Computer Programming Modules in Surrey:

Teaching in Surrey for 2 years in a few modules have given me an experience of managing a limited module objectives in what is defined as a fragmented curricular approach in [1]. The importance of aligning module objectives with program objectives, curriculum topics choices, assessment choices, teaching activities and pedagogic approaches are all affected by best practice available in the literature and new technological advances and directions. This report summarizes my experience as a module leader in one module that changed twice in the two offerings I have managed, than the previous offerings. This module had pre-requisites that I will be discussing their contents. I was also co-teaching in other programming modules that I will discuss briefly below.

COM1032: Operating Systems

This module is newly offered in Surrey this semester and entirely designed by me to suit first year students in their second semester. Last year I led its development as half module with mobile computing topics. Everywhere in the UK and overseas, this module is offered in later years using C programming language mostly, and after algorithms and data structures courses as pre-requisites. It was difficult to simplify the contents to first year students, but main topics were selected, and simulations were applied in labs and coursework using Java language that the students studied in COM1027 Programming Fundamentals modules in their first semester. COM1027 used the building blocks approach [2] to introduce the Java language constructs and syntax, and is overlapping with some student high school curriculum according to [3]. This very relaxed introduction to Higher Education created the expectation by some students that overlap with high school curriculum is the normal case. The question of the first programming language is not fully understood as the reason of failure of a curriculum design if followed by other programming languages according to [4]. However, I believe C is a good beginner language, then generalised to other languages, and other programming paradigms with the theory of computation and the language compiler design to be covered before graduation. Examples are in courses such as Theory of Computation http://moodle.manalhelal.com/course/index.php?categoryid=2 and Advanced Programming Languages (http://moodle.manalhelal.com/course/index.php?categoryid=5).

In COM1032, students were given labs with clear instruction sheet and pointers to further resources to apply the Operating System functions covered during the lecture topic. The students are given the 2 contact hours assisted lab to ask questions, followed by a week before the model answer is released. The release of a form of a model solution to the required deign of the topic of the week, provided a code analysis approach as discussed in [2]. During this week, students can seek further help by sending emails or dropping in in office hours. Usually less than 10% of the students engage by asking questions for further clarifications during lecture and lab contact hours. Less than 2% usually use emails and office hours. These estimated percentages were much reduced using virtual contacts as a result to the pandemic situation. The ideal situation would have been to ask students to submit their weekly attempts for summative feedback with much less weight, but not completely formative carrying no weight. In such a case, more hours are required to provide formative feedback to identify the sources of errors and enable better reading of the model answer upon release, instead of students trying on their own to find out where their logic went wrong.

The summative assessment was based on 40% midterm course in the form of an online exam and 60% programming/report coursework. The exam style coursework was straight forward questions based on

theoretical definitions and straight forward applications of equations students have practiced in the lecture formative assessments. This led to very high scores by all students, particularly for being online at home and open book and resources. This assessment style was almost a test of their ability to search and copy and paste the answers, which is not in any of the learning objectives of higher education. The coursework was based on the full system approach described above, by assembling their understanding of the lab wok during the semester, without limitations to the content given, and with online resources collected and presented to students to learn from. The coursework was flexible enough to be addressed by students differently based on their skills and talents. A Frequently Asked Question document was published to students based on their questions, and similar questions from previous years and suggested by me to further illustrate the various paths that can be followed. A no-code approach using the code analysis approach described above was considered acceptable due to the pandemic circumstances and due to students being spoiled with simplicity in their first semester. This baseline would score the minimum 40% if good explanation of the algorithms with good 2 test cases presented in the report. Every new idea was rewarded by higher marks, and three major bonus requirements were included. Every submission was totally different than the other. The effort in each submission clearly showed who did the labs, who read the lecture slides, who read more from the books and other resources made available to the students, and those on the higher end who are enthusiastic to self-learn new technologies beyond the requirements of the module.

I have three sets of results that I managed to collect during this short period of time to use in measuring the effectiveness of a teaching session. (UKPSF – K5). The first is the students' outcome to the pedagogic approach applied on them, the second is the MEQ survey responses managed by the university which is mostly subjective and not tailored to module content nor pedagogic approaches, and the third is a tailored survey responses designed by me to be objective and informative regarding module content and pedagogic approaches. The first dataset is more rich in term of student number, 146/176 student in the cohort needed to work to achieve their marks. The motivation of the marks in their certificates makes people contribute because of the dual benefits. The optional MEQ response rate is 41/176, which is slightly better than the optional tailored survey which is 11/176.

Table 1: COM1032 Student Outcome per Rubric Criteria

		Percentage of students per Level				
	Level Performance	**Report Discussion Only**	**Attempted Programming with Errors**	**Attempted Programming with Logical Errors**	**Correct Programming**	**Genuine Design**
	Rubric Criteria	**Level E**	**Level D**	**Level C**	**Level B**	**Level A**
Deep: Programmin	**Booting and the Initial Process - 5**	24	13	0	3	60

g Skills and theoretical fundamentals 75%	**Process Scheduler - 25**	10	21	1	16	52
	MMU – 25	55	12	6	14	13
	File System – 25	49	28	10	12	1
	I/O System – 25	62	22	13	3	0
	Multiprocessing – 10	44	19	8	3	26
	Synchronisation and Deadlock Prevention – 10	54	24	16	3	3
	GUI – 10	61	10	17	6	6
Surface: Technical Writing Skills 25%	**Report: Compiling & Running - 3**	5	3	23	48	21
	Report: User Manual - 5	3	6	17	57	17
	Report: Technical Manual - 7	6	5	34	46	9
	Report: Architecture & Instruction Set - 10	24	12	28	28	8
Averages		33.0833333	14.5833333	14.4166667	19.9166667	18

Table 1 summarises student percentages who scored the level in the columns per the rubric criteria in the rows. The first eight rows require programming and evaluated by reviewing the source code submitted. However Level D was considered suitable when the required criteria is well explained in the report with both good explanations of the algorithms and two test cases per criteria. The shading are of grey level representing difficulty (the darker the more difficult). The total marks per criteria is shown in the second column. The 4 subsystems each carrying 25 marks are ordered in the delivery order, in which only 2 subsystems are required and another two if done will be awarded bonus marks. Only Processes Scheduling is covered before going remote, and the rest are after going remote. Figure 1 shows Level A in blue shade at the right side of each criteria bar.

Effects of going remote: It is obvious that much more students, particularly 52%, did the process scheduling in Level A because it was covered before going remote. This is compared to only 13% in Level A for the MMU (Memory Management Unit) subsystem, 1% for the File system and 0% for the I/O subsystem, which were covered after going remote. Multiprocessing was covered before going remote (26% in Level A), but Synchronisation and deadlock after going remote and is more difficult (3% in Level A). GUI (Graphical User Interface) is also an optional bonus, but usually is enjoyed by students, and was done in varying levels as shown in the table, in which level E corresponds to just basic graphical interface, and Level A for charts and visual performance analysis.

Programming vs Technical Writing Skills: The following 4 rows in Table 1 are for the report writing skills, this shows the explanation of the algorithms, the test cases for the implemented code, and the theoretical understanding of the suitability to different architecture and instruction set. Figure 1 again shows that Level B (yellow) is more dominant in the report writing skills, which is very good for first year students, while Level E (Light blue) is more dominant for the programming criteria.

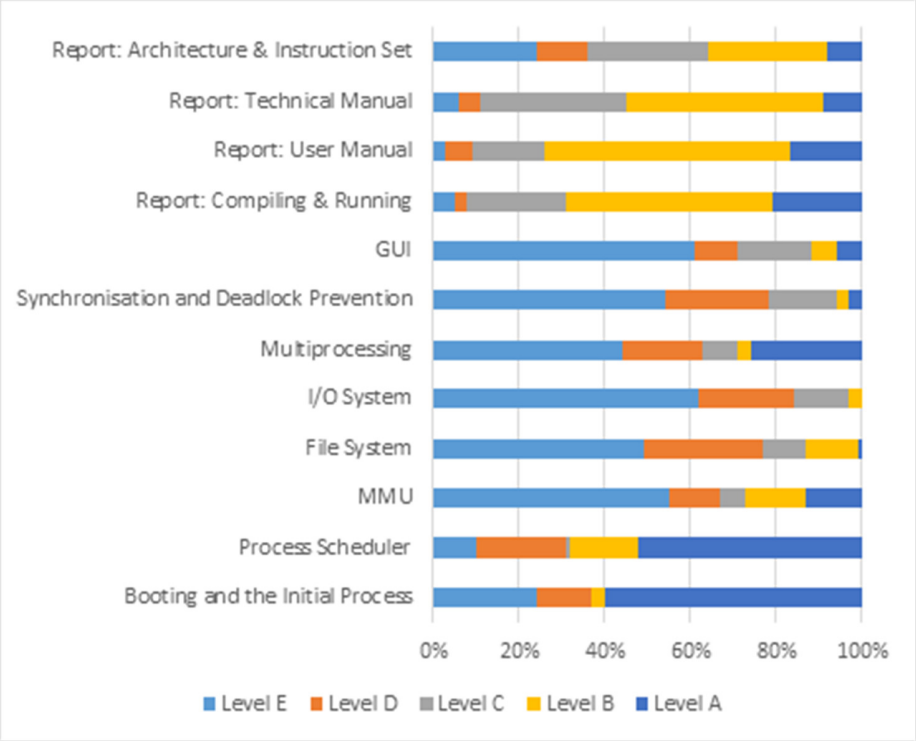

Figure 1: COM1032 Student Outcome for developing Full System in the final Coursework

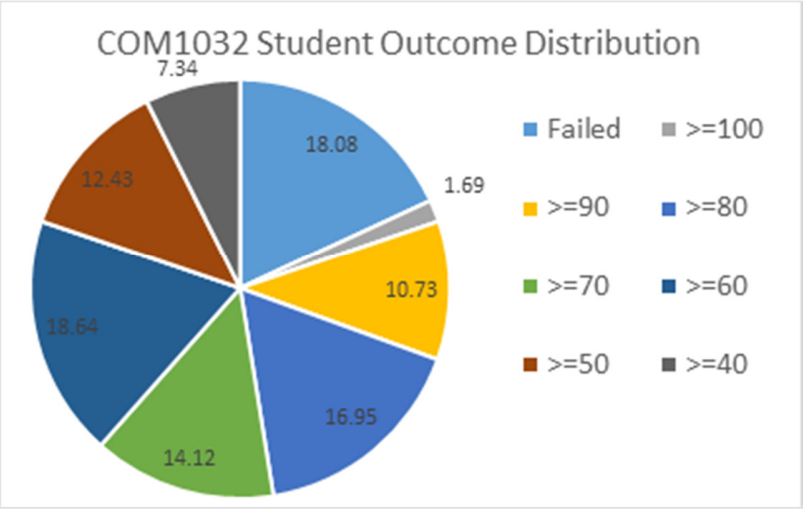

Figure 2: COM1032 Student Outcome Distribution

The last row in Table 1 lists averages per Level. This shows that in this cohort there were 47.6 % students below average, 14.4% average students, and 38% students above average. This will be further discussed in the conclusion section. Figure 2 provides more insights by including the open book midterm exam that was much easier. It shows that 18% failed the module or applied for ECs that when added to the below average (40 to 50%), which is 7.34%, gives 25.34% as below average in this cohort. Average students that scored 50 to 70% are 31.07%, while above average student that scored higher than 70% to be 43.5%.

The MEQ responses are anonymous and included in the accompanying Results Excel Sheet. The scored questions are presented in Table 1. The table shows the averages of the 1 to 5 score, in which 1 is the worst and 5 is the best, across the 41 responses received. The questions are divided into Teacher Support, Learning Experience & Pedagogy (mostly university services), Module Design, and Assessment & Feedback. Although only the first section is subjective about the module leader, all other questions, particularly the free form ones, were used to address the teacher support more than anything else. The results are presented in a format that does not allow tracking the responses of one student to all questions. So what is the best and the worst aspect of the module cannot be linked together to identify the invalid responses, or students who are not engaging. Since the module showed 47.6% of students were below average, then up to 70 students could have used the MEQ to show that their lack of engagement was not their responsibility. Asking first year student about their opinion on module organisation and content is very difficult for them, as they are not comparing with learning objectives, other universities' offerings, nor career long benefits. The overall comments received are almost impossible to consider in perfecting future offerings for not being realistic and not considered constructive criticism.

Table 2: COM1032 Surrey University MEQ Responses for the year 19/20

Question	**Average**	**% +ve**
Teacher Support		
If students had questions or needed assistance, this lecturer generally responded helpfully	3	33

The lecturer was enthusiastic about teaching their subject	2.9	31
The lecturer had an approach to teaching which engaged my interest and motivation	2.1	4
The lecturer explained things clearly	2.2	12
I have received useful advice and guidance in relation to this module	2.5	24
Learning Experience and Pedagogy		
I have found the module intellectually challenging and stimulating	3.4	61
I have gained valuable new knowledge, skills and/or understanding from this module	3	44
Module Design		
The module was well-organised and ran smoothly from a practical point of view (e.g. timetabling, communications)	2.5	24
My learning was well supported by the use of technology such as SurreyLearn, Panopto, electronic voting and/or online resources and activities	3.3	49
Overall, the content and delivery of this module helped motivate me to work hard	2.1	10
It has been straightforward to access the learning materials and other resources needed to support my learning on this module	3.2	51
Assessment and Feedback		
Feedback on my work has been timely (i.e. provided in time to help me with subsequent learning/assessments)	3.2	43
I have found feedback on my work useful	2.7	31

One online tailored survey was conducted during the difficult time of the pandemic, using a sample from COM1032 module cohort. Only 11 student responded and did not answer all questions. This is considered to be statistically insignificant. The details of the questions and the answers are in the accompanying Results Excel Sheet. A number of incoherent replies indicate that the sample responded to the survey do not include the excellent students in the module, particularly 3 out of them are not engaging, and some contradictory answers are highlighted in the accompanying results excel sheet. A general requirement in survey analysis is to eliminate invalid responses and there are various studies about identifying invalid responses and how to handle them such as [5], [6]. In this survey, eliminating any response is not even possible because of the limited responses, but should be considered as a factor that reduces the quality of the analysis.

The tailored survey was analysed using a pivot table, and one student responses are in one row, for the questions in the columns. The objective was to draw students' attention to their own learning and improvements more than using their limited experience in judging teacher's personality nor knowledge contents for the obvious lack of experience, distraction by various sources of bias [7], and irrelevance.

The survey was designed as a preliminary step to gather some information to design student interviews after that. This is why it was more tailored towards the COM1032 approaches, and could have been generalised if more time and effort in the proposal writing were done before the survey design. Experience gained while analysing the responses revealed that the vocabulary used could have been more precise and the sequence of the questions could have been better for the analysis. The remote learning and general mood after the pandemic did not allow the completion of the remaining steps.

The responses to a question on Pedagogic approach preferred by the students are captured in Figure 2. It show that 63.64% of the students prefer Building Blocks approach while being given the design. This is the approach applied on them in the first semester and they are more familiar with it now. There are 4/11 (36.36%) students feeling proud for doing the full system approach followed in the coursework of COM1032, 2 of them are those who preferred the building blocks given the design, and the other 2 preferred the building blocks approach while not given the design. The 18% who prefer Code analysis they prefer also the career paths that do not require writing code.

Figure 3 illustrates student preferences about the assessment type that is most suitable to their study pattern. 63.64% of the students prefer programming course-works. This is justified by another answer to the questions "What makes you feel successful in this module", to be "Building a solid OS simulator that actually works was a great way of implementing everything I was taught in this module. In some areas, such as memory management, I thought my understanding was good, until I went to implement my own MMU, I realised that I had a lot of holes in my knowledge."

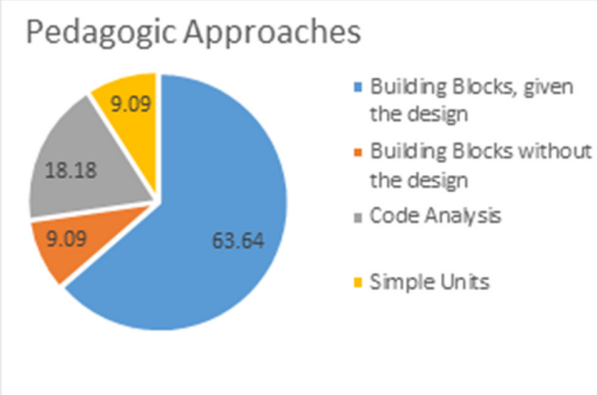

Figure 3: Student Pedagogic Approaches Preferences

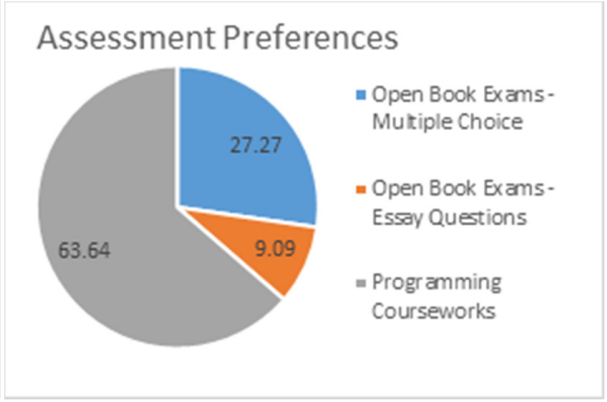

Figure 4: Student Assessment Types Preferences

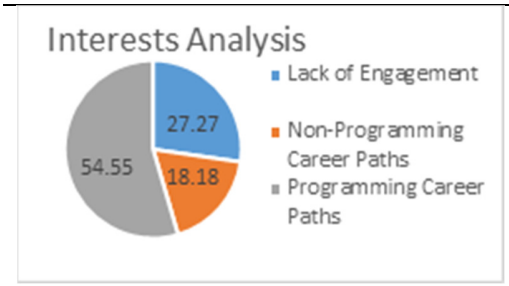

Figure 5: Students Interests analysed from the responses

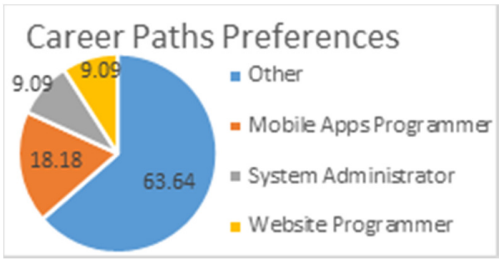

Figure 6: Students Career Paths Preferences

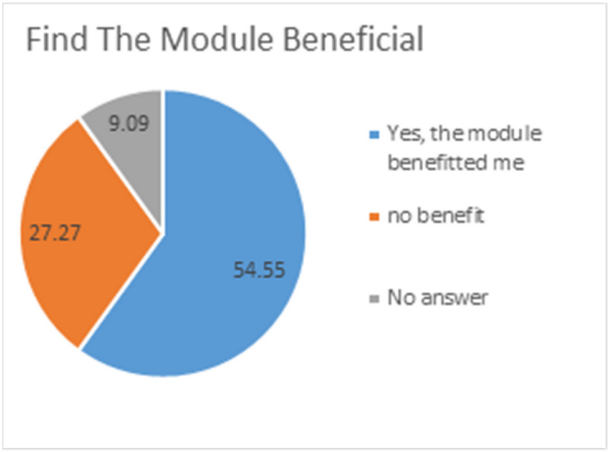

Figure 7: Student Responses about how the modules was useful as delivered

Figure 4 illustrates how I analysed the questions about how the student evaluate their current strengths in order of strength, and where they can improve in order of weakness. This was interpreted as how do they see themselves currently fit to career paths suitable to programming skills. Linking this to the questions about job titles interesting to them as illustrated in Figure 5, it is obvious that the choice of 'other' is chosen by those who lack engagement, or are not interested in developing their programming skills further. Since the module was focused on programming skills at the various levels, attempting to find out if the module as delivered is considered beneficial to the students, their responses are captured in Figure 6, showing 54.55% students finding the module benefitting them. These contradictory findings show unthoughtful some of the responses are. Other inconsistent responses about most and least interesting weeks, and repeated as most and least interesting topic, some unthoughtful answers were given about week numbers that do not correspond to the week topic. Although the questions were focused on the module content and delivery pedagogic approaches, 2 answers tried to move the focus to the voice of the module leader and totally irrelevant and very subjective comments. Table 3 lists statistics about other questions in the survey. 81.81% of the students felt the pandemic event affected their performance. 63.63% of the students did not find the remote learning is suitable to their study pattern. Most of them explained that this is due to home environment as the main distraction, while lecture theatre atmosphere is more motivating. 54.54% of the students felt the remote learning methods applied in the module are not suitable without suggesting what they think could have helped or already applied by other modules.

Table 3: Other questions in the Tailored Survey

	Answer		
Question	Yes	No	Partially
Student feel affected by the Pandemic	81.81	18.18	

Remote Learning Suitability	27.27	63.63	9.09
Remote Learning in COM1032 methods are adequate	18.18	54.54	27.27

COM2031: Advanced Algorithms

This module applies the use of a programming language to practice problem solving approaches to fundamental problems from the literature. The module in Surrey is delivered using 2 hours lecture to cover the theory with practicals only in the 2 hours lab per week. The formative assessments are given in the labs for the students who ask for it, and there is no way the teaching staff can measure how well they performed in these labs. The summative assessments are by a coursework worth 20% that require theoretical tracing of the algorithms by paper and pen or programmatically optionally by the student, then an 80% final exam to cover the theory again. The assessment style does not assess how well the students did in programming and code writing skills. The pre-requisite module of COM1029 Data Structures assess students on 2 exams only with lab work go unassessed as well.

Conclusion:

Student outcomes are not only affected by the pedagogic method applied to them. From the GCLT modules 1 and 2, it is obvious that all learning theories explain aspects of psychological, sociological, economical, historical, and philosophical variance between people that affect both their learning rate and acceptance/rejection of new knowledge (UKPSF – K3). In the following subsections, some the obvious factors that affected my teaching experience in Surrey and I believe did affect the students' outcomes in a hard to measure magnitude are discussed.

Limited Educational Teams and bad staff to student Ratio and Staff motivation:

Since education has become a paid service, it has become a business that target lowering the cost and maximising the revenues by enrolling more students. The cost of education come from teaching staff salaries and lab equipment's and venues maintenance and expansions among many other expenditure types. Lowering the cost by maintaining short term employments for teaching staff with no career paths have been an objective by many departments, which makes only early career staff teaching and no one stays long enough to develop experience in fear of higher costs. The consumable human resource tend to choose the easy options in all the education process from the curriculum design to the marking load because of the demotivation of their short term employments. Even those who maintain internal motivations, very limited hours remain to them to address the increasing number of students with their required individual communications. (UKPSF – A1 & A5).

Teaching staff require retention to have motivation. Teaching staff are maintained as a consumable resource in some universities to reduce the cost. Many HE staff considering job security a main demotivation to their efforts. Watching their hard work repeated by other people later on for the copyright ownership by the university, motivate any teaching staff do the least effort to update any curriculum. Frailty in HE education is discussed in depth in [8].

Motivation from the student side:

Learning need motivation. Students require career paths and examples that manifest the importance of being good programmers. As some career paths do exist for non-coding paths, and students are being taught by some teaching staff who followed these paths already, the importance of this essential skill was reduced to the enthusiasts with many module leaders not assessing this skill at all, or giving it the least weight of marks. Further studies should monitor the skills demand in the job market such as the ranking maintained in [9] showing job advertising count per skill. The historical change over time of the skills demands identify technological trends to update the curriculum with. The current statistics shows that the no-coding career paths such as quality assurance and project management skills, have fewer posts advertised because a few people can do it. On the other hand, software development skills are found in many advertising and more people are needed to do it, creating one form of the computer science crisis exists (the demand is higher than the supply) [10].

Student levels across a wide spectrum:

From the results in COM1032 as a sample from the population, it is obvious that almost one teaching staff is dealing with 176 students across a wide spectrum. As mentioned earlier, there were 47.6% students below average, 14.4% average students, and 38% students above average. A general guideline in Exeter university [11] require 40% of as assessment requirement to be straight forward addressing average students, 10% can be answered adequately by potential first-class students only, and 50% intermediate requirements addressing intermediate level students. While all care was given to apply this balance, a single requirement targeting the above average led to a wave of complains from the below average and average students requesting that 100% of the assessment being tailored to them. On the other hand, a no code route to the final coursework was meant to guarantee 40% mark to the average students, were considered offensive to the above average students and one student complained. (UKPSF – V1, V2). This leads to the next point of managing students expectations and manage complains about the variance in the level of assessments. On the teaching level, using technology to address the different learning rates and levels of students is always suggested in the literature including the work in [12].

Manage student expectations:

Students are now paying for their education and require being treated as customers. This created pressure on teaching staff to apply a kind of customer oriented teaching approaches, in which the customer is the student, ignoring the lack of experience in this early age. To maintain that the society and future employers are the main customers of the education services, students need to be introduced to programming on the same level it is required in companies and research projects from day one in Higher Education (UKPSF – V4). The undergraduate education years should be considered the journey from where they started as novices to where they should be upon graduation as experts. This journey is explained from the psychological transformations perspective as explained in [13] with the feature of every stage Detailed coding experiences are required fundamentals to enable learning independently the new technologies required in the job market as they evolve such as low-code development and integration platforms, for example Microsoft Power Platform. This message need to be reiterated in every module and practiced through assessments for formative and summative feedback.

Curriculum Choices:

Curriculum in computer science is hierarchical by nature. The whole program need to be connected to provide a complete set of skills to enable students for career long self-development. Examples of best

practice is presented by organisations such as IEEE curriculum committee in [14]. Not all universities publish their curriculum online, however many universities do, and many online resources are available such as Coursera. Many HE textbook publishers list all books available under each module topic. All approaches to teaching a topic probably do exist as a complete package of a book, lecture slides with illustrations, lab practical, and even test banks. Finding a new approach to teach a topic, is usually worth publishing a new book, which is hard work of teams working together. These new approaches develop after experience of teaching the same module using existing approaches first. Most reputable universities worldwide follow the details of the most reputable textbook in the module topic that links well to the program objectives and other modules as pre-requisite and future dependent modules. These books mostly cover the best practice for each possible teaching approach and sequence using best examples and practices. Competition is almost impossible with well-established books with several editions over which various errata have been addressed, new technologies are always incorporated in timely manner, and edited by a group of authors with many teaching assistants, editors, and graphic designers. Single handed academics fear innovating new curriculum requirements or teaching activities for possible objections to the validity of their propositions as opposed to existing curriculum. They usually adopt these textbooks for all educational activities, unless exam questions have been exhausted and the need to design new exams or coursework are required. (UKPSF – K1, K6).

Maintaining a curriculum that covers both fundamentals that could last for long decades after graduation, using recent technologies that enable students for the careers of relevance, require all teaching staff in the department to work together to ensure pre-requisites are met, and program objectives are well distributed over all modules in a coherent way. Lack of communication and career instability are usually the main reasons why modules are treated independently with less connections than should be. (UKPSF – K1, A1, A5)

Assessment Styles Choices:

The choice of assessing student outcomes using exams are easier than programming coursework in terms of design requirements and marking load. Some accreditation requirements require closed examination environment to ensure students are solving independently, while coursework are difficult to prove who did them. Plagiarism tools only give a measure of how close any work to another published work, but in undergraduate education, similarity is expected to be high and should be alright as long as due credits are given to the original authors, and reports show understanding depth. Course-works that require a fixed answer that is either correct or wrong are much closer to exam styles and much easier to mark, however tend to measure very little on student understanding and ability to apply the learned concepts using innovative approaches and recent technologies. Course-works that can be applied differently by each student, encouraging the employment of recent technologies require more time to evaluate, and probably learn new technologies by the assessor to keep up with student independent learning. However, these are worth the hard work by both staff and students, because they are true indicators of career long success. (UKPSF – A3).

Student Outcome and Survey Responses:

Student learning approaches (surface vs. deep) are correlated with the valence of their emotional experience (i.e. negative vs. positive), such that students experiencing positive emotions (i.e. hope and pride) are more likely to adopt a deeper approach to learning and achieve higher performance outcomes [15]. Attempting to eliminate personal student bias to evaluate the pedagogic approach used in isolation

of interfering bias is very difficult to achieve. The work in [16] analyses the proper metric to use to evaluate a module offering is the student outcome metrics (how successful they have become in the knowledge content by linking assessments to learning objectives). The book also studies the kind of questions to include in the survey to evaluate students' opinions where it can help in shaping future offering of the module to perform better, and includes example of effective surveys. The work in [17] discusses the bias against women and non-English speaking backgrounds found in student survey data. The choice of the question objective and vocabulary identify the subjectivity in the survey that can target minorities and protected characteristics of the teaching staff rather than the quality of the work offered. (UKPSF – A5, K5).

Complains about the level of assessments by weak students can be reduced by managing their expectations as mentioned earlier. Creating a motivation from the mark distribution and explaining the rationale behind the weight every requirement carries, can reduce potential genuine complains. It is insightful to link the anonymous survey responses to the student outcome average and include this in an anonymous way. This can explain the motivations of the responses and reduce the interpretation from the several questions as attempted in the tailored survey in this work.

The effects of the Pandemic:

From the results above, it is obvious that the pandemic caused both damage to the motivation, and to the communication styles. Although in computer science everything can be done remotely easily, some students found an opportunity in the overall sentiment to be a reason not to engage regardless of the available means. A few students remained well engaged until the last week, and many submitted more than excellent coursework as shown in the results section.

The remote learning created an independent learning style where motivation becomes the only drive. The safety net and other marking concessions made the fear of failure not a pressing motivation in this environment. The feedback and the interaction as main factors in learning were labelled not appropriate by some students, which should be acceptable to be different or less in any remote learning environments as compared to face to face environments. However, technological communication enables virtual office hours and face to face communications with screen sharing. Only a few students attended for these meetings. This lack of engagement was almost the same as before the pandemic event. This leaves the examination style environment to be the only affected factor in remote learning. The closed examination procedures guarantee that the student is answering with his/her genuine understanding accumulated during the semester much better than any coursework submission, which could be assembled work from various resources without much understanding. This can be guaranteed by student presentations to personally explain their genuine course-work submissions.

Bibliography

[1] W. G. Wraga, "Toward a Connected Core Curriculum," *Educ. Horiz.*, vol. 87, no. 2, pp. 88–96, 2009.

[2] C. Selby, "Four approaches to teaching programming," in *Learning, Media and Technology: a doctoral research conference*, London, UK, Jul. 2011, p. 10.

[3] "Computer Science: A curriculum for schools," *Comput. Sci.*, p. 28, 2012.

[4] M. Kolling, "The Pitfalls of Java as a First Programming Language – A Response," Jan. 14, 2008. https://blogs.kcl.ac.uk/proged/2008/01/14/the-pitfalls-of-java-as-a-first-programming-language-a-response/.

[5] J. Coste, L. Quinquis, E. Audureau, and J. Pouchot, “Non response, incomplete and inconsistent responses to self-administered health-related quality of life measures in the general population: patterns, determinants and impact on the validity of estimates. A population-based study in France using the MOS SF-36,” *Health Qual. Life Outcomes*, vol. 11, no. 1, p. 44, 2013, doi: 10.1186/1477-7525-11-44.
[6] “How to Identify and Handle Invalid Responses to Online Surveys,” *CloudResearch*. https://www.cloudresearch.com/resources/guides/ultimate-guide-to-survey-data-quality/how-to-identify-handle-invalid-survey-responses/.
[7] A. Grimes, D. Medway, A. Foos, and A. Goatman, “Impact bias in student evaluations of higher education,” *Stud. High. Educ.*, vol. 42, no. 6, pp. 945–962, Jun. 2017, doi: 10.1080/03075079.2015.1071345.
[8] I. M. Kinchin and N. Winstone, *Pedagogic frailty and resilience in the university.* ROTTERDAM: SENSE, 2017.
[9] “IT Jobs Watch • Tracking the IT Job Market.” https://www.itjobswatch.co.uk/ (accessed Jul. 14, 2020).
[10] H. S. Nwana, “Is Computer Science Education in Crisis?,” *University of Cambridge*.
[11] “Examination Procedures - University of Exeter.” https://newton.ex.ac.uk/handbook/PHY/ExamProcedures.html.
[12] S. Al-Imamy, J. Alizadeh, and M. A. Nour, “On the Development of a Programming Teaching Tool: The Effect of Teaching by Templates on the Learning Process,” *J. Inf. Technol. Educ. Res.*, vol. 5, pp. 271–283, 2006, doi: 10.28945/247.
[13] L. E. Winslow, “Programming pedagogy---a psychological overview,” *ACM SIGCSE Bull.*, vol. 28, no. 3, pp. 17–22, Sep. 1996, doi: 10.1145/234867.234872.
[14] Joint Task Group on Computer Engineering Curricula, *Computer Engineering Curricula 2016 - CE2016. Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering*. Association for Computing Machinery (ACM) IEEE Computer Society, 2016.
[15] K. Trigwell, R. A. Ellis, and F. Han, “Relations between students’ approaches to learning, experienced emotions and outcomes of learning,” *Stud. High. Educ.*, vol. 37, no. 7, pp. 811–824, Nov. 2012, doi: 10.1080/03075079.2010.549220.
[16] M. A. Fox, N. Hackerman, and National Research Council (U.S.), Eds., *Evaluating and improving undergraduate teaching in science, technology, engineering, and mathematics*. Washington, D.C: National Academy Press, 2003.
[17] Y. Fan *et al.*, “Gender and cultural bias in student evaluations: Why representation matters,” *PLOS ONE*, vol. 14, no. 2, p. e0209749, Feb. 2019, doi: 10.1371/journal.pone.0209749.