#from pdf, push pop and print using stack

.MODEL SMALL

.STACK 100H

.DATA

.CODE

MAIN PROC

mov ax,122d ; move z to ax, ascii of z is 122

push ax ;push z to stack

mov ax, 49d ; move 1 to ax

push ax

pop bx ; pop 1(top element) to bx

mov dx, bx ; output content of bx

mov ah, 2

int 21h

pop bx ; pop z next top element to bx

mov dx,bx ; print contents of bx

int 21h

endp

end main

#TASK 3: Input a number and check whether all the digits of that number are unique or not

.MODEL SMALL

.STACK 100H

.DATA

A DW 5 DUP(?)

B DB "NOT UNIQUE $"

C DB "UNIQUE $"

.CODE

MAIN PROC

MOV AX, @DATA

MOV DS , AX

MOV CX, 5

MOV SI,0

MOV AH,1

INPUT:

INT 21H

MOV BL,AL

MOV BH,0

PUSH BX

MOV A[SI],BX

INC SI

LOOP INPUT

MOV CX,5

OUTER:

POP BX

MOV SI,0

; MOV AH,2

; MOV DL,"1"

; INT 21H

MOV AH,0

MOV AL,0

INNER:

CMP AH,5

JE BAIRE

MOV DX,A[SI]

MOV DH,0

CMP BX,DX

JE COUNTER

JMP EXIT:

COUNTER:

INC AL

EXIT:

INC SI

INC AH

JMP INNER

BAIRE:

CMP AL,1

JE NORMAL

JMP NOT\_UNIQUE

NORMAL:

LOOP OUTER

UNIQUE:

MOV AH,9

LEA DX,C

INT 21H

JMP DONE

NOT\_UNIQUE:

MOV AH,9

LEA DX,B

INT 21H

DONE:

MAIN ENDP

END MAIN

# TASK 4: Take five elements in an array and print them in reverse order using stack

x db 1,2,3,4,5

ends

stack segment

dw 128 dup(0)

ends

code segment

start:

; set segment registers:

mov ax, data

mov ds, ax

mov es, ax

; add your code here

mov cx, 5

mov si, 0

begin:

mov dl, x[si]

mov dh, 0

push dx

inc si

loop begin

mov cx, 5

mov si, 0

begin2:

pop dx

add dl, 30h

mov dh, 0

mov ah, 2

int 21h

loop begin2

exit:

; exit to operating system.

ends

end start

#TASK 5: Input a number in a register. Print the number in reverse order using stack.

data segment

; add your data here!

pkey db "press any key...$"

ends

stack segment

dw 128 dup(0)

ends

code segment

start:

; set segment registers:

mov ax, data

mov ds, ax

mov es, ax

; add your code here

mov ax, 678

mov bl, 100

div bl

mov ch, ah

mov ah, 0

push ax

mov al, ch

mov bl, 10

div bl

mov dh, ah

mov ah, 0

push ax

mov al, dh

mov ah, 0

push ax

mov cx, 3

mov ah, 2

begin:

pop dx

add dx, 48

int 21h

loop begin

exit:

; exit to operating system.

ends

end start

#TASK 6: Input a word and check whether the word is palindrome or not

data segment

; add your data here!

m db 'm', 'a', 'd', 'a', 'm'

n db "Palindrome$"

o db "Not a Palindrome$"

ends

stack segment

dw 128 dup(0)

ends

code segment

start:

; set segment registers:

mov ax, data

mov ds, ax

mov es, ax

; add your code here

mov cx, 5

mov si, 0

begin:

mov dl, m[si]

mov dh, 0

push dx

inc si

loop begin

mov cx, 5

mov si, 0

begin2:

pop dx

mov dh, 0

cmp m[si], dl

jne disp

inc si

loop begin2

mov ah, 9

lea dx, n

int 21h

jmp exit

disp:

mov ah, 9

lea dx, o

int 21h

exit:

; exit to operating system.

ends

end start

**Hw4 -**

include "emu8086.inc"

code segment

assume cs:code, ds:code, ss:code

mov di, 20 ;input

mov si, 0

L1:

mov cx, di

sub cx, 2

cmp cx, 0

jle fp

mov bl, 2

L2:

mov ax, di

div bl

cmp ah, 0

je skip

inc bl

loop l2

jmp found

skip:

cmp di, 1

jle fp

dec di

jmp L1

found:

push di

jmp skip

;forced push

fp:

mov ax, 2

push ax

exit:

hlt

code ends

End

**Problem 3 unique not unique**

include "emu8086.inc"

code segment

assume cs:code, ds:code, ss:code

;sorting array

mov ch, 5 ;lenght - 1 of the array

L1:

mov cl, 5 ;lenght - 1 of the array

mov si, 0

L2:

mov ah, string[si]

mov al, string[si+1]

cmp ah, al

jg swap

inc si

dec cl

jnz L2

dec ch

jnz L1

jmp next

swap:

mov string[si], al

mov string[si+1], ah

inc si

dec cl

jnz L2

;duplicate check

next:

mov cx, 5 ;lenght - 1 of the array

mov si, 0

xor ah, ah

mov al, string[si]

push ax

xor di, di

dc:

cmp cx, 0

je dup\_nf

inc si

dec cx

xor ah, ah

mov al, string[si]

pop bx

push bx

cmp ax, bx

je found\_dup

push ax

jmp dc

dup\_nf:

cmp di, 1

je exit

print "Unique"

jmp exit

found\_dup:

cmp di, 1

je dc

print "Not Unique"

mov di, 1

jmp dc

exit:

hlt

string db "ASHIKK"

code ends

end