WHY AND HOW TO MARK AND GIVE FEEDBACK ON CODE

Charlotte Desvages

School of Mathematics

Informatics Teaching Festival, 11th May 2022

PERSPECTIVES FROM MATHEMATICS

- We teach programming largely for scientific computing, statistics, data science.
- Overarching goals:
  - Acquire sufficient fluency with a programming language to implement numerical methods
  - Develop computational thinking
  - Understand computational cost and efficiency
  - Understand the different ways in which "error" guides our choices in mathematical modelling and computation
  - Get a sense of what programming is like "in the real world"

ASSESSING CODE IN TWO DIFFERENT COURSES

PYTHON PROGRAMMING (MSC)

- "Python skills" course
- Range of example applications covered, principally data manipulation and visualisation
- Develop good coding practice, learn to select and use appropriate libraries for practical applications

COMPUTING AND NUMERICS (Y2)

- First course in computational maths
- Mainly covering methods in numerical analysis (differentiation/integration, root-finding, introduction to numerical ODEs)
- Also introduces Python

ASSESSMENT DESIGN

- Any code submitted for assessment is part of the student's evidence for demonstrating learning outcomes.

Penalties applied (**up to half** of marks obtained)

	– 1 pts	Little or no code comments
	– 0.5 pts	Insufficient code comments
✔	**– 1 pts**	No docstring
	– 0.5 pts	Incorrect/incomplete docstring
	– 1 pts	Structure/efficiency/conciseness
✔	**– 0.5 pts**	Structure/efficiency/conciseness
	– 1 pts	Readability (spacing, variable names...)
✔	**– 0.5 pts**	Readability (spacing, variable names...)

Presentation penalties (**up to half** of marks **awarded above**)

– 1 pts	Insufficient code comments
– 0.5 pts	Insufficient code comments
– 1 pts	Unclear/poor presentation of results
– 0.5 pts	Unclear/poor presentation of results

TRANSPARENT ASSESSMENT

- Code is written for machines to execute, and for people to read.
  - Automatic assessment is suitable for (part of) the "machine" side of this.
- Before final projects, students peer-assess each other's code several times, along the same criteria.
  - The marking scheme is given in advance of submission (released with the assessment).
  - Introducing practice examples assessed by the instructor was helpful with the peer-assessment.
- An outline of the "write code for people" side of the marking scheme is also given with final project assignments.

THE IMPORTANT QUESTIONS

- What are the code-related learning outcomes?
- In what ways do we clearly convey to students the important aspects of writing code?
- How does formative assessment align with (and serve) summative assessment?
- Do we award marks for good code, or give penalties for bad code?