Course Proposal Form

Please see Page 2 for instructions on which parts of this form to complete, whom to consult with to avoid unnecessary effort, and where to send the completed form.

Proposer(s): Ben Morse, David Henty, Adam Carter, Mike Jackson Date: 11/02/20

Cover page: Basic permanent course information

Unless otherwise noted, items in this section are entered into EUCLID and **cannot** be changed without creating an entirely new course.

Course Name	Programming Skills
Course Acronym *(used by the School only, e.g., for the Sortable Course List)*	PS
Course Level If the course is **only** available to MSc students, then it must be classed as Postgraduate. All other courses, regardless of level, are Undergraduate.	Y Postgraduate
Normal Year Taken	MSc
Also available in years *[This can be changed later if need be.]*	
SCQF Credit Level Level 8 should normally be used for pre-honours courses. Level 10 should normally be used for optional UG3 courses (so UG4 students may also take them) and for courses aimed mainly at UG4 students. Level 11 should be used for courses aimed mainly at MSc students, whether or not UG4 students can also take them.	11
SCQF Credit Points	10
Delivery Location	On-line Distance Learning
Course Type	Online Distance Learning
Marking Scheme By default, courses use a numerical marking scheme. If you wish to use a grade-only marking scheme, your course proposal below should justify this.	Standard (numerical)

Guidance for remaining sections:

For an initial course proposal, please complete the **cover page and Section 1 (Case for Support)**, which asks you to describe the need for this course and to provide an overview of the course design, including the learning outcomes. **Please discuss your plans as early as possible with the head of Curriculum Review to avoid unnecessary effort.**

Send the form with these sections completed to the BoS Academic Secretary and head of Curriculum Review (listed on the BoS page) to obtain their comments before filling out the remainder of the form.

If a full proposal is invited, please complete the remaining sections and send to **iss-bos@inf.ed.ac.uk**.

2. Student-facing course description and additional feedback and assessment information.
This section provides most of the information students see in the DRPS entry for this course, as well as related details for BoS consideration.

3. Further information for BoS consideration: sample materials.

4. Additional Course Details required for DRPS. *[Administrative information such as delivery timing and prerequisites.]*

5. Placement in degree programme tables. *[Required for all level 9-11 courses; used to determine where the course will be added to existing degree programme tables.]*

6. Comments from colleagues. *[All course proposal should be sent to relevant colleagues in the area as well as to the appropriate year organizer and BoS Academic Secretary for comment in good time before the BoS meeting. Use this section to indicate what feedback has been solicited and received.]*

Colour coding and item-by-item guidance:

Guidance is provided in italics for each item. Please also refer to the guidance for new course proposals at http://www.inf.ed.ac.uk/student-services/committees/board-of-studies/course-proposal-guidelines. Examples of previous course proposal submissions are available on the past meetings page http://web.inf.ed.ac.uk/infweb/admin/committees/bos/meetings-directory but note that the proposal form was updated in Jan 2019.

Sections in gold are for student view and are required before a course can be entered into DRPS. You must complete these sections even if your course has already been approved based on other documentation.
Sections in orange are for School use but are still required for all courses (even those that have already been approved based on other documentation).
Section in gray are for consideration by the Board of Studies. They are normally required for all new course proposals but may be omitted in some circumstances (e.g., for invited course proposals) if you obtain permission in advance.

1. Case for support

This section is for consideration by the Board of Studies. The final two boxes (Learning Outcomes, Graduate Attributes) will also go into the student-facing course description.

Overall contribution to teaching portfolio and relation to existing curriculum

Please explain (a) what motivates the course proposal (e.g. a previous course having become outdated/inappropriate, an emergent or maturing research area or new research activity in the School, offerings of our competitors) and (b) how it relates to existing courses and degree programmes (including any prerequisite courses). Every new course should make an important contribution to the delivery of our Degree Programmes.

Online version of on-campus EPCC compulsory course INFR11177 Programming Skills. This course is required for the Year 1 DPT of the already approved (and recruiting) online MSc programmes in HPC and HPC with Data Science. It will also be made available as an optional course on the DSTI programme (and any other online programmes in the University which expressed interest – some tentative interest already expressed from MVM programmes).

The on-campus equivalent is already taken as an outside choice by students in Bioinformatics (Biology), Physics and Informatics.

Target audience and expected demand

Describe the type of student the course would appeal to in terms of background, level of ability, and interests, and the expected class size for the course based on anticipated demand. A good justification would include some evidence, e.g. by referring to projects in an area, class sizes in similar courses, employer demand for the skills taught in the course, etc

Target audiences are students on the HPC and HPC with Data Science Online MSc programmes for which this will be a compulsory course (unless they already meet the learning outcomes in which case they may apply for a concession to the DPT). It has also been identified as a good course for those on the DSTI programming who already have some programming experience wishing to improve it and as a good follow-up course for those who enjoy introductory programming courses elsewhere on the programme.

Expected demand in first few years is 10-40 (depending on external programme uptake).

Anticipated Resource Requirements

Estimate how much lecturing, tutoring, exam preparation and marking effort will be needed in steady state, and any additional resources needed to set the course up initially. Provide estimates relative to class size where applicable and discuss how support staff will be recruited and supervised, if the class is likely to be very large. Please mention any scaling limits due to equipment or space. If equipment is required, say how it will be procured and maintained.]

There are great synergies in terms of material preparation and delivery with INFR11177 and wider EPCC training activities. The only true 'costs' of running the course are the time spent offering tutorial sessions (~2 hours per-week), query answering, Teaching Assistant time, and marking. Of these marking is the only activity which has a close-to linear scaling of time taken with larger class-size. Some efficiencies can be found between the online and on-campus courses in terms of moderation activities between the marking teams.

Quotas, special arrangements or unusual characteristics

Please specify if this course requires any special arrangements such as quotas or other registration arrangements; is a collaboration with another school or institution, or has other atypical characteristics that may affect finances or student registration. Further justification/information may be requested for such courses.

EPCC course.

Narrative description of the course aims and structure

Please describe the main goals of the course and how the course design will allow students to achieve those goals. This section should be consistent with the student-facing information provided below, but should provide additional information to help colleagues at BoS understand the vision and structure of the course. This description may refer to the learning outcomes and graduate attributes (next two boxes) and should explain how activities such as tutorials, labs, or in-lecture activities will support them, and how the proposed assessments will assess them.

For courses that are important pre-requisites for other courses, this section may also provide content/syllabus information which is too detailed for the student-facing description, such as a lecture-by-lecture syllabus.

This course is designed to help you to produce higher quality code; code that is readable, maintainable, usable, correct and efficient in less time and with less effort. These programming skills are applicable to programs in any language and the course is illustrated with examples from C, Python, Fortran, and Java.

Areas of specific relevance to parallel high-performance computing, including the use of batch systems, scientific libraries, profilers, and performance analysis, are also covered.
The course covers the following:
- Best practices for scientific computing.
- Introduction to operating systems, compilers and batch systems.
- Writing programs for people, programs that are readable, maintainable, and usable.
- Reviewing code to identify bugs and share expertise.
- Debugging using code browsers and debuggers.
- Managing versions and collaborating using revision control.
- Automating common tasks and building programs using a build tool.
- Automated regression and unit testing to help ensure and maintain program correctness.
- Profiling and performance analysis to identify and improve a program's use of processing and memory resources.
- Security, authentication and authorisation concepts.
- REST application programming interfaces.
- Open source software.

Summary of Intended Learning Outcomes (MAXIMUM OF 5)

List the learning outcomes of the course. These must be assessable (i.e., observable), so must specify what the student should be able to do concretely, not simply what they should "understand". Use concrete verbs that indicate (a) what type of assessment would be appropriate, and (b) what level of knowledge/thinking is expected (from recall to analysis to novel creation). ***Example verbs:*** *define, explain, implement, compare, justify. Assessments (described later) should be tied to the learning outcomes.*

Outcomes should typically focus more on the types of thinking/skills developed than on the detailed course content, and the level of thinking should be appropriate to the level of the course: outcomes for a Level 11 course should include more higher-level thinking skills than for a Level 8 course. Further guidance on writing learning outcomes can be found at https://www.ncl.ac.uk/ltds/assets/documents/res-writinglearningoutcomes.pdf

On completion of this course, the student will be able to

1) Understand how to develop code in a Linux/Unix environment.
2) Create code that is modular, robust, reliable, maintainable, efficient and understandable.
3) Analyse code and create unit and regression tests.

4) Evaluate and apply software development tools including integrated development environments, code browsers, debuggers, revision control, build management tools, automated test frameworks, and profilers.
5) Design, implement, analyse and write-up software performance experiments.

Graduate Attributes, Personal & Professional Skills

List the personal attributes and generic transferrable skills this course will help develop. Examples include

Cognitive skills: *problem-solving, critical/analytical thinking, handling ambiguity*

Responsibility, autonomy, effectiveness: *independent learning, self-awareness and reflection, creativity, decision-making, leadership, organization and time management, flexibility and change management, ethical/social/professional awareness and responsibility, entrepreneurship*

Communication: *interpersonal/teamwork skills, verbal and/or written communication, cross-cultural or cross-disciplinary communication*

Project planning.
Effective written and diagrammatic communication.
Technical writing.
Data collection and analysis.
Reflection on learning and practice.
Adaptation to circumstances.
Solution Exploration, Evaluation and Prioritisation.
Programming Skills.

2. Student-facing course description and additional feedback and assessment information

Except where noted, all fields are required and will go into the DRPS entry for the course (for use by students). Important: any text in DRPS is effectively a contract with students, so should not include details that are likely to change from year to year.

Summary Description *Provide a brief official description of the course, around 100 words. This should be worded in a student-friendly way, it is the part of the descriptor a student is most likely to read. If this course replaces another course, please say so in this summary.*	This course is designed to help you to produce higher quality code; code that is readable, maintainable, usable, correct and efficient in less time and with less effort. These programming skills are applicable to programs in any language and the course is illustrated with examples from C, Python, Fortran, and Java.
Keywords *Give a list of searchable keywords.*	Programming,PS,EPCC,Fortran,C,C++,Python,Java,Debugging,Software
Course Description *A more detailed student-facing description of the course, which should normally include (a) a more in-depth academic description of the learning aims, nature and context of the course, (b) a rough outline of the content or syllabus, often as bullet points, and (c) a description of how the course will be taught, how students are expected to engage with their learning and how they will be expected to evidence and demonstrate their achievement of the intended learning outcomes.]*	The course covers the following: - Best practices for scientific computing. - Introduction to operating systems, compilers and batch systems. - Writing programs for people, programs that are readable, maintainable, and usable. - Reviewing code to identify bugs and share expertise. - Debugging using code browsers and debuggers. - Managing versions and collaborating using revision control. - Automating common tasks and building programs using a build tool. - Automated regression and unit testing to help ensure and maintain program correctness. - Profiling and performance analysis to identify and improve a program's use of processing and memory resources. - Security, authentication and authorisation concepts. - REST application programming interfaces. - Open source software.
Assessment Weightings: *These should correspond approximately to the proportion of learning outcomes that each component assesses. More than 30% coursework requires specific justification. The expectation for a 10pt course is 20% coursework with the equivalent of one 15-20hr assessed assignment (but possibly split into smaller pieces). See 'components of assessment' below.*	Coursework 100%

Further Assessment Information *Provide any further information that should go on DRPS for students. E.g., if the assessment includes required group work or if students must pass some individual component of assessment as well as the course overall.*	Coursework 100 %. This is a practical course and is assessed by two pieces of coursework. The assessments are: 1. Developing automated tests (50%) 2. Measuring and reporting performance (50%) Development is in Python. Students will be given access to resources introducing Python programming.
Components of assessment and time spent on assignments (for BoS only) *If not already included in the course narrative description, please describe the type of assessments (oral presentation, report, programming, etc) and **how each component of assessment will assess the intended learning outcomes**. Where coursework involves group work, it is important to remember that every student has to be assessed individually for their contribution to any jointly produced piece of work.* *Also estimate **how many hours** students will spend on assignments. Please see the School policy on Workload and Assessment, which states that students should not be expected to spend more than 6-7 hrs/wk per 10 credits, including contact hours.* *Note that it often desirable to include formative assignments which are not formally assessed but submitted for feedback, often in combination with peer assessment.*	Assessment 1 assesses learning outcomes 1-4. Assessment 2 assesses learning outcome 5. Assessment 1: 28 hours Assessment 2: 21 hours
Feedback Information *Provide a high-level description of how and what type of feedback will be provided to students, for inclusion in DRPS.*	Provided on assessed work within 15 working days of deadlines and through tutorial and discussion sessions. One-to-one appointments can also be arranged for further insight
Additional Feedback Information (for BoS use only) *If not already included in the course narrative, provide further details on planned feedback arrangements. This includes how course feedback is solicited from the class and responded to, as well as what feedback students will get (either on*	Practical exercises undertaken will build students' familiarity with material and prepare them for the assessments. Feedback can be provided on these via the Collaborate practical sessions and course discussion.

work that contributes to their final mark, or not).

The University is committed to a baseline of principles regarding feedback that we have to implement at every level, and the School encourages submission of at least one piece of written work for formative feedback.

In general, formative feedback:

- *Should say how students can improve.*
- *Need not be on individual work (e.g., consider a lecture or document summarizing common issues.)*
- *Can include oral feedback during labs/tutorials*
- *Can include feedback from peers*
- *Clickers/TopHat/equivalents can provide in-class feedback for both students and lecturer.*
- *Is returned in time for other forms of assessment to which it relates, to allow feedforward.*

Breakdown of Learning and Teaching Activities

State how many hours students spend on each part of the course. The total should be 10 x course credits, but please also see the School policy on Workload and Assessment.which states that students should not be expected to spend more than 6-7 hrs/wk per 10 credits, including contact hours.

Assume 10 weeks of lectures slots and 10 weeks of tutorials, but these need not all be used. As a guideline, a 10-pt course typically has 18-20 lecture hours, but should have only around 15 lectures of examinable material; the rest should be used for guest lectures, revision sessions, introductions to assignments, etc.

Contact hours

Hours	Type
9	Lecture Hours
	Seminar/Tutorial Hours
	Dissertation Project Supervision Hours
18	Supervised practical/Workshop/Studio hours
1	Feedback/Feedforward hours
	Summative assessment hours
	Revision Session Hours

Non-contact hours

Hours	Type
69	Directed Learning & Independent Learning hours

Total hours:

Reading List/Learning Resources

You are encouraged to create resource lists using LEGANTO

LEGANTO List also provided

Wilson G, Aruliah D.A., Brown C.T., Chue Hong N.P., Davis M., et al. 'Best Practices for Scientific Computing'. PLoS Biol 12(1): e1001745. doi:10.1371/journal.pbio.1001745.

Wilson, G., Bryan, J., Cranston, K., Kitzes, J., Nederbragt, L. and Teal, T. 'Good enough practices in scientific computing'. PLoS Comput Biol 13(6): e1005510. doi:10.1371/journal.pcbi.1005510.

Sandve G.K., Nekrutenko A., Taylor J., Hovig E. 'Ten Simple Rules for Reproducible Computational Research'. PLoS Comput Biol 9(10):

	e1003285. doi:10.1371/journal.pcbi.1003285. Barnes, N. and Jones, D. 'Clear Climate Code: Rewriting Legacy Science Software for Clarity'. IEEE Software 28(6), pp36-42, Nov-Dec 2011. doi:10.1109/MS.2011.113. Bailey, D.H. 'Twelve Ways to Fool the Masses When Giving Performance Results on Parallel Computers', Supercomputing Review August 1991, pp54-55. Author copy: http://crd-legacy.lbl.gov/~dhbailey/dhbpapers/twelve-ways.pdf. Bailey, D.H. 'Misleading performance in the supercomputing field', Supercomputing '92 Proceedings of the 1992 ACM/IEEE conference on Supercomputing, pp155-158, November 1992. doi:10.1109/SUPERC.1992.236699. The following text books are not essential, but one or the other is recommended, if pursuing a career in software development: McConnell, S. 'Code Complete: A Practical Handbook of Software Construction' (2nd ed.), Microsoft Press, 19 Jun 2004. ISBN-10: 0735619670. ISBN-13: 978-0735619678. Either this or The Pragmatic Programmer below is a good programming best practice book. Hunt, A. 'The Pragmatic Programmer', Addison Wesley, 20 Oct 1999. ISBN-10: 020161622X, ISBN-13: 978-0201616224. Either this or Code Complete above is a good programming best practice book. Additional links to other recommended papers, blog posts, articles and online resources are provided.

3. Further information for BoS consideration: sample materials

A full proposal for a new course must include examples of exercises and assessment. Please provide these below, along with publicity information if the course is to be advertised outwith the School.

Course information and publicity *The course web page (typically the Learn landing page) will be linked from the Sortable Course List, and information such as timetables and assignment deadlines must be made available prior to the start of the academic year. Please specify here if any additional info/publicity is needed for your course: typically only if it is aimed largely at non-SoI students.*	Not required
Sample tutorial/lab sheet questions *Provide a list of tutorial questions and answers and/or samples of lab sheets. These need not be fully fleshed out but should indicate what sort of exercises will be provided to help students learn the material.*	Already used in Programming Skills INFR11177

Sample assessment materials *If the course is primarily assessed by **exam**, provide a sample exam question with model answers. Any non-standard exam format must be justified. The online list of past exam papers gives an idea of typical and alternative exam formats: http://www.inf.ed.ac.uk/teaching/exam_papers/.* *If the course is largely or primarily assessed by **coursework**, provide a sketch of a possible assignment with an estimate of effort against each sub-task and a description of marking criteria.*	Already used by INFR11177
Any other relevant materials *Include anything else that is relevant, possibly in the form of links. If you do not want to specify a set of concrete readings for the official course descriptor, please list examples here.*	

4. Additional Course Details for DRPS

Except where otherwise noted, these fields are required for entry into EUCLID and will be visible to students in the DRPS entry.

Planned Academic Year of Delivery *(The first year you anticipate the course running, e.g. AY 2019-20)*	AY 2020-21
Course Organiser *(By default, the course proposer)*	Mike Jackson is course organiser of INFR11177. An alternative course organiser will be identified before September who will work alongside Mike and will be the primary point of contact for this course.
Intended Delivery Period	Semester 1
Timetable considerations/conflicts *For School use. Please specify any constraints to be considered (e.g. overlap of popular combinations, other specialism courses, external courses etc). Include whether the semester delivery is constrained or could be flexible.*	Online so N/A

Is this course available to visiting students?	No **If no, please provide a justification here:** Online course. INFR11177 available to on-campus visiting students.
Required pre-requisite courses *Use sparingly: these are enforced in PATH and can only be waived by approval from the School's Curriculum Approval Officer. Note that cross-year required pre-requisites may prevent MSc students from registering; consider using recommended pre-requisites or "other requirements" instead.*	No
Recommended pre-requisite courses	No
Required co-requisite courses *Specify any courses that must be taken in parallel with the existing course. Note that this leads to a timetabling constraint that should be mentioned elsewhere in the proposal.*	No
Prohibited Combinations *Specify any courses that may not be taken in combination with the proposed course].*	Yes (please specify full course name(s) and code(s)): INFR11177 – Programming Skills Informatics 2C - Introduction to Software Engineering (INFR08019) AND AI Large Practical (INFR09043) AND Computer Science Large Practical (INFR09044) AND Software Engineering Large Practical (INFR09045) AND Informatics Large Practical (INFR09051)
Other Requirements/Additional Information *This information is often used by MSc students and students from other Schools to see if they have appropriate background without having done our School's courses. So please avoid course titles, instead list specific knowledge and skills (such as mathematical concepts, programming ability or specific languages, etc).* *Also list any other constraints on registration, for example: "Only available to 4th Year Informatics students including those on joint degrees." or "This course is open to all Informatics students including those on joint degrees, and to students in the School of Mathematics. Other external students whose DPT does not list this course should seek permission from the course organiser."*	Yes (please specify): You must have experience in a programming language e.g. C, C++, Python, FORTRAN or Java. You must be familiar with programming concepts including conditionals, loops, arrays and functions. A knowledge of bash shell is highly recommended. No knowledge of parallel programming is required. Some of the above relevant prerequisite material will be made available for self-study, but will not be taught in classes.

Visiting Student Pre-requisites	N/A

5. Placement in degree programme tables: for level 9-11 courses only

This section is for consideration by the Board of Studies and will be used later by ITO to determine where the course will be added to existing degree programme tables.

Is this course restricted to students on a specific degree? *E.g., some courses are only available to students on a specific CDT or MSc.*	___No
Is this course compulsory for students on any degree(s)?	Yes (please specify and provide justification): PTMSCHIPEC1U MSc High Performance Computing 3-6 years Part-time Intermittent PTPGCHIPEC1U PGDip High Performance Computing 2-4 years Part-time Intermittent PTPGDHIPEC1U PGCert High Performance Computing 1-2 years Part-time Intermittent PTMSCHPCDS3P MSc High Performance Computing with Data Science 3-6 years Part-time Intermittent PTPGCHPCDS1P PGDip High Performance Computing with Data Science 2-4 years Part-time Intermittent PTPGDHPCDS1PPGCert High Performance Computing with Data Science 1-2 years Part-time Intermittent
Any issues for part-time students? *Normally, part-time students have access to the same courses as full-time students on the equivalent degree. If you anticipate any problems with this, please specify here.*	Designed for part-time students. No issues

For optional courses:

If this course is available but non-compulsory for students on various degrees (most courses), please fill in this section. The choices here determine where the course appears in degree programme tables (DPTs) and the 2-3 character tags are displayed in the Informatics sortable course list.

Should this course be tagged as 'ML' (machine learning foundations and methods)? *Courses with the ML tag are typically very high-demand and most degrees limit the number of ML credits. If your course might appeal to a similar audience but draw off students from these large courses, please select 'no' and choose one of the tags below.*	No

If you chose 'no', please choose at least one of the following tags… *Ideally, select exactly one, unless there is a good argument for more than one. These three are used in various combinations for many of our degrees.*	**EPCC**
…and also tick if any of the following tags or categories apply. *Do not tick any of these if you selected 'ML' already.*	
If you are not sure which tags are most appropriate or have other questions about this section, please note any comments/issues here.	

6. Comments from colleagues

All course proposal should be sent to relevant colleagues in the area as well as to the appropriate year organizer and BoS Academic Secretary for comment in good time before the BoS meeting. Please indicate here what feedback has been solicited and received.

Additional Comments *Summarise any comments received from relevant individuals prior to proposing the course. If you have not discussed this proposal with others please note this.*	Course required for online degree programmes approved at March 2019 board. Adam Carter has been consulted as programme director of DSTI. He supports this degree
Year Organiser Comments *Year Organisers are responsible for maintaining the official Year Guides for every year of study, which, among other things, provide guidance on available course choices and specialist areas. The Year Organisers of all years for which the course will be offered should be consulted on the appropriateness and relevance on the course. Issues to consider here include balance of course offerings across semesters, subject areas, and credit levels, timetabling implications, fit into the administrative structures used in delivering that year.]*	
BoS Academic Secretary Comments *Proposals must be checked by the Secretary of the Board of Studies prior to discussion at the actual Board meeting. This is a placeholder for their comments, mainly on the formal quality of the content provided above.*	