VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Microprocessors and Interfacing

(CSE – 3002)

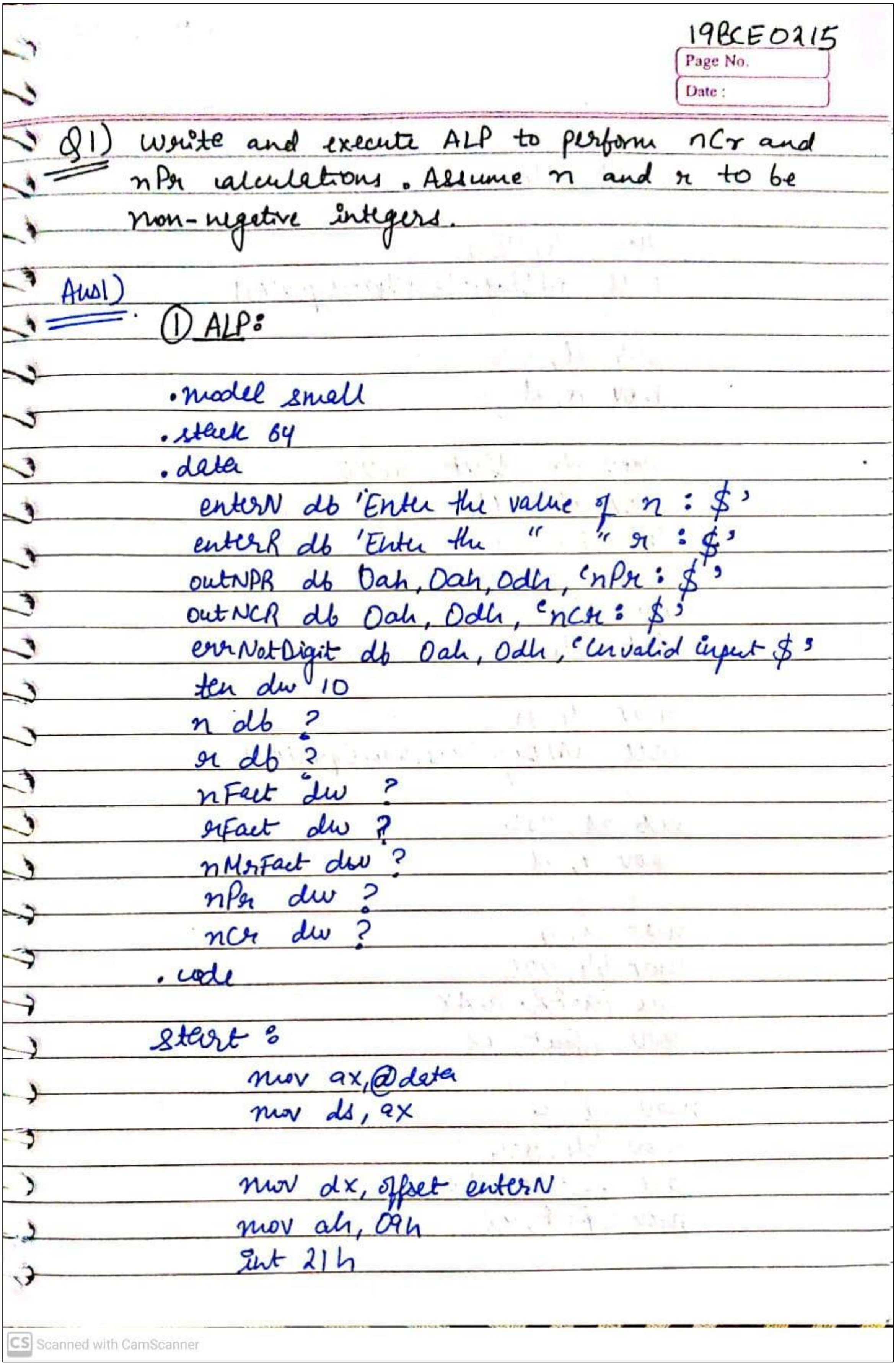
LAB EXPERIMENT– 2

Name: **Vibhu Kumar Singh**

Reg. No: **19BCE0215**

Teacher: **Mr. Konguvel E.**

1. Write and execute ALP to perform nCr and nPr calculations. Assume n and r to be non-negative integers.



19BCE0215

Page No.

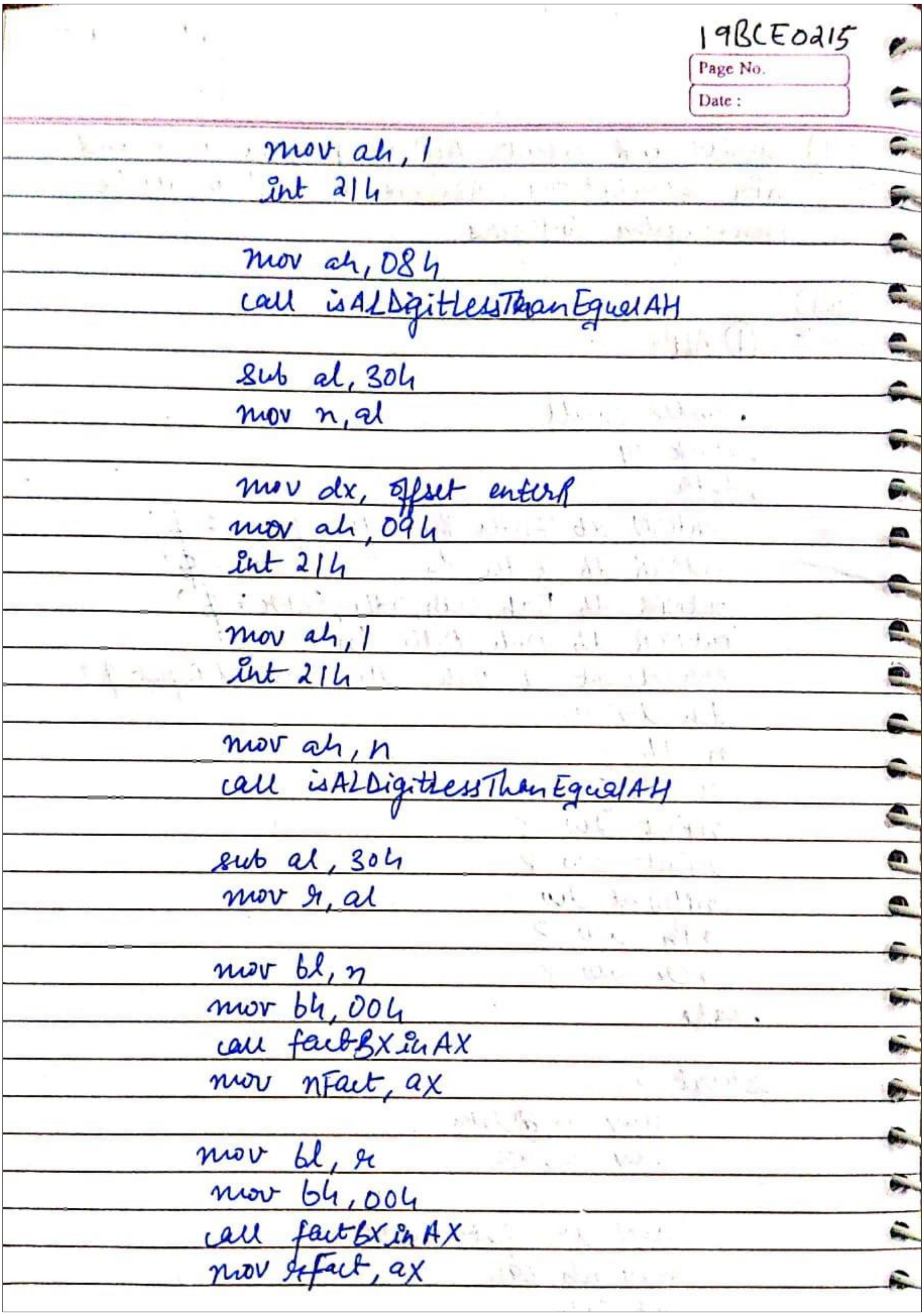
Date :

Q1) Write and execute ALP to perform nCr and nPr calculations. Assume n and r to be non-negative integers.

Ans1)

① ALP:

```
.model small
.stack 64
.data
  enterN db 'Enter the value of n : $'
  enterR db 'Enter the   "    " r : $'
  outNPR db 0ah,0ah,0dh,'nPr : $'
  outNCR db 0ah, 0dh, 'nCr : $'
  errNotDigit db 0ah, 0dh, 'Invalid input $'
  ten dw 10
  n db ?
  r db ?
  nFact dw ?
  rFact dw ?
  nMrFact dw ?
  nPr dw ?
  nCr dw ?
.code

start:
      mov ax,@data
      mov ds, ax

      mov dx, offset enterN
      mov ah, 09h
      int 21h
```

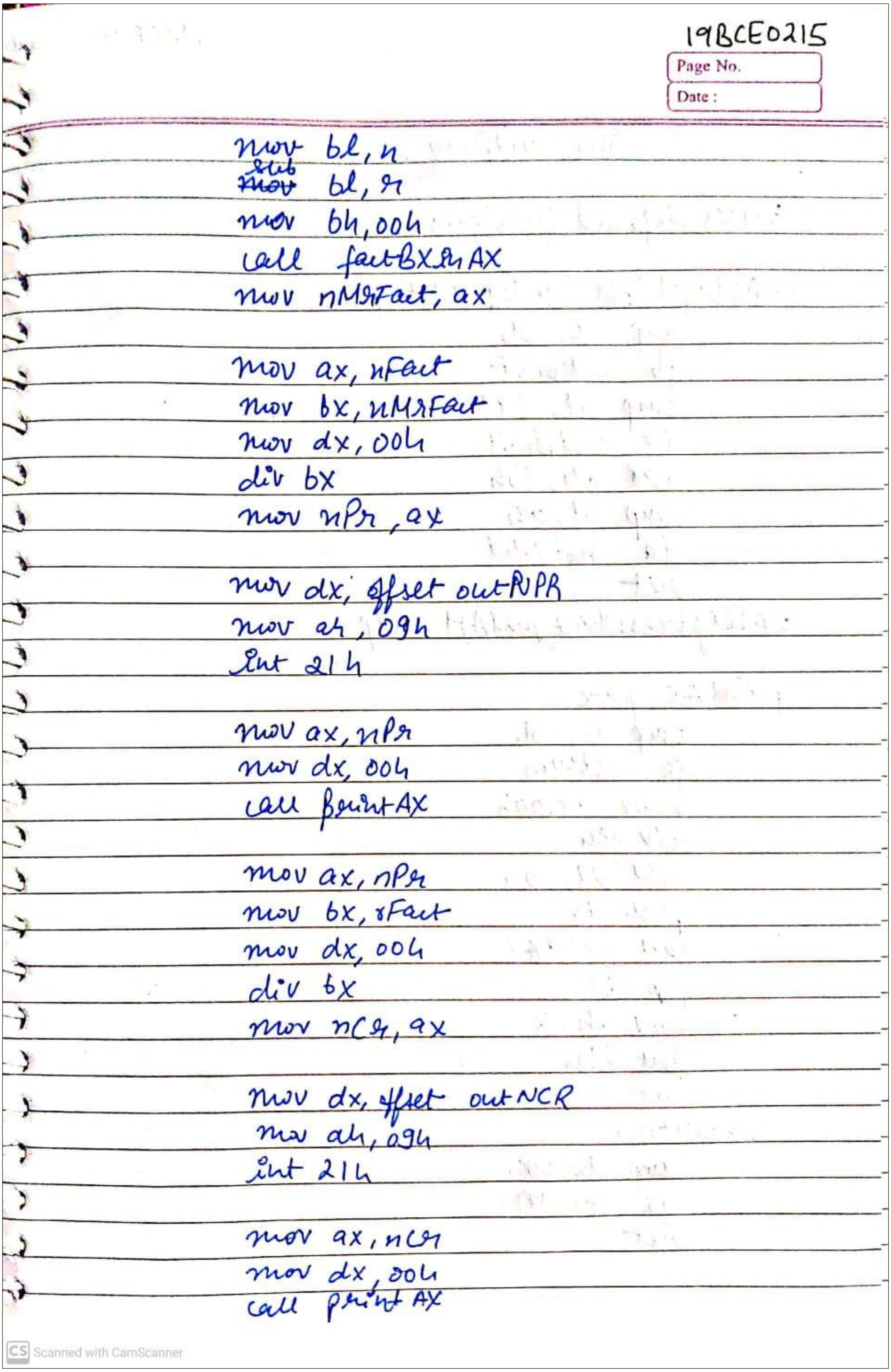


19BCE0215

Page No.

Date :

```
mov ah, 1
int 21h

mov ah, 08h
call isALDgitLessThanEqualAH

sub al, 30h
mov n, al

mov dx, offset enterR
mov ah, 09h
int 21h

mov ah, 1
int 21h

mov ah, n
call isALDigitLessThanEqualAH

sub al, 30h
mov r, al

mov bl, n
mov bh, 00h
call factBXinAX
mov nFact, ax

mov bl, r
mov bh, 00h
call factBXinAX
mov rFact, ax
```

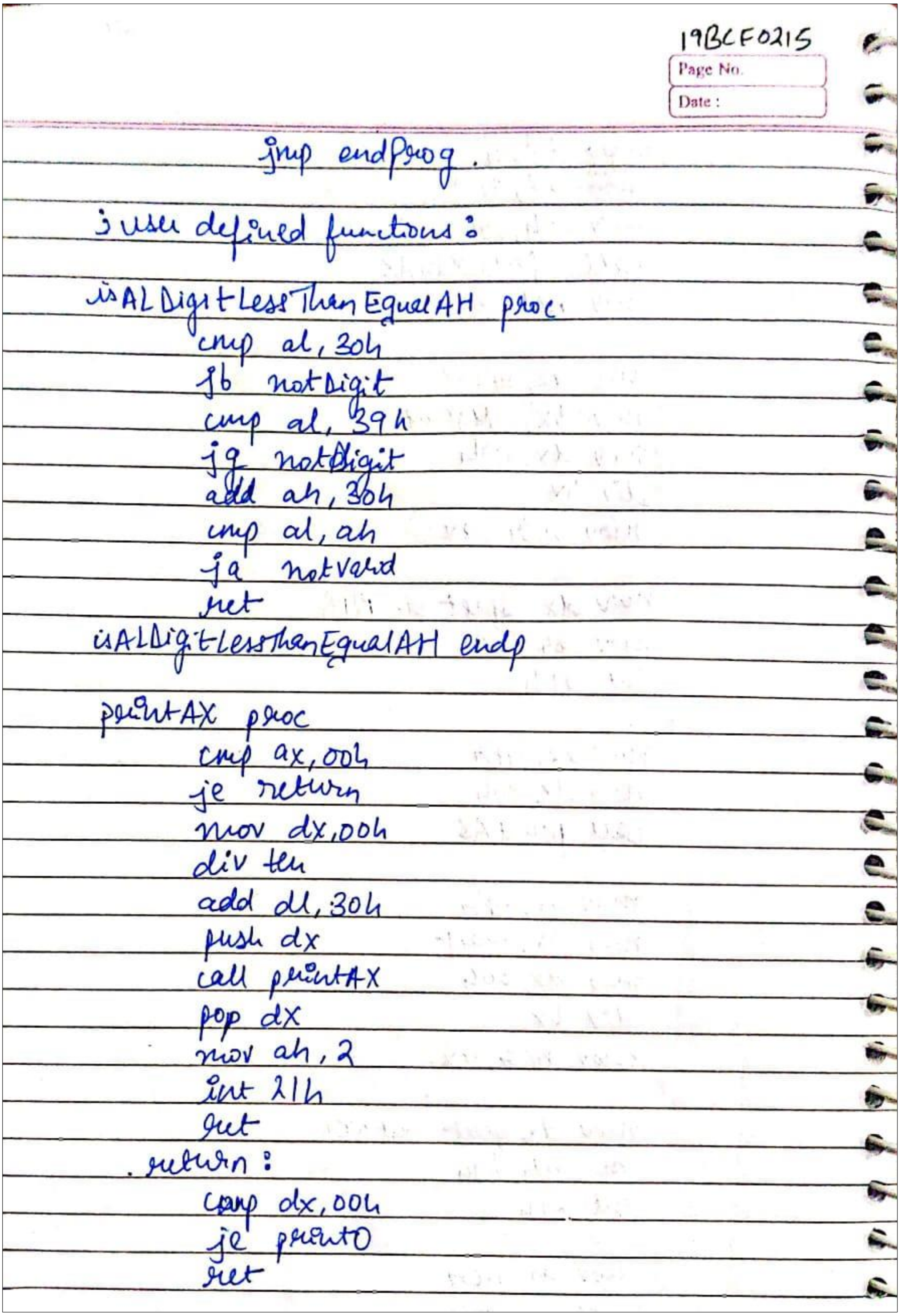


19BCE0215

Page No.

Date :

```
mov bl, n
~~mov~~ sub bl, r
mov bh, 00h
call factBXinAX
mov nMrFact, ax

mov ax, nFact
mov bx, nMrFact
mov dx, 00h
div bx
mov nPr , ax

mov dx, offset outNPR
mov ah, 09h
int 21h

mov ax, nPr
mov dx, 00h
call printAX

mov ax, nPr
mov bx, rFact
mov dx, 00h
div bx
mov nCr, ax

mov dx, offset outNCR
mov ah, 09h
int 21h

mov ax, nCr
mov dx, 00h
call print AX
```

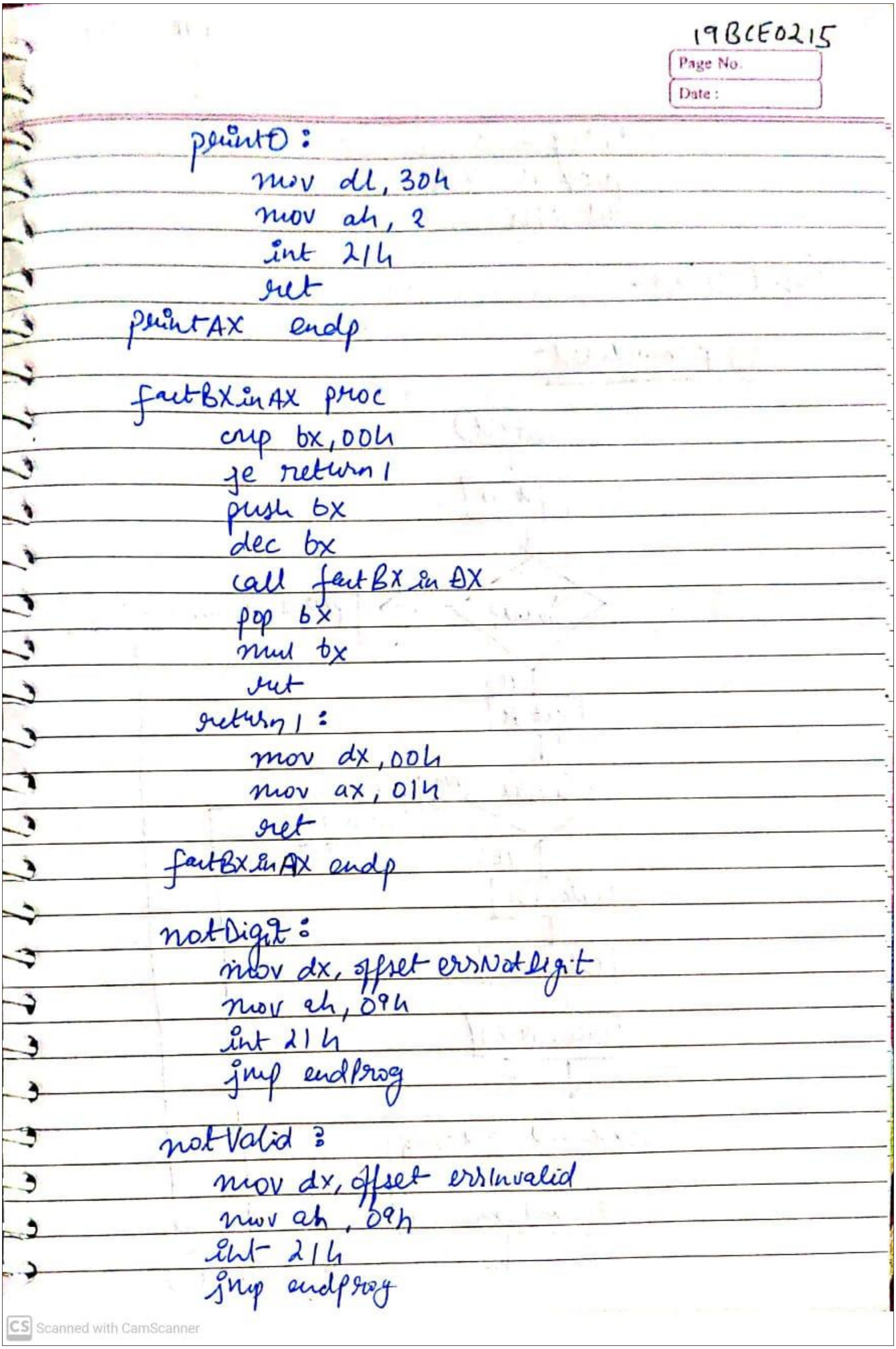


19BCF0215

Page No.

Date :

```
        jmp endProg.
; user defined functions:
isALDigitLessThanEqualAH proc.
        cmp al, 30h
        jb notDigit
        cmp al, 39h
        jg notDigit
        add ah, 30h
        cmp al, ah
        ja notValid
        ret
isALDigitLessthanEqualAH endp

printAX proc
        cmp ax,00h
        je return
        mov dx,00h
        div ten
        add dl,30h
        push dx
        call printAX
        pop dx
        mov ah,2
        int 21h
        ret
    return:
        cmp dx,00h
        je print0
        ret
```



19BCE0215

Page No.

Date :

```
printO :
        mov dl, 30h
        mov ah, 2
        int 21h
        ret
printAX endp

factBXinAX proc
      cmp bx,00h
      je return1
      push bx
      dec bx
      call factBX in AX
      pop bx
      mul bx
      ret
   return1 :
        mov dx,00h
        mov ax,01h
        ret
factBXinAX endp

notDigit :
      mov dx, offset errNotDigit
      mov ah,09h
      int 21h
      jmp endProg

notValid :
      mov dx, offset errInvalid
      mov ah , 09h
      int 21h
      jmp endProg
```

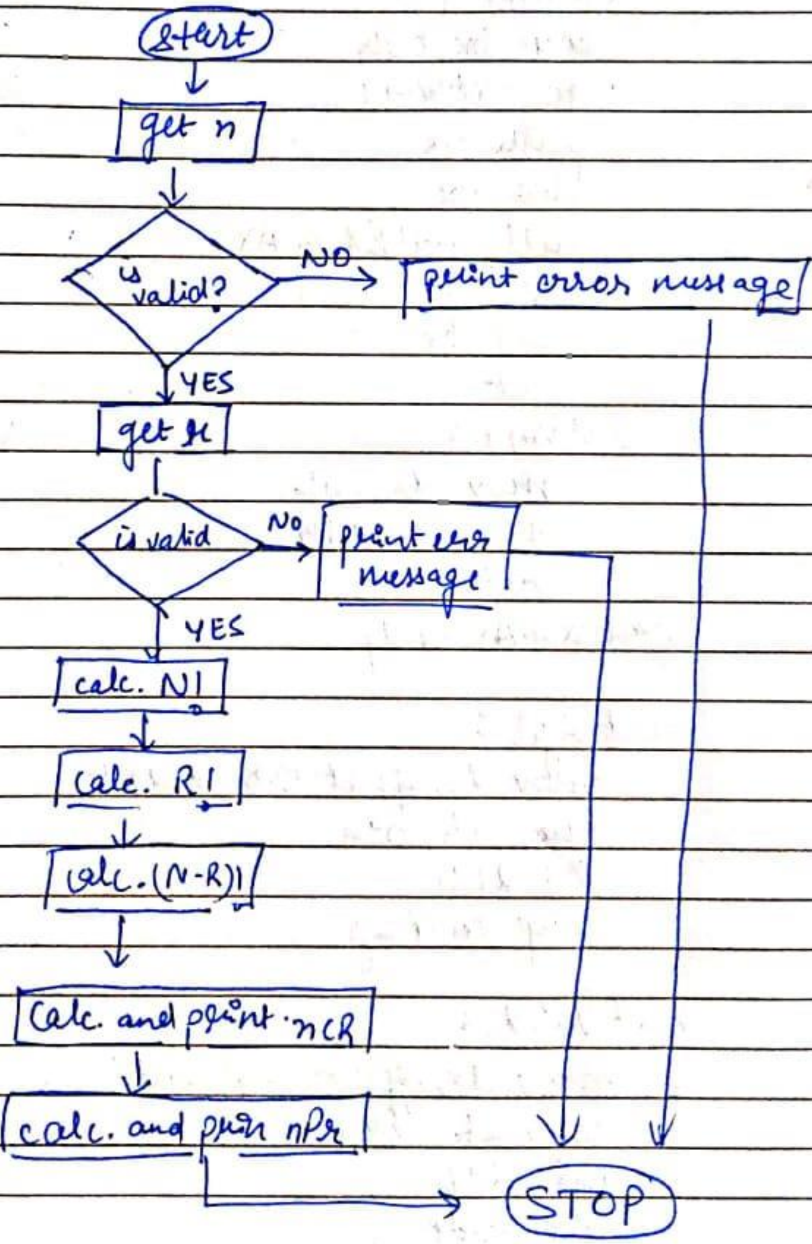
19BCE0215

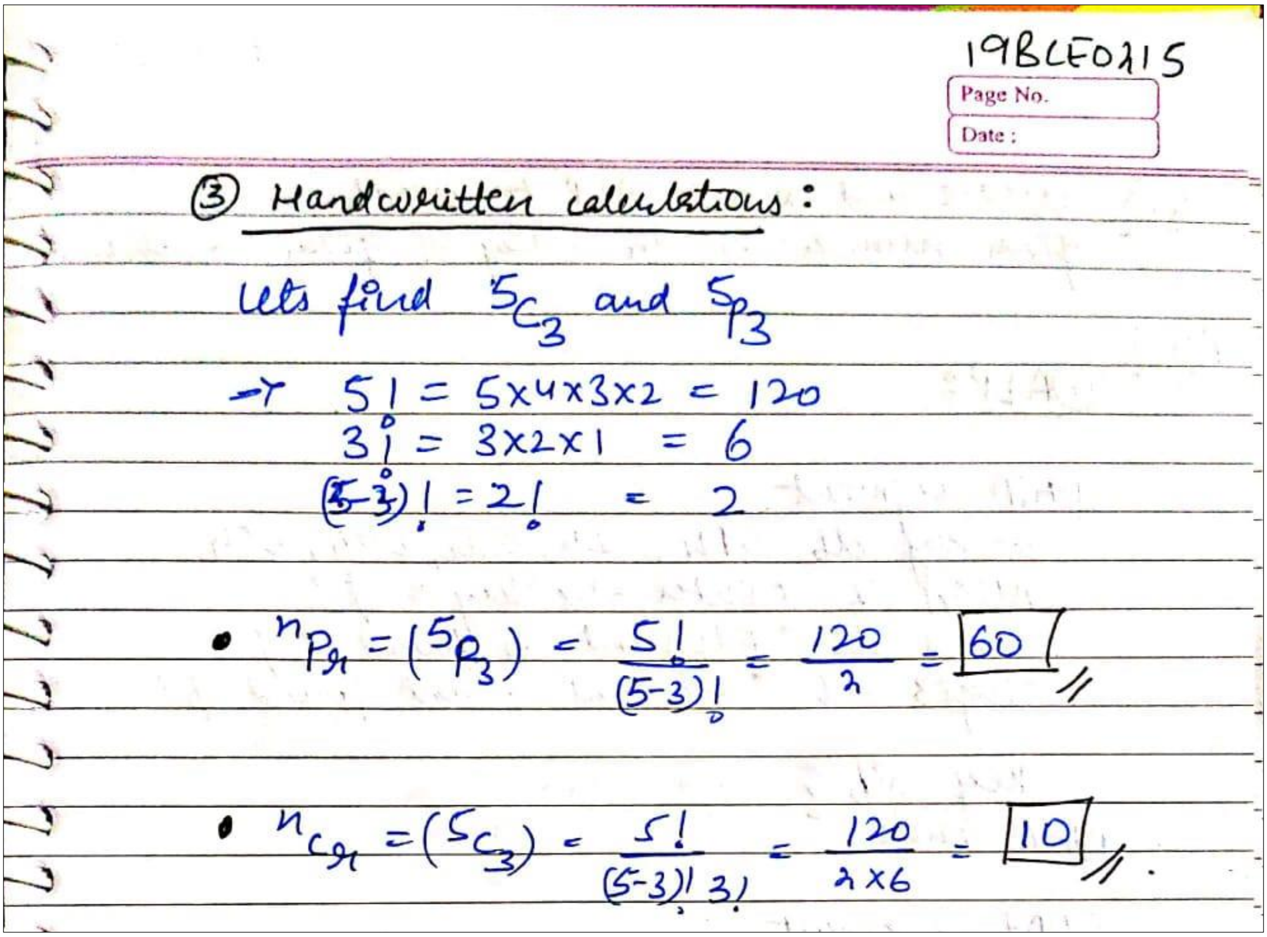
Page No.

Date :

```
endProg :
    mov ah, 04ch
    int 21h

end start
```

② Flowchart





19BCE0215

Page No.

Date :

③ Handwritten calculations :

Lets find ${}^5C_3$ and ${}^5P_3$

→ $5! = 5 \times 4 \times 3 \times 2 = 120$

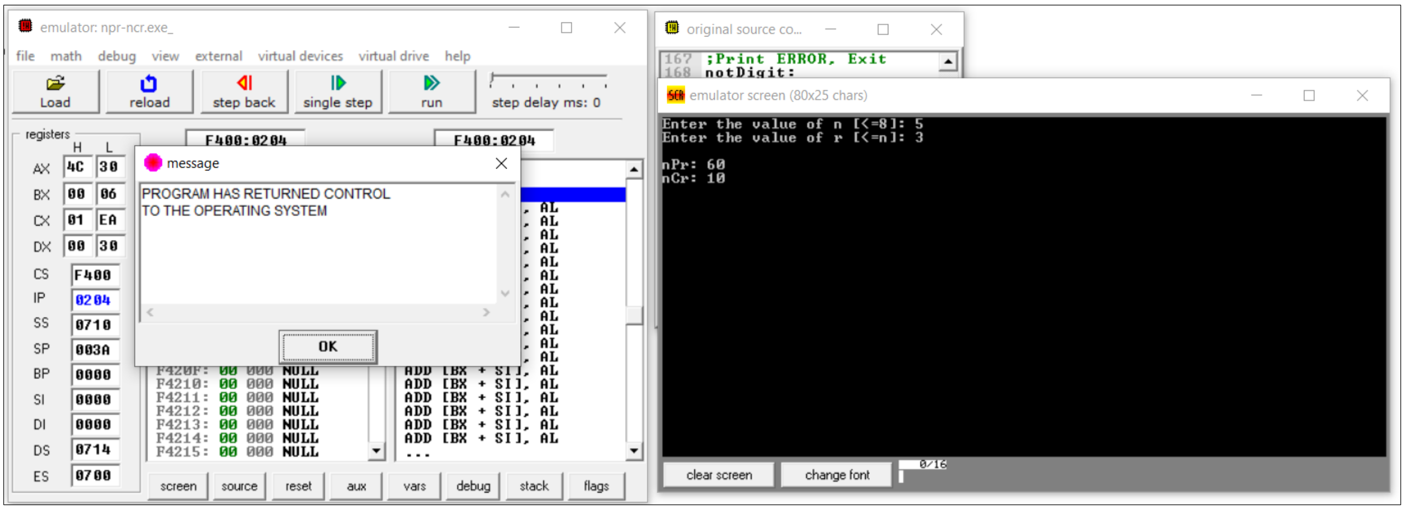
$3! = 3 \times 2 \times 1 = 6$

$(5-3)! = 2! = 2$

- $^nP_r = ({}^5P_3) = \frac{5!}{(5-3)!} = \frac{120}{2} = \boxed{60}$ //

- $^nC_r = ({}^5C_3) = \frac{5!}{(5-3)!\,3!} = \frac{120}{2 \times 6} = \boxed{10}$ //.

Screenshot of Output:



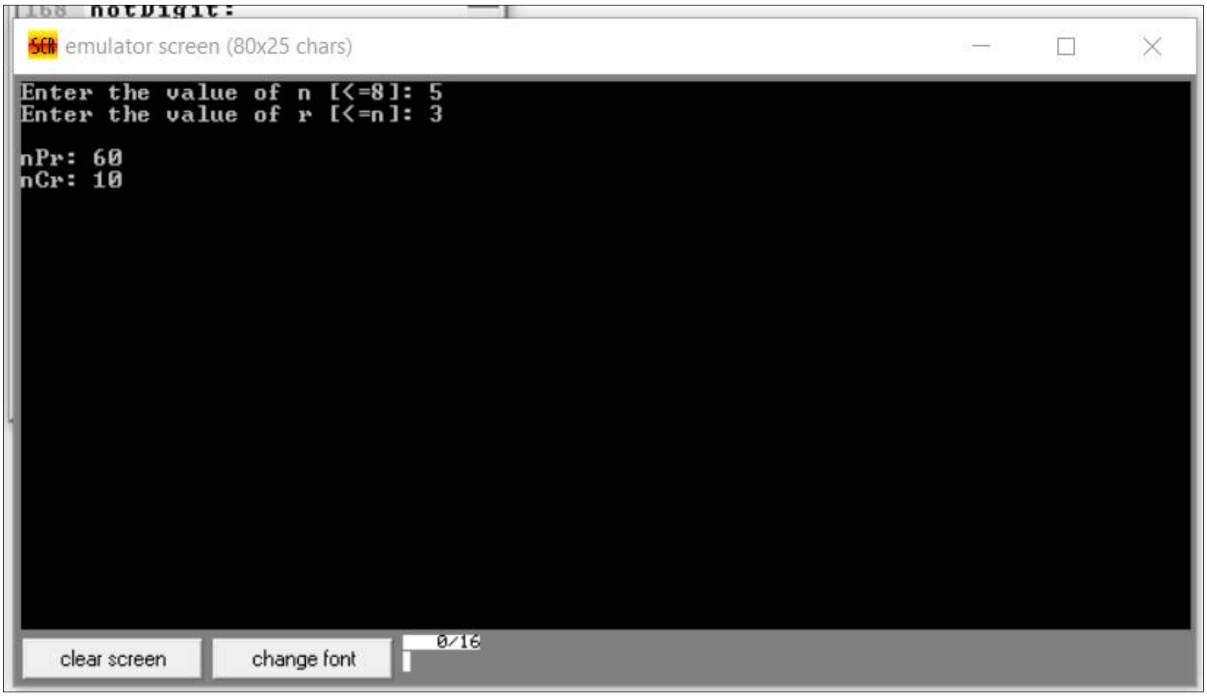
message

```
PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM
```

OK

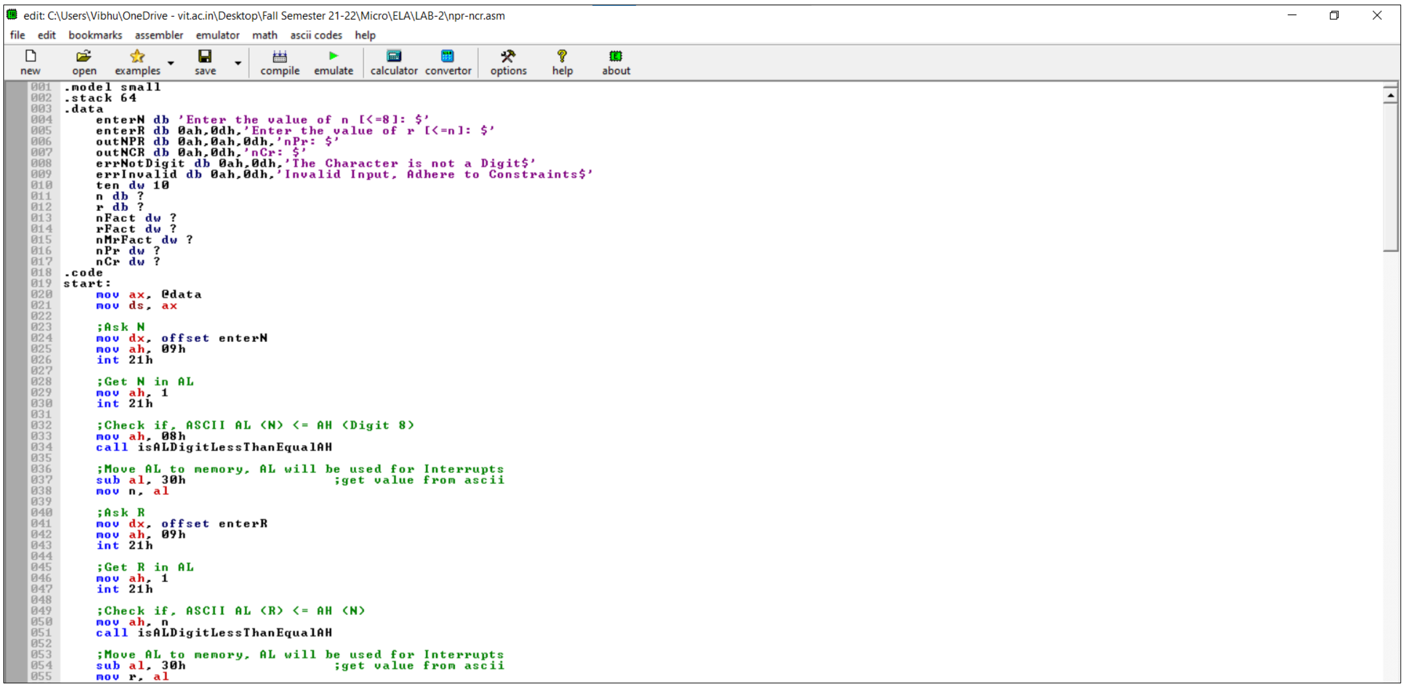
```
167 ;Print ERROR, Exit
168 notDigit:
```

```
Enter the value of n [<=8]: 5
Enter the value of r [<=n]: 3

nPr: 60
nCr: 10
```

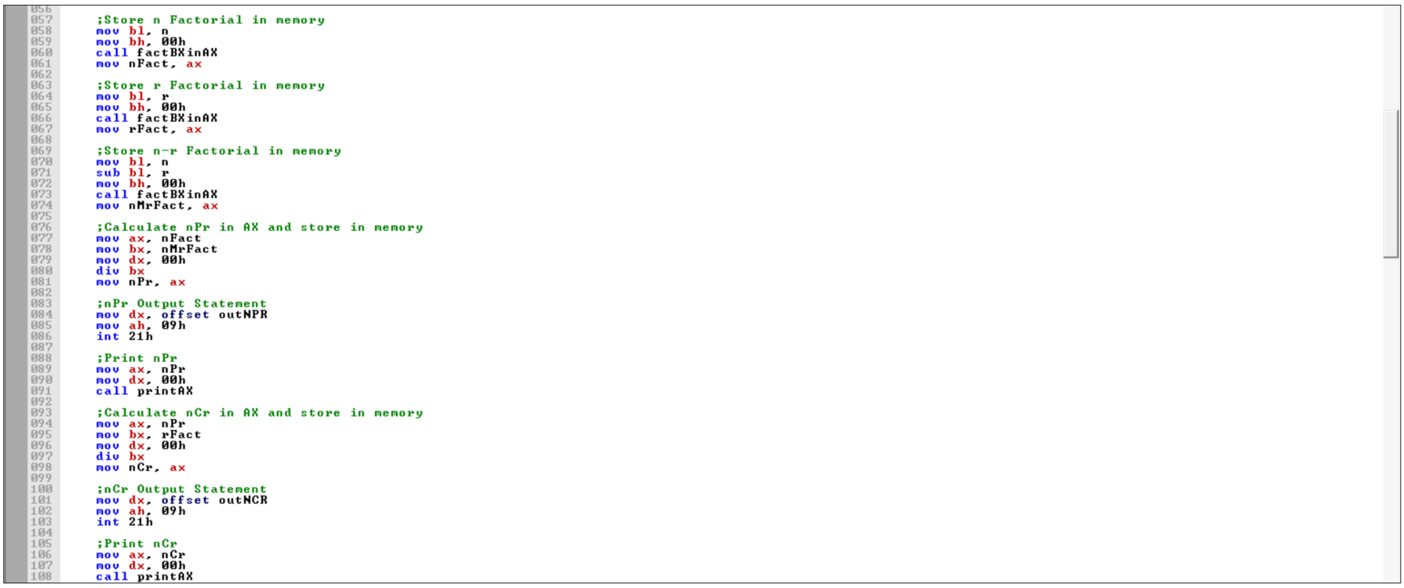


```
Enter the value of n [<=8]: 5
Enter the value of r [<=n]: 3

nPr: 60
nCr: 10
```

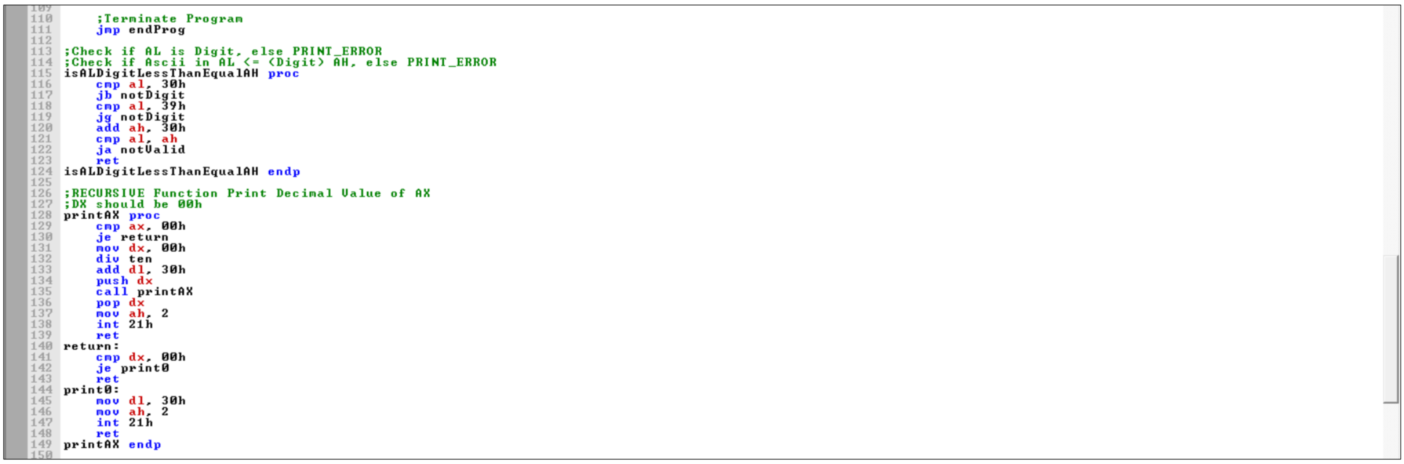
Screenshot of ALP:



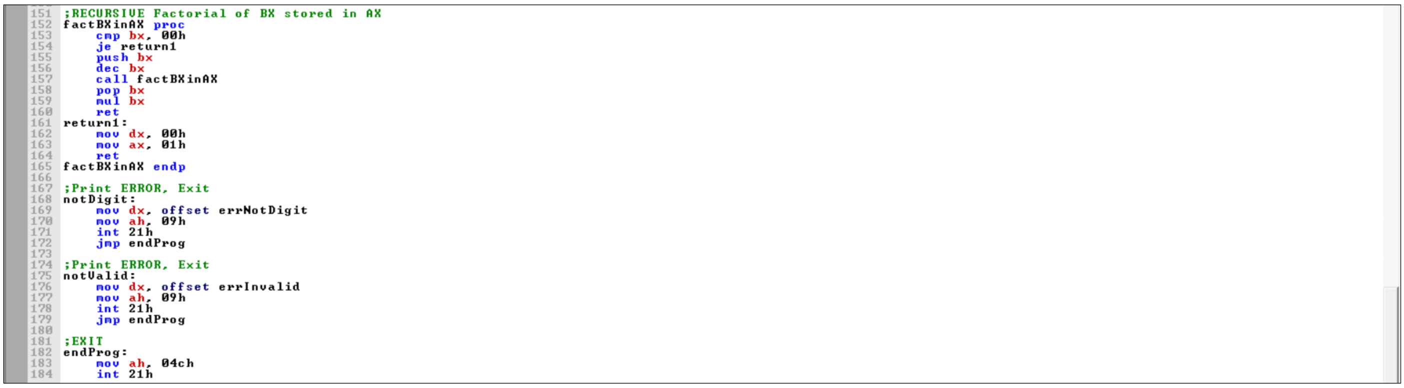
```
.model small
.stack 64
.data
    enterN db 'Enter the value of n [<=8]: $'
    enterR db 0ah,0dh,'Enter the value of r [<=n]: $'
    outNPR db 0ah,0ah,0dh,'nPr: $'
    outNCR db 0ah,0dh,'nCr: $'
    errNotDigit db 0ah,0dh,'The Character is not a Digit$'
    errInvalid db 0ah,0dh,'Invalid Input, Adhere to Constraints$'
    ten dw 10
    n db ?
    r db ?
    nFact dw ?
    rFact dw ?
    nMrFact dw ?
    nPr dw ?
    nCr dw ?
.code
start:
    mov ax, @data
    mov ds, ax

    ;Ask N
    mov dx, offset enterN
    mov ah, 09h
    int 21h

    ;Get N in AL
    mov ah, 1
    int 21h

    ;Check if, ASCII AL (N) <= AH (Digit 8)
    mov ah, 08h
    call isALDigitLessThanEqualAH

    ;Move AL to memory, AL will be used for Interrupts
    sub al, 30h                             ;get value from ascii
    mov n, al

    ;Ask R
    mov dx, offset enterR
    mov ah, 09h
    int 21h

    ;Get R in AL
    mov ah, 1
    int 21h

    ;Check if, ASCII AL (R) <= AH (N)
    mov ah, n
    call isALDigitLessThanEqualAH

    ;Move AL to memory, AL will be used for Interrupts
    sub al, 30h                             ;get value from ascii
    mov r, al
```



```
    ;Store n Factorial in memory
    mov bl, n
    mov bh, 00h
    call factBXinAX
    mov nFact, ax

    ;Store r Factorial in memory
    mov bl, r
    mov bh, 00h
    call factBXinAX
    mov rFact, ax

    ;Store n-r Factorial in memory
    mov bl, n
    sub bl, r
    mov bh, 00h
    call factBXinAX
    mov nMrFact, ax

    ;Calculate nPr in AX and store in memory
    mov ax, nFact
    mov bx, nMrFact
    mov dx, 00h
    div bx
    mov nPr, ax

    ;nPr Output Statement
    mov dx, offset outNPR
    mov ah, 09h
    int 21h

    ;Print nPr
    mov ax, nPr
    mov dx, 00h
    call printAX

    ;Calculate nCr in AX and store in memory
    mov ax, nPr
    mov bx, rFact
    mov dx, 00h
    div bx
    mov nCr, ax

    ;nCr Output Statement
    mov dx, offset outNCR
    mov ah, 09h
    int 21h

    ;Print nCr
    mov ax, nCr
    mov dx, 00h
    call printAX
```

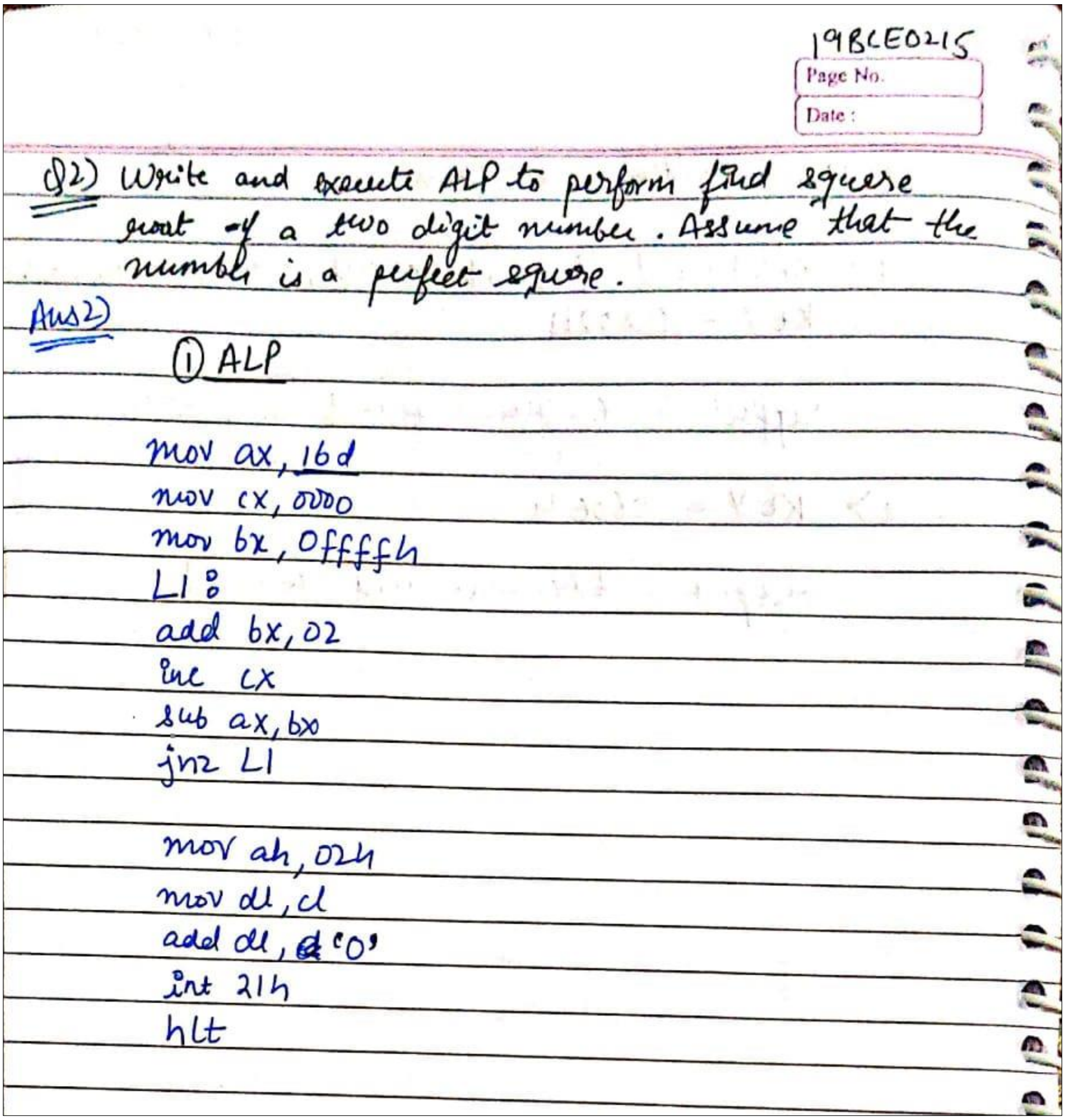


```
    ;Terminate Program
    jmp endProg

;Check if AL is Digit, else PRINT_ERROR
;Check if Ascii in AL <= (Digit) AH, else PRINT_ERROR
isALDigitLessThanEqualAH proc
    cmp al, 30h
    jb notDigit
    cmp al, 39h
    jg notDigit
    add ah, 30h
    cmp al, ah
    ja notValid
    ret
isALDigitLessThanEqualAH endp

;RECURSIVE Function Print Decimal Value of AX
;DX should be 00h
printAX proc
    cmp ax, 00h
    je return
    mov dx, 00h
    div ten
    add dl, 30h
    push dx
    call printAX
    pop dx
    mov ah, 2
    int 21h
    ret
return:
    cmp dx, 00h
    je print0
    ret
print0:
    mov dl, 30h
    mov ah, 2
    int 21h
    ret
printAX endp
```



```
;RECURSIVE Factorial of BX stored in AX
factBXinAX proc
    cmp bx, 00h
    je return1
    push bx
    dec bx
    call factBXinAX
    pop bx
    mul bx
    ret
return1:
    mov dx, 00h
    mov ax, 01h
    ret
factBXinAX endp

;Print ERROR, Exit
notDigit:
    mov dx, offset errNotDigit
    mov ah, 09h
    int 21h
    jmp endProg

;Print ERROR, Exit
notValid:
    mov dx, offset errInvalid
    mov ah, 09h
    int 21h
    jmp endProg

;EXIT
endProg:
    mov ah, 04ch
    int 21h
```

2. Write and execute ALP to perform find square root of a two digit number. Assume that the number is a perfect square.



19BCE0215

Page No.

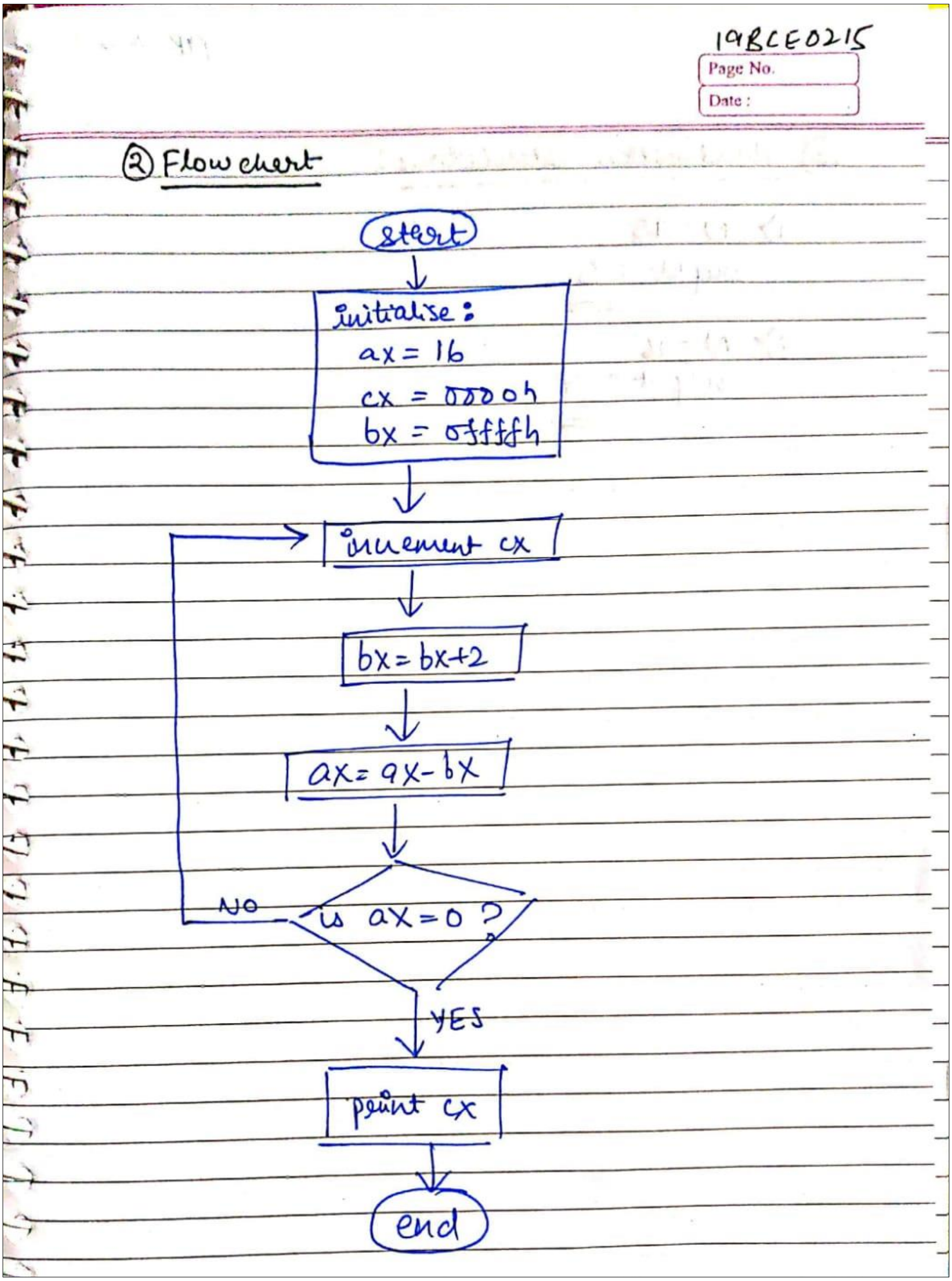
Date :

Q2) Write and execute ALP to perform find squere root of a two digit number. Assume that the number is a perfeet squere.

Ans2)

① ALP

```
mov ax, 16d
mov cx, 0000
mov bx, 0ffffh
L1:
add bx, 02
inc cx
sub ax, bx
jnz L1

mov ah, 02h
mov dl, cl
add dl, '0'
int 21h
hlt
```

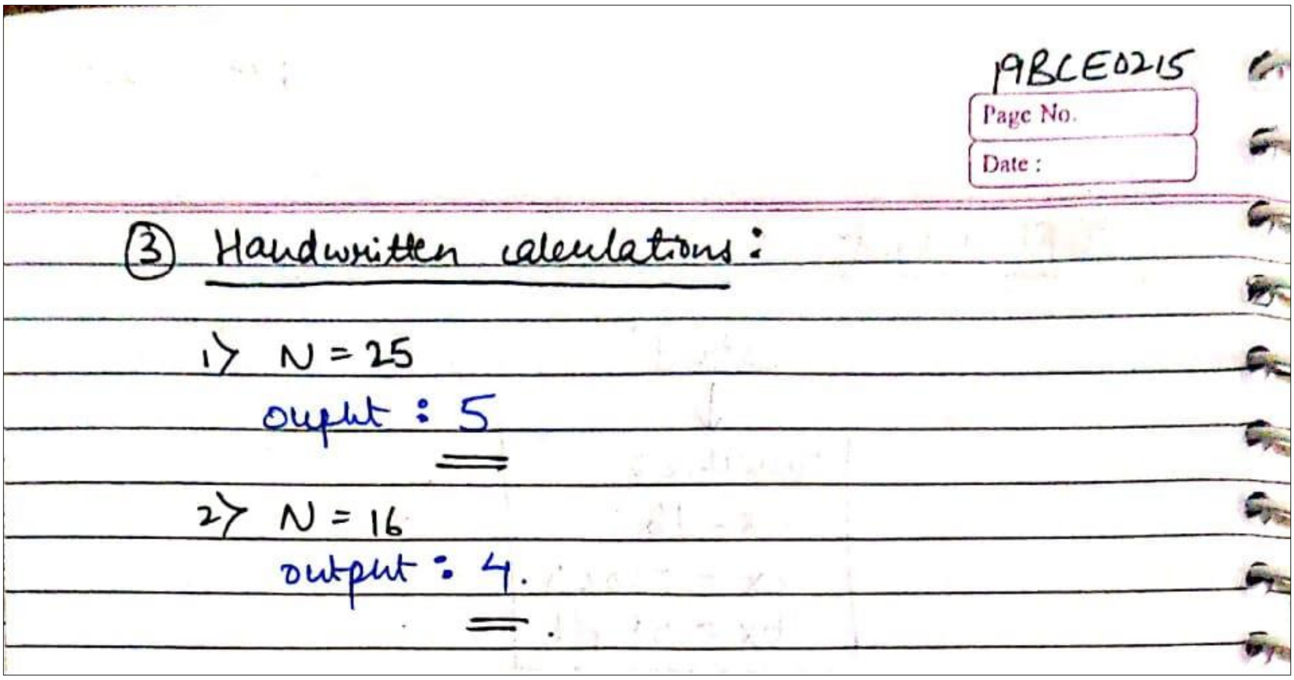


19BCE0215

Page No.

Date :

② Flow chart



19BCE0215

Page No.

Date :

③ Handwritten calculations:

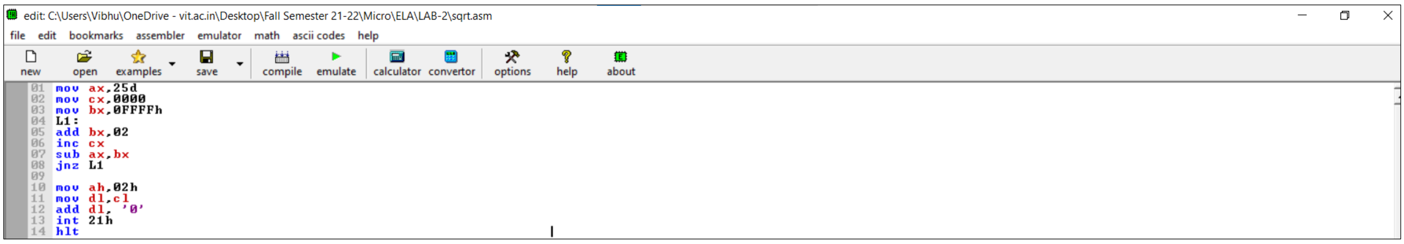
1) N = 25

ouput : 5

2) N = 16

output : 4.

Screenshot of ALP:

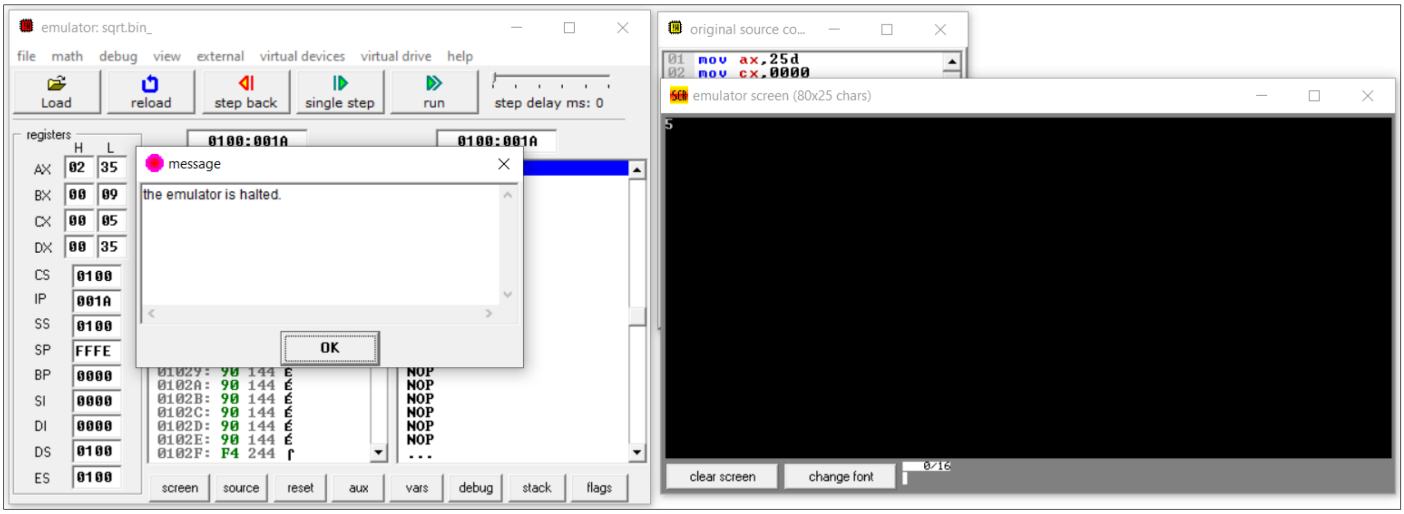


edit: C:\Users\Vibhu\OneDrive - vit.ac.in\Desktop\Fall Semester 21-22\Micro\ELA\LAB-2\sqrt.asm

file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about

```
mov ax,25d
mov cx,0000
mov bx,0FFFFh
L1:
add bx,02
inc cx
sub ax,bx
jnz L1

mov ah,02h
mov dl,cl
add dl, '0'
int 21h
hlt
```

Screenshot of Output:



emulator: sqrt.bin_

file math debug view external virtual devices virtual drive help
Load reload step back single step run
step delay ms: 0
screen source reset aux vars debug stack flags

registers	H	L
AX	02	35
BX	00	09
CX	00	05
DX	00	35
CS	0100	
IP	001A	
SS	0100	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0100	
ES	0100	

```
0100:001A                         0100:001A
01029: 90 144 É                   NOP
0102A: 90 144 É                   NOP
0102B: 90 144 É                   NOP
0102C: 90 144 É                   NOP
0102D: 90 144 É                   NOP
0102E: 90 144 É                   NOP
0102F: F4 244 ſ                   ...
```

message

the emulator is halted.

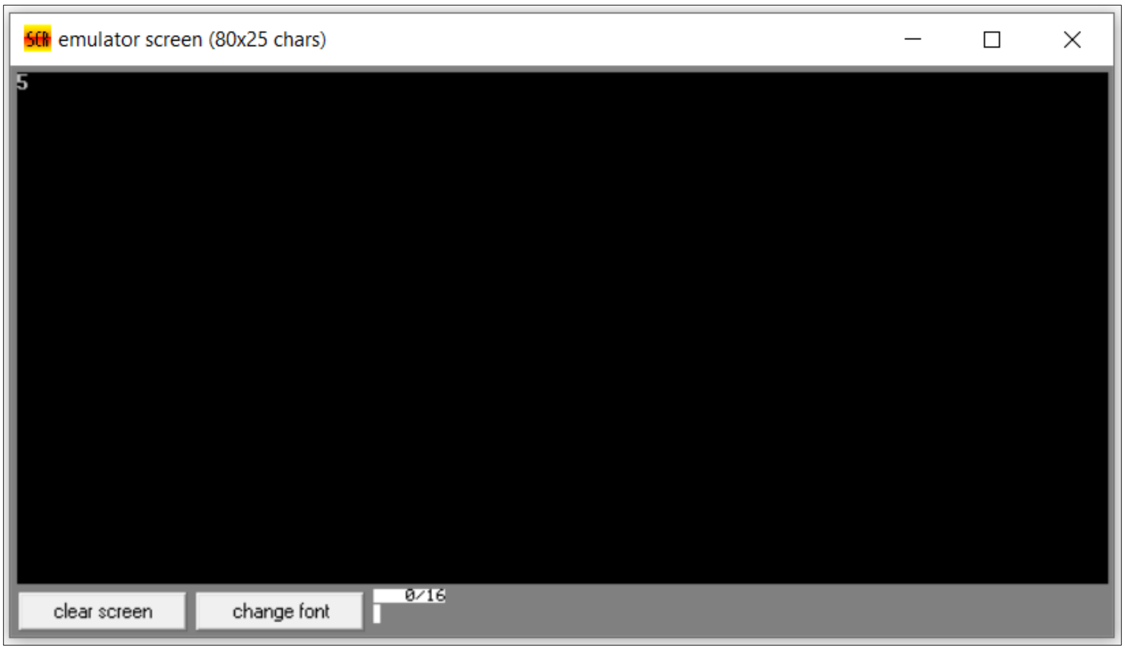
original source co...

```
mov ax,25d
mov cx,0000
```

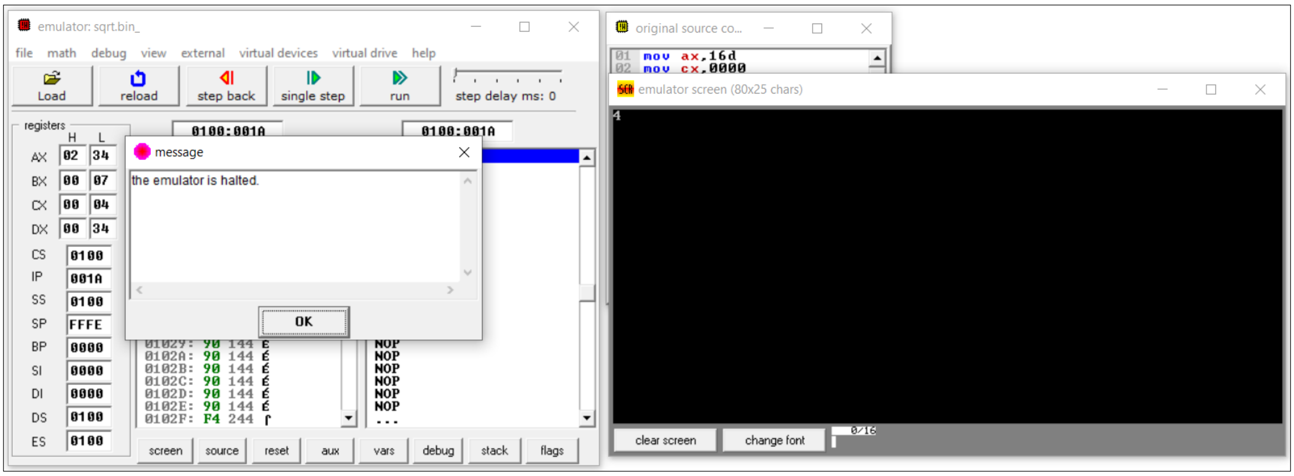
emulator screen (80x25 chars)

```
5
```

clear screen
change font
0/16



```
5
```



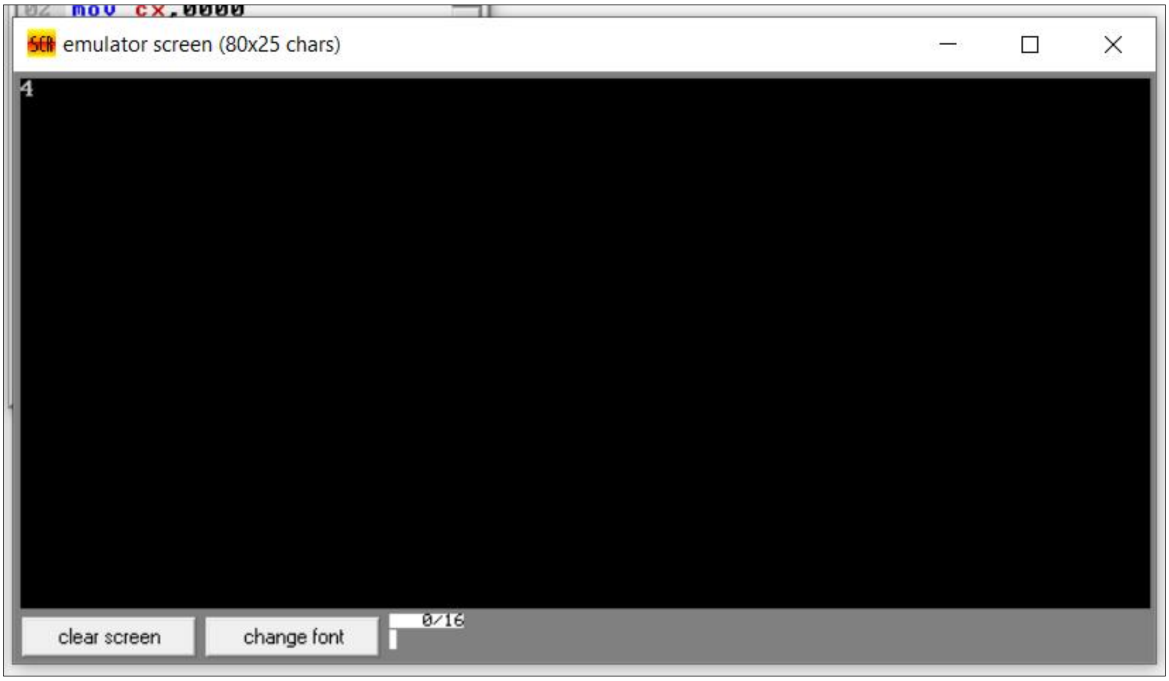
the emulator is halted.

	H	L
AX	02	34
BX	00	07
CX	00	04
DX	00	34
CS	0100	
IP	001A	
SS	0100	
SP	FFFE	
BP	0000	
SI	0000	
DI	0000	
DS	0100	
ES	0100	

```
0100:001A                 0100:001A
01029: 90 144 É           NOP
0102A: 90 144 É           NOP
0102B: 90 144 É           NOP
0102C: 90 144 É           NOP
0102D: 90 144 É           NOP
0102E: 90 144 É           NOP
0102F: F4 244 ſ           ...
```

```
mov ax,16d
mov cx,0000
```

```
4
```



```
mov cx,0000
```

```
4
```