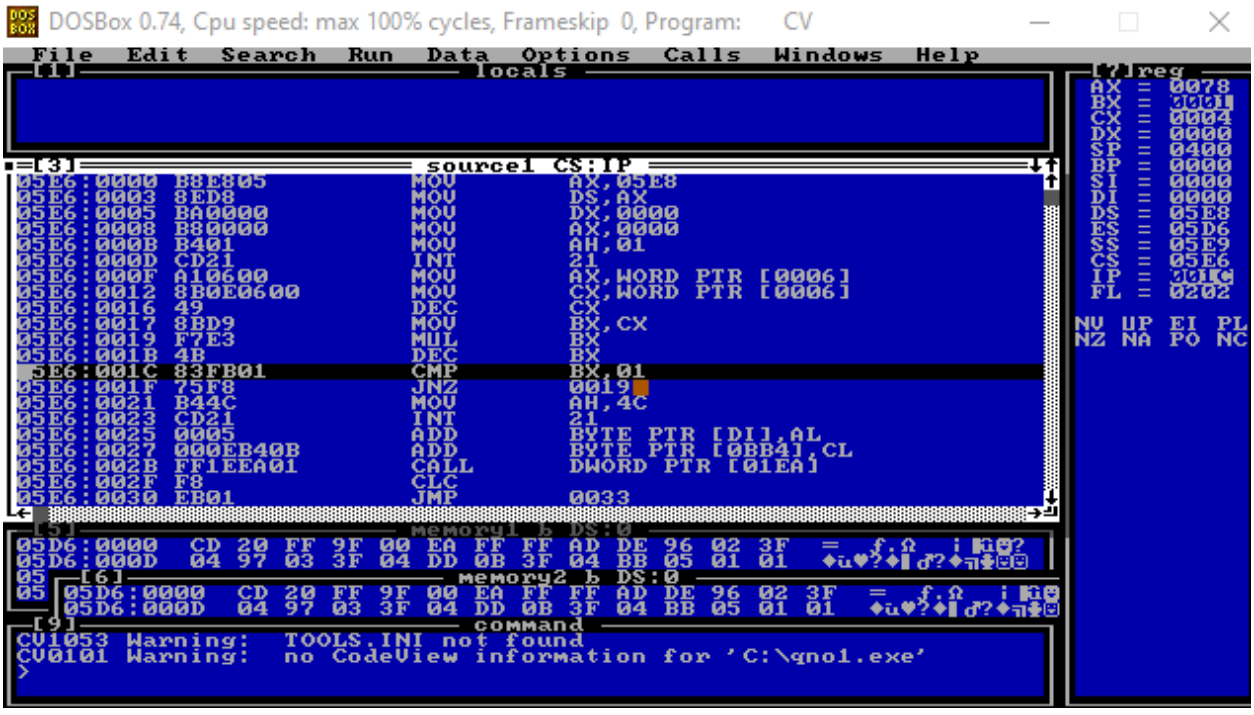
Qno1:

```
.model small
.stack
.data
var1 dw 5
.code

mov ax,@data
mov ds,ax
mov dx,0
mov ax,0

mov ax,var1
mov cx,var1
dec cx
mov bx,cx
l1:
mul bx
dec bx
cmp bx,1
jne l1

exit:
mov ah,4ch
int 21h
end
```



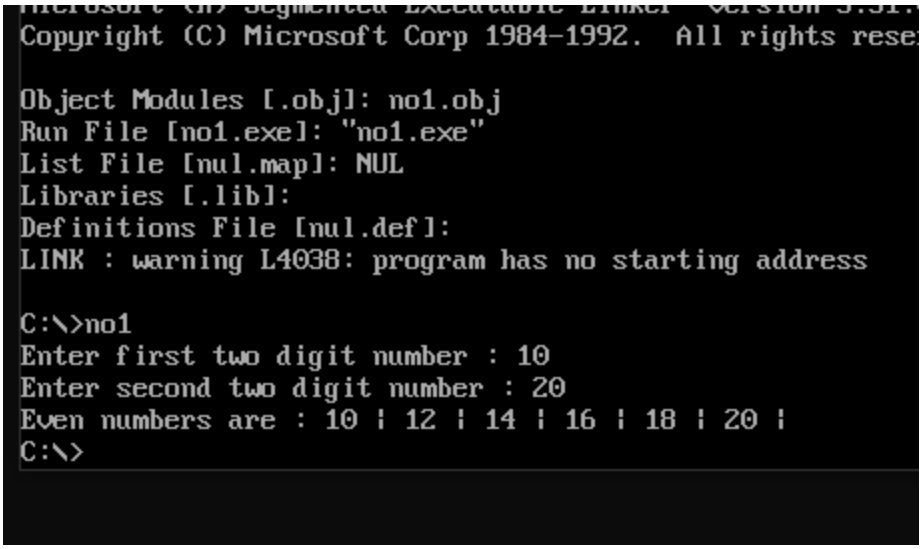
We have stored 5 in the register bx, the multiple of it is being stored in ax then we carry on with our loop until bx is equal to ‘1’ the factorial is stored in ax register.

Question 2:

```
.model small
.stack 100h
.data

msg1 db 'Enter first two digit number : $'
msg2 db 'Enter second two digit number : $'
msg3 db 'Even numbers are : $'
msg4 db ' | $'
digitCount db 0
enteredNumber dw 0
temp1 dw 0
sEven dw 0
digitCount1 db 0
enteredNumber1 dw 0
temp dw 0
.code
mov ax,@data
mov ds,ax
mov ax,0
mov bx,0
mov cx,0

mov dx,0
lea dx, msg1
mov ah,09h
int 21h

mov dx,0
Input:
mov ah,01
int 21H
cmp al,13
JE continue
sub al,48
mov ah,0
mov temp1,ax
mov ax,0
mov ax,enteredNumber
mov bl,10
mul bl
add ax,temp1
mov enteredNumber,ax
```

```
inc digitCount
jmp Input

continue:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h

Input1:
mov ah,01
int 21H
cmp al,13
JE continue1
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,enteredNumber1
mov bl,10
mul bl
add ax,temp
mov enteredNumber1,ax
inc digitCount1
jmp Input1

continue1:
mov dx, offset msg3
mov ah, 09h
int 21h
mov dx,0
mov dx,enterednumber
mov temp1,dx

outforeven:
mov ax,0
mov cx,0
mov dx,0
mov ax , temp1
mov sEven,ax
mov bx,0
mov bx , 2
div bx

mov ax, 0
cmp dx,ax
```

```
je evenval
mov dx,0
inc sEven
mov dx,sEven
mov temp1,dx
cmp enterednumber1,dx
jae outforeven
evenval:
mov ax,temp1
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov temp1,ax
inc cx
cmp temp1,0
jne evenval
display:
pop dx
add dl,48
mov ah,02
int 21h
loop display
mov dx,0
lea dx, msg4
mov ah,09h
int 21h


mov cx,0
mov ax,0
mov dx,0
inc sEven
mov dx,sEven
mov temp1,dx
cmp enterednumber1,dx
jae outforeven
exit:
mov ah,04ch
int 21h
end
```



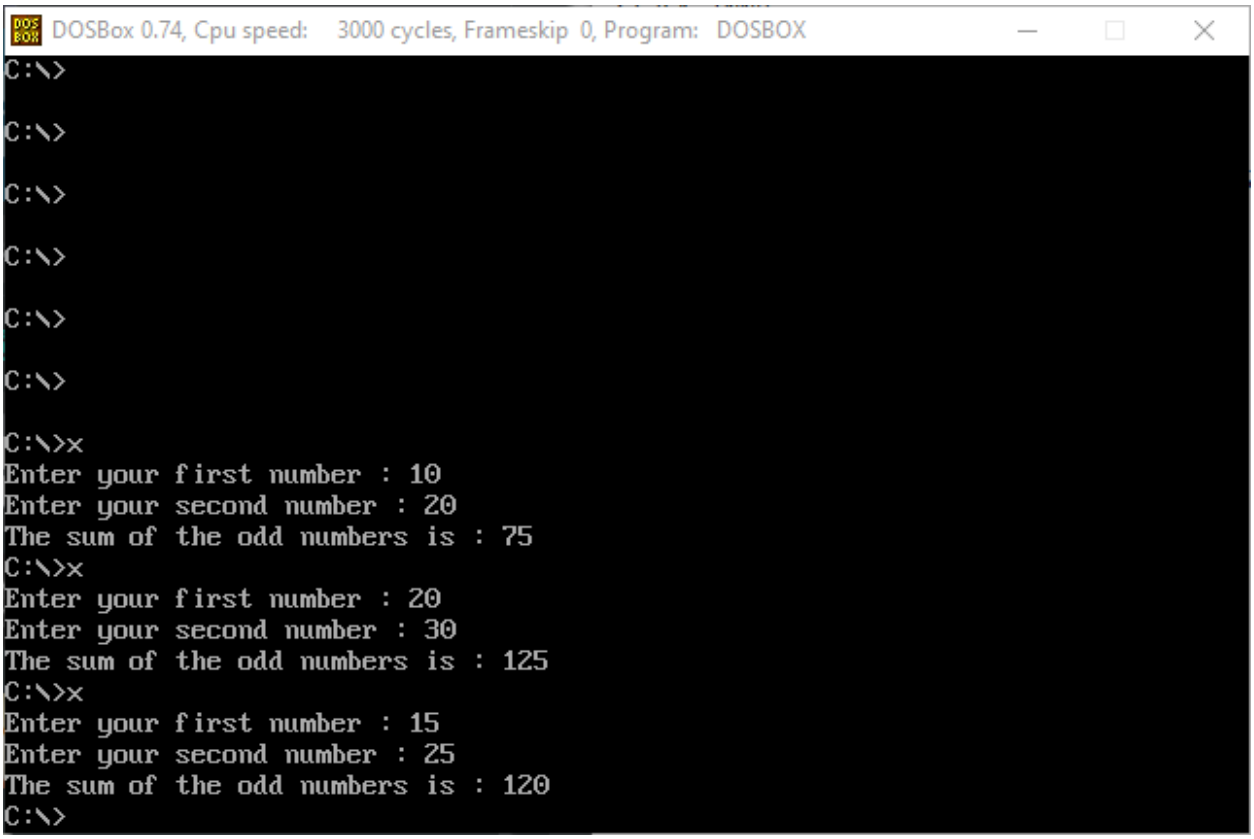
QUESTION 3:

```
.model small
.stack 100h
.data
msg1 db 'Enter your first number : $'
msg2 db 'Enter your second number : $'

msg5 db 'The sum of the odd numbers is : $'
Number1 dw 0
Number2 dw 0
counter1 db 0
temp1 dw 0
sumodd dw 0

.code
mov ax,@data
mov ds,ax
mov ax,0
mov bx,0
mov cx,0
mov dx,0

lea dx, msg1
mov ah,09h
int 21h

Input1:
mov ah,01
int 21H
cmp al,13
JE break1
sub al,48
```

```
mov ah,0
mov temp1,ax
mov ax,0

mov ax,Number1

mov bl,10
mul bl
add ax,temp1
mov Number1,ax
jmp Input1


break1:

lea dx, msg2
mov ah,09h
int 21h

mov temp1,0
mov ax,0
mov bx,0
mov cx,0
mov dx,0

Input2:
mov ah,01
int 21H
cmp al,13
JE checkodd
sub al,48
mov ah,0
mov temp1,ax
mov ax,0
mov ax,Number2
mov bl,10
mul bl
add ax,temp1
mov Number2,ax
jmp Input2

checkodd:
mov counter1,0
mov bx,Number1
mov temp1, bx
mov ax , Number1
```

```
add Number1,1
mov bl , 2
div bl
mov al, ah
cmp al , 1
je sumcal
mov ax,Number1
cmp ax, Number2
jbe checkodd
jmp displaysum


sumcal:

mov bx , temp1
add sumodd, bx

pushodd:

mov bx,0
mov ax,temp1
mov bl ,10
div bl
mov dl,ah
push dx
mov ah,0

mov temp1,ax
inc counter1

cmp temp1,0
jne pushodd

popnum:

pop Dx
Add Dl,48
Mov Ah,02
Int 21H

Cmp counter1,1
je checkodd
Dec counter1
Jmp popnum
```

```
displaysum:

mov dl ,13
mov ah,02h
int 21h

lea dx, msg5
mov ah,09h
int 21h

mov bx, sumodd
mov temp1 ,bx
pushsum:

mov bx,0
mov ax,temp1
mov bl ,10
div bl
mov dl,ah
push dx
mov ah,0

mov temp1,ax
inc counter1

cmp temp1,0
jne pushsum

popsum:

Pop Dx
Add Dl,48
Mov Ah,02
Int 21H

Cmp counter1,1
JE exit
Dec counter1
Jmp popsum

exit:

mov ah,4ch
int 21h
end
```



QUESTION 4:

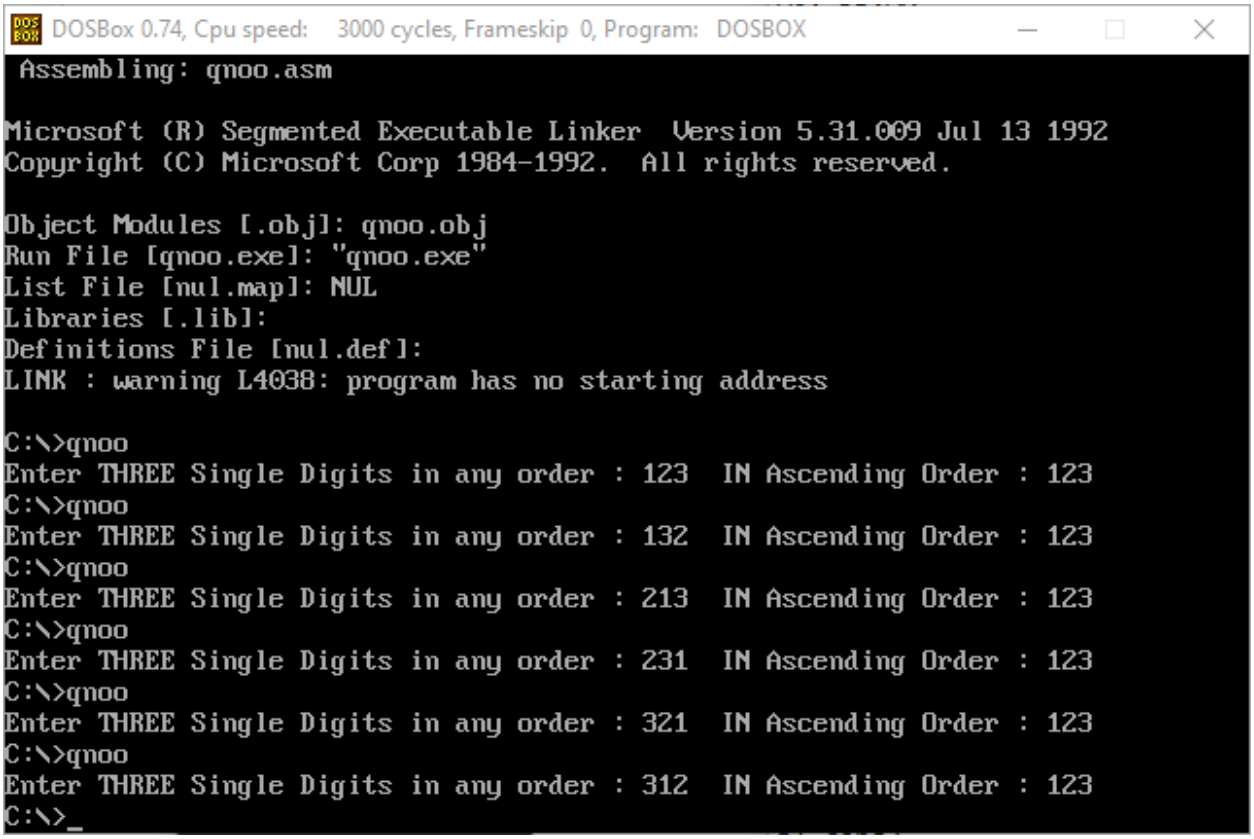
```
.model small
.STACK
.DATA
num1 db ?
num2 db ?
num3 db ?
msg1 db 'Enter THREE Single Digits in any order : $'
msg2 db '  IN Ascending Order : $'
.code
MOV AX,@DATA
MOV DS,AX
mov dx,0
lea dx, msg1
mov ah,09h
int 21h
mov ah,01
int 21h
mov num1,al
mov bl,num1
mov ah,01
int 21h
```

```
mov num2,al
mov cl,num2
mov ah,01
int 21h
mov num3,al


check1:
cmp bl,cl
jl cond1
jmp check2
cond1:
cmp bl,al
jl cond2
jmp check3
cond2:
cmp cl,al
jl out123
jmp out132

check2:
cmp cl,al
jl con1
jmp check3
con1:
cmp al,bl
jl out231
jmp out213

check3:
cmp cl,bl
jl out321
jmp out312

out123:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h
mov dl,num1
mov ah,02
int 21h
mov dl,num2
mov ah,02
int 21h
mov dl,num3
```

```
mov ah,02
int 21h
jmp exit

out132:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h
mov dl,num1
mov ah,02
int 21h
mov dl,num3
mov ah,02
int 21h
mov dl,num2
mov ah,02
int 21h
jmp exit

out213:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h
mov dl,num2
mov ah,02
int 21h
mov dl,num1
mov ah,02
int 21h
mov dl,num3
mov ah,02
int 21h
jmp exit

out231:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h
mov dl,num2
mov ah,02
```

```
int 21h
mov dl,num3
mov ah,02
int 21h
mov dl,num1
mov ah,02
int 21h
jmp exit


out312:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h
mov dl,num3
mov ah,02
int 21h
mov dl,num1
mov ah,02
int 21h
mov dl,num2
mov ah,02
int 21h
jmp exit


out321:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h
mov dl,num3
mov ah,02
int 21h
mov dl,num2
mov ah,02
int 21h
mov dl,num1
mov ah,02
int 21h
jmp exit

exit:
MOV AH,4ch
INT 21h
```

END



QUESTION 5:

```
.model small
.STACK
.DATA
num1 dw ?
num2 dw ?
num3 dw ?
temp dw ?
msg1 db 'Enter First Double Digit Number : $'
msg2 db 'Enter Second Double Digit Number : $'
msg3 db 'Enter Third Double Digit Number : $'
msg4 db '  MAX  : $'
msg5 db '  MIN  : $'
.code
MOV AX,@DATA
MOV DS,AX
mov dx,0
lea dx, msg1
mov ah,09h
int 21h

mov dx,0
```

```
Input:
mov ah,01
int 21H
cmp al,13
JE continue
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,num1
mov bl,10
mul bl
add ax,temp
mov num1,ax
jmp Input
                ;input for the 2nd subject
continue:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h

Input1:
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE continue1
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,num2
mov bl,10
mul bl
add ax,temp
mov num2,ax
jmp Input1
                ;input for the 3rdsubject
continue1:
mov dx, offset msg3
mov ah, 09h
int 21h

Input2:
```

```
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE continue2
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,num3
mov bl,10
mul bl
add ax,temp
mov num3,ax
jmp Input2

continue2:
mov bx,num1
mov cx,num2
mov ax,num3


check1:
cmp bx,cx
jg cond1
jmp check2
cond1:
cmp bx,ax
jg cond2
jmp check3
cond2:
cmp cx,ax
jg out123
jmp out132

check2:
cmp cx,ax
jg con1
jmp check3
con1:
cmp ax,bx
jg out231
jmp out213

check3:
```

```
cmp cx,bx
jg out321
jmp out312

out123:
lea dx, msg4
mov ah,09h
int 21h
val123:
mov ax,num1
mov dx,0
mov bl,10
div bl

mov dl,ah
push dx
mov ah,0
mov num1,ax
mov cx,2
cmp num1,0
jne val123
display:
pop dx
add dl,48
mov ah,02
int 21h
loop display
MOV dl, 10
MOV ah, 02h
INT 21h
MOV dl, 13
MOV ah, 02h
INT 21h
lea dx, msg5
mov ah,09h
int 21h
v123:
mov ax,num3
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num3,ax
inc cx
```

```
cmp num3,0
jne v123
display1:
pop dx
add dl,48
mov ah,02
int 21h
loop display1
jmp exit


out132:
lea dx, msg4
mov ah,09h
int 21h
val132:
mov ax,num1
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num1,ax
mov cx,2
cmp num1,0
jne val132
display2:
pop dx
add dl,48
mov ah,02
int 21h
loop display2
MOV dl, 10
MOV ah, 02h
INT 21h
MOV dl, 13
MOV ah, 02h
INT 21h
lea dx, msg5
mov ah,09h
int 21h
v132:
mov ax,num2
mov bl,10
div bl
```

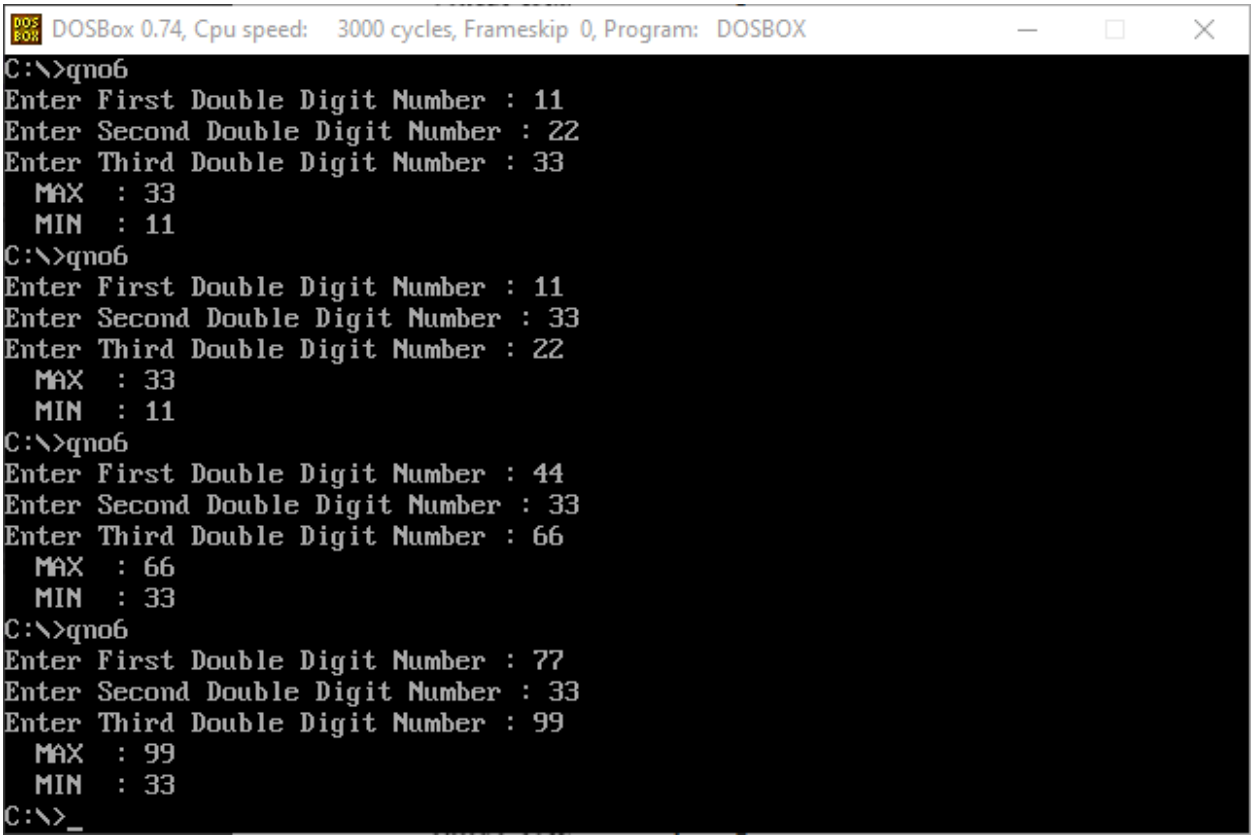
```
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num2,ax
inc cx
cmp num2,0
jne v132
display3:
pop dx
add dl,48
mov ah,02
int 21h
loop display3
jmp exit



out213:
lea dx, msg4
mov ah,09h
int 21h
val213:
mov ax,num2
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num2,ax
mov cx,2
cmp num2,0
jne val213
display4:
pop dx
add dl,48
mov ah,02
int 21h
loop display4
MOV dl, 10
MOV ah, 02h
INT 21h
MOV dl, 13
MOV ah, 02h
```

```
INT 21h
lea dx, msg5
mov ah,09h
int 21h
v213:
mov ax,num3
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num3,ax
inc cx
cmp num3,0
jne v213
display5:
pop dx
add dl,48
mov ah,02
int 21h
loop display5
jmp exit


out231:
lea dx, msg4
mov ah,09h
int 21h
val231:
mov ax,num2
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num2,ax
mov cx,2
cmp num2,0
jne val231
display6:
pop dx
add dl,48
mov ah,02
int 21h
```

```
loop display6
MOV dl, 10
MOV ah, 02h
INT 21h
MOV dl, 13
MOV ah, 02h
INT 21h
lea dx, msg5
mov ah,09h
int 21h
v231:
mov ax,num1
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num1,ax
inc cx
cmp num1,0
jne v231
display7:
pop dx
add dl,48
mov ah,02
int 21h
loop display7
jmp exit


out312:
lea dx, msg4
mov ah,09h
int 21h
val312:
mov ax,num3
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num3,ax
mov cx,2
cmp num3,0
```

```
jne val312
display8:
pop dx
add dl,48
mov ah,02
int 21h
loop display8
MOV dl, 10
MOV ah, 02h
INT 21h
MOV dl, 13
MOV ah, 02h
INT 21h
lea dx, msg5
mov ah,09h
int 21h
v312:
mov ax,num2
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num2,ax
inc cx
cmp num2,0
jne v312
display9:
pop dx
add dl,48
mov ah,02
int 21h
loop display9
jmp exit


out321:
lea dx, msg4
mov ah,09h
int 21h
val321:
mov ax,num3
mov bl,10
div bl
mov dx,0
```

```
mov dl,ah
push dx
mov ah,0
mov num3,ax
mov cx,2
cmp num3,0
jne val321
display11:
pop dx
add dl,48
mov ah,02
int 21h
loop display11
MOV dl, 10
MOV ah, 02h
INT 21h
MOV dl, 13
MOV ah, 02h
INT 21h
lea dx, msg5
mov ah,09h
int 21h
v321:
mov ax,num1
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov num1,ax
inc cx
cmp num1,0
jne v321
display12:
pop dx
add dl,48
mov ah,02
int 21h
loop display12
jmp exit

exit:
MOV AH,4ch
INT 21h
END
```



QUESTION 6:

```
.model small
.stack 100h
.data

msg1 db 'Enter marks of first subject : $'
msg2 db 'Enter marks of second subject :: $'
msg3 db 'Enter marks of third subject : : $'
msg4 db 'Enter marks of fourth subject : $'
msg5 db 'Enter marks of fifth subject : $'
msg6 db 'Percentage is: $'
msg7 db 'Marks obtained out of 500 are: $'
digitCount db 0
temp dw 0
sEven dw 0
digitCount1 db 0
s1 dw 0
s2 dw 0
s3 dw 0
s4 dw 0
s5 dw 0
per dw 0
```

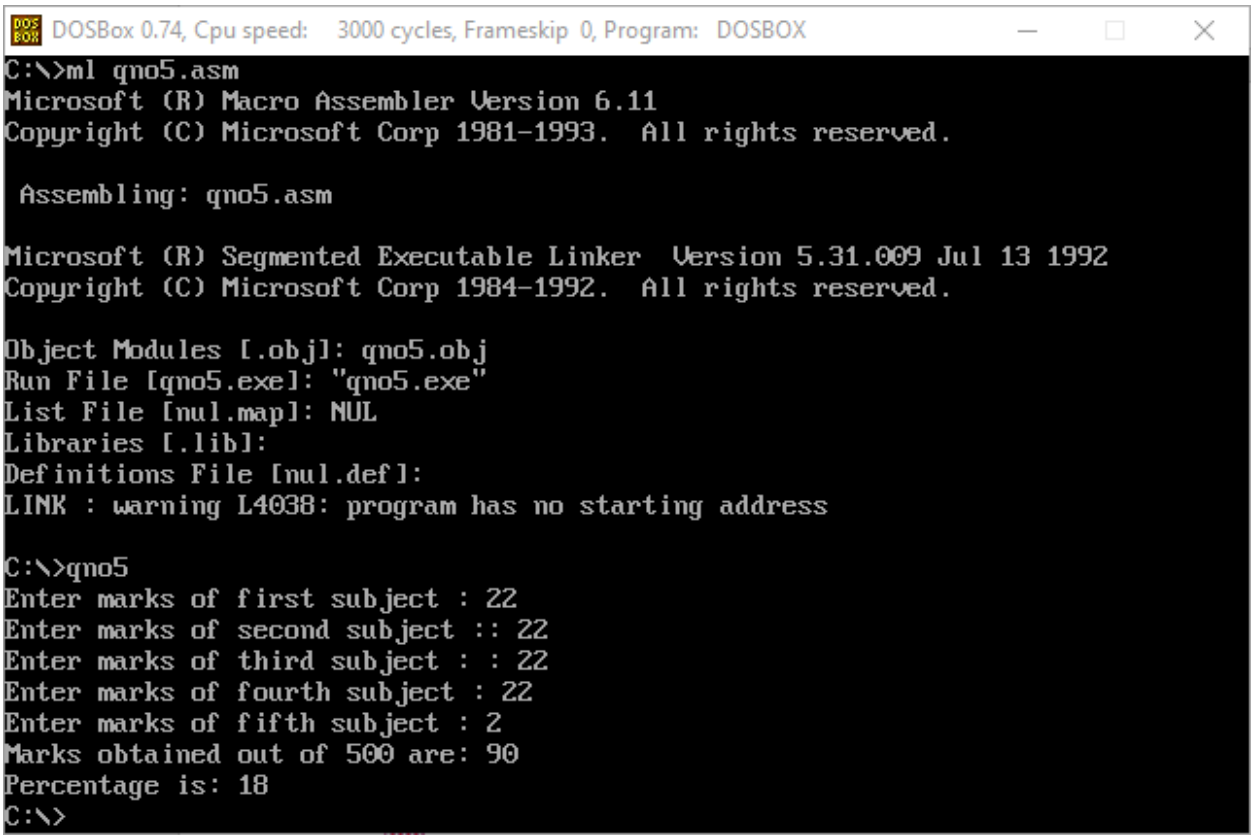
```
tot dw 0
.code
mov ax,@data
mov ds,ax
mov ax,0
mov bx,0
mov cx,0
            ;input for the first subject
mov dx,0
lea dx, msg1
mov ah,09h
int 21h

mov dx,0
Input:
mov ah,01
int 21H
cmp al,13
JE continue
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,s1
mov bl,10
mul bl
add ax,temp
mov s1,ax
inc digitCount
jmp Input
                ;input for the 2nd subject
continue:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h

Input1:
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE continue1
sub al,48
mov ah,0
```

```
mov temp,ax
mov ax,0
mov ax,s2
mov bl,10
mul bl
add ax,temp
mov s2,ax
inc digitCount1
jmp Input1
                    ;input for the 3rdsubject
continue1:
mov dx, offset msg3
mov ah, 09h
int 21h

Input2:
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE continue2
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,s3
mov bl,10
mul bl
add ax,temp
mov s3,ax
inc digitCount1
jmp Input2

continue2:      ;input for the 4th subject
mov dx, offset msg4
mov ah, 09h
int 21h

Input3:
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE continue3
```

```
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,s4
mov bl,10
mul bl
add ax,temp
mov s4,ax
inc digitCount1
jmp Input3
                        ;input for the 5th subject
continue3:
mov dx, offset msg5
mov ah, 09h
int 21h

Input4:
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE marks
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,s5
mov bl,10
mul bl
add ax,temp
mov s5,ax
inc digitCount1
jmp Input4



                        ;total marks obtained
marks:
mov dx, offset msg7
mov ah, 09h
int 21h
mov ax,0
add ax,s1
add ax,s2
```

```
add ax,s3
add ax,s4
add ax,s5
mov tot,ax
mov per,ax
tval:
mov ax,tot
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov tot,ax
inc cx
cmp tot,0
jne tval
display:
pop dx
add dl,48
mov ah,02
int 21h
loop display

MOV dl, 10
MOV ah, 02h
INT 21h
MOV dl, 13
MOV ah, 02h
INT 21h

                              ;percentage
percentage:
mov dx, offset msg6
mov ah, 09h
int 21h
mov ax,0
mov ax,per
mov dx,0

mov bx,0
mov bx,5
div bx
mov per,ax
pval:
mov ax,per
```

```
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov per,ax
inc cx
cmp per,0
jne pval
pdisplay:
pop dx
add dl,48
mov ah,02
int 21h
loop pdisplay



exit:
mov ah,04ch
int 21h
end
```



QUESTION 7:

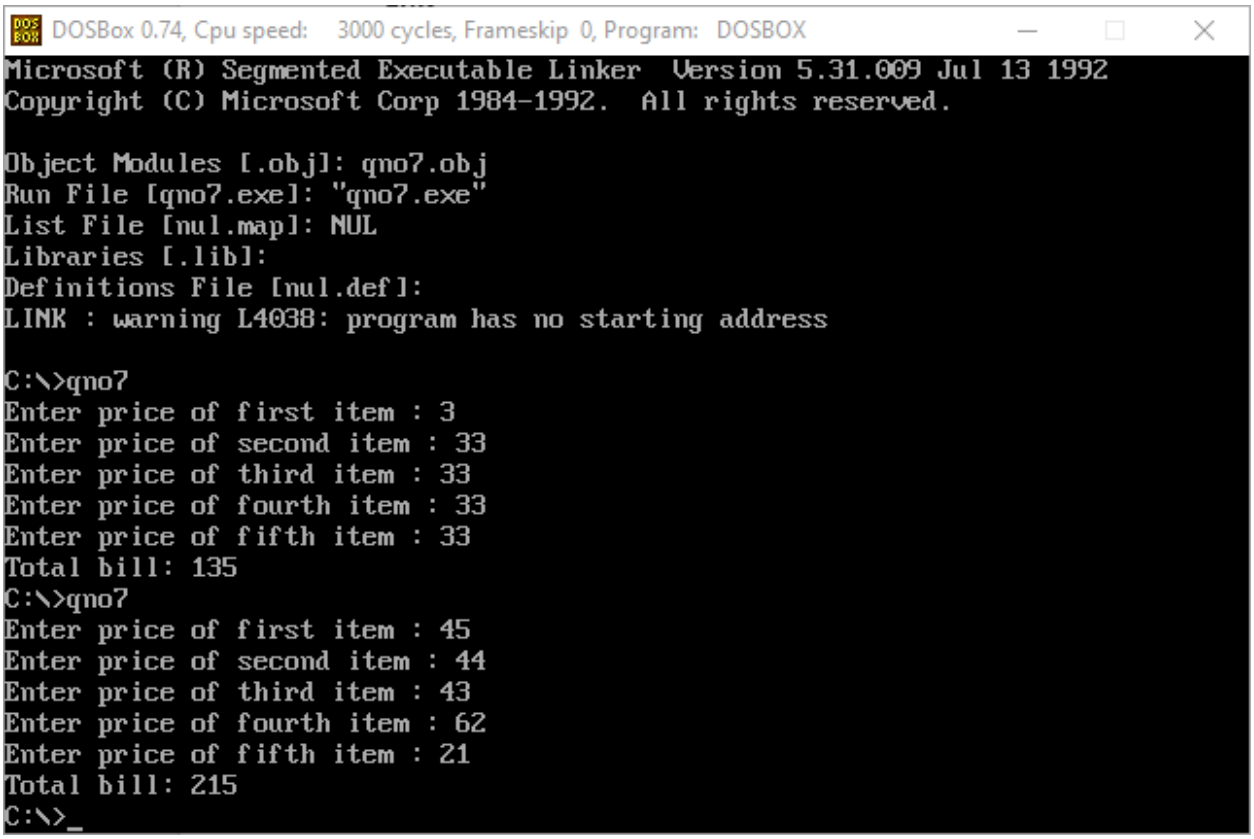
```
.model small
.stack 100h
.data

msg1 db 'Enter price of first item : $'
msg2 db 'Enter price of second item : $'
msg3 db 'Enter price of third item : $'
msg4 db 'Enter price of fourth item : $'
msg5 db 'Enter price of fifth item : $'

msg7 db 'Total bill: $'
digitCount db 0
temp dw 0
sEven dw 0
digitCount1 db 0
s1 dw 0
s2 dw 0
s3 dw 0
s4 dw 0
s5 dw 0
per dw 0
tot dw 0
.code
```

```
mov ax,@data
mov ds,ax
mov ax,0
mov bx,0
mov cx,0
            ;input for the first item
mov dx,0
lea dx, msg1
mov ah,09h
int 21h

mov dx,0
Input:
mov ah,01
int 21H
cmp al,13
JE continue
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,s1
mov bl,10
mul bl
add ax,temp
mov s1,ax
inc digitCount
jmp Input
                    ;input for the 2nd item
continue:
mov dx,0
lea dx, msg2
mov ah,09h
int 21h

Input1:
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE continue1
sub al,48
mov ah,0
mov temp,ax
mov ax,0
```

```
mov ax,s2
mov bl,10
mul bl
add ax,temp
mov s2,ax
inc digitCount1
jmp Input1
                    ;input for the 3rd item
continue1:
mov dx, offset msg3
mov ah, 09h
int 21h

Input2:
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE continue2
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,s3
mov bl,10
mul bl
add ax,temp
mov s3,ax
inc digitCount1
jmp Input2

continue2:      ;input for the 4th item
mov dx, offset msg4
mov ah, 09h
int 21h

Input3:
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE continue3
sub al,48
mov ah,0
```

```
mov temp,ax
mov ax,0
mov ax,s4
mov bl,10
mul bl
add ax,temp
mov s4,ax
inc digitCount1
jmp Input3
                        ;input for the 5th item
continue3:
mov dx, offset msg5
mov ah, 09h
int 21h

Input4:
mov ax,0
mov temp,ax
mov ah,01
int 21H
cmp al,13
JE marks
sub al,48
mov ah,0
mov temp,ax
mov ax,0
mov ax,s5
mov bl,10
mul bl
add ax,temp
mov s5,ax
inc digitCount1
jmp Input4



                        ;total bill obtained
marks:
mov dx, offset msg7
mov ah, 09h
int 21h
mov ax,0
add ax,s1
add ax,s2
add ax,s3
add ax,s4
```

```
add ax,s5
mov tot,ax
mov per,ax
tval:
mov ax,tot
mov bl,10
div bl
mov dx,0
mov dl,ah
push dx
mov ah,0
mov tot,ax
inc cx
cmp tot,0
jne tval
display:
pop dx
add dl,48
mov ah,02
int 21h
loop display



exit:
mov ah,04ch
int 21h
end
```


DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.
Object Modules [.obj]: qno7.obj
Run File [qno7.exe]: "qno7.exe"
List File [nul.map]: NUL
Libraries [.lib]:
Definitions File [nul.def]:
LINK : warning L4038: program has no starting address
C:\>qno7
Enter price of first item : 3
Enter price of second item : 33
Enter price of third item : 33
Enter price of fourth item : 33
Enter price of fifth item : 33
Total bill: 135
C:\>qno7
Enter price of first item : 45
Enter price of second item : 44
Enter price of third item : 43
Enter price of fourth item : 62
Enter price of fifth item : 21
Total bill: 215
C:\>_