list_manual_inbuilt_fun & method

List-related functions without using inbuilt functions

```
def list_length(lst):
    count = 0
    for _ in lst:
        count += 1
    return count
```

```
def list_sum(lst):
    total = 0
    for num in lst:
        total += num
    return total
```

```
def list_max(lst):
    max_val = lst[0]
    for num in lst:
        if num > max_val:
            max_val = num
    return max_val
```

```
def list_min(lst):
    min_val = lst[0]
    for num in lst:
        if num < min_val:
            min_val = num
    return min_val
```

```
def count_occurrences(lst, target):
    count = 0
    for item in lst:
        if item == target:
            count += 1
    return count
```

```
def reverse_list(lst):
    reversed_lst = []
    for i in range(list_length(lst) - 1, -1, -1):
        reversed_lst.append(lst[i])
    return reversed_lst
```

```
def is_palindrome(lst):
    n = list_length(lst)
    for i in range(n // 2):
        if lst[i] != lst[n - i - 1]:
            return False
    return True
```

```
def remove_duplicates(lst):
    unique = []
    for item in lst:
        found = False
        for u in unique:
            if u == item:
                found = True
                break
        if not found:
            unique.append(item)
    return unique
```

```
def sort_list(lst):
    n = list_length(lst)
    for i in range(n):
        for j in range(i + 1, n):
            if lst[i] > lst[j]:
                lst[i], lst[j] = lst[j], lst[i]
    return lst
```

```
def merge_lists(lst1, lst2):
    result = []
    for item in lst1:
        result.append(item)
    for item in lst2:
        result.append(item)
    return result
```

```
def second_max(lst):
    max1 = lst[0]
    for num in lst:
        if num > max1:
            max1 = num
    max2 = None
    for num in lst:
        if num != max1:
            if max2 is None or num > max2:
```

```
                max2 = num
    return max2
```

```
def common_elements(lst1, lst2):
    common = []
    for i in lst1:
        for j in lst2:
            if i == j:
                already_added = False
                for k in common:
                    if k == i:
                        already_added = True
                if not already_added:
                    common.append(i)
    return common
```

```
def rotate_left(lst, k=2):
    n = list_length(lst)
    rotated = []
    for i in range(k, n):
        rotated.append(lst[i])
    for i in range(k):
        rotated.append(lst[i])
    return rotated
```

```
def split_even_odd(lst):
    even = []
    odd = []
    for num in lst:
        if num % 2 == 0:
            even.append(num)
        else:
            odd.append(num)
    return even, odd
```

```
def are_equal(lst1, lst2):
    if list_length(lst1) != list_length(lst2):
        return False
    for i in range(list_length(lst1)):
        if lst1[i] != lst2[i]:
            return False
    return True
```

--- Additional List Examples Without Inbuilt Functions ---

```
def list_to_string(lst):
    result = ''
    for ch in lst:
        result += ch
    return result
```

```
def string_to_list(s):
    result = []
    for ch in s:
        result.append(ch)
    return result
```

```
def remove_duplicate_characters(s):
    result = ''
    for ch in s:
        found = False
        for r in result:
            if ch == r:
                found = True
                break
        if not found:
            result += ch
    return result
```

```
def alternate_case(s):
    result = ''
    upper = True
    for ch in s:
        if 'a' <= ch <= 'z' or 'A' <= ch <= 'Z':
            if upper:
                if 'a' <= ch <= 'z':
                    result += chr(ord(ch) - 32)
                else:
                    result += ch
                upper = False
            else:
                if 'A' <= ch <= 'Z':
                    result += chr(ord(ch) + 32)
                else:
                    result += ch
                upper = True
        else:
            result += ch
    return result
```

```python
def string_length(s):
    count = 0
    for _ in s:
        count += 1
    return count
```

```python
def reverse_each_word(s):
    result = ''
    word = ''
    for ch in s + ' ':
        if ch != ' ':
            word += ch
        else:
            for i in range(string_length(word)-1, -1, -1):
                result += word[i]
            result += ' '
            word = ''
    return result.strip()
```