Hala Mallak

202010001

Assembly 2

; This program is designed to perform the following tasks:

; 1. Number Input and Manipulation:

; - Prompts the user to enter a university ID number (up to 10 digits)

; - Displays the entered number on the screen

; - Rearranges the digits of the number in descending order and displays the result

; - This is achieved using a bubble sort algorithm on the array that stores the digits

; 2. Digit Analysis and Cubing:

; - Finds the largest digit in the 10-digit number entered in Task 1

; - This is done by iterating through the array of digits and keeping track of the largest one

; - Calculates the cube of the largest digit and displays the result

; - The cube operation is performed by multiplying the largest digit by itself twice

; The program uses various assembly language instructions and procedures to accomplish these tasks, including:

; - Input/output operations (e.g., displaying prompts, reading user input)

; - Array manipulation (e.g., storing digits, sorting in descending order)

; - Arithmetic operations (e.g., finding the largest digit, calculating the cube)

; Finally, it displays the author's name.

.model small

.stack 100h

.data

prompt db 'Please enter your university ID [be attention at most 10 digits]: $' ; Prompt message

input db 11 dup(0) ; Buffer for user input, including space for null terminator

output db 'Your ID is: $' ; Output message

sortedID db 'This is your ID sorted in descending order: $' ; Message for sorted ID

array db 10 dup(0) ; Array to store the digits of the ID

largest db 'The largest digit is: $' ; Message for largest digit

largest_digit db 0 ; Variable to store the largest digit

cube_msg db 'The cube of the largest digit is: $' ; Message for cube of the largest digit

cube_result dw 0 ; Variable to store the cube result

newline db 0Dh, 0Ah, '$' ; Newline characters

Owner db 'Created by [Hala Mallak]$', 0 ; Author's name

```
intro db 'Assignment 2: Number Manipulation and Digit Analysis in Assembly Language$', 0 ; Intro message

.code

main proc

; Initialize data segment

mov ax, @data

mov ds, ax

; Display intro message

lea dx, intro

mov ah, 09h

int 21h

; Display newline

lea dx, newline

mov ah, 09h

int 21h

; Display prompt message

lea dx, prompt

mov ah, 09h

int 21h

; Read up to 10 digits from user input

mov ah, 01h

lea si, input
```

```
lea di, array
mov cx, 0
read_char:
int 21h            ; Read a character
cmp al, 13         ; Check for Enter key (carriage return)
je process_input    ; If Enter key, jump to process_input
mov [si], al       ; Store character in input buffer
sub al, '0'        ; Convert ASCII to integer
mov [di], al       ; Store integer in array
inc si             ; Increment input buffer pointer
inc di             ; Increment array pointer
inc cx             ; Increment digit count
cmp cx, 10         ; Check if 10 digits are entered
jb read_char       ; If not, continue reading
process_input:
mov byte ptr [si], '$' ; Null-terminate the input string

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

```
; Display output message with entered ID
lea dx, output
mov ah, 09h
int 21h
lea dx, input
mov ah, 09h
int 21h
; Sort digits in descending order using bubble sort
mov bx, cx
dec bx
outer_loop:
mov si, 0
mov di, 1
mov cx, bx
inner_loop:
mov al, [array + si]
cmp al, [array + di]
jge skip_swap
xchg al, [array + di] ; Swap if the next digit is larger
mov [array + si], al
skip_swap:
inc si
```

```
inc di
loop inner_loop
dec bx
jnz outer_loop
; Display newline
lea dx, newline
mov ah, 09h
int 21h
; Display sorted ID message
lea dx, sortedID
mov ah, 09h
int 21h
; Display sorted digits
lea si, array
mov cx, 10
display_loop:
mov dl, [si]
add dl, '0'         ; Convert integer to ASCII
mov ah, 02h
int 21h
inc si
loop display_loop
```

```
; Display newline
lea dx, newline
mov ah, 09h
int 21h
; Find the largest digit
mov al, [array]      ; Initialize largest_digit to the first digit
mov cx, 9            ; Loop 9 times (since we already checked the first digit)
mov si, 1            ; Start from the second digit
find_largest:
   cmp al, [array + si]
   jae skip_update ; Jump if current largest is greater than or equal to the current digit
   mov al, [array + si] ; Update largest_digit
skip_update:
   inc si
   loop find_largest
mov largest_digit, al

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

```
int 21h



; Display largest digit message
lea dx, largest
mov ah, 09h
int 21h

; Display the largest digit
mov al, largest_digit
add al, '0' ; Convert digit to ASCII
mov ah, 02h
mov dl, al ; Move the largest digit to DL for printing
int 21h

; Display newline
lea dx, newline
mov ah, 09h
int 21h
; Calculate the cube of the largest digit
mov al, largest_digit
mov ah, 0
```

```
mul al              ; Square the digit
mov bx, ax
mul largest_digit   ; Multiply the result by the digit again to get the cube
mov cube_result, ax
; Display cube message
lea dx, cube_msg
mov ah, 09h
int 21h
; Display the cube result
mov ax, cube_result
call print_number
; Display newline
lea dx, newline
mov ah, 09h
int 21h
; Display author's name
lea dx, Owner
mov ah, 09h
int 21h
; End the program
mov ah, 4Ch
```

```
int 21h
ret
main endp
; Procedure to print a number stored in AX
print_number proc
push ax
push bx
push cx
push dx
mov cx, 10
mov bx, 0
convert_loop:
xor dx, dx
div cx              ; AX = AX / 10, DX = AX % 10
push dx             ; Push remainder on stack
inc bx
cmp ax, 0
jne convert_loop
print_digits:
pop dx
add dl, '0'         ; Convert integer to ASCII
mov ah, 02h
```

```
int 21h
dec bx
jnz print_digits
pop dx
pop cx
pop bx
pop ax
ret
print_number endp
end main
```