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

Experiment 9

AIM: Study of various methods of passing parameters to a procedure

1. **Write an assembly language program to convert a 4-digit BCD number to binary. Use procedure and stack to pass parameters.**

Rules for Operands: You have to use your roll-no/registration no. as 4-digit BCD number.

E.g. IT025, so BCD input should be 0025H.

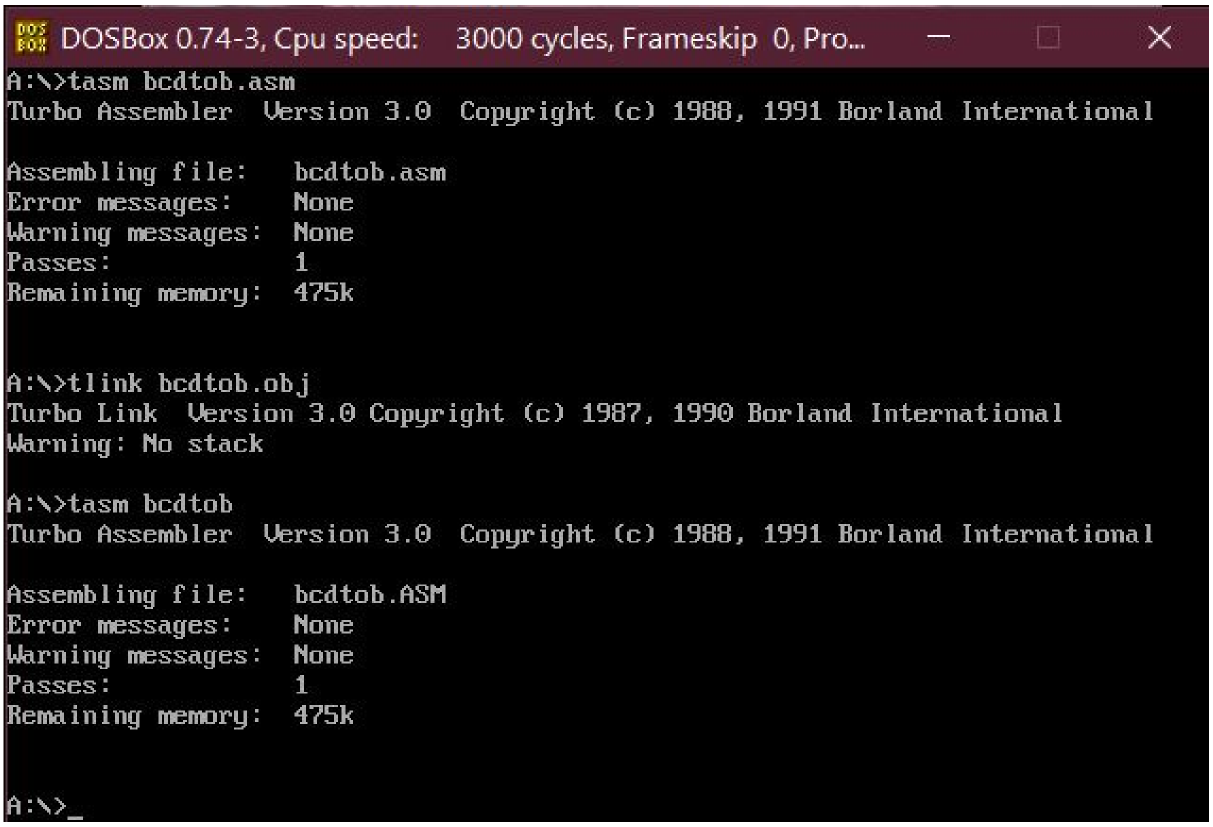
e.g. for repeater student ID=18ITUOS103, BCD input should be 0103h

Write your code here:

```
data_here SEGMENT

        bcd_n DW 0081H

        bin_n DW ?

data_here ENDS


stack_here SEGMENT STACK

        DW 50 DUP(0)

        Stack1 LABEL WORD

stack_here ENDS


code_here SEGMENT


        ASSUME CS:code_here, ES:data_here, DS:data_here, SS:stack_here

START : MOV AX,data_here

                MOV DS,AX

                MOV ES,AX

                MOV AX,stack_here

                MOV SS,AX
                LEA SP,Stack1

                MOV AX, bcd_n
                PUSH AX             ;store value in stack
                CALL CONVERT1                   ;call procedure
                POP AX              ;store the result of procedure will be popped from stack
                MOV bin_n,AX        ;copy result in bin_n

                INT 3h

        CONVERT1 PROC NEAR

                PUSHF
                PUSH BX
                PUSH CX
                PUSH BP

                MOV BP,SP
                MOV AX,[BP+10]

                MOV BX,AX
                AND AX,000FH      ;by this and operation last digit will be stored at last position
                MOV BP,AX

                MOV AX,BX
                AND AX,00F0H          ;to store at second last position
                MOV CL,04H
                SHR AX,CL                   ;shift by 4 right position
                MOV SI,000AH
                MUL SI                                ;digit will be multiplied by 10
                MOV SI,AX
                MOV AX,BX
                AND AX,0F00H          ; to store at third from last position
                MOV CL,08H
                SHR AX,CL       ;shift by 8 right position
                MOV DI,0064H
                MUL DI          ;digit will be multiplied by 100
                MOV DI,AX

                MOV AX,BX
                AND AX,0F000H     ;to store at fourth from last position
                MOV CL,0CH
                SHR AX,CL        ;shift by 12 right position
                MOV CX,03E8H
                MUL CX           ;digit will be multiplied by 1000
                ADD AX,SI
                ADD AX,DI
                ADD AX,BP        ; add all digits
                MOV BP,SP
                MOV [BP+10], AX    ;storing result in stack

                POP BP
                POP CX
                POP BX
                POPF
                RET
        CONVERT1 ENDP

    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)

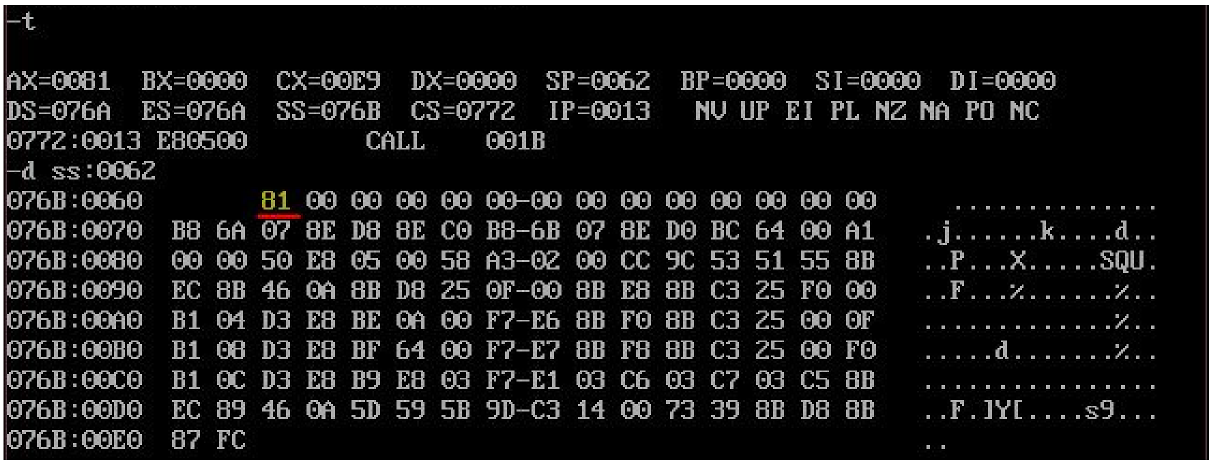


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Pro...

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

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


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

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

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


A:\>_
```

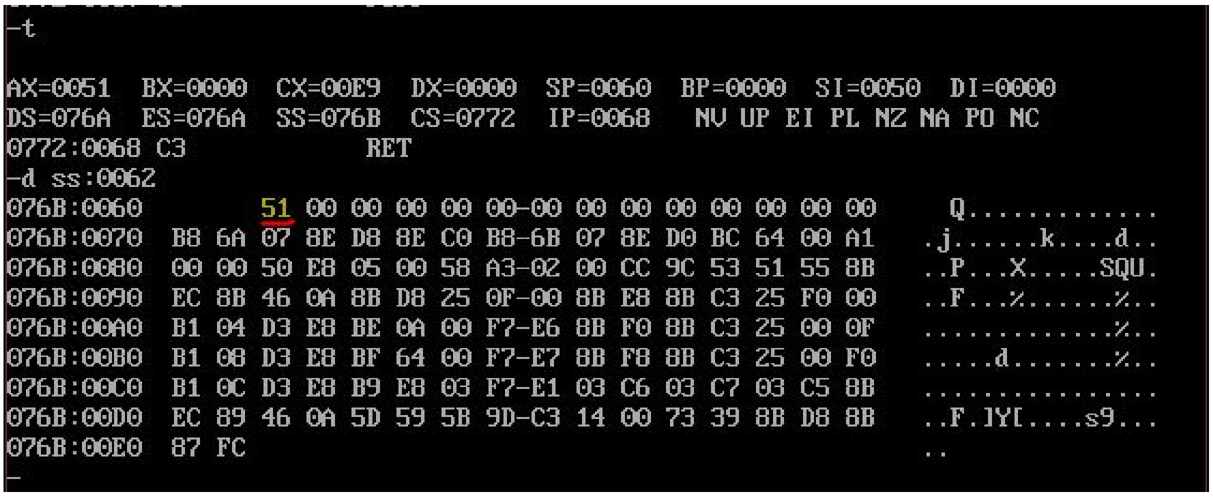
Output:

1. Put a screenshot of stack memory content (immediately after CALL instruction). Mark/highlight the parameter which you have passed from main program to procedure.



```
-t

AX=0081  BX=0000  CX=00E9  DX=0000  SP=0062  BP=0000  SI=0000  DI=0000
DS=076A  ES=076A  SS=076B  CS=0772  IP=0013   NV UP EI PL NZ NA PO NC
0772:0013 E80500        CALL    001B
-d ss:0062
076B:0060        81 00 00 00 00 00-00 00 00 00 00 00 00 00    ..............
076B:0070  B8 6A 07 8E D8 8E C0 B8-6B 07 8E D0 BC 64 00 A1   .j......k....d..
076B:0080  00 00 50 E8 05 00 58 A3-02 00 CC 9C 53 51 55 8B   ..P...X.....SQU.
076B:0090  EC 8B 46 0A 8B D8 25 0F-00 8B E8 8B C3 25 F0 00   ..F...%......%..
076B:00A0  B1 04 D3 E8 BE 0A 00 F7-E6 8B F0 8B C3 25 00 0F   .............%..
076B:00B0  B1 08 D3 E8 BF 64 00 F7-E7 8B F8 8B C3 25 00 F0   .....d.......%..
076B:00C0  B1 0C D3 E8 B9 E8 03 F7-E1 03 C6 03 C7 03 C5 8B   ................
076B:00D0  EC 89 46 0A 5D 59 5B 9D-C3 14 00 73 39 8B D8 8B   ..F.]Y[....s9...
076B:00E0  87 FC                                             ..
```

2. Put a screenshot of stack memory content (immediately after RET instruction). Mark/highlight the parameter which you have passed from procedure to program.



```
-t

AX=0051  BX=0000  CX=00E9  DX=0000  SP=0060  BP=0000  SI=0050  DI=0000
DS=076A  ES=076A  SS=076B  CS=0772  IP=0068   NV UP EI PL NZ NA PO NC
0772:0068 C3            RET
-d ss:0062
076B:0060        51 00 00 00 00 00-00 00 00 00 00 00 00     Q.............
076B:0070  B8 6A 07 8E D8 8E C0 B8-6B 07 8E D0 BC 64 00 A1   .j......k....d..
076B:0080  00 00 50 E8 05 00 58 A3-02 00 CC 9C 53 51 55 8B   ..P...X.....SQU.
076B:0090  EC 8B 46 0A 8B D8 25 0F-00 8B E8 8B C3 25 F0 00   ..F...%......%..
076B:00A0  B1 04 D3 E8 BE 0A 00 F7-E6 8B F0 8B C3 25 00 0F   .............%..
076B:00B0  B1 08 D3 E8 BF 64 00 F7-E7 8B F8 8B C3 25 00 F0   .....d.......%..
076B:00C0  B1 0C D3 E8 B9 E8 03 F7-E1 03 C6 03 C7 03 C5 8B   ................
076B:00D0  EC 89 46 0A 5D 59 5B 9D-C3 14 00 73 39 8B D8 8B   ..F.]Y[....s9...
076B:00E0  87 FC                                             ..
-
```

2. Write an assembly language program to count the number of 1's in the binary representation of 16-bit number using procedure and registers as parameter passing method.

Rules for Operands: You have to use binary representation of your roll-no/registration no. as 16-bit binary input.

E.g. IT025, so Binary input should be 0025H. i.e 0000 0000 0010 0101.

For repeater studentID=18ITUOS103, binary input should be 0103H. i.e. 0000 0001 0000 0011

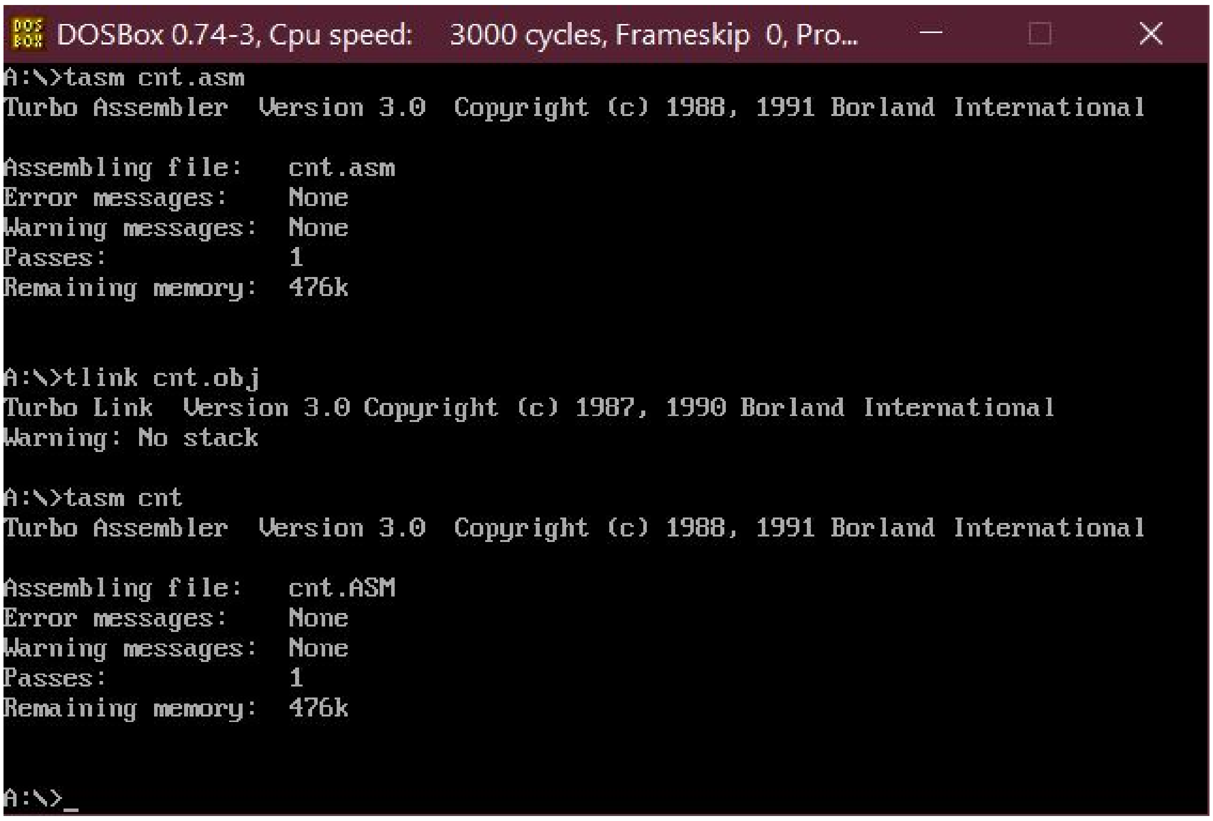
Write your code here:

```
data_here segment
        input1 DW 0081H ;0081 == 0000 0000 1000 0001
        ans DB ?
data_here ENDS

stack_here segment STACK
```

```
        DW 50 DUP(0)
        stack1 LABEL WORD
stack_here ENDS

code_here segment
ASSUME CS:code_here ,SS:stack_here ,DS:data_here
START : MOV AX,data_here
                MOV DS,AX
                MOV AX,stack_here
                MOV SS,AX
                LEA SP ,stack1

                MOV AX,input1
                CALL cnt1
                MOV ans,AL
                INT 3H

cnt1 PROC NEAR
                MOV BL,00H
                MOV CL ,10H
NEXT:   SHR AX,1            ;at a time shift reg AX
                JNC POS             ;if carry not generated jump to POS
                INC BL          ;if generated BL++

POS:    DEC CL
                JNZ NEXT
                MOV AL,BL
                RET
cnt1  ENDP
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)

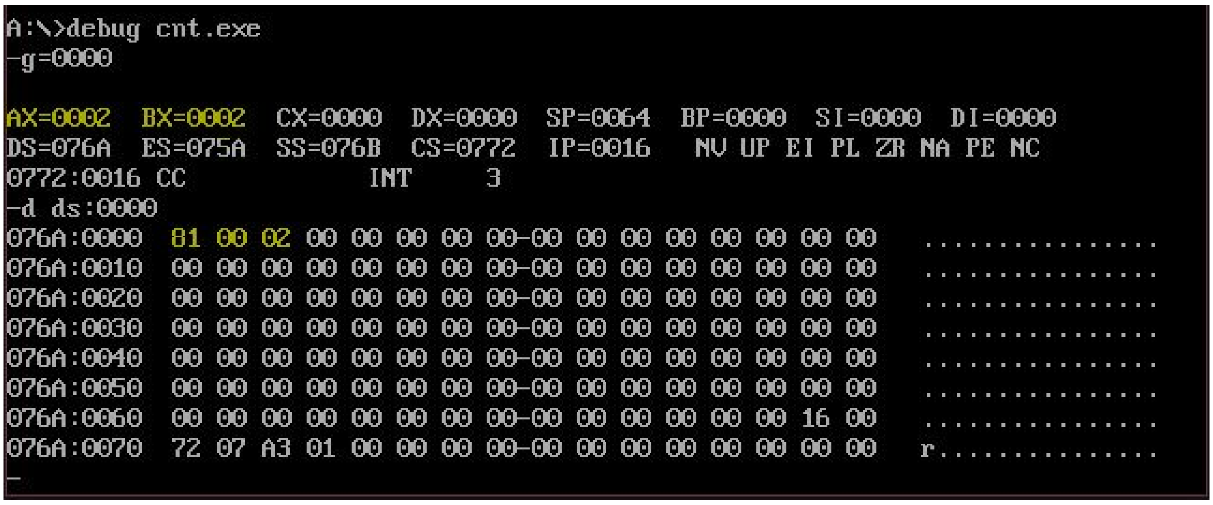


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Pro...

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

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


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

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

Assembling file:   cnt.ASM
Error messages:    None
Warning messages:  None
Passes:            1
Remaining memory:  476k


A:\>_
```

Output:

Screenshots of the memory/registers, which you are using to store your answer.



```
A:\>debug cnt.exe
-g=0000

AX=0002  BX=0002  CX=0000  DX=0000  SP=0064  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=076B  CS=0772  IP=0016   NV UP EI PL ZR NA PE NC
0772:0016 CC            INT     3
-d ds:0000
076A:0000  81 00 02 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
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 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 16 00   ................
076A:0070  72 07 A3 01 00 00 00 00-00 00 00 00 00 00 00 00   r...............
-
```