Narrowing and Stretching: Addressing the Challenge of Multi-track Programming

Steven Bradley
Eleni Akrida
s.p.bradley@durham.ac.uk
eleni.akrida@durham.ac.uk
Department of Computer Science, Durham University
Durham, UK

ABSTRACT

Given the different amount of programming experience that students have arriving at university, some universities have introduced alternative multiple streams to teach programming. This approach was exemplified by Harvey Mudd College, who successfully used it as part of a range of measures to increase gender diversity within computing. One of the challenges with having multiple streams is how to narrow the experience gap without holding back those students who arrive with more experience of programming. In this paper we discuss one potential solution, which is to offer more experienced students the opportunity to choose their own topic of study and their own project. Rather than having technical constraints on their work, students are instead required to demonstrate how they have managed their own learning and worked collaboratively, developing within Bloom's affective domain.

CCS CONCEPTS

• **Social and professional topics** → **Computer science education**; **Student assessment**.

KEYWORDS

education, programming, assessment, diversity

ACM Reference Format:
Steven Bradley and Eleni Akrida. 2022. Narrowing and Stretching: Addressing the Challenge of Multi-track Programming. In *Computing Education Practice 2022 (CEP 2022), January 6, 2022, Durham, United Kingdom.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 WHY? INTRODUCTION

Over the last decade a key focus of education policy for schools in the UK has been on narrowing the attainment gap for disadvantaged pupils, with substantial funding provided through Pupil Premium (Pupil Deprivation Grant in Wales) [13]. Disadvantaged groups have also been a major concern in Higher Education, but here measures have been more focused on access to or participation in Higher Education, with fee levels in England being contingent upon access agreements being made with the Office for Fair Access (OFFA) [18]. The inclusion of learning gain as a measure of success in the Teaching Excellence Framework (TEF) for Higher Education is both provisional and controversial [19] whereas satisfaction measures, particularly the National Student Survey (NSS), are central to league tables and university management across the UK.

In Higher Education, however, Computer Science seems to run counter to the broader policy agenda. HESA [5] (Table WP2) reports that in 2017/18, Computer Sciences had 95.4% of its UK domiciled entrants coming from under-represented groups (defined as state schools and low participation neighbourhoods), very close to the top subject area of Education with 97.4%[1]. If you were to ask a Computer Scientist what the main issues were around access and participation, they would very likely refer to gender. This is borne out by the HESA statistics from 2019/20 [4], which show that only 19.9% of Computing students in the UK were female, with only Engineering and technology (19.8%) doing worse. Overall 57.0% of UK students were female, so there is no national priority to increase female participation in undergraduate education as a whole.

Of course this problem is not specific to the UK, and many people have tried many ways to increase female participation in computing. Harvey Mudd College have reported great success with a range of measures [6], including providing different routes for students with different experience: the Black (experienced) and Gold (not experienced) sections. This should be advantageous to all students, but because girls have less experience with programming on average, and experience is important [20], this should benefit female students on average.

The key question that arises from this separation, and that we address in this this paper is

> **How can we narrow the learning gap for less experienced students while still stretching and challenging those with more experience?**

At Harvey Mudd the approach taken was that " the Black section explores more challenging applications of the same fundamental concepts. As just one example, advanced students implement Huffman coding instead of, or in addition to, run-length encoding. [6] " However this still means that the experienced group are exposed to more curriculum-focused material in greater depth, making them more prepared for the academic content of subsequent teaching.

Some have advocated addressing issues of fairness by using an ipsative approach to assessment [11], where students' grades depend not on their attainment, but on the progress that they make from their starting position. This mirrors the Progress 8 measure [3], used in English secondary schools. While this approach, used in

CEP 2022, January 6, 2022, Durham, United Kingdom
2022. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

[1] In comparison, "Combined subjects" came bottom with 73.2%

HE, can address issues in fairness of assessment grades, it does not impact on the attainment gap that students exhibit.

An alternative approach, that we have adopted, is to consider a different kind of learning entirely. Most educationalists are familiar with Bloom's taxonomy, but for many people this means just one of three domains that Bloom's group identified: the Cognitive Domain. The other domains receive much less attention and it is the Affective Domain [2] — sometimes described as emotional intelligence — that we consider here. As Pierre and Oughton identify, many of the affective competencies are closely related to soft skills that are central to professional computing practice: " They include self-awareness, analytical thinking, leadership skills, team-building skills, flexibility, acceptance of diversity, the ability to communicate effectively, creativity, problem-solving skills, listening skills, diplomacy and change-readiness. [17] "

Because assessment in schools is very individualised, some of the negative stereotypes associated with computing [9] — social isolation and technology focus — are institutionalised. So students that arrive with a lot programming experience (cognitive domain) may well be less well developed in the affective domain. Our approach then is to focus more on building on affective skills — receiving, responding, valuing, organising — through collaborative self-guided learning, assessed by reflective writing. This could prove more challenging for male students because females are generally happier with group work [21].

How we implemented these changes is described in the next section and then evaluated in Section 3. The curriculum development and delivery of the module was carried out by the first author, with the second author completing the evaluation.

2 WHAT? SELF-GUIDED LEARNING

Curriculum and Teaching

In the academic year 2020/21 we introduced Black and Gold routes for our first year Programming module[2]. The technical content for the two modules was the same, covering client-side and server-side programming with JavaScript, combined with development tools for source code management, code quality and automated unit-testing. Similar material had been covered for the previous two years, but in a single module taken by all first year undergraduate computer scientists. The extra content in module descriptions for Programming (Black) is given in Table 1, other than that they were the same: notably there was no difference in the Content or the intended learning outcomes related to Subject-specific Knowledge.

Students were advised that the Black route was for those with A-level Computer Science or equivalent experience, and were initially allocated on that basis. A lightweight web-based self-evaluation questionnaire [12] was administered in the first week and the distribution of results was shared with students so that they could identify their experience level compared with the rest of the cohort. Students were given the opportunity to change between the two modules without constraint during the first three weeks of teaching.

The Gold route was delivered over the same timescale as the previous provision, but with more lectures scheduled to go over the material with more examples. Practical laboratory sessions were run for the whole academic year. For the Black route the technical content was covered more quickly, with fewer examples, and delivered over one term rather than two.

In the second term (in Black) there were no new content lectures or scheduled practicals, but optional drop-in sessions were provided for students. The whole of the second term was given over to self-guided learning, working towards the final assessment.

Item	Extra content for Programming (Black)
Aims	to apply programming principles to real-world problems
Subject-specific Skills	an ability to apply software development tools and skills in real-world scenarios e.g. open-source projects, hackathons, competitions
Key Skills	an ability to plan and work independently

Table 1: Differences between module specifications for Programming (Gold) and Programming (Black)

Assessment

Both modules were assessed solely by two pieces of coursework (one for each term), with the second assignment for Gold being fairly equivalent to the first assignment for Black, except that Black used some peer assessment of the first assignment [7]. While constrained in terms of the techniques used, students were free to choose the application domain [8].

The second assignment for Programming (Black) is where the self-guided learning was implemented. Students were required to choose a skill to develop (JavaScript or non-JavaScript) and to choose a collaborative project to contribute to e.g. Open-Source Software, hackathon or (collaborative) competition. They were assessed on the basis of a written learning log (recorded with git) and a guide for other learners. The learning log was essentially a piece of reflective writing, so the assessment criteria were chosen from a toolkit provided by Edinburgh University [1]:

- Number, timing and word count of entries (use of git)
- Appropriate development and monitoring of goals
- Evidence of increased understanding
- Evidence of collaboration
- Evidence of criticality about your own actions and assumptions

Context

Our institution is small/medium sized research intensive university in the UK, and the students are studying for BSc/MEng in Computer Science Degree, which has a high entry tariff (top 10 in the UK). On Programming (Black) there were 88 students in Academic Year (AY) 2020/21, with 71 taking the Gold route. The AY 2020/21 was very unusual in that, because of the Covid pandemic, all teaching took place on-line.

What we expected to happen

When we previously introduced non-standard forms of assessment it proved unpopular with students [7], so we expected there to be

[2]This is not the only programming undertaken by the first years — python is covered separately elsewhere.

some discomfort expressed by students, who are " instinctively wary of approaches with which they are not familiar or that might be more demanding ... unhappy about assessment methods where the outcomes might be less predictable. [14] " In particular, others have identified the introduction of reflection into computing classes as challenging [10]. Our experience in other modules, and from colleagues in other institutions, is that students in general do not like doing group/collaborative work. We anticipated that in general students would enjoy the opportunity to have freedom in what they studied, and hoped that they would find it interesting and challenging, but that some would prefer more direction for the topic to be studied. Whether or not our expectations were realised is the subject of the next section.

3 DOES IT WORK? EVALUATION

We collected students' perceptions using Module Evaluation Questionnaires (MEQ) and performed quantitative and qualitative analysis on them. MEQ are routinely collected online across the University towards the end of each Academic Year (AY) for every module delivered, with invitations for participation sent only to students taking the module during that AY. Participation in the MEQ was voluntary and in total, there were 39 student participants for Programming (Black) in 2020/21. A Likert scale was used, ranging from 1 = Definitely Disagree to 5 = Definitely Agree, to allow students to score various aspects of the module; the MEQ report provides comparative data for each question showing the mean scores for other contexts where that question was used (department, faculty, or university). Students were also encouraged to provide free text responses to highlight positive aspects (12 responses received) as well as areas where the module could improve (13 responses received).

We compared the data from AY 2020/21 to historical data from MEQ prior to the split to Programming (Gold) and Programming (Black). Table 2 shows a selection of questions on the MEQ and the respective average scores that the module received in 2018/19[3] and 2020/21 (columns titled 'Module'), with departmental averages shown for comparison. The questions were selected based on how directly related they are to the quality of the module, including quality of assessment, and as such we excluded questions that are more directly related to the perception of teaching staff by students.

	2018/19		2020/21	
	Module	Dept	Module	Dept
The module was well-organised and ran smoothly	2.92	3.97	4.15	4.05
The module has challenged me to achieve my best work	3.55	3.98	4.08	4.03
Staff have made the subject interesting	3.60	3.70	4.21	3.87

Table 2: Mean values of selected MEQ.

It can be seen that the responses to the statement "The module was well-organised and ran smoothly" were neutral for the module in 2018/19, while the 2020/21 module average has dramatically increased to 'definitely agree'. Furthermore, the statements "The module has challenged me to achieve my best work" and "Staff have made the subject interesting" had average results in the 'agree' section, while the 2020/21 average has increased to 'definitely agree'. For all three questions, the new average in 2020/21 is higher than the corresponding departmental mean values, contrary to 2018/19. The results from the MEQ scores align with the free text responses and suggest that the students who took Programming (Black) in 2020/21 are much happier with the content and structure of the module overall compared to those who took Programming in 2018/19. However, it is worth noting that in 2018/19, free text responses indicated a lot of dissatisfaction with peer-assessment, possibly creating an 'anti-halo effect' that brought down all satisfaction results within the module during that year.

We also performed a qualitative analysis with NVivo using a blended coding approach [15] on the textual data from 2020/2021. When asked to highlight positive aspects of the module, students responded with comments such as:

> " It was incredibly motivating to try your best with the summative projects. "
>
> " The ability to decide what my second term project is on was amazing as that allowed me the freedom to explore my interests. "
>
> " Really enjoyed the freedom to learn whatever aspect of programming that was given in the second assignment. "

Two students commented on the intensity of learning they were required to do during the first term, with one of those students commenting on the positive effect that had on completing the assignment during the second term. This provides evidence for the hoped-for jump from being a 'learner' to being 'self-reliant when working independently and when cooperating' [2].

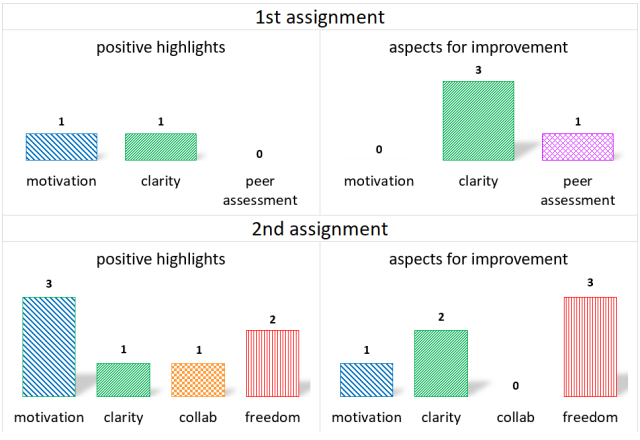

Figure 1: Textual MEQ data references classified based on assessment (1st / 2nd term) and sentiment (positive / negative).

Figure 1 summarises our analysis of the free text responses, presenting the coding of positive and negative comments associated with assessments. Regarding the second assignment in particular, students mentioned positively that they found it interesting and

[3]No MEQ data were collected in 2019/20 due to the Covid pandemic

challenging, and that they enjoyed the freedom they were allowed, which indicates that students were getting the anticipated benefits of self-guided learning. However, the downside of allowing students, and perhaps particularly year one students, to work on their topic of choice in an assessment context is often exposed by the lack of clarity of aims and ambiguity that students report. Indeed, when considering aspects of the module that could be improved, three students asked for more obvious and specific goals for the self-guided assessment, with two students suggesting more clarity in the specification. It comes as a surprise that no students left negative comments regarding collaborative work or reflective writing; instead, there is one positive comment mentioning collaboration. There was a single comment regarding peer feedback, unlike previous years that saw a large amount of negative comments regarding peer assessment; this appears to be resolved in 2020/21 through careful motivation of this type of assessment and explanation of its usefulness in the context of a programming module.

Threats to validity

This was not a controlled trial, and there are threats to the validity of any conclusion that we might try to draw from the data. Firstly the sample size of responses was relatively small, particularly in the free text. When the MEQ were submitted the students had not yet submitted (and probably not completed in full) the second assignment. Comparisons between the 2018/19 and 2020/21 data are difficult, not least because of the different circumstances in which they were delivered: in-person vs on-line. More rigorous introduction was provided for peer assessment in 2020/21, which probably affected the overall student sentiment for the module. Lastly, in 2018/19 the full cohort was taught in the same way, so the class size was larger and more mixed in ability on entry.

4 CONCLUSIONS

Overall we have found the approach to be successful, inasmuch as it has proved relatively popular with students. Many of the anticipated benefits were observed: enjoyment of the freedom in study choices; the students found it interesting and challenging. Surprisingly few of the expected issues were raised — some would have preferred more direction in the topics that were studied but collaborative working and reflective writing did not appear to be a problem. A more detailed study of the outcomes of both modules, and how they relate to gender and self-evaluation, is the subject of further work. Similarly, we plan to further investigate student opinion about being allowed freedom within these modules and what factors influenced their choices. As well as narrowing the gap for students with less experience, we hope that the broader adoption of this kind of approach could be a step towards " re-structuring the computer science field and the rules of the participation game so as to legitimise diverse forms of participation.[16] "

REFERENCES

[1] [n. d.]. Assessing reflection. https://www.ed.ac.uk/reflection/facilitators-toolkit/assessment

[2] [n. d.]. Bloom's Taxonomy: The Affective Domain. http://www.nwlink.com/~donclark/hrd/Bloom/affective_domain.html

[3] [n. d.]. Secondary accountability measures (including Progress 8 and Attainment 8). https://www.gov.uk/government/publications/progress-8-school-performance-measure

[4] [n. d.]. What do HE students study? | HESA. https://www.hesa.ac.uk/data-and-analysis/students/what-study

[5] [n. d.]. Widening participation: UK Performance Indicators 2017/18 | HESA. https://www.hesa.ac.uk/news/07-02-2019/widening-participation-tables

[6] Christine Alvarado, Zachary Dodds, and Ran Libeskind-Hadas. 2012. Increasing women's participation in computing at Harvey Mudd College. *ACM Inroads* 3, 4 (Dec. 2012), 55–64. https://doi.org/10.1145/2381083.2381100

[7] Steven Bradley. 2019. Addressing Bias to Improve Reliability in Peer Review of Programming Coursework. In *Proceedings of the 19th Koli Calling International Conference on Computing Education Research (Koli Calling '19)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3364510.3364523

[8] Steven Bradley. 2020. Creative Assessment in Programming: Diversity and Divergence. In *Proceedings of the 4th Conference on Computing Education Practice 2020 (CEP 2020)*. Association for Computing Machinery, New York, NY, USA, 1–4. https://doi.org/10.1145/3372356.3372369

[9] Sapna Cheryan, Allison Master, and Andrew N. Meltzoff. 2015. Cultural stereotypes as gatekeepers: increasing girls' interest in computer science and engineering by diversifying stereotypes. *Developmental Psychology* 6 (2015), 49.

[10] Sue Inn Chng. 2018. Incorporating reflection into computing classes: models and challenges. *Reflective Practice* 19, 3 (May 2018), 358–375. https://doi.org/10.1080/14623943.2018.1479686 Publisher: Routledge _eprint: https://doi.org/10.1080/14623943.2018.1479686.

[11] Ryan Crosby, Marie Devlin, and Lindsay Marshall. 2019. Computing Ipsative Assessment. Improving the Student Experience in Higher Education Using Personalised Assessment and Feedback. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '19)*. Association for Computing Machinery, New York, NY, USA, 300. https://doi.org/10.1145/3304221.3325565

[12] Rodrigo Duran, Jan-Mikael Rybicki, Juha Sorva, and Arto Hellas. 2019. Exploring the Value of Student Self-Evaluation in Introductory Programming. In *Proceedings of the 2019 ACM Conference on International Computing Education Research (ICER '19)*. Association for Computing Machinery, New York, NY, USA, 121–130. https://doi.org/10.1145/3291279.3339407

[13] David Foster, Nerys Roberts, and Robert Long. 2021. The Pupil Premium. (Dec. 2021). https://commonslibrary.parliament.uk/research-briefings/sn06700/

[14] Graham Gibbs. 2006. How assessment frames student learning. In *Innovative Assessment in Higher Education*. Routledge. Num Pages: 14.

[15] Melissa E. Graebner, Jeffrey A. Martin, and Philip T. Roundy. 2012. Qualitative data: Cooking without a recipe. *Strategic Organization* 10, 3 (Aug. 2012), 276–284. https://doi.org/10.1177/1476127012452821 Publisher: SAGE Publications.

[16] Maria Kallia and Quintin Cutts. 2021. Re-Examining Inequalities in Computer Science Participation from a Bourdieusian Sociological Perspective. In *Proceedings of the 17th ACM Conference on International Computing Education Research*. Association for Computing Machinery, New York, NY, USA, 379–392. https://doi.org/10.1145/3446871.3469763

[17] Eleanor Pierre and John Oughton. 2007. The Affective Domain: Undiscovered Country. *College Quarterly* 10, 4 (2007), 1–7. https://eric.ed.gov/?id=EJ813766 Publisher: Seneca College of Applied Arts and Technology.

[18] Office for Students. 2018. Monitoring outcomes: OFFA access agreements and HEFCE funding for widening access - Office for Students. https://www.officeforstudents.org.uk/publications/monitoring-outcomes-offa-access-agreements-and-hefce-funding-for-widening-access/ Archive Location: Worldwide Publisher: Office for Students.

[19] Office for Students. 2020. The extent to which providers effectively demonstrate the learning gain of their students - Office for Students. https://www.officeforstudents.org.uk/about/measures-of-our-success/experience-performance-measures/the-extent-to-which-providers-effectively-demonstrate-the-learning-gain-of-their-students/ Archive Location: Worldwide Publisher: Office for Students.

[20] Chris Wilcox and Albert Lionelle. 2018. Quantifying the Benefits of Prior Programming Experience in an Introductory Computer Science Course. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18)*. Association for Computing Machinery, New York, NY, USA, 80–85. https://doi.org/10.1145/3159450.3159480

[21] James Wolf. 2011. A reexamination of gender-based attitudes toward group projects: Evidence from the Google Online Marketing Challenge. *Computers in Human Behavior* 27, 2 (March 2011), 784–792. https://doi.org/10.1016/j.chb.2010.11.001