**Algorithm:**

Step 1 : Start

Step 2 : read s

Step 3 : call the function palindrome(s)

3.1 : check if s == s[::-1]

3.1.1 : if yes, then display ‘palindrome’

3.1.2 : if no, then display ‘not palindrome’

Step 4 : Stop

**Program:**

# program to check string is palindrome or not

def check\_palindrome(s):

if s==s[::-1]:

print(s,'is palindrome')

else:

print(s,'is not palindrome')

str1=input('Enter a string : ')

check\_palindrome(str1)

**Output:**

Enter a string : son

son is not palindrome

**Algorithm:**

Step 1 : Start

Step 2 : read s

Step 3 : Assign rev=call the function reverse(str1)

3.1 : return s[::-1]

Step 4 : display rev

Step 5 : Stop

**Program:**

# program to reverse the string

def reverse(s):

return s[::-1]

str1=input('Enter a string : ')

rev=reverse(str1)

print('The reversed string is',rev)

**Output:**

Enter a string : alex

The reversed string is xela

**Algorithm:**

Step 1 : Start

Step 2 : read s

Step 3 : call the function str\_len(s)

3.1 : assign l=0

3.2 : for i in s and go to step 3.2.1

3.2.1 : Update l+=1 and go to step 3.2

3.3 : display l

Step 4 : Stop

**Program:**

# program to find the length of the string

def str\_len(s):

l=0

for i in s:

l+=1

return l

str1=input('Enter a string : ')

print(str\_len(str1))

**Output:**

Enter a string : pandian

7

**Algorithm:**

Step 1 : Start

Step 2 : read s

Step 3 : call the function replace(s)

3.1 : Assign a=’’

3.2 : for i in s and go to step 3.2.1

3.2.1 : check if i in ['a','e','i','o','u','A','E','I','O','U']

3.2.1 : if yes then update a+=’@’

3.2.2 : if no then update a+=i

3.3 : display a

Step 4 : Stop

**Program:**

# replacing the vowel by @

def replace(s):

a=’’

for i in s:

if i in ['a','e','i','o','u','A','E','I','O','U']:

a+=’@’

else:

a+=i

return a

str1=input('Enter a string : ')

print(replace(str1))

**Output:**

Enter a string : alex

@l@x