Tasks – Lists in Python

Practice creating lists, indexing, `len()`, modifying (append, change by index), looping over lists, and building lists from input. Create each file, run it, and check the output.

Run scripts with: `python3 script_name.py`

Part 1 – Creating and accessing

Task 1.1 – Create and print a list (`list_basic.py`)

- Create `list_basic.py`.
- Assign a list of numbers to a variable, e.g. `nums = [10, 20, 30, 40, 50]`. Print the list with `print(nums)`. Then print the first element (index 0) and the last element (index -1).

Expected output:

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

Task 1.2 – List with different types (`list_mixed.py`)

- Create `list_mixed.py`.
- Create a list that contains an integer, a float, a string, and a boolean, e.g. `[1, 2.5, "hello", True]`. Assign it to a variable and print the list. Then print the element at index 2 (the string).

Expected output:

```
[1, 2.5, 'hello', True]
hello
```

Task 1.3 – len() and index (`list_len_index.py`)

- Create `list_len_index.py`.
- Assign `items = ["apple", "banana", "cherry", "date"]`. Print the length of the list with `len(items)`. Print the first item, the last item, and the item at index 2. Use variables or direct indexing.

Expected output:

```
4
apple
date
cherry
```

Task 1.4 – Empty list and append (`list_append.py`)

- Create `list_append.py`.
- Start with an empty list: `my_list = []`. Use `my_list.append(5)`, then `my_list.append(10)`, then `my_list.append(15)`. Print the list. Then print its length.

Expected output:

```
[5, 10, 15]
3
```

Part 2 – Modifying lists

Task 2.1 – Change element by index (`list_change.py`)

- Create `list_change.py`.
- Assign `vals = [1, 2, 3, 4, 5]`. Change the element at index 2 to 100: `vals[2] = 100`. Print the list. Then change the last element to 99 (use index -1) and print again.

Expected output:

```
[1, 2, 100, 4, 5]
[1, 2, 100, 4, 99]
```

Task 2.2 – Append in a loop (`list_append_loop.py`)

- Create `list_append_loop.py`.
- Start with an empty list. Use a for loop: `for i in range(1, 6):` and append `i * 10` to the list. After the loop print the list. You should get [10, 20, 30, 40, 50].

Expected output:

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

Task 2.3 – Sum and average of a list (`list_sum_avg.py`)

- Create `list_sum_avg.py`.
- Assign `numbers = [10, 20, 30, 40, 50]`. Use a for loop to compute the sum (e.g. `for n in numbers: total += n`). Print the sum. Then compute the average (sum / len(numbers)) and print it. Use variables for total and average.

Expected output:

```
150
30.0
```

Part 3 – Looping over lists

Task 3.1 – Print each element (`list_loop_print.py`)

- Create `list_loop_print.py`.
- Assign `fruits = ["apple", "banana", "cherry"]`. Use `for fruit in fruits:` and print each fruit (one per line). Then use a second loop with `for i in range(len(fruits)):` and print the index and the fruit, e.g. "0: apple".

Expected output:

```
apple
banana
cherry
0: apple
1: banana
2: cherry
```

Task 3.2 – Double each number in a list (`list_double.py`)

- Create `list_double.py`.
- Assign `nums = [1, 2, 3, 4, 5]`. Use a for loop with index: `for i in range(len(nums)):` and do `nums[i] = nums[i] * 2`. Print the list after the loop. The list should be [2, 4, 6, 8, 10].

Expected output:

```
[2, 4, 6, 8, 10]
```

Task 3.3 – Check if value in list (`list_in.py`)

- Create `list_in.py`.
- Assign `allowed = ["yes", "y", "ok"]`. Read a string from the user with `input()`. If the string is in the list (`if answer in allowed:`), print "Allowed". Else print "Not allowed". Use a variable for the input.

Expected output (example – input "y"):

```
Enter response: y
Allowed
```

Part 4 – Lists and input

Task 4.1 – Build list from N inputs (`list_from_input.py`)

- Create `list_from_input.py`.
- Ask "How many numbers?" and read n. Start with an empty list. Use a for loop (range(n)): read a number with input() and append it to the list (convert to int or float). After the loop print the list and its length.

Expected output (example – 3 numbers: 5, 10, 15):

```
How many numbers? 3
Number 1: 5
Number 2: 10
Number 3: 15
[5, 10, 15]
3
```

Task 4.2 – Sum and max of entered list (`list_input_sum_max.py`)

- Create `list_input_sum_max.py`.
- Ask how many numbers, read n. Build a list of n numbers (same as 4.1). Then use a for loop to compute the sum. Use another loop (or the same idea) to find the maximum (variable max_val, start with first element, then if item > max_val: max_val = item). Print "Sum: X, Max: Y". Use variables and str().

Expected output (example – 4 numbers 3, 7, 2, 9):

```
How many numbers? 4
Number 1: 3
Number 2: 7
Number 3: 2
Number 4: 9
Sum: 21, Max: 9
```

Task 4.3 – List of names and greet each (`list_names.py`)

- Create `list_names.py`.

- Ask "How many names?" and read n. Build a list: in a for loop, ask for a name each time and append it to the list. After the loop, use a for loop to print "Hello, [name]!" for each name in the list. Use variables and string concatenation or print with sep.

Expected output (example – 2 names):

```
How many names? 2
Name 1: Ali
Name 2: Bo
Hello, Ali!
Hello, Bo!
```

Part 5 – A bit more

Task 5.1 – First and last element (`list_first_last.py`)

- Create `list_first_last.py`.
- Read a list size n, then read n numbers into a list (same as 4.1). If the list is not empty, print "First: [first], Last: [last]" using index 0 and index -1. If the list is empty (n was 0), print "Empty list". Use if/else and variables.

Expected output (example – 3 numbers 10, 20, 30):

```
How many? 3
Number 1: 10
Number 2: 20
Number 3: 30
First: 10, Last: 30
```

Task 5.2 – Count how many are positive (`list_count_positive.py`)

- Create `list_count_positive.py`.
- Assign `numbers = [3, -1, 5, -2, 0, 4]`. Use a variable count = 0 and a for loop over the list. For each number, if it is greater than 0, add 1 to count. After the loop print "Positive count: " + str(count).

Expected output:

```
Positive count: 3
```

Done

You've used: creating lists, indexing (0 and -1), len(), append(), changing by index, for loop over list, for loop with range(len(list)), building a list from input, in operator, and simple list processing (sum, max, count).