dw ? ; set double word variable POS DW ? ; set double word variable

.code ; start of code segment

.startup ; Generates program start-up code

MOV DX, OFFSET MESS01 MOV AH, 09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV DX, OFFSET MESS02 MOV AH, 09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV DX,OFFSET MESS1 MOV AH, 09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV AH, 01

INT 21H ; input from user CMP al,39h

JBE abc ; jump if below or equal to

MOV DX, OFFSET ErrMess MOV AH, 09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

JMP myexit

abc: AND AL, 0FH

MOV AH, 0

MOV number, AX

MOV CX, AX ; SET COUNTER AL TIMES MOV DI, 0

; INPUT ELEMENTS IN ARRAY

MYLOOP:

MOV DX, OFFSET MESS0 MOV AH, 09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

; Tens digit

MOV AH, 01

INT 21H ; input from user CMP AL, 39H

JBE abc2 ;; jump if below or equal to

MOV DX, OFFSET ErrMess MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

JMP myexit

abc2: AND al,0fh

SHL AL, 4 ; multiply by 16 MOV BL, AL

; Units digit

MOV AH,01 INT 21H

cmp al,39h

jbe abcx ; jump if below or equal to

MOV DX,OFFSET ErrMess MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 jmp myexit

abcx:

AND al,0fh ADD al, bl

MOV ARRAY[DI], AL INC DI

LOOP MYLOOP

;INPUT ELEMENT TO BE SEARCHED

MOV DX,OFFSET MESS2 MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

; Tens digit

MOV AH,01 INT 21H

cmp al,39h

jbe abcl ; jump if below or equal to

MOV DX,OFFSET ErrMess MOV AH,09

INT 21H

jmp myexit

abcl:

and al,0fh

shl al,4 ; multiply by 16 mov bl,al

; Units digit

MOV AH,01

INT 21H ; input from user cmp al,39h

jbe abcm ; jump if below or equal to

MOV DX,OFFSET ErrMess MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 jmp myexit

abcm: and al,0fh

add al,bl mov DAT,AL

; SEARCH PROCESS

MOV AX, DS MOV ES, AX

MOV AL, DAT

CLD ; Auto-Increment Mode MOV CX, number

MOV DI, OFFSET ARRAY REPNE SCASB

CMP CX, 0

JE NOTFOUND

MOV DX, OFFSET MESS02 MOV AH, 09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV DX, OFFSET MESS3 MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

SUB NUMBER, CX

ADD NUMBER,30H ; convert the hexadecimal digits into its equivalent ASCII

MOV DX, NUMBER MOV AH, 02

INT 21H ; Output a character present in DL , as AH value is 2

JMP myexit

NOTFOUND:

MOV DX,OFFSET MESS4 MOV AH,09

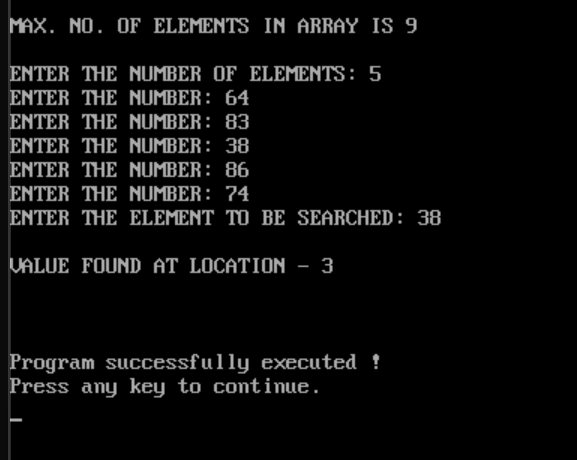
INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

myexit:

MOV DX, OFFSET MESS02 MOV AH, 09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

.EXIT END



Binary-Search :-

.model small ; contain two segment data and code

.stack ; tells the assembler to reserve storage

.386 ;instructions for the 80386 processor

.data ; start of data segment

ARRAY DB 10 DUP(?); Declaring an array with garbage

MESS0 DB 13,10,"ENTER THE NUMBER: $" ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

MESS1 DB 13,10,"ENTER THE NUMBER OF ELEMENTS: $" MESS2 DB 13,10,"ENTER THE ELEMENT TO BE SEARCHED: $" MESS3 DB 13,10,"VALUE FOUND AT LOCATION- $"

MESS4 DB 13,10,"VALUE NOT FOUND!!!$"

ErrMess DB 13,10,"ERROR IN INPUT DIGIT$"

DAT DB ? ;set byte size variable

number dw ? ; set double word size variable

.code ; start of code segment

.startup ; Generates program start-up code

MOV DX,OFFSET MESS1 MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV AH,01

INT 21H ; input from user cmp al,39h

jbe abc ; jump if below or equal to

MOV DX,OFFSET ErrMess MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 jmp myexit

abc:

and al,0fh mov ah,0

mov number,ax MOV CX,AX MOV DI,0

MYLOOP:

MOV DX,OFFSET MESS0 MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 MOV AH,01

INT 21H ; input from user cmp al,39h

jbe abc2 ; jump if below or equal to MOV DX,OFFSET ErrMess

MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 jmp myexit

abc2:

and al,0fh

MOV ARRAY[DI],AL INC DI

LOOP MYLOOP

MOV DX,OFFSET MESS2 MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 MOV AH,01

INT 21H ; input from user cmp al,39h

jbe abc3 ; jump if below or equal to MOV DX,OFFSET ErrMess

MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 jmp myexit

abc3:

and al,0fh MOV DAT,AL

mov ax,ds mov es,ax mov al,dat

CLD ; auto increment mode mov cx,number

INC CX

mov DI, offset ARRAY

repne SCASB ; scan the memory for AL

CMP CX,0

JE NTFOUND

MOV DX,OFFSET MESS3 MOV AH,09

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 SUB NUMBER,CX ;FIND ELEMENT LOCATION

ADD NUMBER,30H MOV DX,NUMBER INC DX

MOV AH,02

INT 21H ; Output a character present in DL , as AH value is 2 JMP myexit

NTFOUND:

MOV DX,OFFSET MESS4 MOV AH,09

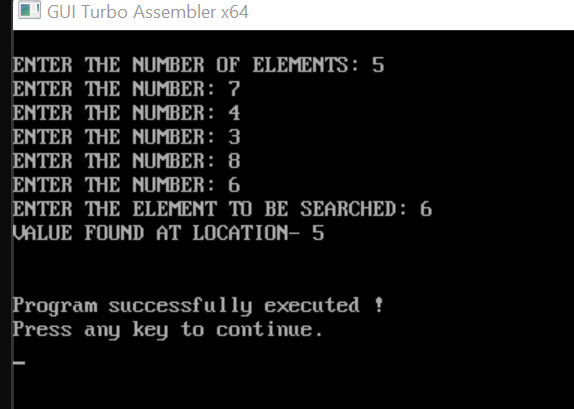
INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

myexit:

MOV AH,4CH

INT 21H ; causes the current process to terminate

END



**Question5:- Write a program to add and subtract two array.** Solution:-

Addition of Two Array:

.model small ; contain two segment data and code

.data ; start of data segment

mat1 db 12h, 11h, 12h, 10h, 11h, 12h, 10h, 11h, 12h ; defined variable mat2 db 13h, 02h, 02h, 02h, 02h, 02h, 02h, 02h, 02h

res3 dw 9 dup(?); Declaring an array with garbage

.code ; start of code segment

mov ax, @data ; loading starting address of data segment in ax

mov ds, ax mov cx, 09h

mov di, offset mat1 mov bx, offset mat2 mov si, offset res3

back : mov ah, 0 mov al, [di]

add al, [bx] adc ah, 00 mov [si], ax inc di

inc bx inc si inc si

loop back

mov si, offset res3 mov dh, 9

l10: mov ch, 04h mov cl, 04h

mov bx, [si]

l2: rol bx, cl ; rotates the bits within the destination operand to the left mov dl, bl

and dl, 0fh cmp dl, 09 jbe l4

add dl, 07

l4: add dl, 30h

mov ah, 02 int 21h dec ch

jnz l2

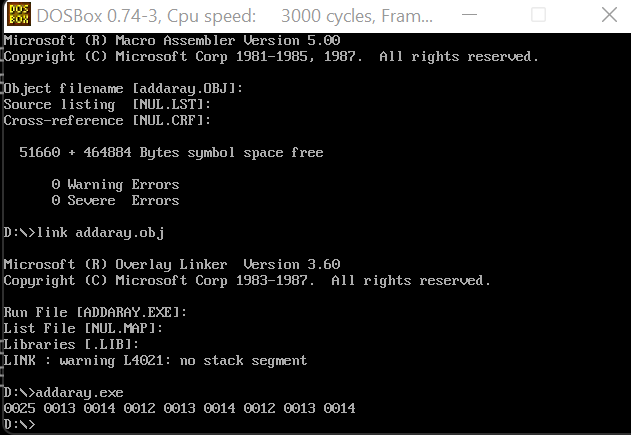
mov dl, ' ' ;This is a whitespace int 21h

inc si inc si dec dh jnz l10

mov ah, 4ch

int 21h ; causes the current process to terminate

end



Subtraction of two array:-

.model small ; contain two segment data and code

.data ; start of data segment

mat1 db 12h, 11h, 12h, 10h, 11h, 12h, 10h, 11h, 12h ; defining byte size variable

mat2 db 13h, 02h, 02h, 02h, 02h, 02h, 02h, 02h, 02h res3 dw 9 dup(?); Declaring an array with garbage

.code ; start of code segment

mov ax, @data ;loading starting address of data segment in ax mov ds, ax

mov cx, 09h

mov di, offset mat1 mov bx, offset mat2 mov si, offset res3 back : mov ah, 0 mov al, [di]

sub al, [bx] adc ah, 00 mov [si], ax inc di

inc bx inc si inc si

loop back

mov si, offset res3 mov dh, 9

l10: mov ch, 04h mov cl, 04h

mov bx, [si]

l2: rol bx, cl ; rotates the bits within the destination operand to the left mov dl, bl

and dl, 0fh cmp dl, 09

jbe l4

add dl, 07

l4: add dl, 30h mov ah, 02

int 21h ; display single character as ah = 2 dec ch

jnz l2

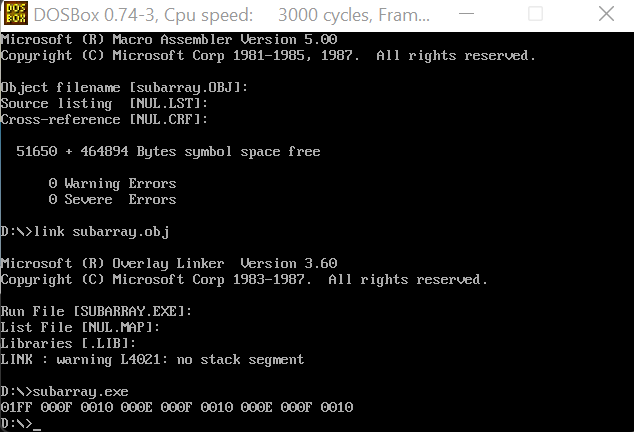
mov dl, ' ' ;This is a whitespace int 21h

inc si inc si dec dh jnz l10

mov ah, 4ch

int 21h ; causes the current process to terminate

end



**Question6:-write a program for binary to ascii conversion.**

.MODEL SMALL ; contain two segment data and code

.DATA ; start of data segment

INPUT DB 10,13 , 'ENTER BINARY NO: $' ; 10 is the ASCII control code for line feed while 13 is the code for carriage return

DB 10,13, 'THE ASCII CHARACTER IS:$'

ARR DB ? ; define byte size variable

.CODE ; start of code segment

.STARTUP ; Generates program start-up code MOV AH,09H

MOV DX,OFFSET INPUT

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV BL, 00H MOV CL,08H

INPUT1: MOV AH,01H

INT 21H ; input from user

# SUB AL,30H ; convert the hexadecimal digits into its equivalent ASCII

SHL BL,1 ; multiply by 16 ADD BL,AL

LOOP INPUT1 ; go-to label INPUT

MOV AH,09H LEA DX,

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9 MOV AH,02H

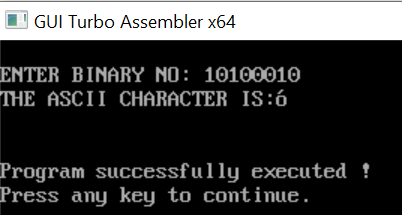
MOV DL,BL

INT 21H ; Output a character present in DL , as AH value is 2

MOV AH,4CH

INT 21H ; causes the current process to terminate

END



**Question 7:-Write a program for ascii to binary conversion.**

.model small ; contain two segment data and code

.stack 100h ; tells the assembler to reserve 100 byte storage

.data ; start of data segment

input db "Enter an ASCII character :$"

output db 10,13,"Binary Equiv: $"; 10 is the ASCII control code for line feed while 13 is the code for carriage return

.code ; start of code segment

MOV AX ,@DATA ; loading starting address of data segment in ax MOV DS ,AX

MOV DX ,OFFSET input ; loading the offset address MOV AH, 09H

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV AH,01H

INT 21H ; input from the user

MOV BL,AL

MOV DX,OFFSET output MOV AH,09H

INT 21H ; Output a string terminated by '$’ stored in DX, as AH =9

MOV CX,8

BIN\_EQUIV:

SHL BL,1 ; multiply by 16 JC PRINTONE PRINTZERO:

MOV DL,30H JMP PRINT

PRINTONE:

MOV DL,31H PRINT:

MOV AH,02H

INT 21H ; Output a character present in DL , as AH value is 2 LOOP BIN\_EQUIV

MOV AH,4CH

INT 21H ; causes the current process to terminate END

