Experiment No:-

21-08-24

Bubble Sort

Aim:- Implement Bubble sort in MASM programming

Algorithm

1. START
2. Define data segment
3. Intialize prompt1 "Enter the no of elements:$" prompt 2 "Enter the numbers:$" result "Sorted array is:$"
4. Define n, array 9 bytes, Data segments ends
5. Define macro display with parameter msg
6. Used to display msg
7. Define macro read Digit
8. Used to read data from user and convert ASCII value to numeric value.
9. Define macro printDigit
10. Convert numeric value to ASCII and print data.
11. Define code segment
12. Assume cs:code ds:data
13. Intialize label start
14. Point ds register to beginning of data segment
15. readDigit read value of n transfer to cl
16. Display prompt 2
17. Set si with offset of array
18. Intialize label read
19. Read the array of numbers.
20. Intialize label loop1:
21. Represent outer loop for bubble sort
22. Intialize label loop2
23. Represent inner loop for bubble sort

24. Intialize label swap
25. Used to swap values of two index of a array
26. Intialize swapped
27. To check whether loop is over or not
28. If not over jump to corresponding loops
29. Else Display the array jump to print.
30. Intialize label print
31. To display the array one by one
32. Terminate the program
33. Code segment ends
34. Stop

<u>Result</u>: MASM program to implement Bubble sort executed successfully and output verified.

28/10/24

PROGRAM

```
data segment
m1 db 0ah,0dh,"enter the no of elements:$"
m2 db 0ah,0dh,"enter the number :$"
result db 0ah,0dh,"sorted array is :$"
n db 09h dup(?)
array db 09h dup(?)
data ends
display macro msg
lea dx,msg
mov ah,09h
int 21h
endm
readDigit macro
mov ah,01h
int 21h
sub al,30h
endm
printDigit macro
add dl,30h
mov ah,02h
int 21h
endm
code segment
assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
display m1
readDigit
mov n,al
mov cl,n
display m2
mov si,offset array
read:
readDigit
mov [si],al
inc si
dec cl
jnz read
mov cl,n
loop1:
```

```
mov ch,n
mov si,offset array
loop2:
mov dl,[si]
cmp dl,[si+1]
jnc swap
jmp swapped
swap:
mov dl,[si]
xchg dl,[si+1]
mov [si],dl
swapped:
inc si
dec ch
jnz loop2
dec cl
jnz loop1
display result
mov si,offset array
mov cl,n
inc si
print:
mov dl,[si]
printDigit
inc si
dec cl
jnz print
mov ah,4ch
int 21h
code ends
end start
```

OUTPUT:

```
enter the no of elements: 6
enter the number: 773874
sorted array is: 3477778
```

✓

Experiment No:-

23-8-24

Reverse of a String

Aim:- Implement a string reverser in MASM programming

Algorithm

1. START
2. Define data segment
3. Intialize msg1 "Enter the string: $" msg 2 "The reverse:$"
4. Define str1 20 byte rev 20 byte
5. End of data segment
6. Define macro display with parameter msg
7. Used to display msg
8. Define code segment
9. assume cs: code, ds: data
10. Intialize label start
11. Point ds register to beginning of data segment
12. clear cx with 0
13. load effective address of str1 to si
14. Intialize label loop1
15. read the character
16. Intialize label loop2
17. end str1 with "$" symbol
18. move value of si to di
19. load effective address of rev to si
20. Intialize label loop3
21. move value of [di] to bl
22. move value of bl to [si]
23. increment si decrement di
24. display msg 2
25. display rev

26. Terminate the program
27. Code segment ends
28. Stop

<u>Result</u> :- MASM program to implement reverse of a string executed successfully and output obtained.

28/10/24

PROGRAM

```
data segment
msg1 db 0ah,0dh,"Enter the string:$"
msg2 db 0ah,0dh,"the reverse is:$"
str1 db 20 dup(?)
rev db 20 dup(?)
data ends
display macro msg
mov ah,09h
lea dx,msg
int 21h
endm
code segment
assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
display msg1
mov cx,0000h
lea si,str1
loop1:
mov ah,01h
int 21h
mov [si],al
cmp al,0dh
je loop2
inc si
inc cx
jmp loop1
loop2:
mov bl, "$"
mov [si],bl
mov di,si
lea si,rev
dec di
loop3:
mov bl,[di]
mov [si],bl
dec di
inc si
dec cx
jnz loop3
```

```
mov bl,"$"
mov [si],bl
display msg2
display rev
mov ah,4ch

int 21h
code ends
end start
```

OUTPUT:

```
Enter the key: key
the reverse: yek
```

23-8-24

Experiment No:-
Palindrome or Not

Aim: Implement a palindrome checker in MASM programming.

Algorithm

1. START
2. Define data segment
3. Intialize msg1 "Enter the string $" msg2 "The string is Palindrome: $", msg3 "The string is not palindrome: $"
4. Define str1 50 bytes
5. End of data segment
6. Define macro display with parameter msg
7. Used to display msg
8. Define code segment
9. Assume cs:code ds:data
10. Intialize label start
11. Point ds register to beginning of data segment
12. Display msg1, clear cx with 0000h
13. load effective address of str1 to si
14. Intialize label loop1
15. Read the characters if character ends jump to loop 2
16. Intialize label loop2
17. Decrement si and load effective address of str1 to di
18. Intialize label loop3
19. compare the str1
20. from start to end and end to start
21. If comparison between character not same.

22. Jump to loop4

23. else jump to loop5 display msg2

24. Intialize loop4 display msg3

25. Intialize loop 5

26. Terminate program

27. code segment ends

28. Stop

Result :- MASM program to implement palindrome of a string executed successfully and output obtained.

28/10/24

PROGRAM

```
data segment
msg1 db 0ah,0dh,"enter the string:$"
msg2 db 0ah,0dh,"the string is palindrome:$"
msg3 db 0ah,0dh,"the string is not palindrome:$"
str1 db 50 dup(?)
data ends
display macro msg
mov ah,09h
lea dx,msg
int 21h
endm
code segment
assume cs:code,ds:data
start:
mov ax,data
mov ds,ax
display msg1
mov cx,0000h
lea si,str1
loop1:
mov ah,01h
int 21h
mov [si],al
cmp al,0dh
je loop2
inc si
inc cx
jmp loop1
loop2:
dec si
lea di,str1
loop3:
mov bl,[di]
mov al,[si]
cmp al,bl
jnz loop4
inc di
dec si
dec cx
jnz loop3
display msg2
```

```
jmp loop5
loop4:
display msg3
loop5:
mov ah,4ch

int 21h
code ends
end start
```

OUTPUT:

```
enter the string: ardra
the string is palindrome
enter the string: anu
the string is not palindrome
```

✓