**TASK 05**

.model small

.stack 010h

.data

c1 db 4

c1cpy db 3

bool db 0

index db 6

indcpy db 5

Rows1 db 4

spp db 1

sppcopy db 1

spp1 db 3

sppcopy1 db 3

c2 db 1

c1cpy1 db 1

index1 db 1

indcpy1 db 1

bool1 db 0

Rows2 db 3

.code

mov ax,@data

mov ds,ax

mov cl,Rows1

L6:

mov bool1 ,0

mov bl,c1cpy1

mov c2,bl

mov bl,indcpy1

mov index1,bl

mov bl,sppcopy1

mov spp1 ,bl

start4:

cmp spp1,0

je start5

mov ah,02h

mov dl ,020h

int 21h

dec spp1

jmp start4

start5:

mov ah,02h

mov dl,48

add dl,c2

int 21h

cmp c2 ,1

jne ne1

mov bool1,1

ne1:

cmp bool1 ,1

je add11

dec c2

jmp end1

add11:

inc c2

end1:

dec index1

cmp index1,0

jne start5

mov ah,02h

mov dl,10

int 21h

inc indcpy1

inc indcpy1

inc c1cpy1

dec sppcopy1

loop l6

mov cl,Rows2

start2:

mov bool ,0

mov bl,c1cpy

mov c1,bl

mov bl,indcpy

mov index,bl

mov bl,sppcopy

mov spp ,bl

start3:

cmp spp,0

je start1

mov ah,02h

mov dl ,020h

int 21h

dec spp

jmp start3

start1:

mov ah,02h

mov dl,c1

add dl,'0'

int 21h

cmp c1 ,1

jne nex

mov bool,1

nex:

cmp bool ,1

je add1

dec c1

jmp endloop

add1:

inc c1

endloop:

dec index

cmp index,0

jne start1

mov ah,02h

mov dl,10

int 21h

dec indcpy

dec indcpy

dec c1cpy

inc sppcopy

loop start2

mov ah,4ch

int 21h

end

**TASK 03**

.model small

.stack 100h

.data

str1 db " Enter Number: "

uv dw (0)

creating\_space db (0)

count dw 0

n dw 0

n2 dw 0

multiplier db 1

armstrong db ":---> THE NUMBER ENTERD IS AN ARMSTRONG."

Not\_armstrong db "\*:---> THE NUMBER ENTERD IS NOT AN ARMSTRONG."

.code

jmp main

Arms\_check proc

mov cx,sizeof str1

mov di,0

mov ah,02h

prr1:

mov dl,str1[di]

int 21h

inc di

loop prr1

mov di,offset uv

;Multy Digit Input

Start1:

Mov ah, 01h

Int 21h

cmp al,0dh

je ex

sub al, 48

mov [di], al

inc count

inc di

jmp start1

ex:

mov di, offset count

mov cx ,3

start3:

dec di

cmp [di],al

je start3

mov ax,0

mov al,[di]

mul multiplier

add n2, ax

mov ax,0

mov al, multiplier

mov bl, 0ah

mul bl

mov multiplier,al

dec cx

cmp cx ,0

jne start3

mov di, offset uv

mov cx ,count

back:

push cx

mov cx ,count

mov ax,0

mov bx,0

mov dx,0

mov al, [di]

dec cx

start2:

mov dl, [di]

mul dx

loop start2

add n, ax

inc di

pop cx

loop back

mov bx,n

cmp bx, n2

jne c1

;Printing string

mov cx,sizeof armstrong

mov di,0

mov ah,02h

ass:

mov dl,armstrong[di]

int 21h

inc di

loop ass

jmp c2

c1:

;Printing string

mov cx,sizeof Not\_armstrong

mov di,0

mov ah,02h

as:

mov dl,Not\_armstrong[di]

int 21h

inc di

loop as

c2:

mov ah, 02h

mov dl, 10

int 21h

ret

Arms\_check endp

endl:

main proc

mov ax, @data

mov ds, ax

call Arms\_check

mov ax,0

main endp

mov ah,04ch

int 21h

end