**## Pseudocode that determines whether given word is palindrome or not**

Step 1: BEGIN PROCDURE Palindrome

Step 2: DECLARE word

Step 3: DISPLAY ‘Enter a word’

Step 4: DECLARE word\_reversed[]

Step 5: char[ ] word\_reversed is equal to word.toCharArray()

Step 6: Declare new\_word

Step 7: for i is equal to reversed\_word.length-1 and greater than or equal to 0

new\_word is equal to new\_word+word\_reversed[i]

Step 8: if word is equal to new\_word

Display ‘The entered word is palindrome’

Display ‘new\_word’

Step 9: else

Display ‘The word is not palindrome’

Display ‘new\_word’

End if

Step 10: END

**Time Complexity Using Big O notation**

Time complexity of any program is defined as the time required for its execution. Big O notation is used for the time complexity analysis of a program whenever it requires maximum time for execution.

f(n)<=c.g(n)

n>=n0 =>O(g(n))

where c>0,n0>=1

n should be greater than or equal to n0

n0 always greater than or equal to 1

worst case complexity O(n)

worst case: maximum time required for program execution

Input Size

n0

f(n)

c.g(n)-> upper bound

Time