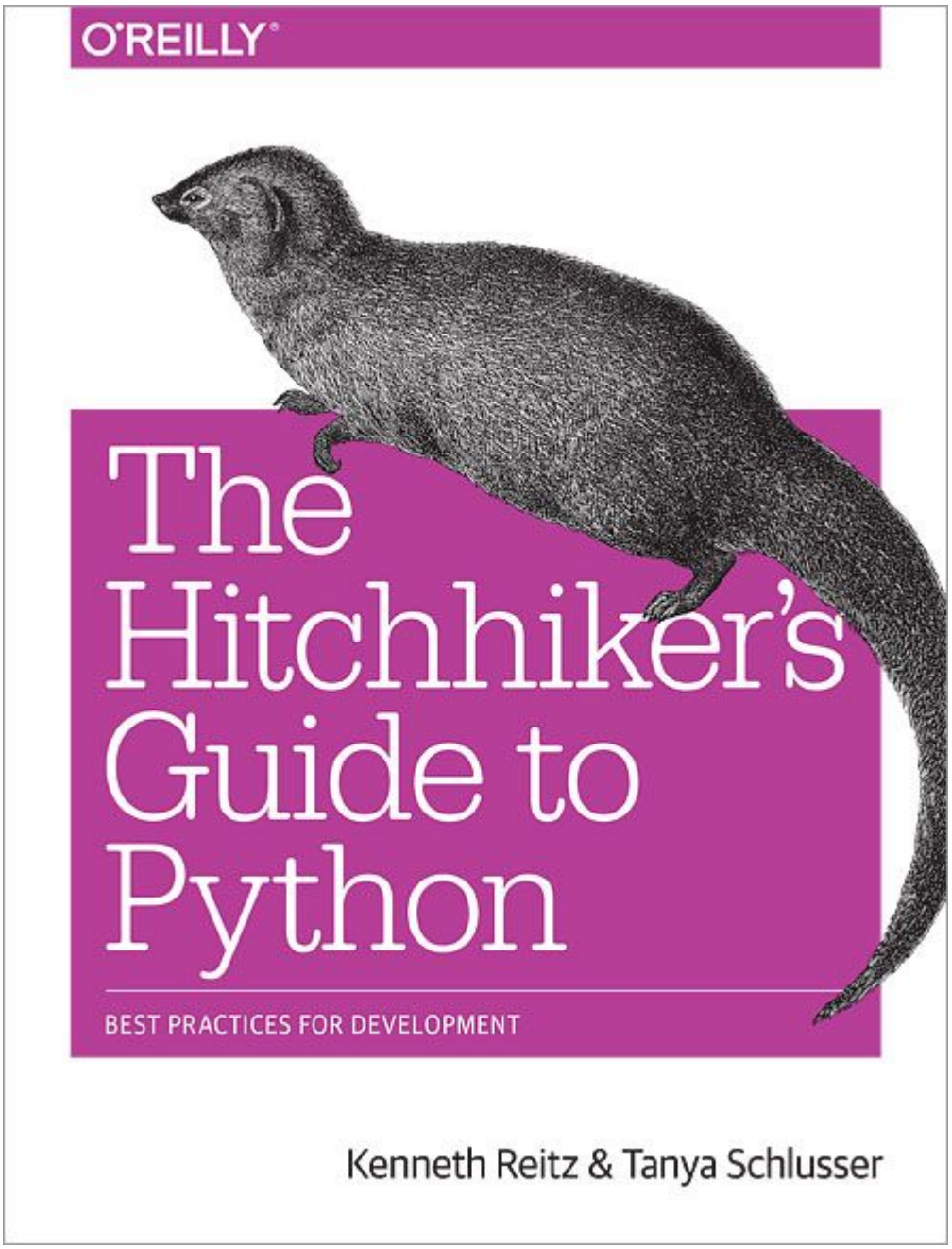
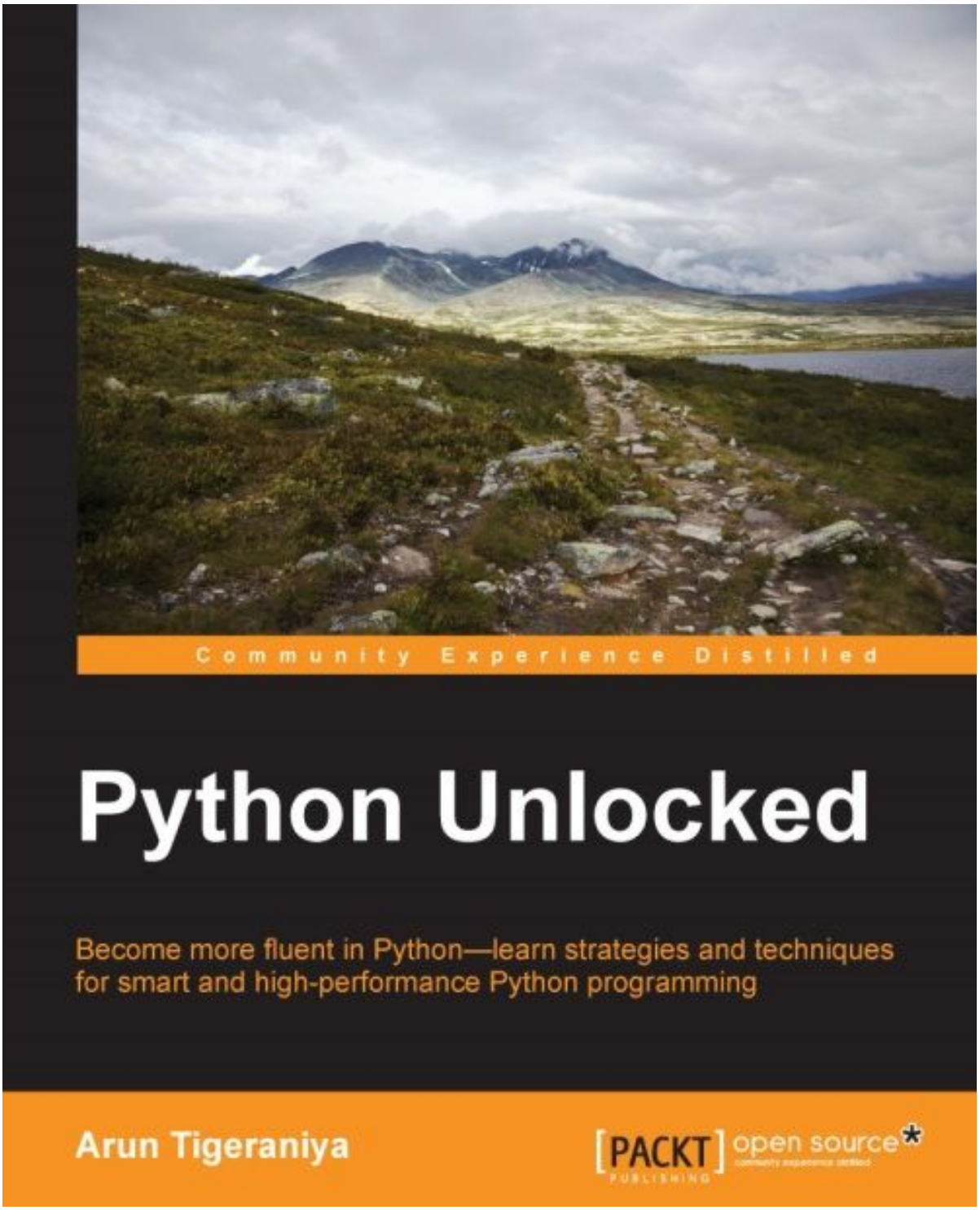
Framework na vytváranie testových úloh v Pythone

- **Reitz, Schlusser: The Hitchhiker's Guide to Python**; Best Practices for Development, O' Reilly Media, 2016



- **Tigeraniya: Python Unlocked**; Packt Publishing 2015



- Michal Forišek, On the Suitability of Programming Tasks for Automated Evaluation, Informatics in Education 5 (2006)

On the suitability of programming tasks for automated evaluation

Michal Forišek
Department of Informatics
Faculty of Mathematics, Physics and Informatics
Comenius University, Bratislava, Slovakia

Abstract

For many programming tasks we would be glad to have some kind of automatic evaluation process. As an example, most of the programming contests use an automatic evaluation of the contestants' submissions. While this approach is clearly highly efficient, it also has some drawbacks. Often it is the case that the test inputs are not able to "break" all flawed submissions. In this article we show that the situation is not pleasant at all – for some programming tasks it is impossible to design good test inputs. Moreover, we discuss some ways how to recognize such tasks, and discuss other possibilities for doing the evaluation. The discussion is focused on programming contests, but the results can be applied for any programming tasks, e.g., assignments in school.

1 Introduction

Competitions similar to the International Olympiad in Informatics (IOI, [12]) and the ACM International Collegiate Programming Contest (ACM ICPC, [1]) have been going on for many years. In the ACM ICPC contest model the contestants are given feedback on the correctness of their submissions during the contest. Due to a vast amount of submitted programs this is almost always done automatically. (Often there is a human supervising the testing process.) At the IOI the submitted programs are only tested after the contest ends, and the submissions are awarded a partial score for solving each of the test inputs.

We will now describe this canonical IOI scoring model in more detail. Each of the tasks presented to the contestants is worth 100 points. Before the competition the author of the task prepares his own solution, a set of test inputs, and an output correctness checker. (The output correctness checker can be replaced by a set of correct output files, if they are unique.) The 100 points are distributed among the test inputs. After the contest ends, each of the contestants' programs is compiled and run on each test input. For each test input

Ďalšie základné informácie

- Python Unit Testing Tutorial in https://cgoldberg.github.io/python-unittest-tutorial/
- https://www.python.org/dev/peps/pep-0008/
- http://www.practicepython.org/
- https://www.codewars.com/