Python Basic Exercise #8

Total Question: **4 Question**
Topic: **Problem Solving**

1. Number Validation

Create a Function for Integer Input Validation:

- Create a function named `get_valid_integer` that prompts the user to enter an integer.
- The function should validate that the input is an integer and within a specified range (e.g., 1 to 100).
- If the input is invalid, the function should print an error message and prompt the user to enter a valid integer again.

```
integer_prompt = "Enter an integer between 1 and 100: "
valid_integer = get_valid_integer(integer_prompt, 1, 100)

# output
Enter an integer between 1 and 100: 150
Error: Input must be between 1 and 100.
Enter an integer between 1 and 100: abc
Error: Invalid input. Please enter an integer.
Enter an integer between 1 and 100: 50
```

2. Create a Function with Variable-Length Keyword Arguments

Create a function named `print_student_info` that takes a student's name and variable-length keyword arguments.
The function should print the student's name and additional information (like age, grade, and city) passed as keyword arguments.

```
# Printing student information
print_student_info("Charlie", age=15, grade="10th", city="New York")

#output
Student Name: Charlie
Age: 15
Grade: 10th
City: New York
```

3. Create a Function to Sort Data with Custom Key

Create a function named `sort_strings_by_length` that takes a list of strings as an argument.
The function should return a new list with the strings sorted by their length in ascending order.

```
strings = ["apple", "banana", "orange", "kiwi", "grape"]
sorted_strings = sort_strings_by_length(strings)
print(f"Strings sorted by length: {sorted_strings}")

# output
Strings sorted by length: ['kiwi', 'apple', 'grape', 'banana', 'orange']
```

4. Create a Function to Transform Data with Conditional Logic

Create a function named `classify_numbers` that takes a list of integers as an argument.
The function should return a dictionary with keys "even" and "odd", where each key contains a list of even or odd numbers, respectively.

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
classified_numbers = classify_numbers(numbers)
print(f"Classified numbers: {classified_numbers}")

#output
Classified numbers: {'even': [2, 4, 6, 8, 10], 'odd': [1, 3, 5, 7, 9]}
```

5. Create a Function to Aggregate Data with Custom Operation

Create a function named calculate_product that takes a list of integers as an argument.
The function should return the product of all the numbers in the list.

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
product = calculate_product(numbers)
print(f"Product of numbers: {product}")

#output
Product of numbers: 3628800
```

6. Create a Function to Combine Data

Create a function named `merge_lists` that takes two lists as arguments.
The function should return a new list containing elements from both lists, with duplicates removed.

```
numbers = [1, 2, 4, 5, 6, 7, 8, 9, 10]
odd_numbers = [1, 3, 5, 7, 9, 11]
merged_list = merge_lists(numbers, odd_numbers)
print(f"Merged list without duplicates: {merged_list}")

#output
Merged list without duplicates: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]
```