Four-column template

Author name(s): Dan Grose and Jamie Fairbrother	**Module title: STOR609 Programming for Reproducible Research**	
Programme(s) contributes to: MRes Statistics and Operational Research	**Level (UG,6 /PG,7) and year of study: PG7**	**Core / optional: core**

Template for aligning learning outcomes, assessment and teaching and learning approaches - consider how you use technology to deliver this

Intended learning outcomes On successful completion of the module students should be able to… (How do these contribute to the PLOs?)	**Assessment and feedback** The summative and formative assessment tasks designed to enable and to demonstrate the achievement of the learning outcomes of the module.	**Teaching hours / class contact** Synchronous activities and approaches designed to support the students in achieving the learning outcomes.	**Directed independent learning** Asynchronous activities students are expected to undertake outside contact hours to progress towards achievement of the learning outcomes.
Students who pass this module should be able to... A1. to analyse an algorithm in terms of its computational complexity and determine appropriate data structures for its implementation in software A2. to use the three main models of programming (imperative, object-oriented and functional), and identify applicable design patterns. A3. to use software engineering tools such as profilers, debuggers, testing frameworks and environment management systems A4: Understand basic computer architecture and operating systems and be able to parallelize code where appropriate A5. to use tools and mechanisms for collaborative programming, and distribution and support of code within the wider research community A6. to produce scientific software that is replicable, reproducible, reusable, re-runnable, and repeatable.	**Programming Assessment (A1, A2)**: • Timing: after session on Introduction to Python • Weighting: 25% • Form: individual coursework with coding and short report • Assignment: ◦ Analyse an algorithm ◦ Choose and justify a suitable programming paradigm(s) for implementing the algorithm ◦ Implement the algorithm in code ◦ Assess the performance of the code **Software Engineering Assessment (A2-A4):** • Timing: after session on Computer Architecture and Parallel Programming • Weighting: 25% • Form: Individual assessment with coding and short report • Assignment: ◦ Study and analyse a selection of third party code, one of which includes some element of parallelization	• Analysis of algorithms : 4 hours (lecture and workshops) • Introduction to Python: 4 hours (workshops) • 5 R's for Reproducible Research (1 hour lecture) • Introduction to software engineering: 3 hours (lecture + workshop) • Further Python + Design Patterns: 4 hours (lecture and workshops) • Computer architecture and models for Parallel Programming: 3 hours (lecture and workshop) • Python Packaging and Environments (4 hours workshop) • Version Control with Git (3 hours workshop)	Students are required to carry out any activities required to successfully complete their coursework assessments such as coding and debugging. In addition to this, the following activities may help in understanding the contents of the course • Read suggested papers • Complete short programming challenges to improve coding skills • Independently download packages and run some simple examples • Research, discover and understand several examples of the analysis of algorithms

Intended learning outcomes On successful completion of the module students should be able to… (How do these contribute to the PLOs?)	**Assessment and feedback** The summative and formative assessment tasks designed to enable and to demonstrate the achievement of the learning outcomes of the module.	**Teaching hours / class contact** Synchronous activities and approaches designed to support the students in achieving the learning outcomes.	**Directed independent learning** Asynchronous activities students are expected to undertake outside contact hours to progress towards achievement of the learning outcomes.
Graduate Attributes/General Learning Outcomes Students who pass this module should be able to… G1. Make informed choices regarding programming systems and software engineering tools G2. Collaborate effectively on programming projects G3. Produce and make available publicly quality software G4. Communicate and justify the design and organisation of software and its components	◦ Debug, profile and refactor code to improve its quality with respect to the five R's ◦ Discuss and justify refactoring of code in a short report and submit refactored code **Final Group Assessment (A1-A6):** • Timing: after all teaching sessions complete • Weighting: 50% • Form: group work coursework with coding, presentation and peer assessment • Assignment: ◦ Programming reproduction exercise where groups will have to implement a methodology from the stats/OR literature as a package ◦ Students will be required to collaborate using version control software, the activity logs from this will be part of the assessment	• Group work Workshop (3 hours) Total teaching hours: 29	