Tasks – List Methods (Basics Part 2): insert, append, swap, sort, reverse

Practice **append**, **insert**, **swapping** two elements, **sort()**, and **reverse()**. Do lab **18_lists** first. This is lists basics part 2, before the intermediate lab. Create each file, run it, and check the output.

Run scripts with: `python3 script_name.py`

Part 1 – append (review) and insert

Task 1.1 – append then insert (`list_append_insert.py`)

- Create `list_append_insert.py`. Start with `nums = [10, 20, 30]`. Use `nums.append(40)` and print the list. Then use `nums.insert(1, 15)` to insert 15 at index 1 (so 15 goes before 20). Print the list again. Result should be [10, 15, 20, 30, 40].

Expected output:

```
[10, 20, 30, 40]
[10, 15, 20, 30, 40]
```

Task 1.2 – insert at start and at end (`list_insert_positions.py`)

- Create `list_insert_positions.py`. Start with `words = ["banana", "cherry"]`. Use `words.insert(0, "apple")` to add "apple" at the beginning. Print the list. Then use `words.insert(len(words), "date")` to add "date" at the end (same as append). Print the list. Result: ["apple", "banana", "cherry", "date"].

Expected output:

```
['apple', 'banana', 'cherry']
['apple', 'banana', 'cherry', 'date']
```

Task 1.3 – insert in the middle (`list_insert_middle.py`)

- Create `list_insert_middle.py`. Assign `vals = [1, 2, 4, 5]`. Insert 3 at index 2 so the list becomes [1, 2, 3, 4, 5]. Use `vals.insert(2, 3)`. Print the list.

Expected output:

```
[1, 2, 3, 4, 5]
```

Part 2 – Swapping two elements

Task 2.1 – Swap first and last (`list_swap_first_last.py`)

- Create `list_swap_first_last.py`. Assign `items = [10, 20, 30, 40, 50]`. Swap the first and last elements: use a temporary variable, e.g. `temp = items[0]`, then `items[0] = items[-1]`, then `items[-1] = temp`. Print the list. Result: [50, 20, 30, 40, 10].

Expected output:

```
[50, 20, 30, 40, 10]
```

Task 2.2 – Swap two given indices (`list_swap_indices.py`)

- Create `list_swap_indices.py`. Assign `data = ["a", "b", "c", "d", "e"]`. Swap the elements at index 1 and index 3 (b and d). Use a temporary variable. Print the list. Result: ["a", "d", "c", "b", "e"].

Expected output:

```
['a', 'd', 'c', 'b', 'e']
```

Task 2.3 – Swap with input (`list_swap_input.py`)

- Create `list_swap_input.py`. Start with `nums = [5, 10, 15, 20, 25]`. Ask the user for two indices (two inputs, convert to int). If both indices are valid (0 <= i < len(nums)), swap the elements at those indices and print the list. If either index is invalid, print "Invalid index" and do not change the list. Use variables and if/else.

Expected output (example – indices 0 and 4):

```
Enter first index: 0
Enter second index: 4
[25, 10, 15, 20, 5]
```

Part 3 – sort and reverse

Task 3.1 – sort() (`list_sort.py`)

- Create `list_sort.py`. Assign `nums = [30, 10, 50, 20, 40]`. Use `nums.sort()` to sort the list in place (ascending). Print the list. Then assign `letters = ["banana", "apple", "date", "cherry"]` and use `letters.sort()`. Print letters (alphabetical order).

Expected output:

```
[10, 20, 30, 40, 50]
['apple', 'banana', 'cherry', 'date']
```

Task 3.2 – reverse() (`list_reverse.py`)

- Create `list_reverse.py`. Assign `vals = [1, 2, 3, 4, 5]`. Use `vals.reverse()` to reverse the list in place. Print the list. Result: [5, 4, 3, 2, 1].

Expected output:

```
[5, 4, 3, 2, 1]
```

Task 3.3 – sort then reverse (descending) (`list_sort_reverse.py`)

- Create `list_sort_reverse.py`. Assign `nums = [30, 10, 50, 20, 40]`. First use `nums.sort()` (ascending), then use `nums.reverse()` to get descending order. Print the list. Result: [50, 40, 30, 20, 10].

Expected output:

```
[50, 40, 30, 20, 10]
```

Task 3.4 – Build list, then sort (`list_input_sort.py`)

- Create `list_input_sort.py`. Ask "How many numbers?" and read n. Read n numbers into a list (append each). After the loop use `nums.sort()` and print the sorted list. Use variables. Test with e.g. 4 numbers: 30, 10, 20, 5.

Expected output (example):

```
How many numbers? 4
Number 1: 30
Number 2: 10
Number 3: 20
Number 4: 5
[5, 10, 20, 30]
```

Part 4 – Quick combined

Task 4.1 – insert, swap, sort in one script (`list_combine.py`)

- Create `list_combine.py`. Start with `data = [2, 4, 6]`. Insert 1 at index 0. Print the list. Swap the elements at index 0 and index 3 (first and last). Print the list. Then sort the list and print. Result after all steps: [1, 2, 4, 6] then [6, 2, 4, 1] then [1, 2, 4, 6].

Expected output:

```
[1, 2, 4, 6]
[6, 2, 4, 1]
[1, 2, 4, 6]
```

Done

You've used: **append**, **insert(i, x)**, **swapping** with a temp variable, **sort()** (in place), **reverse()** (in place), and combined them. Next: after more list labs (20–22), do **23_lists_intermediate** (negative index, del, and harder scenarios).