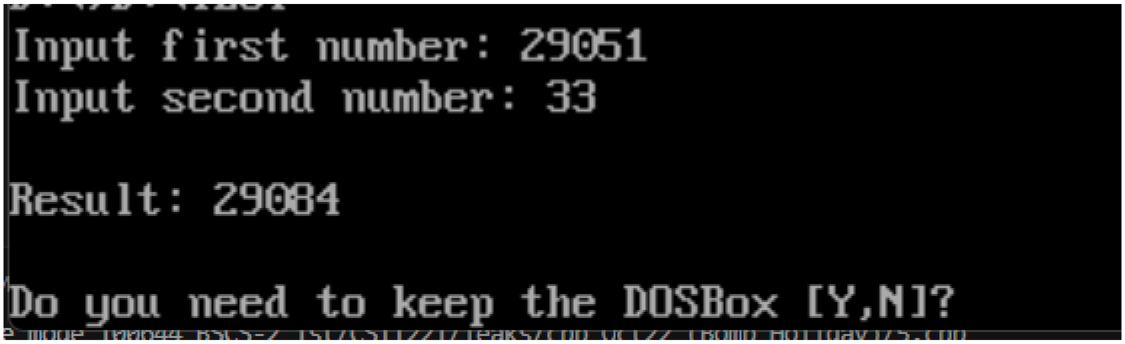
1. EXER40.ASM

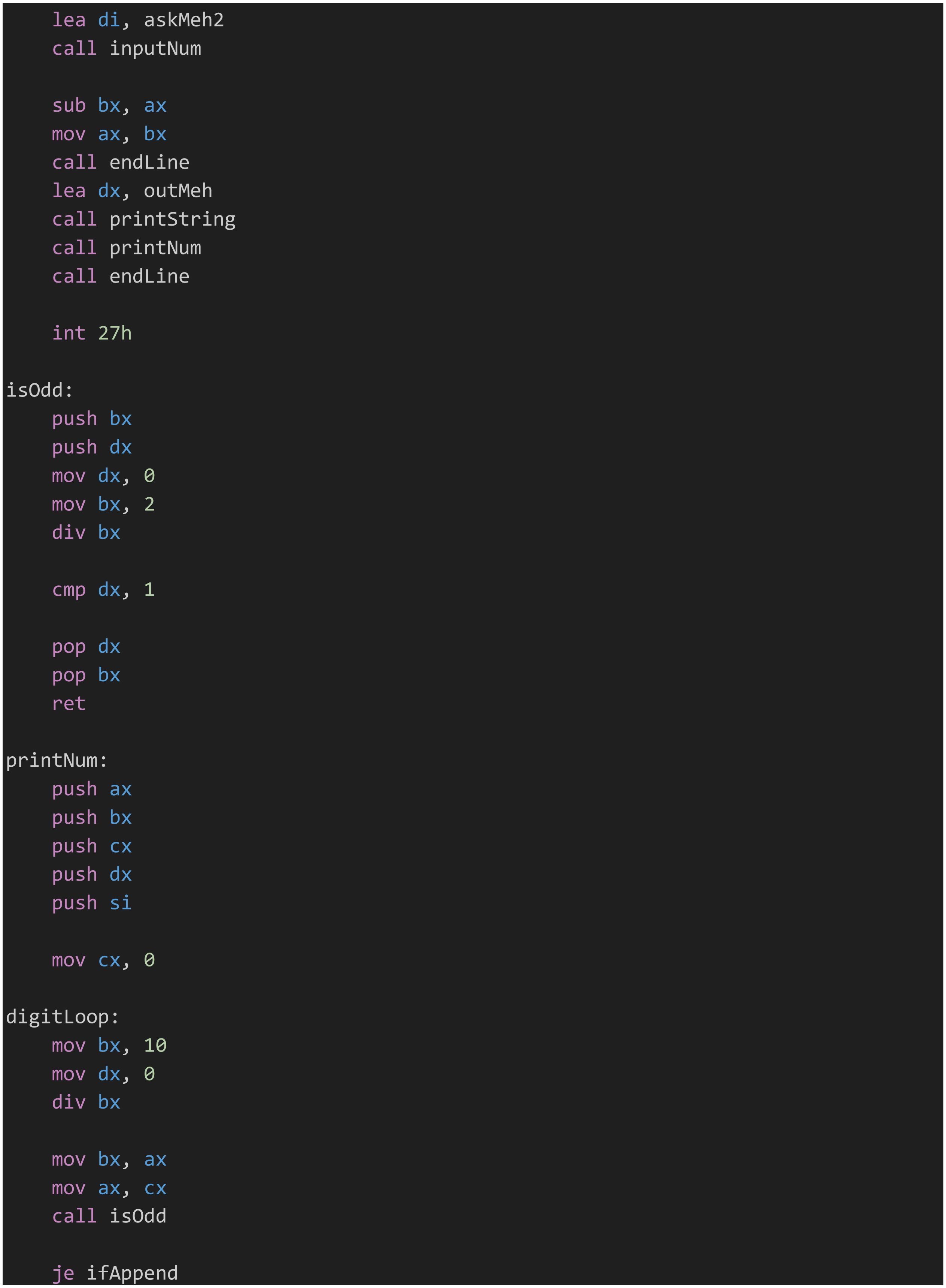
```
; Filename: EXER40.ASM
; Programmer Name: JOHN ZILLION C. REYES
; Date: October 25, 2024


.model small
.stack 100
.data
    askMeh db "Input first number: $"
    askMeh2 db "Input second number: $"
    outMeh db "Result: $"
    cls db "                             ", 0Dh ,
"                             ", 0Dh, "$"
.code
start:
    mov ax, @data
    mov ds, ax

    lea di, askMeh
    call inputNum
    mov bx, ax
    lea di, askMeh2
    call inputNum

    add ax, bx
    call endLine
    lea dx, outMeh
    call printString
    call printNum
    call endLine

    int 27h

isOdd:
    push bx
    push dx
    mov dx, 0
    mov bx, 2
    div bx

    cmp dx, 1

    pop dx
    pop bx
    ret

printNum:
    push ax
    push bx
    push cx
    push dx
    push si

    mov cx, 0

digitLoop:
    mov bx, 10
    mov dx, 0
    div bx

    mov bx, ax
    mov ax, cx
    call isOdd

    je ifAppend
    mov ah, dl
    mov al, 0
    jmp endifAppend
ifAppend:
    pop ax
    mov al, dl
endifAppend:
    push ax
    inc cx

    mov ax, bx

    cmp ax, 0
    jne digitLoop

printLoop:

    mov ax, cx
    call isOdd

    pop ax
    je ifPrint
    mov dl, al
    push ax
    jmp endifPrint
ifPrint:
    mov dl, ah
endifPrint:
    add dl, '0'
    mov ah, 02h
    int 21h

    loop printLoop

    pop si
    pop dx
    pop cx
    pop bx
    pop ax

    ret

inputNum:
    push bx
    push cx
    push dx
    push si
    mov cx, 0
    mov si, 10
    mov bx, 0
    mov dx, di
    call printString
inputLoop:
    mov ah, 7
    int 21h

    cmp al, 8
    je inputRem
    cmp al, '0'
    jl exitInputNum
    cmp al, '9'
    jg exitInputNum
    sub al, '0'
    mov cl, al
    mov ax, bx
    mul si

    add ax, cx
    mov bx, ax
    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum

    jmp inputLoop
inputRem:
    mov ax, bx
    mov dx, 0
    div si
    mov bx, ax

    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum

    jmp inputLoop

exitInputNum:
    call endLine
    mov ax, bx

    pop si
    pop dx
    pop cx
    pop bx
    ret

printString:
    push ax

    mov ah, 09h
    int 21h

    pop ax
    ret

endLine:
    push ax
    push dx

    mov ah, 02h
    mov dl, 0Ah
    int 21h

    pop dx
    pop ax
    ret
end start
```



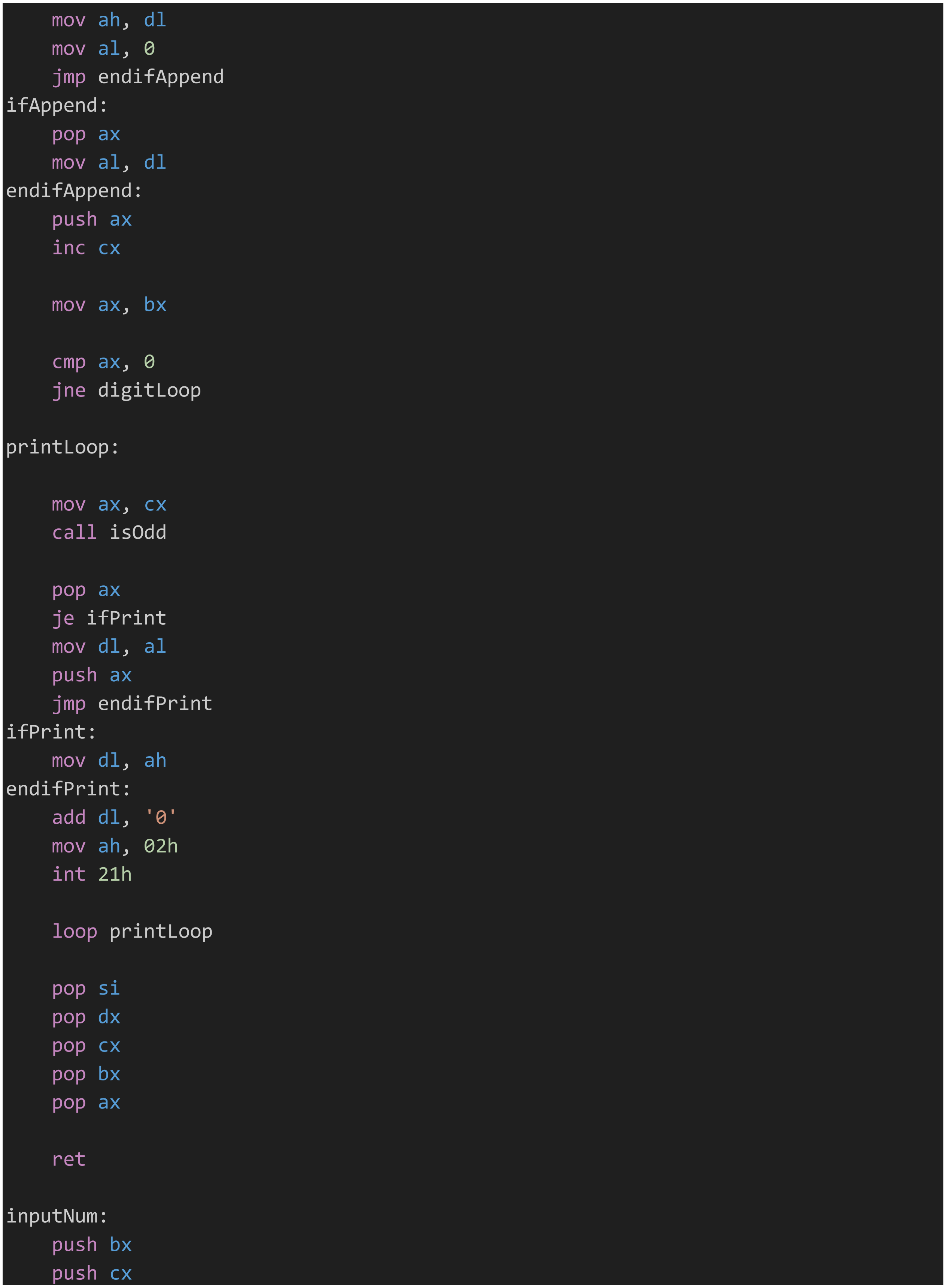
```
Input first number: 29051
Input second number: 33

Result: 29084

Do you need to keep the DOSBox [Y,N]?
```

2. EXER41.ASM

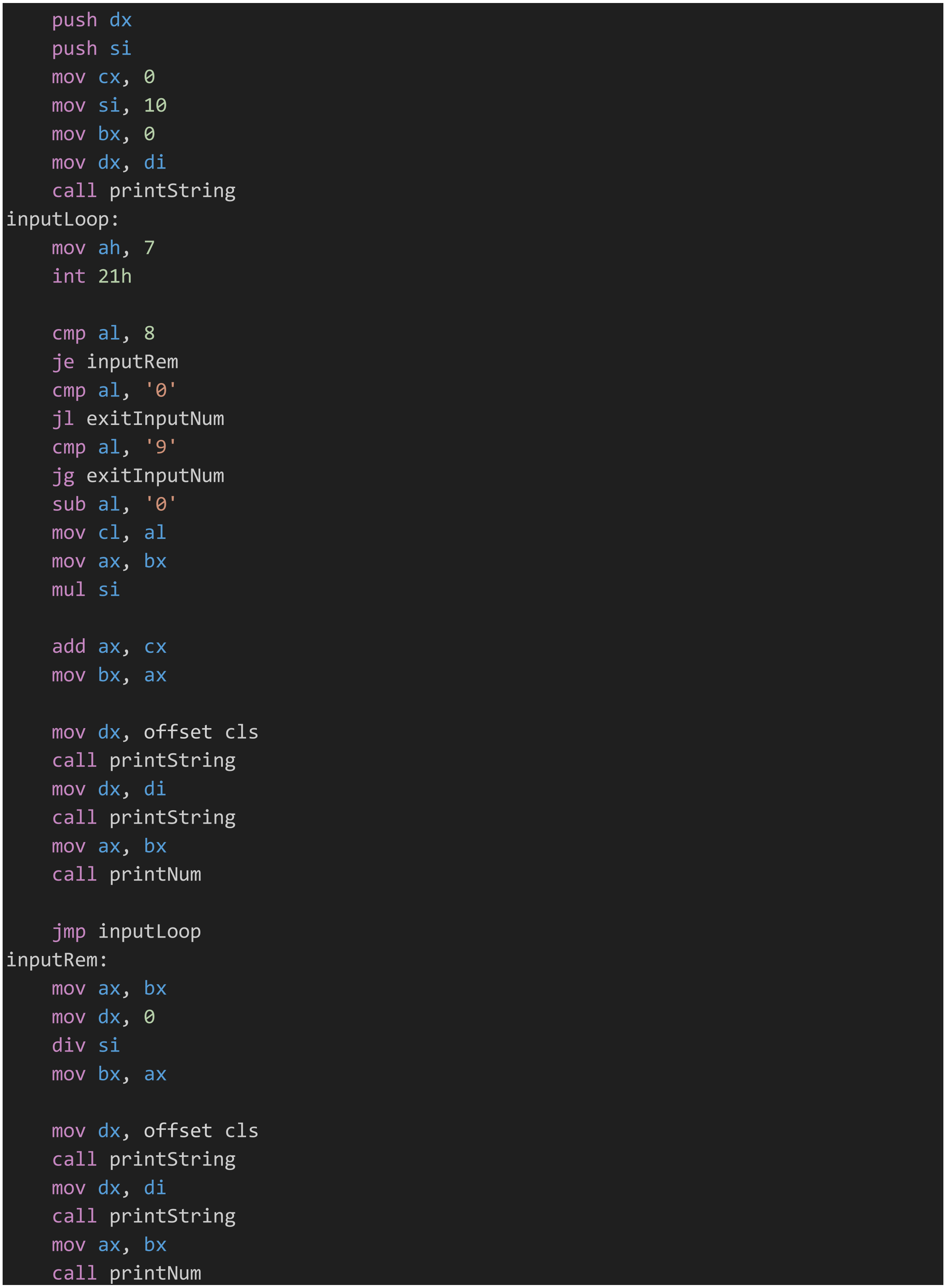
```
; Filename: EXER41.ASM
; Programmer Name: JOHN ZILLION C. REYES
; Date: October 25, 2024


.model small
.stack 100
.data
    askMeh db "Input first number: $"
    askMeh2 db "Input second number: $"
    outMeh db "Result: $"
    cls db "                             ", 0Dh ,
"                             ", 0Dh, "$"
.code
start:
    mov ax, @data
    mov ds, ax

    lea di, askMeh
    call inputNum
    mov bx, ax
```



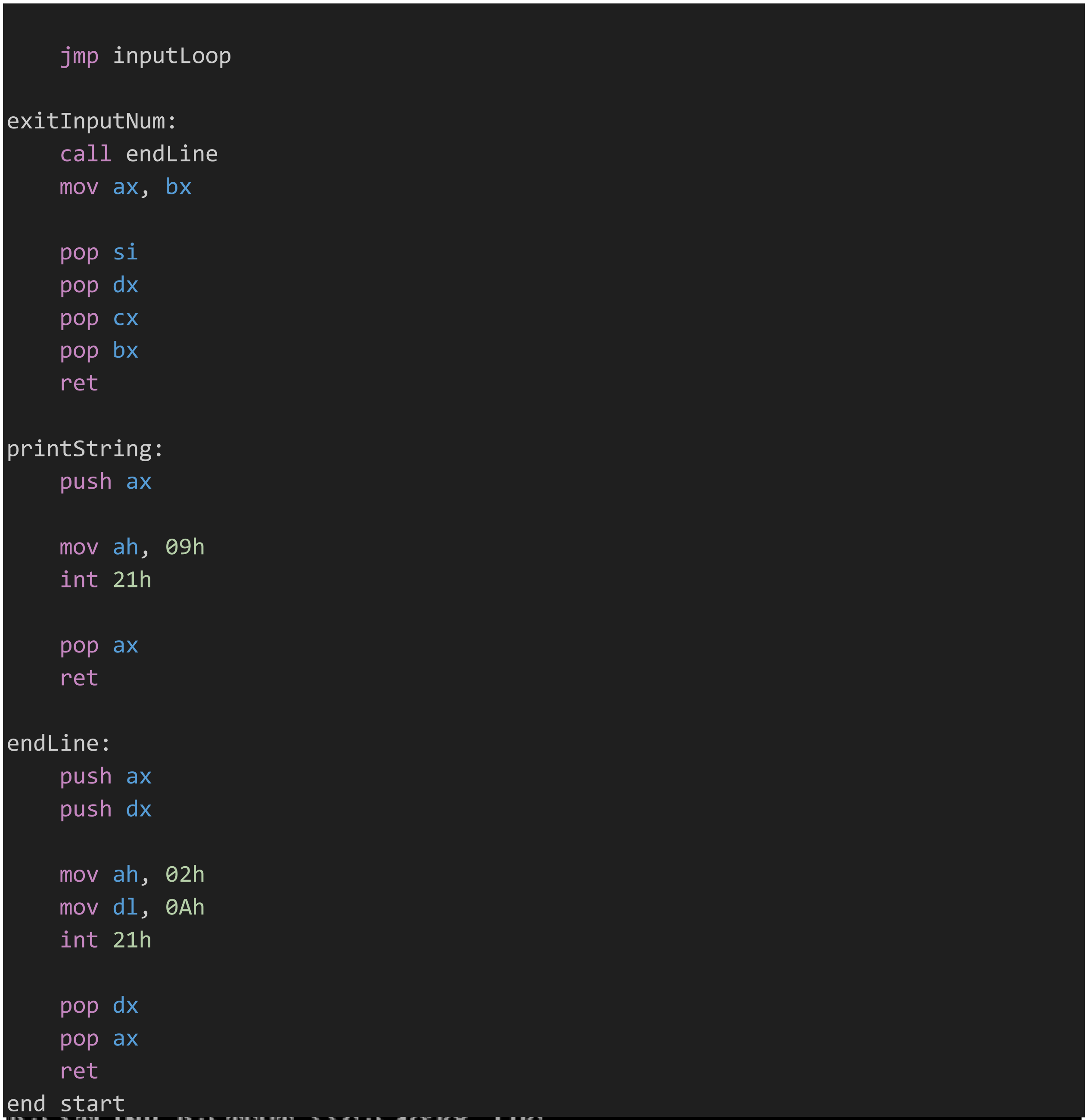
```
    lea di, askMeh2
    call inputNum

    sub bx, ax
    mov ax, bx
    call endLine
    lea dx, outMeh
    call printString
    call printNum
    call endLine

    int 27h

isOdd:
    push bx
    push dx
    mov dx, 0
    mov bx, 2
    div bx

    cmp dx, 1

    pop dx
    pop bx
    ret

printNum:
    push ax
    push bx
    push cx
    push dx
    push si

    mov cx, 0

digitLoop:
    mov bx, 10
    mov dx, 0
    div bx

    mov bx, ax
    mov ax, cx
    call isOdd

    je ifAppend
```



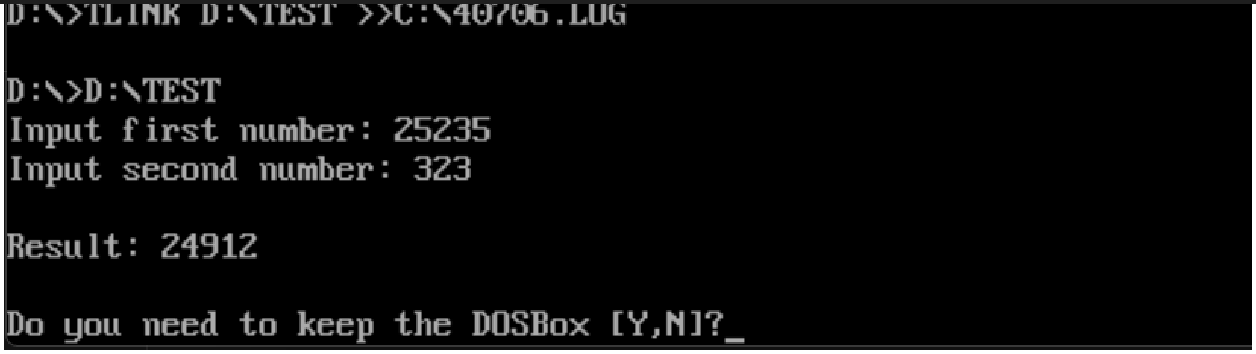
```
    mov ah, dl
    mov al, 0
    jmp endifAppend
ifAppend:
    pop ax
    mov al, dl
endifAppend:
    push ax
    inc cx

    mov ax, bx

    cmp ax, 0
    jne digitLoop

printLoop:

    mov ax, cx
    call isOdd

    pop ax
    je ifPrint
    mov dl, al
    push ax
    jmp endifPrint
ifPrint:
    mov dl, ah
endifPrint:
    add dl, '0'
    mov ah, 02h
    int 21h

    loop printLoop

    pop si
    pop dx
    pop cx
    pop bx
    pop ax

    ret

inputNum:
    push bx
    push cx
```



```
    push dx
    push si
    mov cx, 0
    mov si, 10
    mov bx, 0
    mov dx, di
    call printString
inputLoop:
    mov ah, 7
    int 21h

    cmp al, 8
    je inputRem
    cmp al, '0'
    jl exitInputNum
    cmp al, '9'
    jg exitInputNum
    sub al, '0'
    mov cl, al
    mov ax, bx
    mul si

    add ax, cx
    mov bx, ax

    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum

    jmp inputLoop
inputRem:
    mov ax, bx
    mov dx, 0
    div si
    mov bx, ax

    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum
```

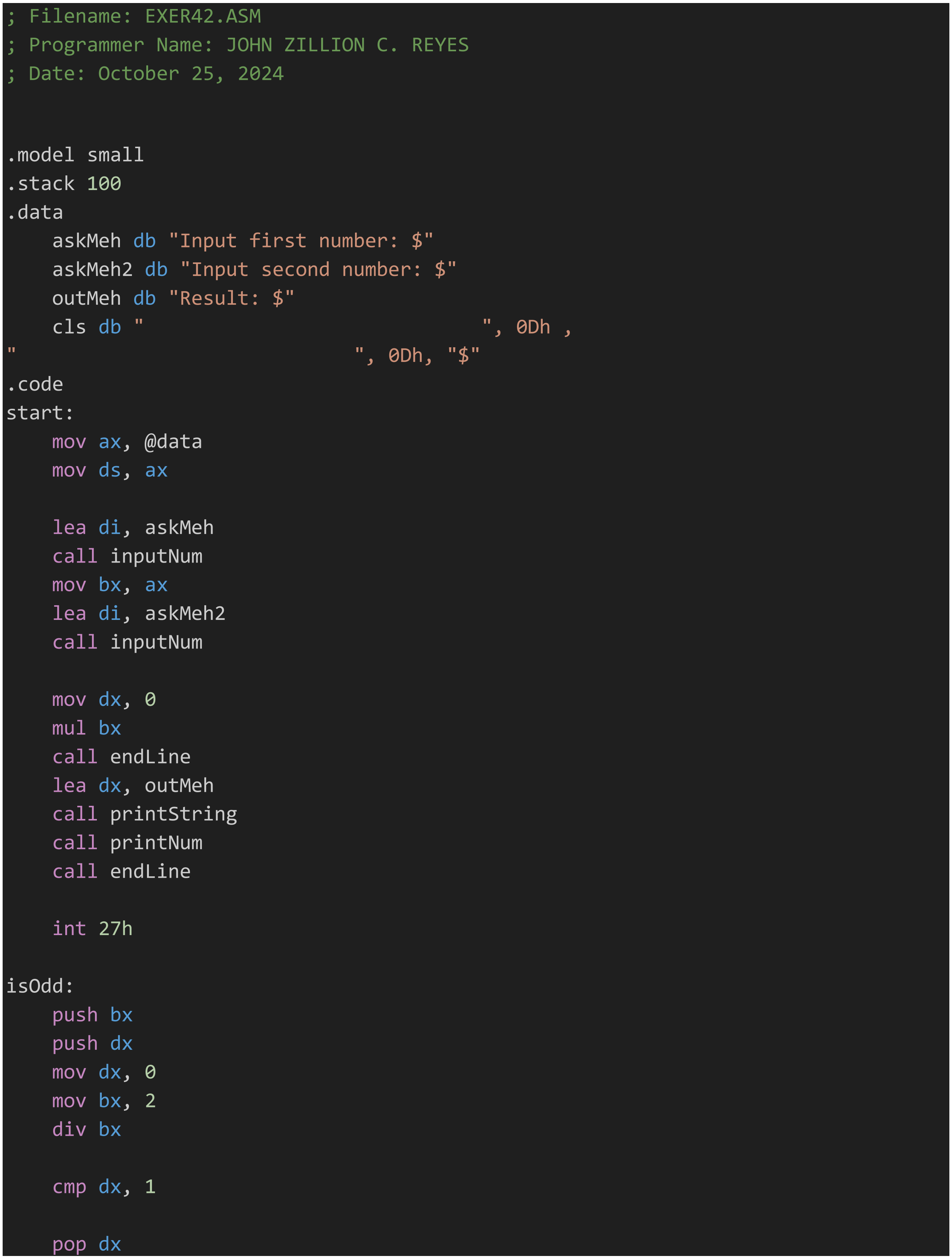


```
    jmp inputLoop

exitInputNum:
    call endLine
    mov ax, bx

    pop si
    pop dx
    pop cx
    pop bx
    ret

printString:
    push ax

    mov ah, 09h
    int 21h

    pop ax
    ret

endLine:
    push ax
    push dx

    mov ah, 02h
    mov dl, 0Ah
    int 21h

    pop dx
    pop ax
    ret
end start
```

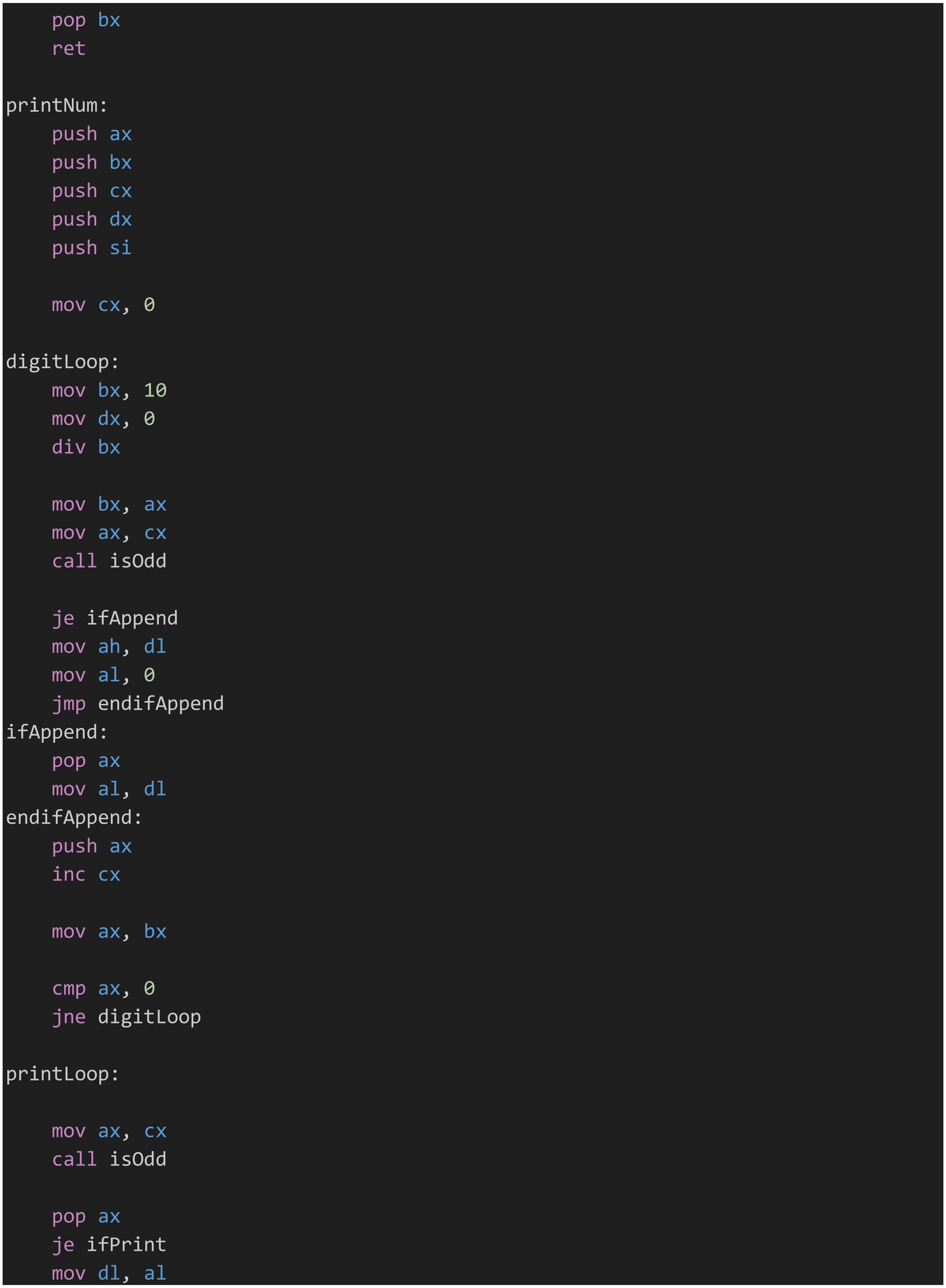


```
D:\>TLINK D:\TEST >>C:\40706.LOG

D:\>D:\TEST
Input first number: 25235
Input second number: 323

Result: 24912

Do you need to keep the DOSBox [Y,N]?_
```

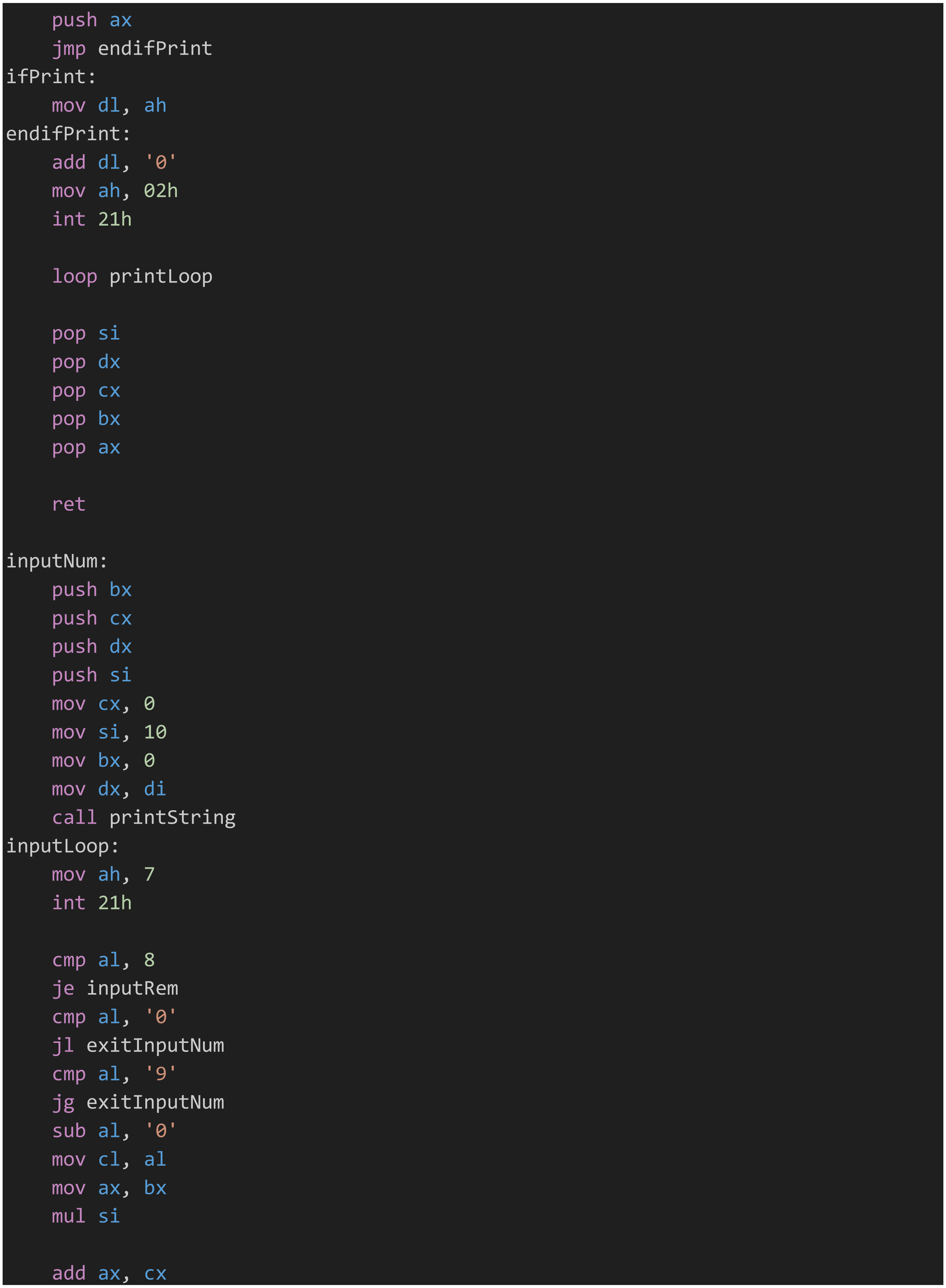
3. EXER42.ASM



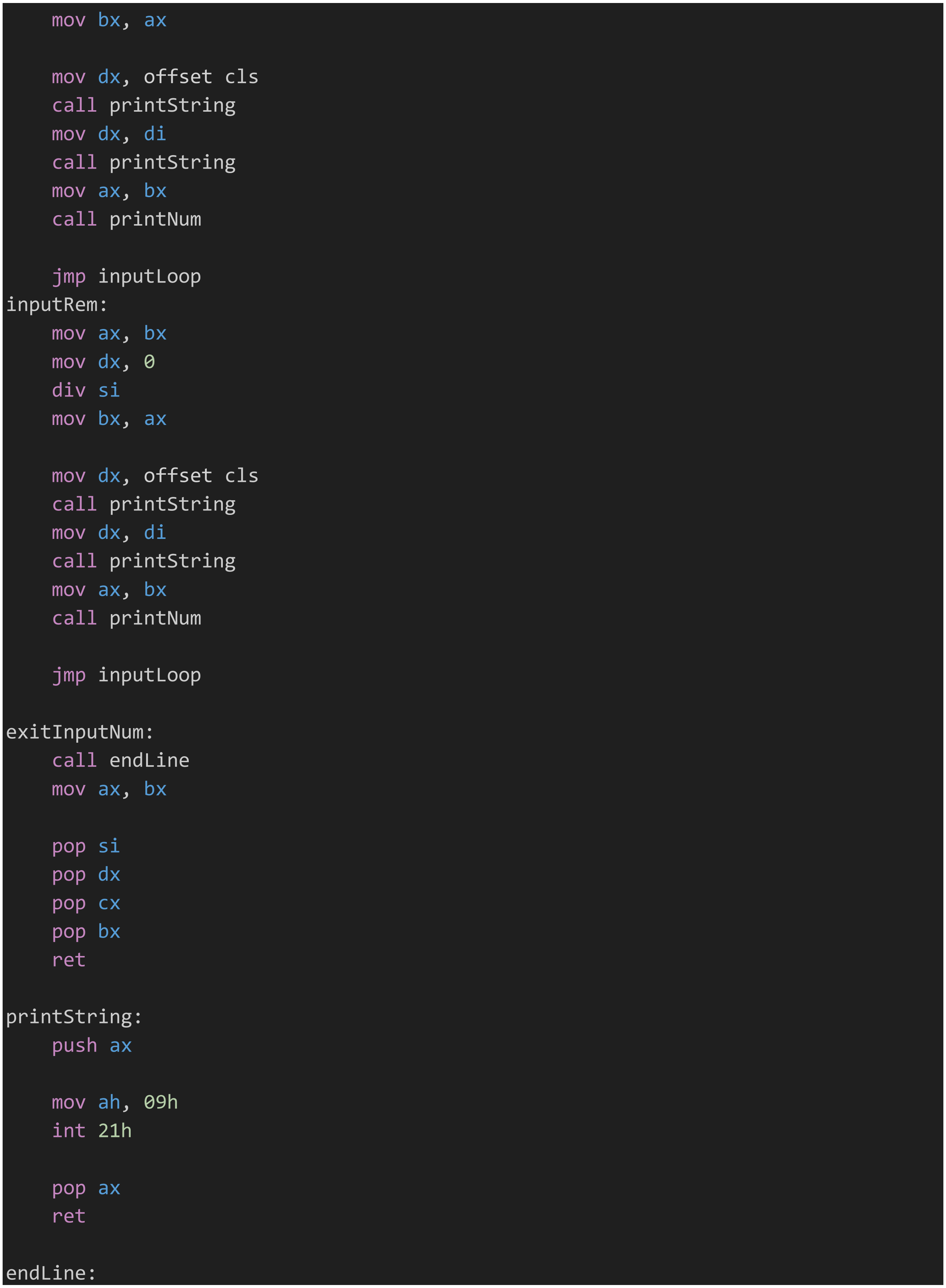
```
; Filename: EXER42.ASM
; Programmer Name: JOHN ZILLION C. REYES
; Date: October 25, 2024


.model small
.stack 100
.data
    askMeh db "Input first number: $"
    askMeh2 db "Input second number: $"
    outMeh db "Result: $"
    cls db "                             ", 0Dh ,
"                             ", 0Dh, "$"
.code
start:
    mov ax, @data
    mov ds, ax

    lea di, askMeh
    call inputNum
    mov bx, ax
    lea di, askMeh2
    call inputNum

    mov dx, 0
    mul bx
    call endLine
    lea dx, outMeh
    call printString
    call printNum
    call endLine

    int 27h

isOdd:
    push bx
    push dx
    mov dx, 0
    mov bx, 2
    div bx

    cmp dx, 1

    pop dx
```



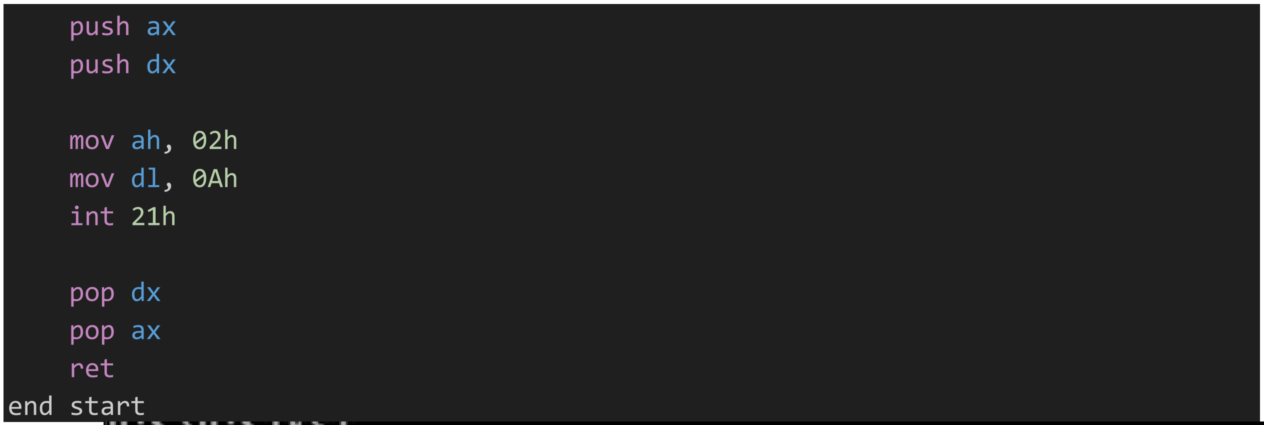
```
    pop bx
    ret

printNum:
    push ax
    push bx
    push cx
    push dx
    push si

    mov cx, 0

digitLoop:
    mov bx, 10
    mov dx, 0
    div bx

    mov bx, ax
    mov ax, cx
    call isOdd

    je ifAppend
    mov ah, dl
    mov al, 0
    jmp endifAppend
ifAppend:
    pop ax
    mov al, dl
endifAppend:
    push ax
    inc cx

    mov ax, bx

    cmp ax, 0
    jne digitLoop

printLoop:

    mov ax, cx
    call isOdd

    pop ax
    je ifPrint
    mov dl, al
```



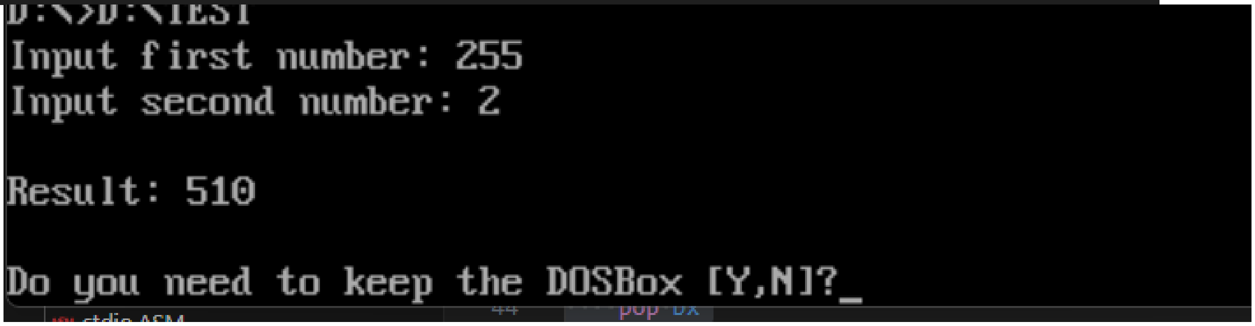
```
    push ax
    jmp endifPrint
ifPrint:
    mov dl, ah
endifPrint:
    add dl, '0'
    mov ah, 02h
    int 21h

    loop printLoop

    pop si
    pop dx
    pop cx
    pop bx
    pop ax

    ret

inputNum:
    push bx
    push cx
    push dx
    push si
    mov cx, 0
    mov si, 10
    mov bx, 0
    mov dx, di
    call printString
inputLoop:
    mov ah, 7
    int 21h

    cmp al, 8
    je inputRem
    cmp al, '0'
    jl exitInputNum
    cmp al, '9'
    jg exitInputNum
    sub al, '0'
    mov cl, al
    mov ax, bx
    mul si

    add ax, cx
```



```
    mov bx, ax

    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum

    jmp inputLoop
inputRem:
    mov ax, bx
    mov dx, 0
    div si
    mov bx, ax

    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum

    jmp inputLoop

exitInputNum:
    call endLine
    mov ax, bx

    pop si
    pop dx
    pop cx
    pop bx
    ret

printString:
    push ax

    mov ah, 09h
    int 21h

    pop ax
    ret

endLine:
```

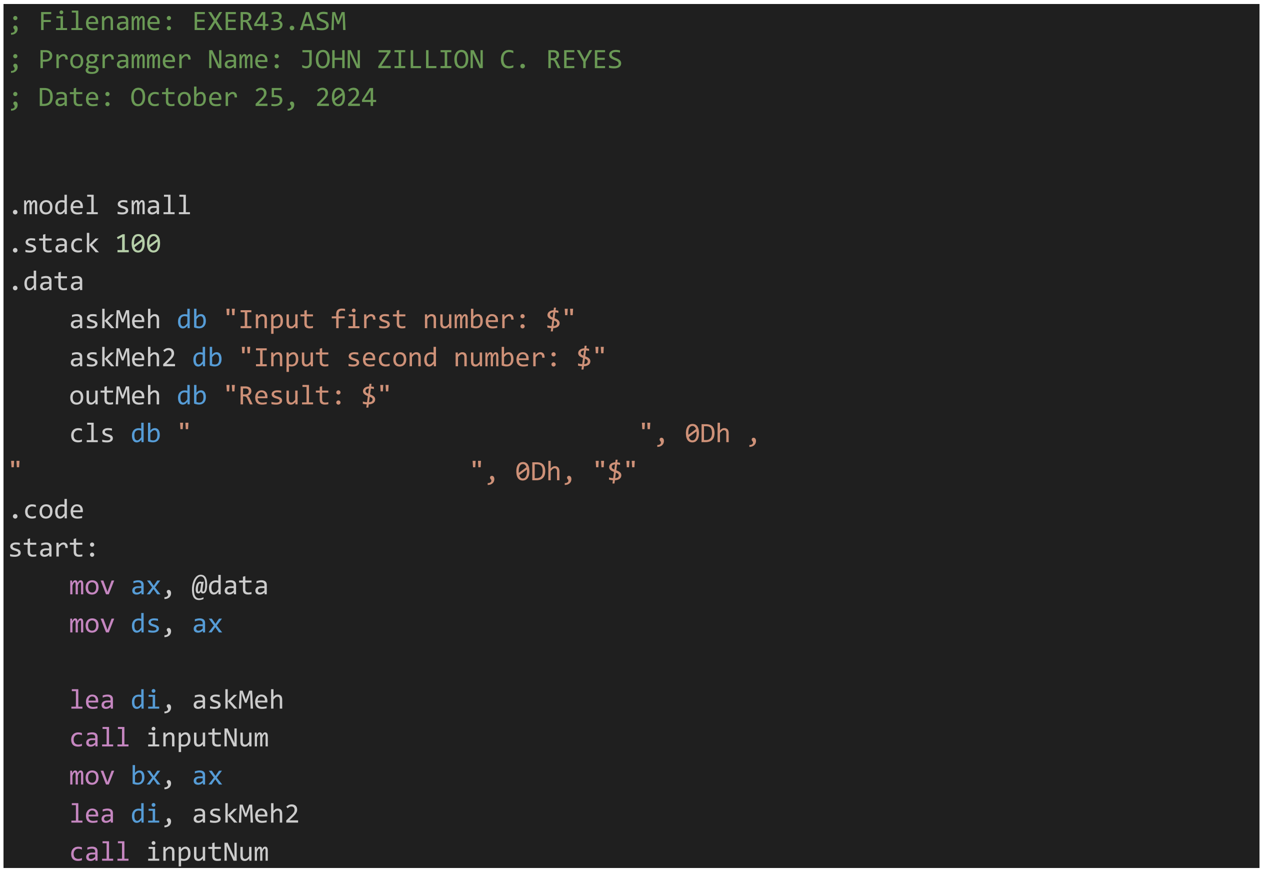


```
    push ax
    push dx

    mov ah, 02h
    mov dl, 0Ah
    int 21h

    pop dx
    pop ax
    ret
end start
```

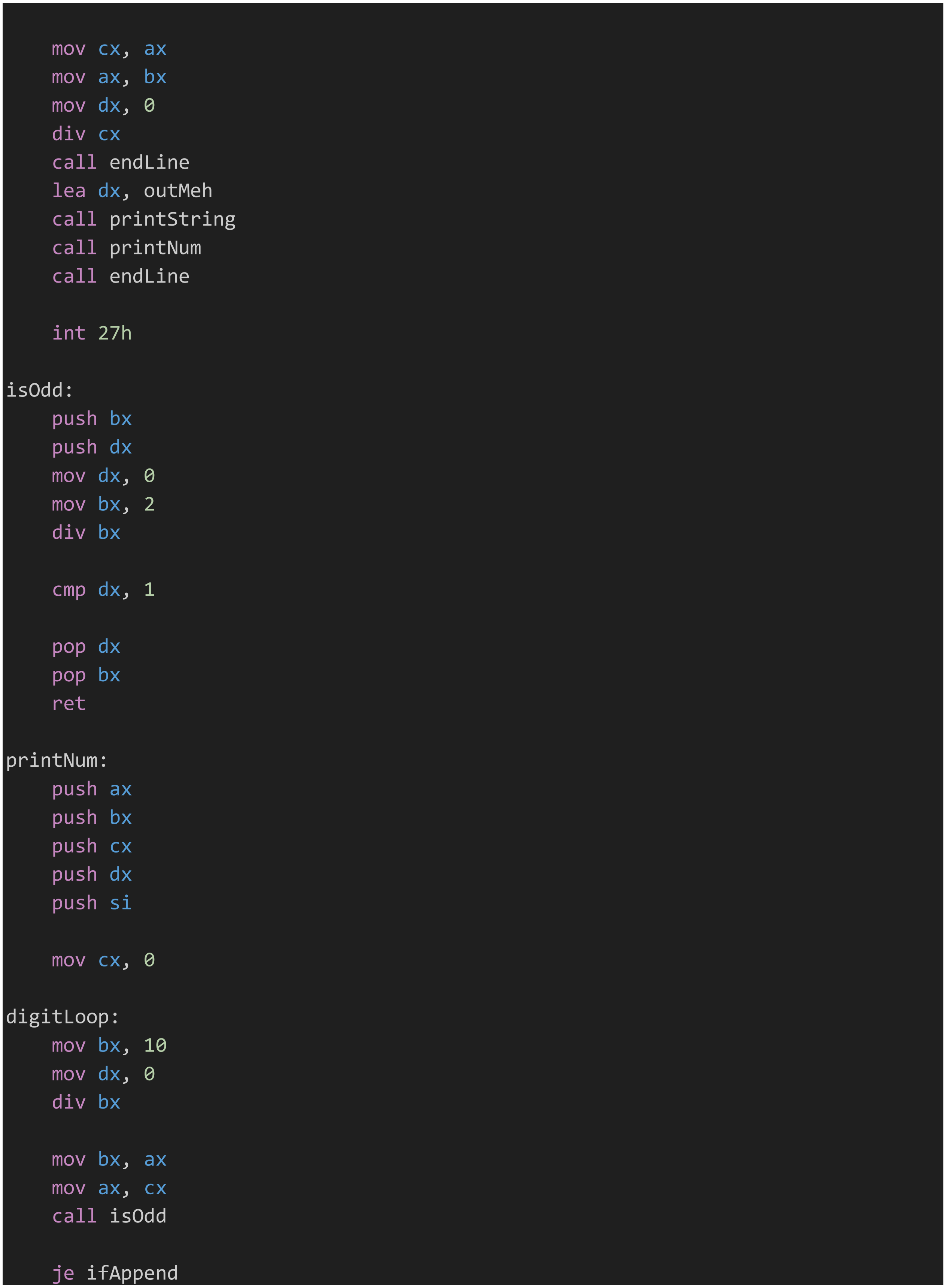


```
D:\>D:\TEST
Input first number: 255
Input second number: 2

Result: 510

Do you need to keep the DOSBox [Y,N]?_
```

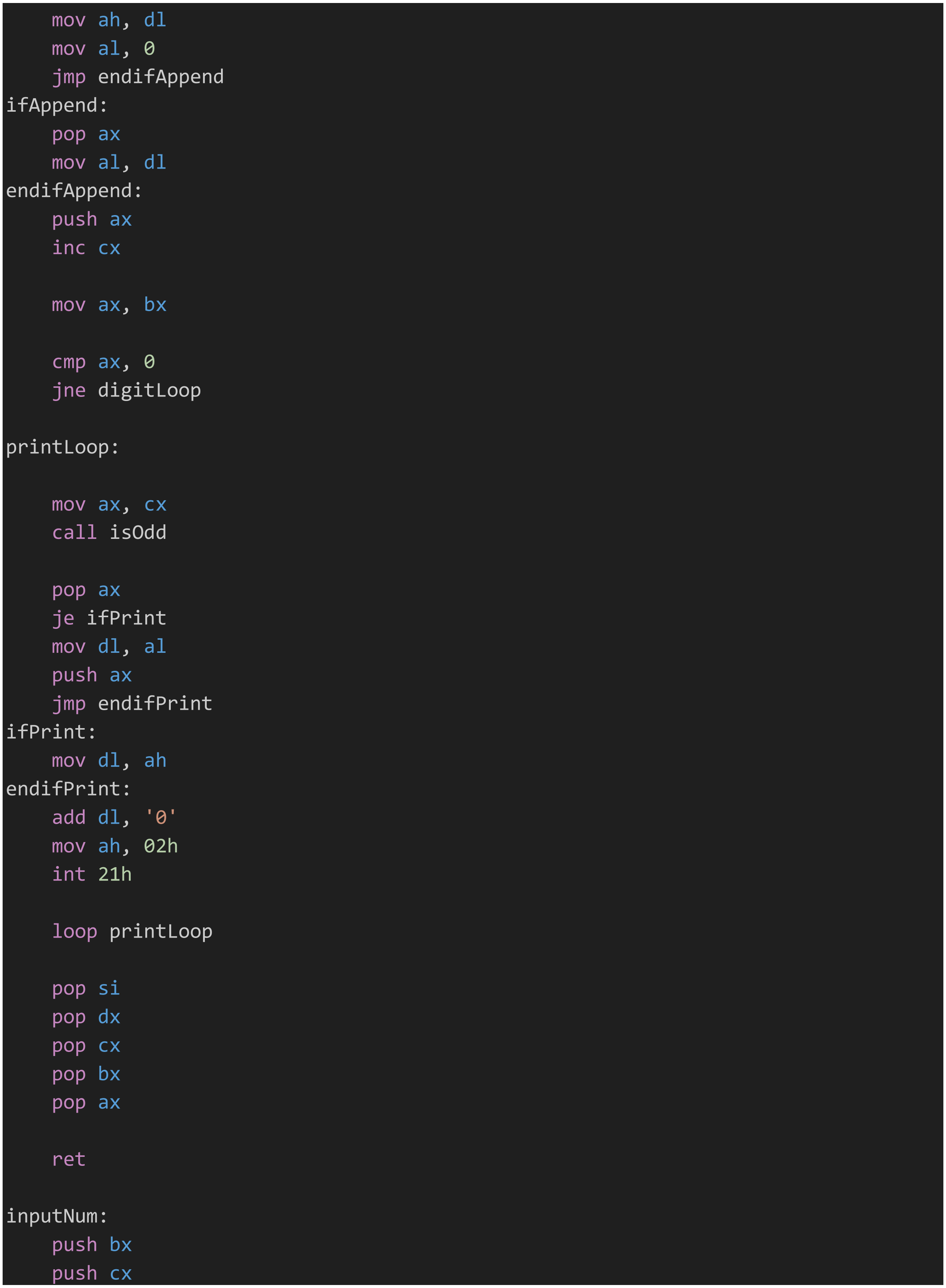
4. EXER43.ASM



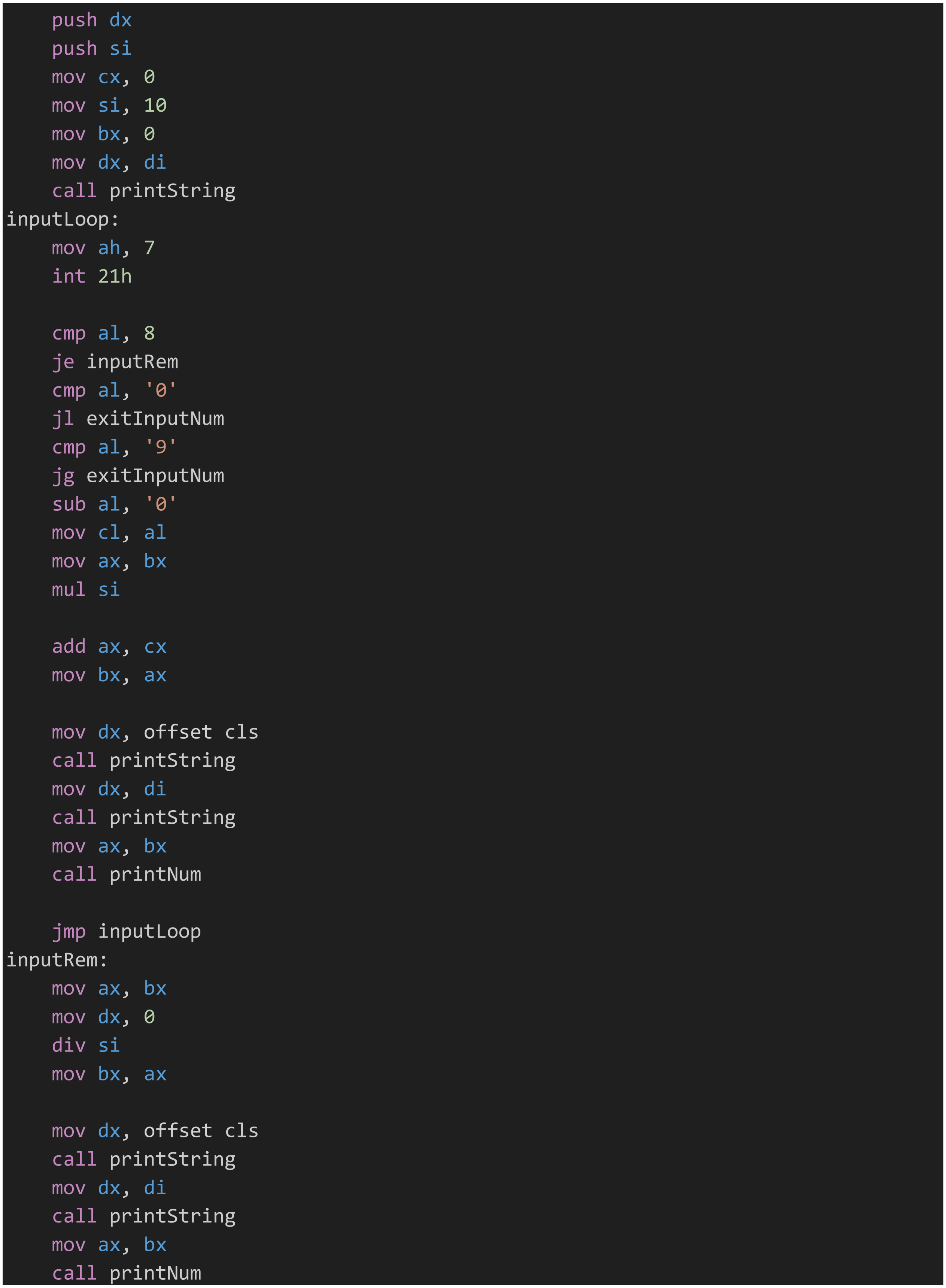
```
; Filename: EXER43.ASM
; Programmer Name: JOHN ZILLION C. REYES
; Date: October 25, 2024


.model small
.stack 100
.data
    askMeh db "Input first number: $"
    askMeh2 db "Input second number: $"
    outMeh db "Result: $"
    cls db "                             ", 0Dh ,
"                              ", 0Dh, "$"
.code
start:
    mov ax, @data
    mov ds, ax

    lea di, askMeh
    call inputNum
    mov bx, ax
    lea di, askMeh2
    call inputNum
```



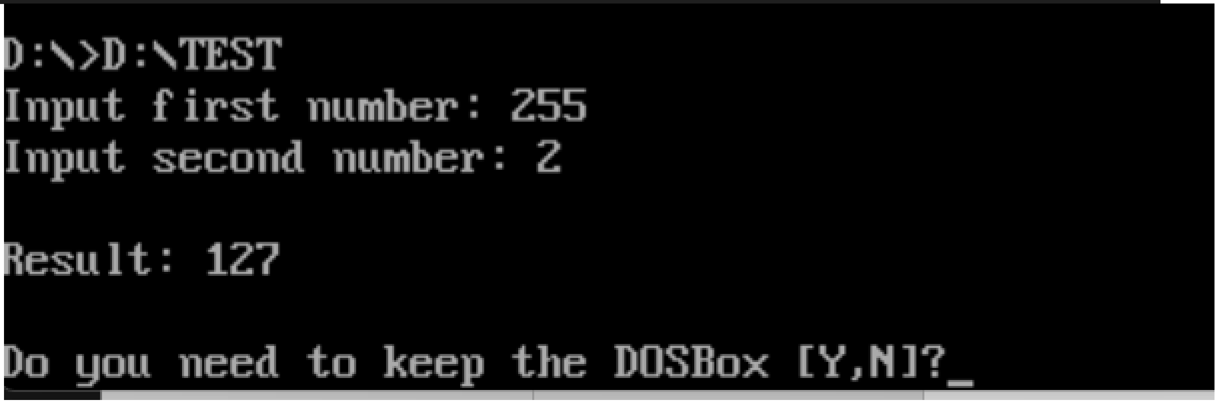
```
    mov cx, ax
    mov ax, bx
    mov dx, 0
    div cx
    call endLine
    lea dx, outMeh
    call printString
    call printNum
    call endLine

    int 27h

isOdd:
    push bx
    push dx
    mov dx, 0
    mov bx, 2
    div bx

    cmp dx, 1

    pop dx
    pop bx
    ret

printNum:
    push ax
    push bx
    push cx
    push dx
    push si

    mov cx, 0

digitLoop:
    mov bx, 10
    mov dx, 0
    div bx

    mov bx, ax
    mov ax, cx
    call isOdd

    je ifAppend
```



```
    mov ah, dl
    mov al, 0
    jmp endifAppend
ifAppend:
    pop ax
    mov al, dl
endifAppend:
    push ax
    inc cx

    mov ax, bx

    cmp ax, 0
    jne digitLoop

printLoop:

    mov ax, cx
    call isOdd

    pop ax
    je ifPrint
    mov dl, al
    push ax
    jmp endifPrint
ifPrint:
    mov dl, ah
endifPrint:
    add dl, '0'
    mov ah, 02h
    int 21h

    loop printLoop

    pop si
    pop dx
    pop cx
    pop bx
    pop ax

    ret

inputNum:
    push bx
    push cx
```

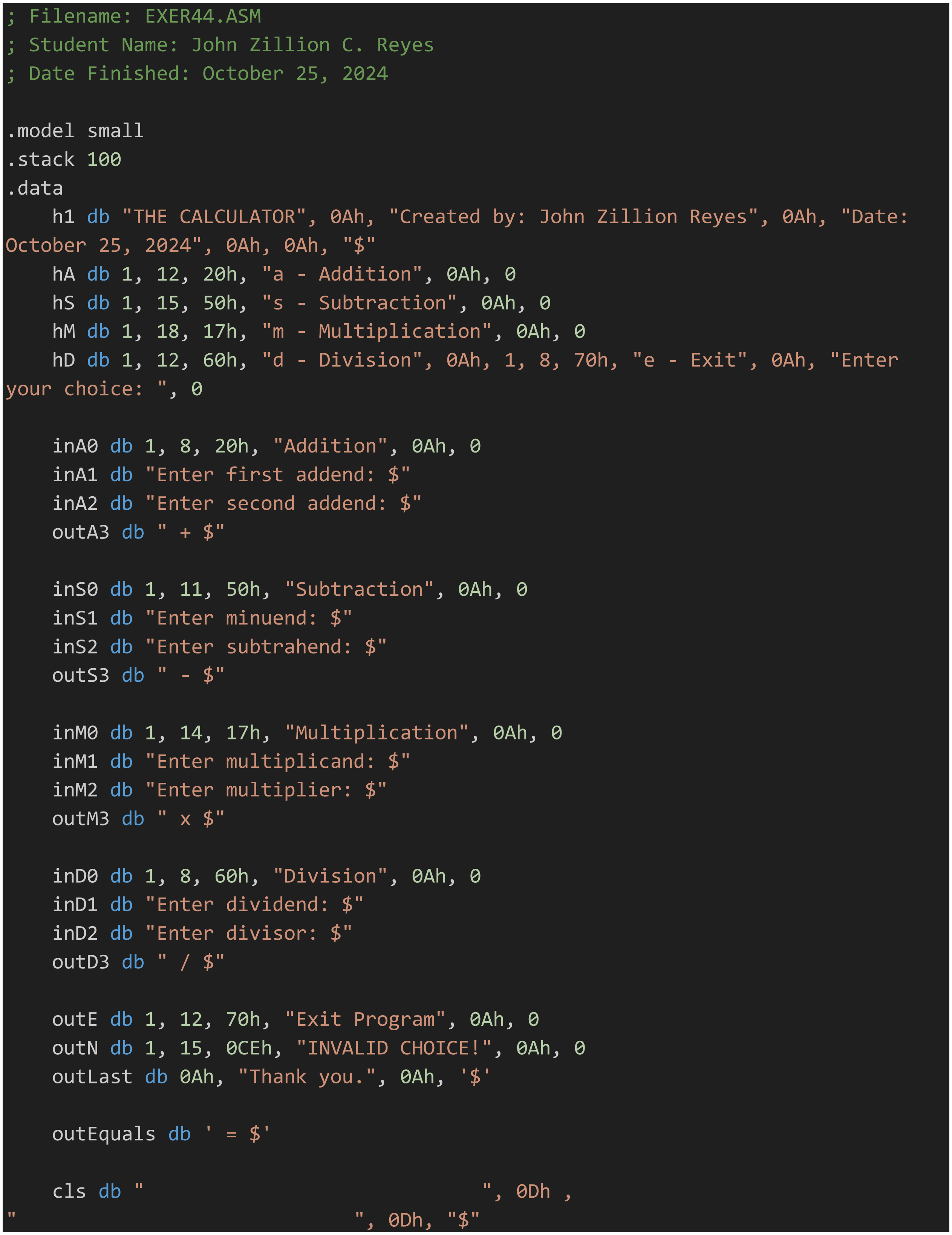


```
    push dx
    push si
    mov cx, 0
    mov si, 10
    mov bx, 0
    mov dx, di
    call printString
inputLoop:
    mov ah, 7
    int 21h

    cmp al, 8
    je inputRem
    cmp al, '0'
    jl exitInputNum
    cmp al, '9'
    jg exitInputNum
    sub al, '0'
    mov cl, al
    mov ax, bx
    mul si

    add ax, cx
    mov bx, ax

    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum

    jmp inputLoop
inputRem:
    mov ax, bx
    mov dx, 0
    div si
    mov bx, ax

    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum
    jmp inputLoop

exitInputNum:
    call endLine
    mov ax, bx

    pop si
    pop dx
    pop cx
    pop bx
    ret

printString:
    push ax

    mov ah, 09h
    int 21h

    pop ax
    ret

endLine:
    push ax
    push dx

    mov ah, 02h
    mov dl, 0Ah
    int 21h

    pop dx
    pop ax
    ret
end start
```

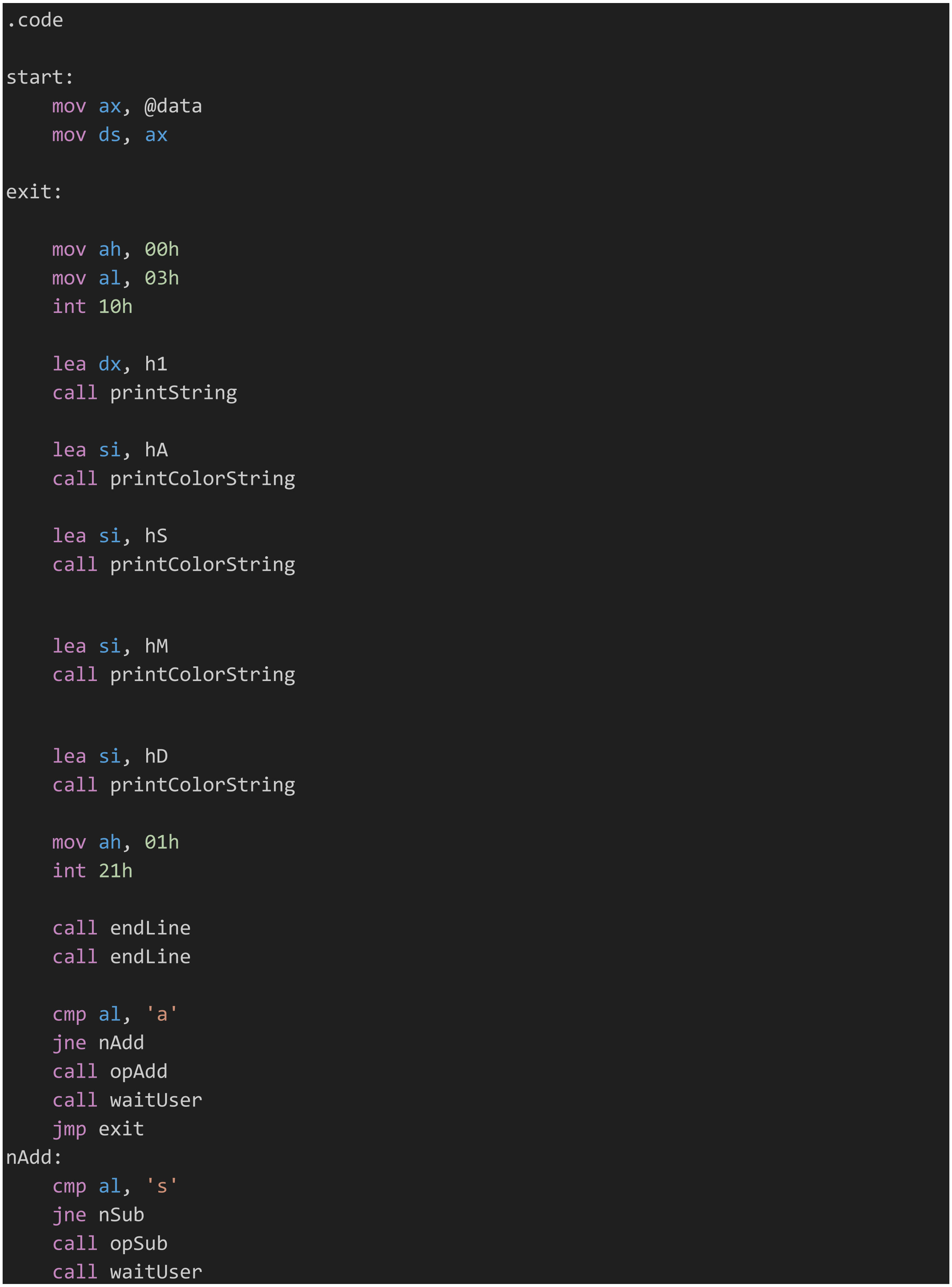


```
D:\>D:\TEST
Input first number: 255
Input second number: 2

Result: 127

Do you need to keep the DOSBox [Y,N]?_
```

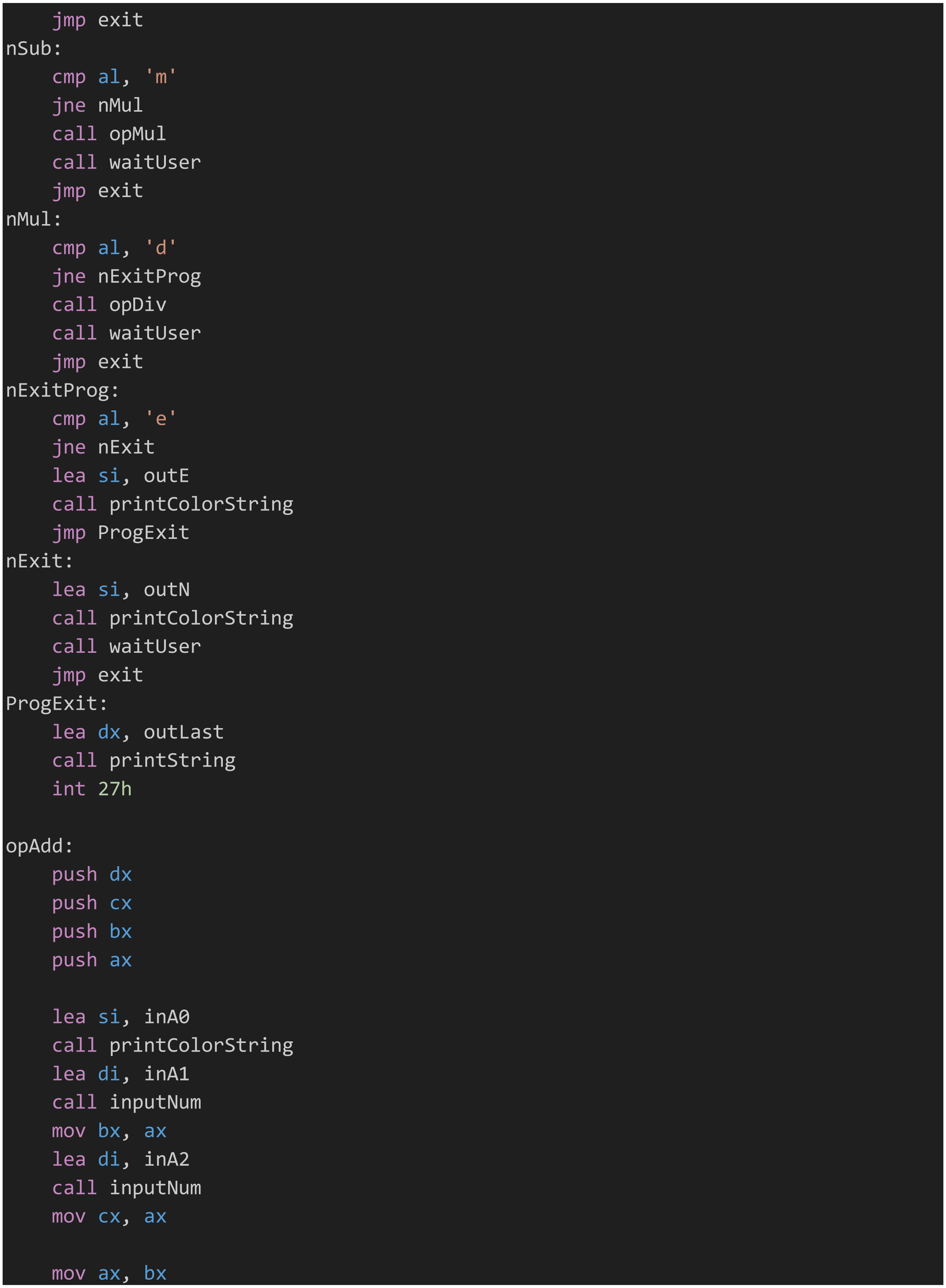
5. EXER44.ASM



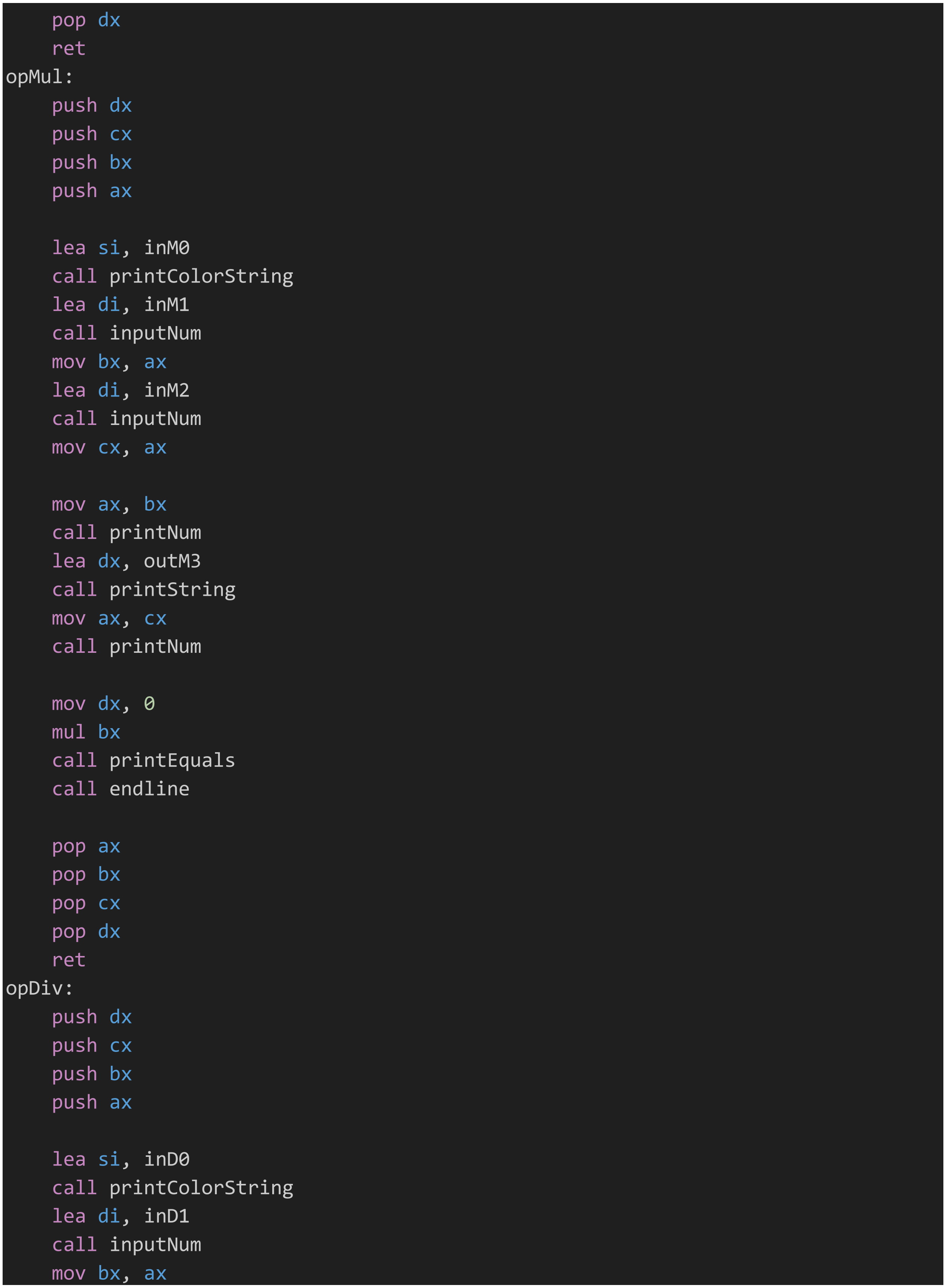
```
; Filename: EXER44.ASM
; Student Name: John Zillion C. Reyes
; Date Finished: October 25, 2024

.model small
.stack 100
.data
    h1 db "THE CALCULATOR", 0Ah, "Created by: John Zillion Reyes", 0Ah, "Date:
October 25, 2024", 0Ah, 0Ah, "$"
    hA db 1, 12, 20h, "a - Addition", 0Ah, 0
    hS db 1, 15, 50h, "s - Subtraction", 0Ah, 0
    hM db 1, 18, 17h, "m - Multiplication", 0Ah, 0
    hD db 1, 12, 60h, "d - Division", 0Ah, 1, 8, 70h, "e - Exit", 0Ah, "Enter
your choice: ", 0

    inA0 db 1, 8, 20h, "Addition", 0Ah, 0
    inA1 db "Enter first addend: $"
    inA2 db "Enter second addend: $"
    outA3 db " + $"

    inS0 db 1, 11, 50h, "Subtraction", 0Ah, 0
    inS1 db "Enter minuend: $"
    inS2 db "Enter subtrahend: $"
    outS3 db " - $"

    inM0 db 1, 14, 17h, "Multiplication", 0Ah, 0
    inM1 db "Enter multiplicand: $"
    inM2 db "Enter multiplier: $"
    outM3 db " x $"

    inD0 db 1, 8, 60h, "Division", 0Ah, 0
    inD1 db "Enter dividend: $"
    inD2 db "Enter divisor: $"
    outD3 db " / $"

    outE db 1, 12, 70h, "Exit Program", 0Ah, 0
    outN db 1, 15, 0CEh, "INVALID CHOICE!", 0Ah, 0
    outLast db 0Ah, "Thank you.", 0Ah, '$'

    outEquals db ' = $'

    cls db "                             ", 0Dh ,
"                             ", 0Dh, "$"
```



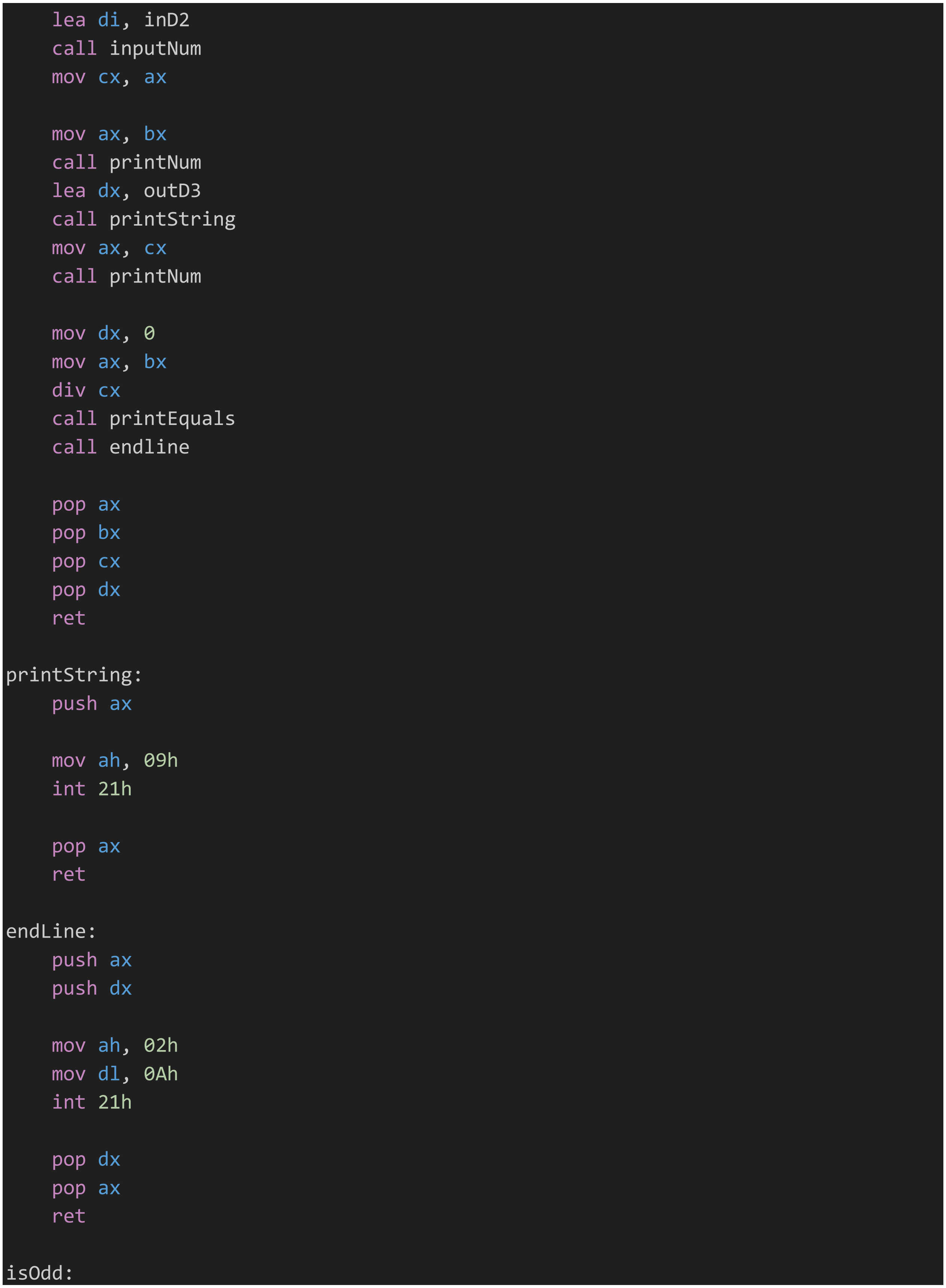
```
.code

start:
    mov ax, @data
    mov ds, ax

exit:

    mov ah, 00h
    mov al, 03h
    int 10h

    lea dx, h1
    call printString

    lea si, hA
    call printColorString

    lea si, hS
    call printColorString


    lea si, hM
    call printColorString


    lea si, hD
    call printColorString

    mov ah, 01h
    int 21h

    call endLine
    call endLine

    cmp al, 'a'
    jne nAdd
    call opAdd
    call waitUser
    jmp exit
nAdd:
    cmp al, 's'
    jne nSub
    call opSub
    call waitUser
```



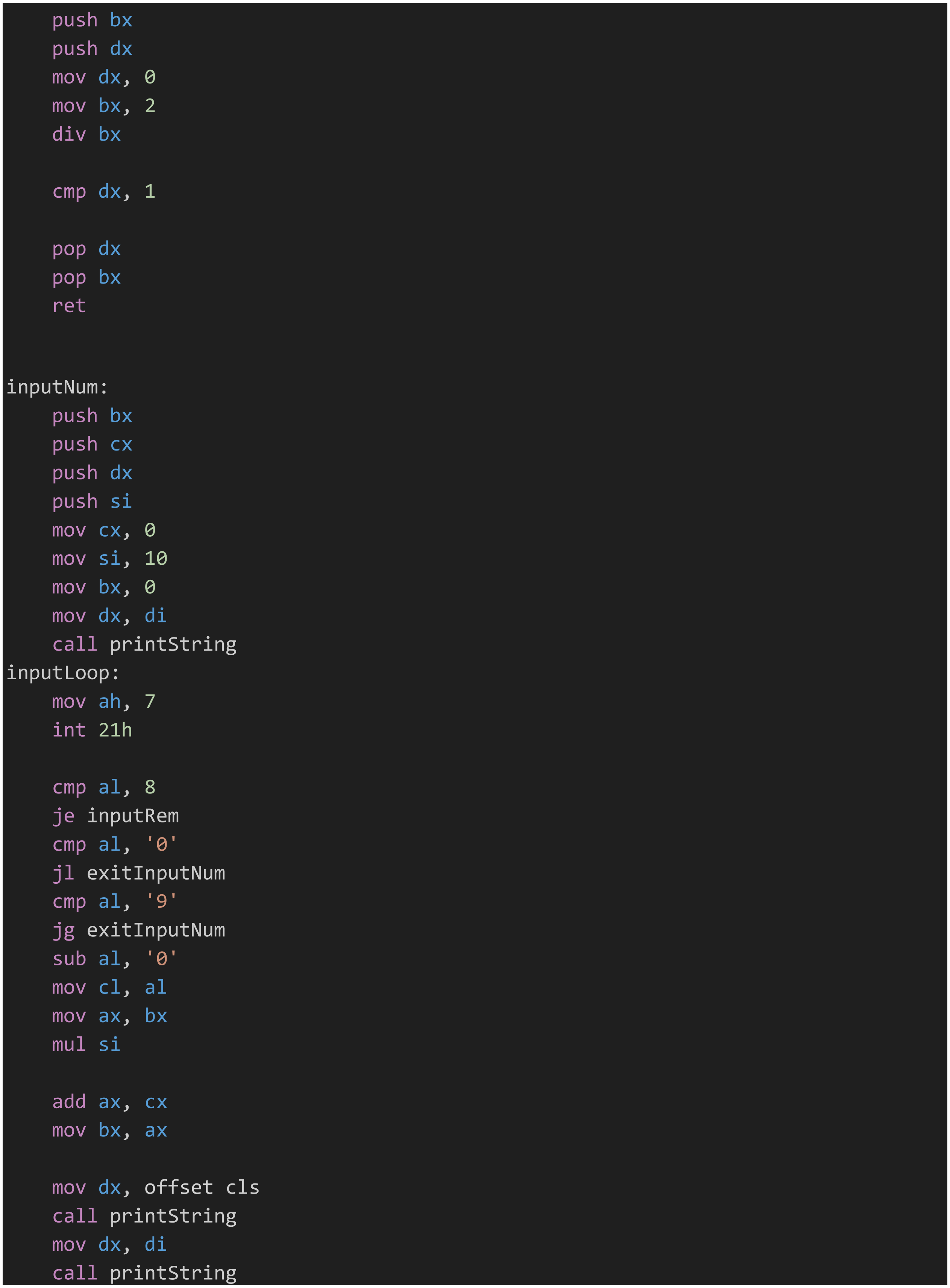
```
    jmp exit
nSub:
    cmp al, 'm'
    jne nMul
    call opMul
    call waitUser
    jmp exit
nMul:
    cmp al, 'd'
    jne nExitProg
    call opDiv
    call waitUser
    jmp exit
nExitProg:
    cmp al, 'e'
    jne nExit
    lea si, outE
    call printColorString
    jmp ProgExit
nExit:
    lea si, outN
    call printColorString
    call waitUser
    jmp exit
ProgExit:
    lea dx, outLast
    call printString
    int 27h

opAdd:
    push dx
    push cx
    push bx
    push ax

    lea si, inA0
    call printColorString
    lea di, inA1
    call inputNum
    mov bx, ax
    lea di, inA2
    call inputNum
    mov cx, ax

    mov ax, bx
    call printNum
    lea dx, outA3
    call printString
    mov ax, cx
    call printNum

    add ax, bx
    call printEquals
    call endline

    pop ax
    pop bx
    pop cx
    pop dx
    ret
opSub:
    push dx
    push cx
    push bx
    push ax

    lea si, inS0
    call printColorString
    lea di, inS1
    call inputNum
    mov bx, ax
    lea di, inS2
    call inputNum
    mov cx, ax

    mov ax, bx
    call printNum
    lea dx, outS3
    call printString
    mov ax, cx
    call printNum

    sub bx, ax
    mov ax, bx
    call printEquals
    call endline

    pop ax
    pop bx
    pop cx
```



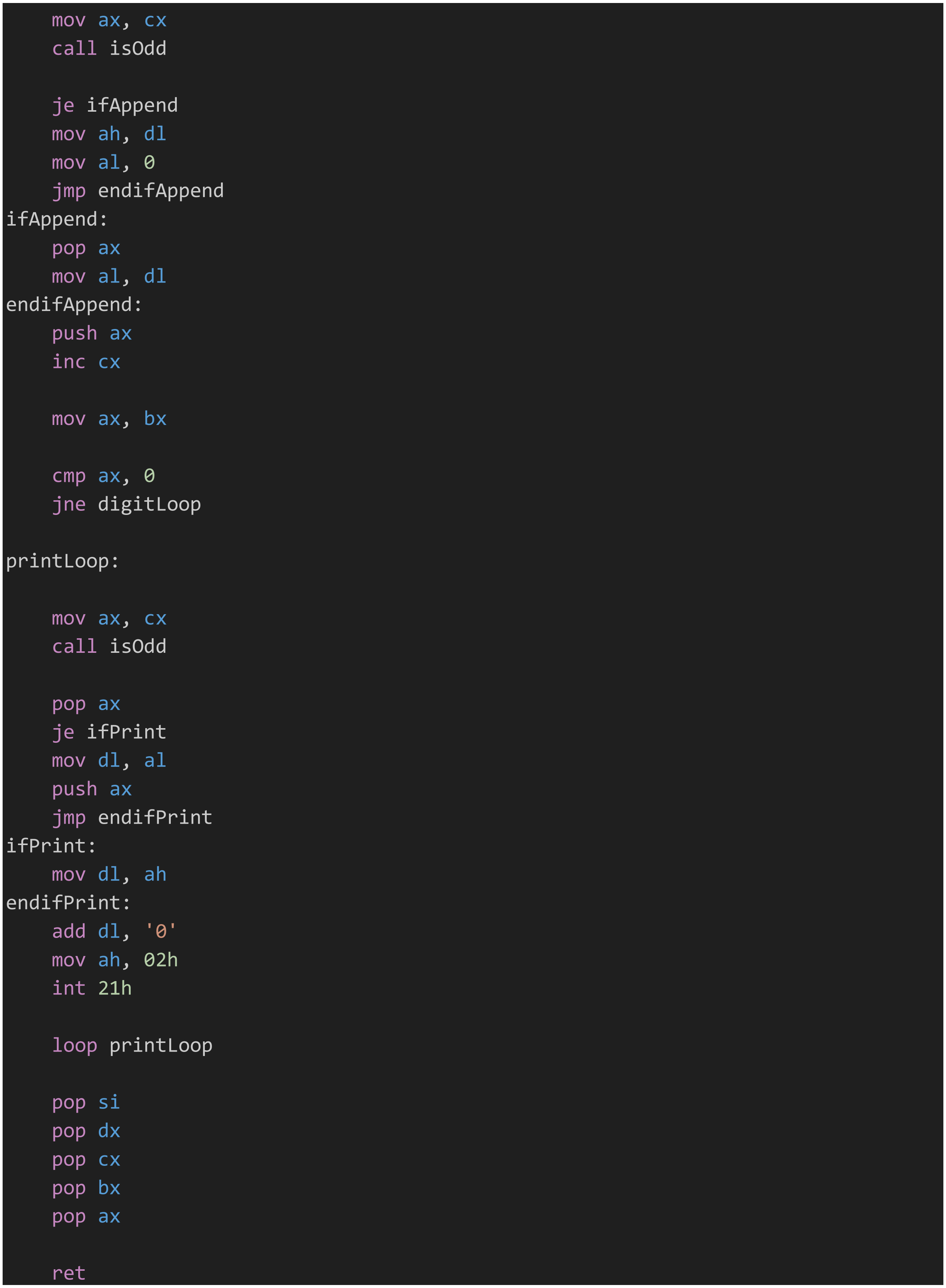
```
    pop dx
    ret
opMul:
    push dx
    push cx
    push bx
    push ax

    lea si, inM0
    call printColorString
    lea di, inM1
    call inputNum
    mov bx, ax
    lea di, inM2
    call inputNum
    mov cx, ax

    mov ax, bx
    call printNum
    lea dx, outM3
    call printString
    mov ax, cx
    call printNum

    mov dx, 0
    mul bx
    call printEquals
    call endline

    pop ax
    pop bx
    pop cx
    pop dx
    ret
opDiv:
    push dx
    push cx
    push bx
    push ax

    lea si, inD0
    call printColorString
    lea di, inD1
    call inputNum
    mov bx, ax
```



```
    lea di, inD2
    call inputNum
    mov cx, ax

    mov ax, bx
    call printNum
    lea dx, outD3
    call printString
    mov ax, cx
    call printNum

    mov dx, 0
    mov ax, bx
    div cx
    call printEquals
    call endline

    pop ax
    pop bx
    pop cx
    pop dx
    ret

printString:
    push ax

    mov ah, 09h
    int 21h

    pop ax
    ret

endLine:
    push ax
    push dx

    mov ah, 02h
    mov dl, 0Ah
    int 21h

    pop dx
    pop ax
    ret

isOdd:
```

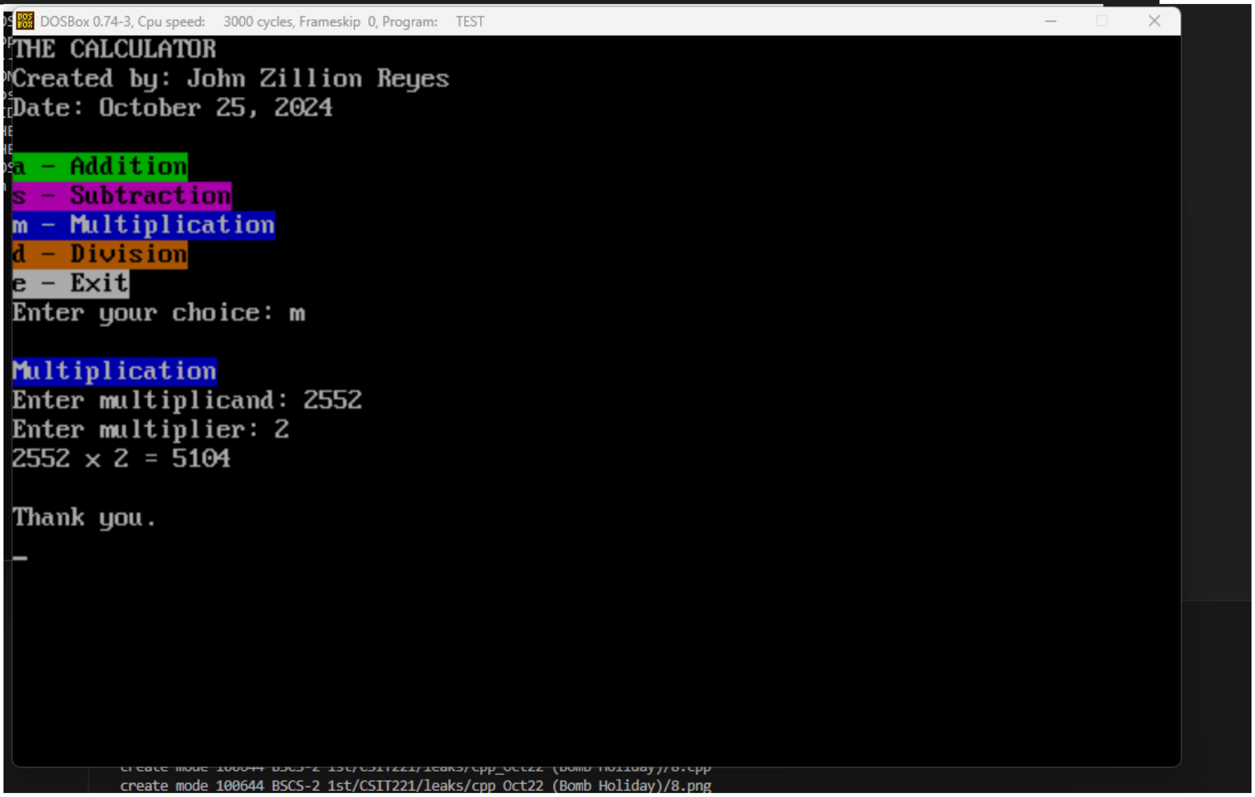


```
    push bx
    push dx
    mov dx, 0
    mov bx, 2
    div bx

    cmp dx, 1

    pop dx
    pop bx
    ret


inputNum:
    push bx
    push cx
    push dx
    push si
    mov cx, 0
    mov si, 10
    mov bx, 0
    mov dx, di
    call printString
inputLoop:
    mov ah, 7
    int 21h

    cmp al, 8
    je inputRem
    cmp al, '0'
    jl exitInputNum
    cmp al, '9'
    jg exitInputNum
    sub al, '0'
    mov cl, al
    mov ax, bx
    mul si

    add ax, cx
    mov bx, ax

    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum

    jmp inputLoop
inputRem:
    mov ax, bx
    mov dx, 0
    div si
    mov bx, ax

    mov dx, offset cls
    call printString
    mov dx, di
    call printString
    mov ax, bx
    call printNum

    jmp inputLoop

exitInputNum:
    call endLine
    mov ax, bx

    pop si
    pop dx
    pop cx
    pop bx
    ret


printNum:
    push ax
    push bx
    push cx
    push dx
    push si

    mov cx, 0

digitLoop:
    mov bx, 10
    mov dx, 0
    div bx

    mov bx, ax
```



```
    mov ax, cx
    call isOdd

    je ifAppend
    mov ah, dl
    mov al, 0
    jmp endifAppend
ifAppend:
    pop ax
    mov al, dl
endifAppend:
    push ax
    inc cx

    mov ax, bx

    cmp ax, 0
    jne digitLoop

printLoop:

    mov ax, cx
    call isOdd

    pop ax
    je ifPrint
    mov dl, al
    push ax
    jmp endifPrint
ifPrint:
    mov dl, ah
endifPrint:
    add dl, '0'
    mov ah, 02h
    int 21h

    loop printLoop

    pop si
    pop dx
    pop cx
    pop bx
    pop ax

    ret
```

```
printEquals:
    lea dx, outEquals
    call printString
    call printNum
    ret

printColorString:
    push ax
    push bx
    push cx
    push dx
printColorLoop:
    mov bl, 0
    cmp [si], bl
    je printColorExit
    mov bl, 1
    cmp [si], bl
    jne notColor
    mov ah, 09h
    inc si
    mov cl, [si]
    mov ch, 0
    inc si
    mov bl, [si]
    int 10h
    inc si
    jmp printColorLoop
notColor:
    mov ah, 02h
    mov dl, [si]
    int 21h
    inc si
    jmp printColorLoop
printColorExit:
    pop dx
    pop cx
    pop bx
    pop ax
    ret

waitUser:
    push ax
    push dx
    lea dx, outLast
    call printString
    mov ah, 7
    int 21h
    pop dx
    pop ax
    ret

end start
```


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: TEST

```
THE CALCULATOR
Created by: John Zillion Reyes
Date: October 25, 2024

a - Addition
s - Subtraction
m - Multiplication
d - Division
e - Exit
Enter your choice: m

Multiplication
Enter multiplicand: 2552
Enter multiplier: 2
2552 x 2 = 5104

Thank you.
_
```

```
create mode 100644 BSCS-2 1st/CSIT221/leaks/cpp_Oct22 (Bomb Holiday)/8.png
```