Q4 – 10 Points

Instructions:

1. Open the file `Q4_template_code.py` from the provided folder for this question.
2. Complete the functions in the provided template by replacing `<YOUR_REGEX_HERE>` with appropriate regular expressions for each pattern description.
3. Run the `test_functions()` function to validate your regular expressions with the provided test cases.
4. Ensure your code handles both matching and non-matching strings correctly.

Requirements

1. **Function `match_pattern_a()`**
   - Matches the strings: `1`, `12`, `123`, `13`
   - Test cases: `["1", "12", "123", "13", "14", "112"]`
2. **Function `match_pattern_b()`**
   - Matches the strings: `1`, `12`, `123`
   - Test cases: `["1", "12", "123", "13", "0", "124"]`
3. **Function `match_pattern_c()`**
   - One or more `1`s, followed by one or more `2`s, followed by one or more `3`s (e.g., `11223`, `111222333`)
   - Test cases: `["123", "1123", "1112233", "111222333", "113", "223"]`
4. **Function `match_pattern_d()`**
   - Non-empty strings consisting **only** of uppercase and lowercase letters (a-z, A-Z)
   - Test cases: `["abc", "ABC", "aBc", "123", "abc123", "!@#", " "]`
5. **Function `match_pattern_e()`**
   - Strings that contain **at least one blank space** and uppercase or lowercase letters (e.g., `"Hello World"`, `"A B C"`)
   - Test cases: `["Hello World", "A B C", "abc", "HelloWorld", " "]`
6. **Function `match_pattern_f()`**
   - String representations of **non-empty Python lists** containing positive integers (e.g., `[1, 2, 3]`, `[5]`, `[10, 20, 30]`)
   - Test cases: `["[1, 2, 3]", "[5]", "[10, 20, 30]", "[]", "[1, -2, 3]", "[abc]"]`