Small Projects with Python: Structured Academic Exercises for Foundational Programming Skills and Open Science Practices

v1.0.0, Zenodo DOI: 10.5281/zenodo.15548859

Mohammad Rasoul Sohrabi (Rasoul Unlimited)

Department of Biomedical Engineering

Islamic Azad University, Central Tehran Branch

ORCID: 0009-0004-7177-2080

Email: Rasoul.Unlimited@gmail.com

July 30, 2025

Abstract

This article presents ”Small Projects with Python,” a systematically organized set of foundational Python programming exercises originally developed as undergraduate coursework assignments at Sahand University of Technology (Tabriz, Iran). Each exercise focuses on core algorithmic principles, structured problem-solving, and code readability, promoting foundational programming competence. The open publication of this project, complete with comprehensive semantic metadata, version control, and DOI archival via Zenodo, exemplifies how academic assignments can transition into fully citable and reusable software resources adhering to FAIR and open science principles.

1 Introduction

Python’s prominence in education has made it an essential skill in engineering and scientific disciplines. ”Small Projects with Python” emerged as a carefully curated compilation of introductory algorithmic exercises, emphasizing structured code development and readability. This collection addresses a notable gap in openly archived, structured, and metadata-rich educational resources, thereby enhancing citation potential and global academic visibility.

2 Background and Context

The exercises originated from undergraduate coursework in a Python programming module at Sahand University of Technology, encompassing fundamental computational concepts such as conditional logic, iterative methods, numerical analysis, and prime number theory. Transitioning these classroom exercises into a structured, publicly archived software repository illustrates the potential of academic coursework to contribute meaningfully to educational infrastructure.

3 Software Description

3.1 Content Overview

The repository comprises eight individual Python scripts demonstrating elementary algorithmic and numerical methods:

* Part1: Summation from user input
* Part2: Cumulative sum calculation (while loop)
* Part3: Interval overlap detection
* Part4: Integer square root determination
* Part5: Prime number identification (naive method)
* Part6: Prime factorization
* Part7: Square root approximation (Newton-Raphson method)
* Part8: Analysis of Stirling's factorial approximation

3.2 Repository Structure

Scripts are organized in a flat directory with clearly defined metadata files (CITATION.cff, codemeta.json, zenodo.json) and supporting documentation, ensuring compliance with scholarly citation standards.

3.3 Dependencies

The exercises require only Python's standard library, ensuring compatibility and straightforward execution in diverse computational environments.

3.4 Licensing and Archiving

All materials are licensed under the MIT License, archived with Zenodo (DOI: 10.5281/zenodo.15548859), and systematically versioned for scholarly reference.

4 Functional Capabilities

The educational value stems from:

* Logical separation and clear code organization
* Transparent algorithmic reasoning
* Descriptive variable naming and extensive comments
* Compatibility with automated validation frameworks

An additional validation script (TestExample.py) illustrates execution correctness, demonstrating reliability for educational usage.

5 Applications and Educational Value

This software serves as:
* Instructional examples for introductory programming courses
* Reference code for algorithmic fundamentals
* Benchmarking tools for educational numerical methods
* Reusable materials for digital educational platforms (MOOCs, workshops)

6 Comparative Analysis

Unlike typical online repositories, this project distinguishes itself by integrating scholarly standards, semantic metadata, structured DOI registration, and formal citation infrastructure. It addresses critical needs in reproducibility and academic reuse.

7 Software Validation

Scripts have undergone thorough manual testing, demonstrating robustness and reliability within standard Python environments. Future versions will include automated testing badges and detailed validation documentation to enhance reproducibility.

8 Academic Impact and FAIR Compliance

The inclusion of structured metadata (CITATION.cff, codemeta.json, JSON-LD schema) facilitates indexing in academic databases, automatic citation generation, and interoperability with scholarly platforms (ORCID, OpenAIRE, Google Dataset Search). This project exemplifies best practices for FAIR (Findable, Accessible, Interoperable, Reusable) principles in educational software publication.

9 Author Declaration

The author confirms the originality and independent development of this work, without derivation from previously published resources, conducted solely within an academic context.

10 Acknowledgments

Gratitude is extended to the course instructor at Sahand University of Technology for inspiring these algorithmic exercises and supporting their scholarly dissemination.

11 Open Science and Accessibility

All software resources and metadata are openly accessible on GitHub and Zenodo under the MIT License, reinforcing commitments to reproducibility, transparency, and open science.

References

[1] Wilson, G. et al. (2014). Best practices for scientific computing. *PLoS biology*, 12(1), e1001745.

[2] Katz, D. S. et al. (2021). Recognizing the value of software in research. *F1000Research*, 9, 86.

[3] Spaaks, J. H., & Kluyver, T. (2021). The citation file format: formalizing citation metadata for software. *PeerJ Computer Science*, 7, e623.

[4] Jones, M. B., et al. (2017). CodeMeta: An exchange schema for software metadata. *PeerJ Preprints*, 5, e3217v1.

[5] Smith, A. M., Katz, D. S., Niemeyer, K. E., et al. (2016). Software citation principles. *PeerJ Computer Science*, 2, e86.