Python Exercises Report

Hussein Hussein
Student ID: 58301

Repository Information

GitHub Repository: https://github.com/Leo-devv/cloud-programming-labs
Online Editor: https://replit.com/@replit-userhy/cloud-programming-labs

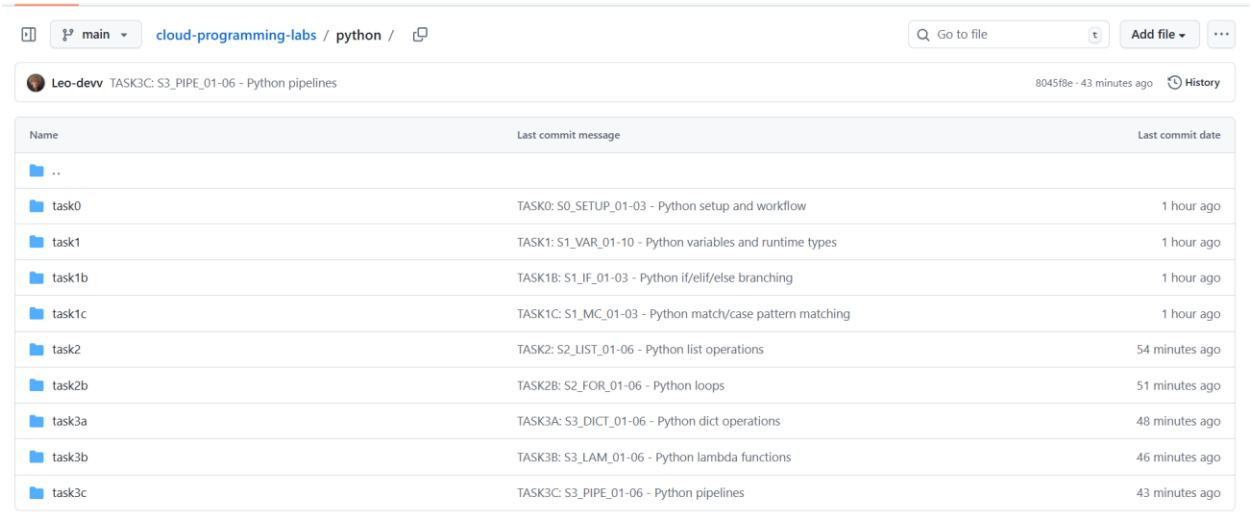
Project Overview

This report documents 50 Python exercises covering Python fundamentals (Python 3.10+), organized into 9 tasks including setup and optional bonus.

Repository Structure

The project structure follows course requirements:

```
python/ ├── task0/      Setup (3 files) ├── task1/      Variables and Types
(10 files) ├── task1b/     Conditional Statements (3 files) ├── task1c/
Match/Case Statements (3 files) ├── task2/      List Operations (6 files) ├──
task2b/     Loop Constructs (6 files) ├── task3a/     Dict Operations (6
files) ├── task3b/     Lambda Functions (6 files) └── task3c/     Pipelines
(7 files including bonus)
```



Files follow the naming convention S[stage]_[type]_[number].py.

Task 0: Setup and Workflow

File	Description
`S0_SETUP_01.py`	Create project and print "It works"
`S0_SETUP_02.py`	File structure setup

S0_SETUP_03.py	Minimal test helper eq()

Task 1: Variables and Runtime Types

File	Description
S1_VAR_01.py	Catalog of values and type()
S1_VAR_02.py	Rebinding and dynamic typing
S1_VAR_03.py	Mutability: list vs tuple
S1_VAR_04.py	Identity vs equality (is vs ==)
S1_VAR_05.py	Truthiness
S1_VAR_06.py	Safe conversion: int() / float()
S1_VAR_07.py	NaN and math.isnan
S1_VAR_08.py	Big integers
S1_VAR_09.py	Type hints are not runtime enforcement
S1_VAR_10.py	Mini inspector

Task 1B: if/elif/else Branching

File	Description
S1_IF_01.py	Shipping cost with validation
S1_IF_02.py	Score to grade (A-F)
S1_IF_03.py	Normalize user name

Task 1C: match/case Pattern Matching (Python 3.10+)

File	Description
S1_MC_01.py	Day name using match/case
S1_MC_02.py	Command router using match/case
S1_MC_03.py	Simple calculator using match/case

Task 2: List Operations

File	Description

S2_LIST_01.py	Clean number strings
S2_LIST_02.py	Deduplicate without set
S2_LIST_03.py	Chunk a list
S2_LIST_04.py	Flatten one level
S2_LIST_05.py	Stats (min, max, avg, sum)
S2_LIST_06.py	Transform records

Task 2B: Loop Constructs

File	Description
S2_FOR_01.py	FizzBuzz+
S2_FOR_02.py	Find first even
S2_FOR_03.py	Sum until threshold
S2_FOR_04.py	Count occurrences (histogram)
S2_FOR_05.py	10x10 multiplication table
S2_FOR_06.py	Sum nested lists (matrix)

Task 3A: Dict Operations

File	Description
S3_DICT_01.py	Safe get by dotted path
S3_DICT_02.py	Merge defaults (shallow)
S3_DICT_03.py	Pick keys
S3_DICT_04.py	Omit keys
S3_DICT_05.py	Invert with collisions
S3_DICT_06.py	Group by property

Task 3B: Lambda Functions

File	Description
S3_LAM_01.py	Convert to lambdas
S3_LAM_02.py	Sort by key
S3_LAM_03.py	Closure factory (make_adder)
S3_LAM_04.py	Sum of squares of even numbers
S3_LAM_05.py	Higher-order predicate (at_least)
S3_LAM_06.py	Map values in a dict

Task 3C: Pipelines

File	Description
S3_PIPE_01.py	Implement pipe()
S3_PIPE_02.py	Implement compose()
S3_PIPE_03.py	String normalization pipeline
S3_PIPE_04.py	Iterable pipeline (generator-based)
S3_PIPE_05.py	Log line pipeline
S3_PIPE_06.py	Safe pipeline (pipe_safe)
S3_PIPE_07.py	Bonus: itertools (islice, groupby) + memory explanation

Git Workflow

Commit History

Development progressed through incremental commits:

Commit	Message
f8276e2	TASK0: S0_SETUP_01-03 - Python setup and workflow
bdb4e30	TASK1: S1_VAR_01-10 - Python variables and runtime types
d5f5252	TASK1B: S1_IF_01-03 - Python if/elif/else branching
acc9fa8	TASK1C: S1_MC_01-03 - Python match/case pattern matching
61cd5b1	TASK2: S2_LIST_01-06 - Python list operations
24bfdbc	TASK2B: S2_FOR_01-06 - Python loops
3ea4280	TASK3A: S3_DICT_01-06 - Python dict operations
f9c3d00	TASK3B: S3_LAM_01-06 - Python lambda functions
8045f8e	TASK3C: S3_PIPE_01-07 - Python pipelines

Tags

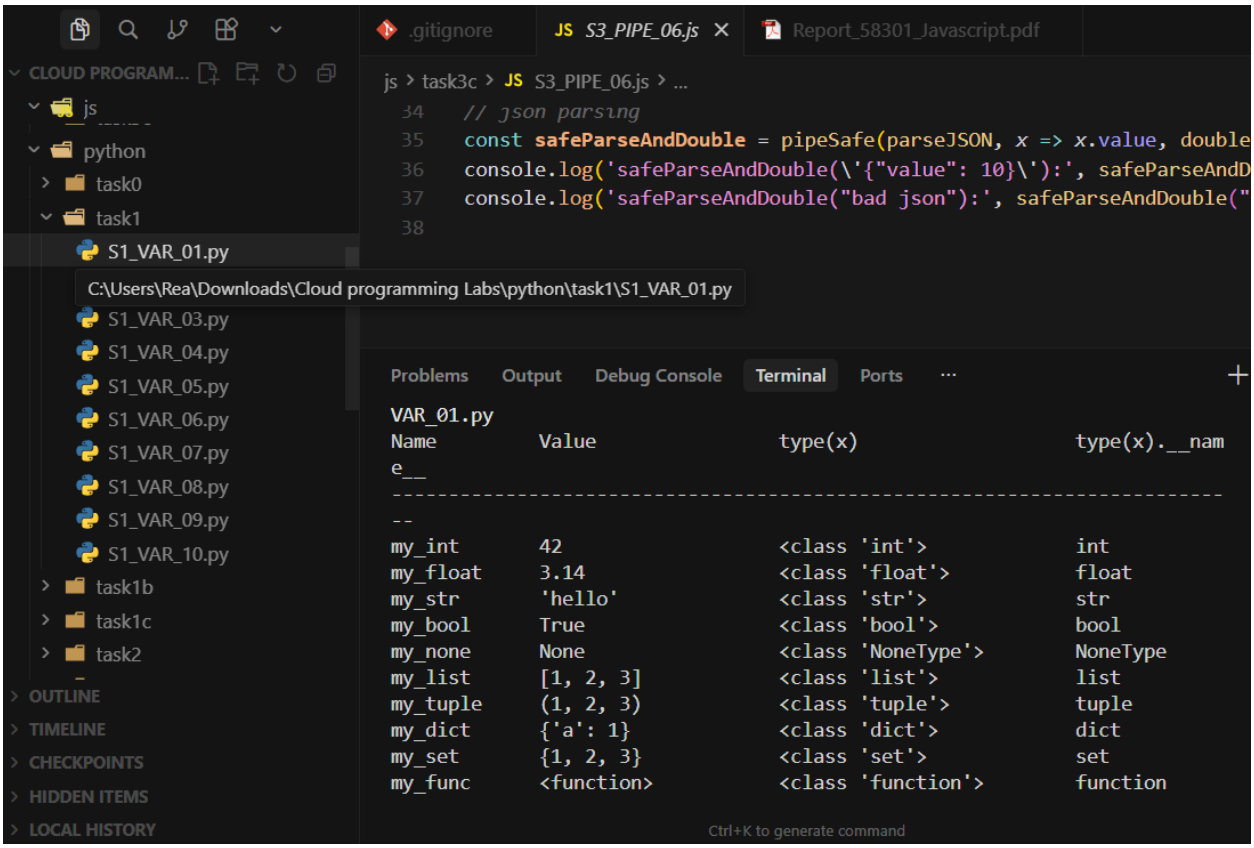
Tag	Commit
py-task0-done	f8276e2
py-task1-done	bdb4e30
py-task1b-done	d5f5252
py-task1c-done	acc9fa8
py-task2-done	61cd5b1
py-task2b-done	24bfdbc
py-task3a-done	3ea4280
py-task3b-done	f9c3d00
py-task3c-done	8045f8e
py-all-done	8045f8e

Testing

All exercises were tested using Python 3.10+. Each file contains executable code with output verification.

To run any exercise:

```
python python/task1/S1_VAR_01.py
```



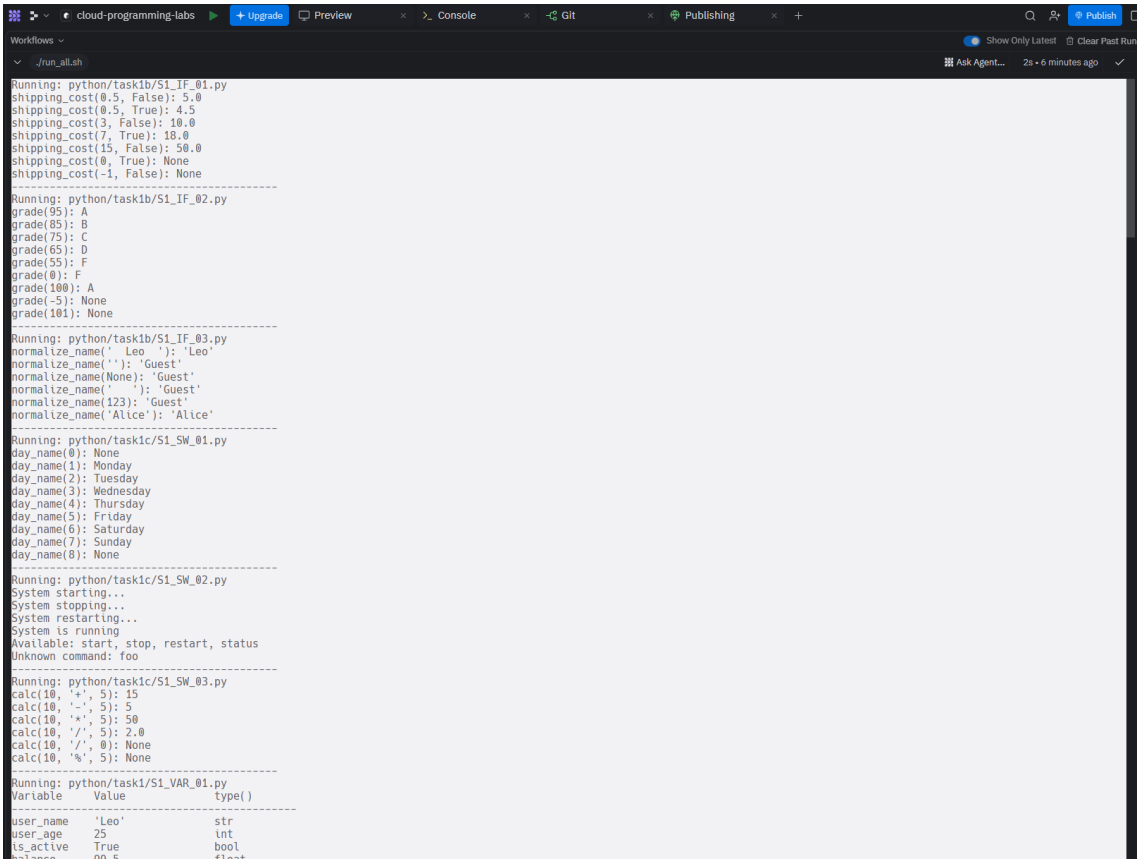
The repository can be tested in Replit using the link above.

Screenshots

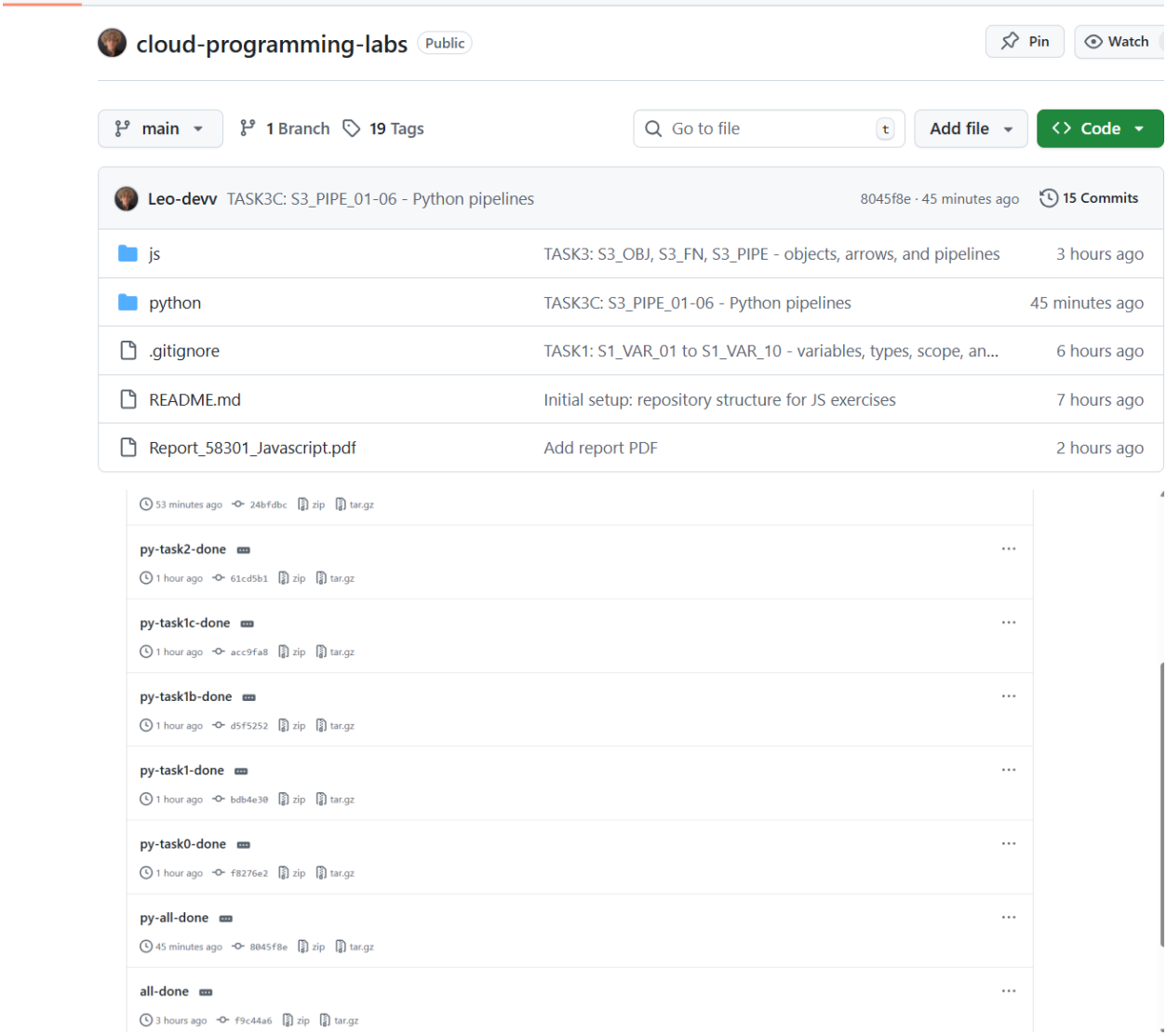
Extra Task

```
_PIPE_07.py
Logs grouped by level:
  DEBUG: 1 entries
  ERROR: 2 entries
  INFO: 4 entries

First 3 logs using islice:
  {'level': 'INFO', 'msg': 'Started'}
  {'level': 'INFO', 'msg': 'Processing'}
  {'level': 'ERROR', 'msg': 'Failed'}

Items 100-105 from infinite generator:
[100, 101, 102, 103, 104, 105]
```

Replit Interface



Git Tags



Summary

All pyexercises are complete and tested (Task 0: 3 files, Task 1: 10 files, Task 1B: 3 files, Task 1C: 3 files, Task 2: 6 files, Task 2B: 6 files, Task 3A: 6 files, Task 3B: 6 files, Task 3C: 7 files including bonus). The project follows course guidelines for structure, file naming, and version control.