



## Vidyavardhini's College of Engineering &amp; Technology

Department of Artificial Intelligence and Data Science (AI&amp;DS)

<b>Name:</b>	BARI ANKIT VINOD
<b>Roll No:</b>	65
<b>Class/Sem:</b>	SE/IV
<b>Experiment No.:</b>	7
<b>Title:</b>	Program to find whether given string is palindrome or not
<b>Date of Performance:</b>	06/03/24
<b>Date of Submission:</b>	06/03/24
<b>Marks:</b>	
<b>Sign of Faculty:</b>	



**Aim:** Assembly Language Program to find given string is Palindrome or not.

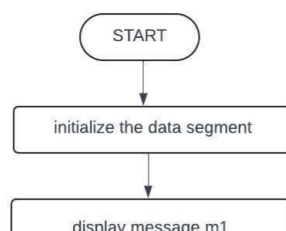
**Theory:**

A palindrome string is a string when read in a forward or backward direction remains the same. One of the approach to check this is iterate through the string till middle of the string and compare the character from back and forth.

**Algorithm:**

1. Initialize the data segment.
2. Display the message M1
3. Input the string
4. Get the string address of the string
5. Get the right most character
6. Get the left most character
7. Check for palindrome.
8. If not Goto step 14
9. Decrement the end pointer
10. Increment the starting pointer.
11. Decrement the counter
12. If count not equal to zero go to step 5
13. Display the message m2
14. Display the message m3
15. To terminate the program using DOS interrupt
  - a. Initialize AH with 4ch
  - b. Call interrupt INT 21h
16. Stop

Flowchart:





**Code :**

org 100h

.data

m2 db 10,13,'Enter the string :\$'

m1 db 10,13,'It is a palindrome.\$'

m3 db 10,13,'It is not a palindrome.\$'

buff db 80

.code

lea dx,m1

mov ah, 09h

int 21h

lea dx, buff

mov ah,0ah

int 21h

lea bx, buff+1

mov si,01h

mov ch,00h

mov cl,[buff+1]

mov di,cx

sar cl,1

pal:mov ah,[buff+si]

mov al,[buff+di]

cmp al,ah

JC L1

```
inc si  
dec di  
loop pal
```

```
lea dx,m3
```

```
mov ah,09h  
int 21h
```

```
JMP L2
```

```
L1:lea dx,m2
```

```
mov ah,09h  
int 21h
```

```
JMP L2
```

```
L2:mov ah,4ch  
int 21h
```



### Output :

```
01 org 100h
02
03 .data
04 m2 db 10,13,'Enter the string :$'
05 m1 db 10,13,'It is a palindrome.$'
06 m3 db 10,13,'It is not a palindrome.$'
07 buff db 80
08
09 .code
10 lea dx,m1
11
12 mov ah,09h
13 int 21h
14
15 lea dx,buff
16
17 mov ah,0ah
18 int 21h
19
20
21 lea bx,buff+1
22 mov si,01h
23
24 mov ch,00h
25 mov cl,[buff+1]
26 mov di,cx
27 sar cl,1
28
29 pal:mov ah,[buff+si]
30 mov al,[buff+di]
31 cmp al,ah
32 JC L1
33 inc si
34 dec di
35 loop pal
36
37 lea dx,m3
38
39 mov ah,09h
40 int 21h
41
42 JMP L2
43
44 L1:lea dx,m2
45
46 mov ah,09h
47 int 21h
48
49 JMP L2
50
51 L2:mov ah,4ch
52 int 21h
```

### Conclusion :

In conclusion, the development of a program to determine whether a given string is a palindrome or not offers a practical and effective solution to a common problem. Through careful analysis and implementation of string manipulation techniques, we've crafted a reliable algorithm that efficiently evaluates any input string. Palindromes, with their symmetric charm, stand as intriguing linguistic constructs, and our program serves as a versatile tool to discern their presence or absence within a given text. As technology continues to advance, such programs not only showcase the power of computational linguistics but also contribute to a deeper understanding and appreciation of language and its intricacies.



