


/ Homework 1 - C Programming Test Suite /

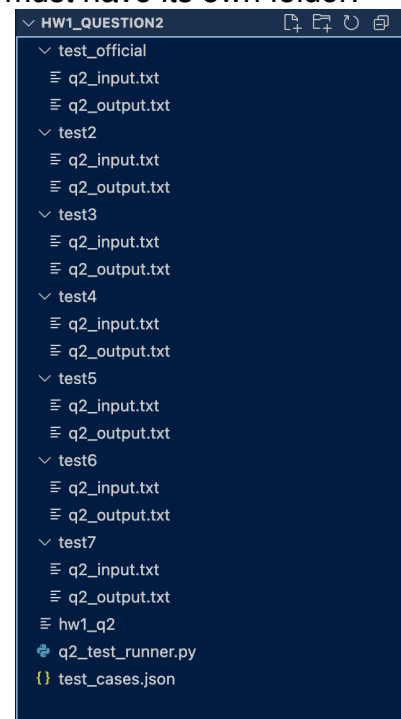
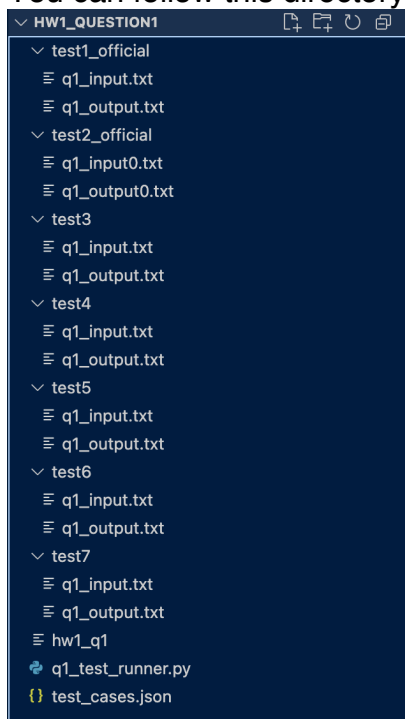
Short explanation:

- 1  Compile your C file: `clang q1.c -o q1.exe //or gcc q1.c -o q1.exe`
- 2 Add test folders to `hw1_questionX/`
- 3 Update `test_cases.json` with paths to each test's input/output
- 4 Run the tester: `python3 q1_test_runner.py`

Thoroughly explained process:

— Directory Structure

You can follow this directory structure; each question must have its own folder:



Base files:

1. Test folder
 - a. Input.txt (the input file in .txt format)
 - b. Output.txt (the output file in .txt format)
2. Question.exe or Question (the compiled file)
3. Test_cases.json (json file to set the correct file paths)
4. Question_test_runner.py (python runner file)

You can add your test files in the main directory.

If you change the location of the test files Edit the json with the new path.

There is no need to change the python runner file its pretty much plug-and-play.

How to Run the Tests:

—Step 1: Compile your C file

For example:

Bash – (paste in your terminal):

```

```
clang q1.c -o q1.exe # For Question 1
```

```
clang q2.c -o q2.exe # For Question 2
```

// on mac you can use this format:

```
// gcc q1.c -o q1
```

```
//without the .exe extension (It is not necessary)
```

```

—Step 2: Run the test runner

Make sure Python 3 is installed, then:

Bash – paste in the terminal:

```


```
Python3 q1_test_runner.py # Tests Q1 using JSON and input/output files
```

```
python3 q2_test_runner.py # Tests Q2 using JSON and input/output files
```

```

—Step 3: Check output

Each test will show:

-  if passed

-  if failed, with details of input, expected, and actual output

— Notes

- The `.json` files contain paths to all the test case folders

- File-based input/output makes test management easier for complex cases

- No additional libraries are required in your C programs (only ``)

Question 1 - Reverse Digits

—Behavior

- Takes an integer as input
- Prints its digits in reverse order
- Handles 0 and negative values

— Test Cases

| Test | Input | Output |

|-----|-----|-----|

| 1 | 0 | Reverse of the number is: 0 |

| 2 | 7 | Reverse of the number is: 7 |

| 3 | 1000 | Reverse of the number is: 1 |

| 4 | -123 | Reverse of the number is: -321 |

| 5 | -1200 | Reverse of the number is: -21 |

| 6 | 214748364 | Reverse of the number is: 463847412 |

| 7 | -1000000000 | Reverse of the number is: -1 |

Question 2 - Print Divisors

—Behavior

- Takes a natural number or 0
- If input is 0: prints "0 has no Divisors!"
- If input > 0: prints all positive divisors in ascending order

—Test Cases

| Test | Input | Output |

|-----|-----|-----|

| 1 | 0 | 0 has no Divisors! |

| 2 | 1 | Divisors of 1 are: 1 |

| 3 | 7 | Divisors of 7 are: 1 7 |

| 4 | 36 | Divisors of 36 are: 1 2 3 4 6 9 12 18 36 |

| 5 | 1000 | All divisors of 1000 |

| 6 | 7919 | Divisors of 7919 are: 1 7919 |

| 7 | 2147483647 | Divisors of 2147483647 are: 1 2147483647 |

 Author: Aiman Abed

Course: "C Programming"

Homework 1 - Test Automation

Feel free to expand or reuse this format for future assignments!