```python
#1. Write a Python program to sum all the items in a list.

def sum_list(items):
  """Sums all the items in a list.

  Args:
    items: A list of numbers.

  Returns:
    The sum of all the items in the list.
  """
  sum_numbers = 0
  for x in items:
    sum_numbers += x
  return sum_numbers

# Get input from the user
numbers_str = input("Enter a list of numbers, separated by spaces: ")

# Convert the input string to a list of numbers
numbers = [int(x) for x in numbers_str.split()]

# Calculate the sum of the numbers
total_sum = sum_list(numbers)

# Print the sum
print("The sum of the numbers is:", total_sum)
```

```
Enter a list of numbers, separated by spaces: 2 4 3 1 5 6 7 5 8 7
The sum of the numbers is: 48
```

```python
#2. Write a Python program to get the largest and smallest number from
a list without builtin functions


def find_min_max(numbers):
  """Finds the minimum and maximum numbers in a list.

  Args:
    numbers: A list of numbers.

  Returns:
    A tuple containing the minimum and maximum numbers.
  """

  # Initialize minimum and maximum to the first element of the list
  minimum = numbers[0]
  maximum = numbers[0]
  # Iterate through the list, updating minimum and maximum as needed
  for number in numbers:
    if number < minimum:
      minimum = number
    if number > maximum:
      maximum = number

  return minimum, maximum

# Example usage
numbers = [1, 2, 3, 4, 5]
minimum, maximum = find_min_max(numbers)
print("Minimum:", minimum)
print("Maximum:", maximum)
```

```
Minimum: 1
Maximum: 5
```

```python
#3. Write a Python program to find duplicate values from a list and
display those.

def find_duplicates(items):
    """
    Finds duplicate values in a list.

    Args:
        items: A list of items.

    Returns:
        A list of duplicate items.
    """

    duplicates = []
    seen = set()

    for item in items:
        if item in seen:
            if item not in duplicates:  # Check to avoid adding the
same duplicate multiple times
                duplicates.append(item)
        else:
            seen.add(item)

    return duplicates

# Get user input
while True:
    try:
        input_str = input("Enter a list of items separated by spaces:
")
        items = input_str.split()
        # Attempt to convert items to numbers if possible
        try:
            items = [int(x) for x in items]
        except ValueError:
            # If conversion to numbers fails, keep items as strings
            pass

        break  # Exit loop if input is valid
    except ValueError:
        print("Invalid input. Please enter items separated by
spaces.")

# Find and print duplicates
duplicate_items = find_duplicates(items)
print("Duplicate items:", duplicate_items)
```

```
Enter a list of items separated by spaces: 1 2 2 3 4 4 5
Duplicate items: [2, 4]
```

```
# 4. Write a Python program to split a given list into two parts where
the length of the first part of the list is given. Original list: [1,
1, 2, 3, 4, 4, 5, 1] Length of the first part of the list: 3 Splitted
the said list into two parts: ([1, 1, 2], [3, 4, 4, 5, 1])

def split_list(original_list, length_of_first_part):
  """Splits a list into two parts.

  Args:
    original_list: The original list to split.
    length_of_first_part: The length of the first part of the list.

  Returns:
    A tuple containing the two parts of the list.
  """
  first_part = original_list[:length_of_first_part]
  second_part = original_list[length_of_first_part:]
  return first_part, second_part

# Example usage
original_list = [1, 1, 2, 3, 4, 4, 5, 1]
length_of_first_part = 3
first_part, second_part = split_list(original_list,
length_of_first_part)
print("Splitted the said list into two parts:", (first_part,
second_part))
```

```
Splitted the said list into two parts: ([1, 1, 2], [3, 4, 4, 5, 1])
```

```python
#5. Write a Python program to traverse a given list in reverse order,
and print the elements with the original index. Original list: ['red',
'green', 'white', 'black'] Traverse the said list in reverse order:
black white green red

def traverse_reverse(colors):
  """Traverses a list in reverse order and prints elements with their
original indices.

  Args:
    colors: The list to traverse.
  """

  for i in range(len(colors) - 1, -1, -1):
    print(colors[i], end=" ")
  print()  # Add a newline for better formatting


# Example usage
colors = ['red', 'green', 'white', 'black']
print("Traverse the said list in reverse order: ", end="")
traverse_reverse(colors)
```

```
Traverse the said list in reverse order: black white green red
```