Name:	Pankhania Aanandi R.
Roll No:	IT081
Batch:	I1

Experiment 4

AIM: To study multi module program

1. **Write a multi module assembly program to divide 32-bit number by 16-bit number and return a 32-bit quotient.**

Rules for Operands:

1. You have to use ascii values of the first 4-letters of your name as a **DIVIDEND**. E.g. According to my name (sunil), my DIVIDEND is: 73756E69h

LETTER (use lowercase letters)	ASCII Value in Hex
s	73h
u	75h
n	6Eh
i	69h

Note: If your name is having only 2/3 letters then consider the remaining letters as 00h e.g. "jay" so the dividend will be 6A617900h.

2. You have to use the ascii value of the first 2-letters of your name as a **DIVISOR**. E.g. According to my surname (VITHLANI), my DIVISOR is: 5649h

LETTER (use UPPERCASE letters)	ASCII Value in Hex
V	56h
I	49h

3. Clearly mention ascii values of your name and surname and then write your program.

Write your code here:

According to my name (aanandi), my DIVIDEND is: 61616E61h

LETTER (use lowercase letters)	ASCII Value in Hex
a	61h
a	61h
n	6Eh
a	61h

According to my surname (PANKHANIA), my DIVISOR is: 5041h

LETTER (use UPPERCASE letters)	ASCII Value in Hex
P	50h
A	41h

CODE:

```
;MAIN PROGRAM : FARPRO
data_here segment word public
        dvd dw 6E61h,6161h
        dvs dw 5041h
data_here ends

data1_here segment word
        quotient dw 2 dup(0)
        reminder dw 0
data1_here ends

stack_here segment stack
        dw 30 dup(0)
        t1 label word
stack_here ends

public dvs

procedure_here segment public
        extrn division:Far
procedure_here ends

code_here segment word public
```

```
        assume cs:code_here,ds:data_here, ss:stack_here

start:  mov ax,data_here
                mov ds,ax
                mov ax,stack_here
                mov ss,ax
                mov sp,offset t1
                mov ax,dvd
                mov dx,dvd+2
                mov cx,dvs

                call division
                jnc X
                jmp q

                assume ds:data1_here

  X:    push ds
                mov bx,data1_here
                mov ds,bx

                mov quotient,ax
                mov quotient+2,dx
                mov reminder,cx

                assume ds:data_here
                pop ds

q:      int 3h
code_here ends
end start
```

```
;MODULE
data_here segment public
        extrn dvs:word
data_here ends
public division
procedure_here segment public

        division proc far
                assume cs:procedure_here,ds:data_here
                cmp dvs,0
                je carry

                mov bx,ax
                mov ax,dx
                mov dx,0000h
                div cx
                mov bp,ax
                mov ax,bx
                div cx
                mov cx,dx
                mov dx,bp
                clc
                jmp q

carry:  stc
q:              ret
division endp

procedure_here ends
end
```

Compilation /Running and Debugging steps:

- Clearly mention each step

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Pro...
Drive A is mounted as local directory f:\tasm\

Z:\>a:

A:\>tasm farpro.asm
Turbo Assembler  Version 3.0  Copyright (c) 1988, 1991 Borland International

Assembling file:   farpro.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  475k


A:\>tasm div.asm
Turbo Assembler  Version 3.0  Copyright (c) 1988, 1991 Borland International

Assembling file:   div.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  476k


A:\>
```

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Pro...
A:\>tasm div.asm
Turbo Assembler  Version 3.0  Copyright (c) 1988, 1991 Borland International

Assembling file:   div.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  476k


A:\>tlink farpro.obj div.obj
Turbo Link  Version 3.0 Copyright (c) 1987, 1990 Borland International
Warning: No stack

A:\>tasm farpro
Turbo Assembler  Version 3.0  Copyright (c) 1988, 1991 Borland International

Assembling file:   farpro.ASM
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  475k


A:\>_
```

- Put a screenshot of the mapping file. (Generated after linkage of object files)

```
CNT.MAP

 Start  Stop   Length Name              Class

 00000H 00002H 00003H DATA_HERE
 00010H 00073H 00064H STACK_HERE
 00080H 000A7H 00028H CODE_HERE

Program entry point at 0008:0000
Warning: No stack
```

Output:

Screenshots of memory that contains values of DIVIDENT, DIVISOR, QUOTIENT and REMINDER (output of -d ds:offset_addres command.)

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Pro...
Turbo Assembler  Version 3.0  Copyright (c) 1988, 1991 Borland International

Assembling file:   farpro.ASM
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  475k


A:\>debug farpro.exe
-g=0000

AX=36A1  BX=076B  CX=3F80  DX=0001  SP=003C  BP=0001  SI=0000  DI=0000
DS=076A  ES=075A  SS=076C  CS=0770  IP=0034   NV UP EI PL NZ NA PE NC
0770:0034 CC            INT     3
-d ds:0000 DIVIDEND   DIVISOR
076A:0000  61 6E 61 61 41 50 00 00-00 00 00 00 00 00 00 00   anaaAP..........
076A:0010  A1 36 01 00 80 3F 00 00-00 00 00 00 00 00 00 00   .6...?..........
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 01 00 34 00-70 07 A3 01 00 00 00 00   ......4.p.......
076A:0060  B8 6A 07 8E D8 B8 6C 07-8E D0 BC 3C 00 A1 00 00   .j....l....<....
076A:0070  8B 16 02 00 8B 0E 04 00-9A 40 00 70 07 73 03 EB   .........@.p.s..
-          QUOTIENT   REMAINDER
```

Hexadecimal Calculation—Add, Subtract, Multiply, or Divide

Result

Hex value:
61616E61 ÷ 5041 = **136A1 Remainder : 3F80**

2. **Write an assembly language program to develop a far procedure to find whether the given number is EVEN or ODD and print message appropriately. Write a main program to call this far procedure and pass the roll_no as a parameter to the far procedure.**

   Rules for Operands:
   1. You have to pass your Roll_NO/ID_no for repeater students as a parameter to the procedure.
   2. You can use multi module program or single module program **(If you are using single module then first define procedure segment and then code segment in your program)**

Write your code here:

```
Data_here segment
        num DW 0081H ; REQ. INPUT : IT081(MY ROLL NUM)
        msg1 db 'Given number is ODD$'
        msg2 db'Given number is EVEN$'
Data_here ends

Stack_here segment stack
        dw 50 dup(0)
        stk1 label word
Stack_here ends

msgproc segment
Check proc far
Assume  cs:msgproc
                PUSHF
```

```
                PUSH DX
                shr ax,01H
                jnc evn

odd :   MOV AH,09h
                MOV DX,offset msg1
                INT 21h
                JMP q

evn :  MOV AH,09h
                MOV DX,offset msg2
                INT 21h
                JMP q

q       :       POP DX
                POPF
                RET
                Check endp
msgproc ends
Code_here segment
Assume cs:Code_here ,ds:Data_here ,ss:Stack_here
Start : mov ax,Data_here
                mov ds,ax
                mov ax,Stack_here
                mov ss,ax
                LEA SP,stk1
                mov ax,num
                CALL Check
                INT 3h
Code_here ends
End Start
```

Compilation /Running and Debugging steps:

(As given in lab manual as an example of multiplication program on page no:5 of lab manual)

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Pro...
A:\>tasm oddeven.asm
Turbo Assembler  Version 3.0  Copyright (c) 1988, 1991 Borland International

Assembling file:   oddeven.asm
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  475k


A:\>tlink oddeven.obj
Turbo Link  Version 3.0 Copyright (c) 1987, 1990 Borland International
Warning: No stack

A:\>tasm oddeven
Turbo Assembler  Version 3.0  Copyright (c) 1988, 1991 Borland International

Assembling file:   oddeven.ASM
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  475k


A:\>
```

Output:

1. Screenshot of the memory where you have stored your number.
2. Screenshot of the output message. (e.g. “Your roll_no is EVEN”)

```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Pro...

A:\>debug oddeven.exe
-g
Given number is ODD
AX=0940  BX=0000  CX=00D6  DX=0000  SP=0064  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=076D  CS=0776  IP=0015   NV UP EI PL NZ NA PO NC
0776:0015 CC            INT     3
-d ds:0000
076A:0000  81 00 47 69 76 65 6E 20-6E 75 6D 62 65 72 20 69   ..Given number i
076A:0010  73 20 4F 44 44 24 47 69-76 65 6E 20 6E 75 6D 62   s ODD$Given numb
076A:0020  65 72 20 69 73 20 45 56-45 4E 24 00 00 00 00 00   er is EVEN$.....
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```