BOLSTERING DEEP LEARNING WITH METHODS AND PLATFORMS FOR TEACHING PROGRAMMING

[a]MÁRK CSÓKA, [B]DÁVID PAKSI, [C]KRISZTINA CZAKÓOVÁ

J. Selye University, Bratislavská cesta 3322, 94501, Komárno, Slovakia
email: [a]csokam@ujs.sk, [b]paksid@ujs.sk, [c]czakoovak@ujs.sk

The paper was supported by project KEGA 013TTU-4/2021 "Interactive animation and simulation models in for deep learning".

Abstract: After decades of effort informatics and programming are part of the high school curriculum in Slovakia. The demand and popularity of IT impacts the education and continuously conquers space. Nowadays informatics is present from primary schools in the examined region. Although the students are more exposed to IT compared to previous generations learning programming offers numerous challenges for the beginners. Our goal is to streamline and deepen the learning process by evaluating the learning habits and observe the usage of a newly implemented platform. To improve the understanding of the currently prioritised learning styles we combined the advantages of survey with more direct interview methods. Based on the multilateral data collected from students' and teachers' perspective we provide methodological recommendations on improving the programming education to facilitate deep learning.

Keywords: deep learning, Jupyter Notebook, JupyterHub, notetaking, computer science, programming

1 Introduction

With the introduction of Information Technology (IT) as a school subject, Slovakia faced new challenges. The integration of IT to the educational curriculum was welcomed by students and teachers alike because of the rapid development of technology the basic computer skills turned into expected ones. Despite the positive reception teaching and learning IT, especially programming, has its own challenges. The content of the IT subject is constantly changing which makes suitable teaching materials hard to find. In addition, not all teaching methods seem suitable for this subject. IT is an umbrella term which can cover everything from basic computer skills to programming in a specific language. The study deals with the challenges of teaching and learning programming from beginner level, introduces an old new platform in the university environment and presents the results based on the data collected. The remainder of the paper is organised as follows: Section 2 introduces the implemented research methods and local situation. Section 3 outlines the desired learning outcomes and details the Jupyter ecosystem. Finally, Section 4 contains the results of the qualitative and quantitative data collected.

2 Methodology

The primary tool of the research was a questionnaire consisting of 21 questions which included 3 main sections besides the introductory demographic questions. The sections provided insights to students' learning habits in general, the involvement of notetaking throughout learning programming and finally their opinion towards the newly implemented JupyterHub service. The questionnaire was limited to the attendees of informatics-related studies and the responses were collected during summer and winter semester of 2022. The summer semester granted 50, while the latter 55 valid responses. The questionnaires ended with a voluntary prompt if the respondent would have liked to participate in a more detailed interview regarding the same topic. The mentioned interview was carried out with student and teacher participants after the evaluation of quantitative data.

3 The implemented teaching workflow

The content of informatics (or subjects dealing with informatics) taught in the schools of the examined regions is constantly changing and expanding. (Paksi, Csóka, Annuš 2022) In the last decade, teaching of programming received a lot of attention, computers became essential parts of our civilization and are present nearly everywhere. In order to keep up with the demand and the rapid expansion of technology the field of information technology gained its own school subject version. One way or another it is present in the teaching materials from primary school age. (Végh, Takáč 2014) Depending on the region, knowledge of a descriptive programming language is becoming more common while high-level programming language is already expected at secondary school level. These are accompanied by a relatively low number of lessons, on average 1 lesson per week. (Paksi, Csóka, Annuš 2022) Due to these restrictions and ambitious objectives, there is great demand for effective teaching of the curriculum which can facilitate fast learning and deep understanding. If the mentioned goals are not fulfilled, the teaching process fails. The small number of lessons combined with the high amount of educational material in the field of programming often peaks in superficial knowledge or completely omitted topics. (Willis, Charlton, Hirst 2020) The problem is also present at our university. The Applied Informatics study programme shows a high number of attritions which got even worse when the students who graduated during the pandemic began their university studies. Our goal is to explore the root causes and reduce the student non-continuation rate by improving the learning process with tools and methods. (Czakóová, Stoffová 2020) In order to overcome the obstacles of programming there are a lot of teaching methods and approaches available, but not all of them are suitable for the above-mentioned requirements. (Végh, Takáč 2016), (Ilter 2017) The cornerstones of the solution we developed for teaching programming are based on the deep learning teaching approach and the Jupyter note-taking platform (Sáiz-Manzanares, Consuelo, García, César, Díez-Pastor, Martín, Luis 2019). To evaluate if the solution is applicable and effective, first let us look at the pillars.

3.1 Deep learning with notetaking

Almost five decades ago Marton and Säljö discovered the two different learning processes. (Marton, Säljö 1976) Surface learning refers to rote learning and memorising the text. Deep learning refers to meaningful learning and to understand the text's meaning and significance. (Choithram, Suwimon, Nonglak 2014)

Understanding and thinking are the basis of a deep learning method. It is defined as a significant understanding of the basic content waiting to be mastered, accompanied by critical thinking and the ability to solve problems. These core competencies are joined by collaboration, communication, and the ability to control one's own learning. The possession of positive beliefs and attitudes about oneself can motivate continuous learning. (Paksi, Csóka, Annuš 2022), (Czakóová 2020)

The essence of the approach is the in-depth study of a given topic. An excellent tool for encouraging the favoured behaviour is a notetaking platform where the users can take notes, run, and experiment with program codes at the same time. This is supported by the (Dong 2021) study, where it was discovered that the data scientists who used notebooks kept continuously adding more and more lines of code to their notes, thus they were able to experiment and propose alternative solutions. In the beginning similar platforms were developed for professional usage, for example in the field of data science. (Zhong, Wei, Yao, Deng, Wang, Tong 2020) The arrival of computational notebooks and their functions satisfied demand in the industry. Nowadays such products are more widely known and thanks to their favoured properties the inclusion to education gradually began. (Asikainen, Gijbels 2017), (Czakóová, Udvaros 2021)

Bruner's Spiral curriculum also fits to the mentioned concepts and to the nature of programming. Spiral curriculum is a design where the key concepts are not just presented, but often reintroduced throughout the curriculum. Every reappearance of the familiar concepts gradually increases difficulty. (Bruner, 1960)

3.2 The Jupyter Ecosystem

Project Jupyter started as the successor of the notebook interface parts of the original IPython (Interactive Python) platform. IPython was originally developed as a command shell for interactive computing but over the years as the project got more attention, the original developer decided to move some functionality under a new name thus creating a clearer distinction among the solutions. (Csóka 2021)

Jupyter Notebook is a web-based interactive computational environment and as the name suggests it is for creating programming-oriented documents. It is capable of running codes, displaying visualisations and handling markdown text in one place. To achieve this, they use their own .ipynb file type, which nowadays is supported by many popular IDEs (integrated developer environments). The application supports Python, Julia, R languages out of the box, but the functionality can be further expanded by installing additional kernels (programming languages), which we previously prepared and tested. (Paksi, Csóka 2022)

JupyterLab includes all functionality of Jupyter Notebook, adds a modular interface to create experience similar to IDEs and the possibility to install extensions. Furthermore, added features like better handling of .csv files and other improvements make it favourable. (Lee, Lan, Hamman, Hendricks 2008)

The idea of visualisation and interactivity in education is not new at all. It helps to interpret complex information and find relations between data. (Svitek, Annuš, Filip 2020) (Czakóová 2019)

JupyterHub is a server-side, centralised solution usually running on high performance resources. Table 1 compares key features of selected platforms It is intended to deliver multiple clients the same environments and kernels set up by the hosting institute, or organisation. From users' perspective the process is drastically simplified, their only task is to open the institute's JupyterHub website and log in. Upon successful authentication the users land on the familiar Jupyter Notebook or JupyterLab menu which was set up by the provider of the server. The operation and maintenance of such service requires advanced skills. The appointed people with necessary experience are capable of finetuning most aspects of the service, installing the necessary extensions and kernels globally or individually. (Siegel 2018)

Table 1 - Comparison of available programming platforms

	Visual Studio Code	JupyterLab	Google Colaboratory	JupyterHub
Multiple clients	×	×	✓	✓
Centralised	×	×	✓	✓
Multiple programming languages	✓	✓	×	✓
Automatic grading	×	✓	×	✓
High customisability	✓	✓	×	✓

In our region it is customary and expected that the students are going through summative assessments where their performance and work are expressed in grades. From teachers' perspective there is a constantly growing demand to make and improve the reliability, validity, and speed of the evaluating process. Regarding assessment of programming there are two main groups present in the region. The representatives of the traditional method are assessing the students by solving the practical programming test with pen and paper. This approach helps to pinpoint knowledge gaps during the correction process which can be supplemented with comments. Secondly, it excludes a huge number of cheating or helping opportunities otherwise offered by modern development environments (IDE) and computers in general. On the other hand, representatives of the modern methods prefer to conduct the process digitally. From handing out the problems waiting to be solved to publishing the test results. In this case it is more difficult to attach comments to mistakes for every individual and ensure the desired pedagogical effect of the feedback. Another often missed opportunity by teachers is the omission of automatic test evaluation with the help of applications. In contrast to other subjects the nature of programming assessments made automatization more difficult but nowadays multiple solutions are available. Jupyter has developed an indirect solution to handle assessments. The platform by default does not include functions regarding assigning, collecting, and evaluating tasks. However, as the programming-oriented document editing environments began to appear in classrooms the demand for digital assessment solutions started to grow. Nowadays, there are many plugins, such as the nbgrader which we also tested (Paksi, Csóka 2022). In addition, we could also mention the Web-CAT, CourseMaker (Manzoor, Naik, Shaffer, North, Edwards 2020) and the UNCode Notebook (González-Carrillo, Restrepo-Calle, Ramírez-Echeverry, González 2021). Each of these tools approaches the problem differently and from a different angle, but they all have huge potential.

3.3 Local situation

Despite the soon two-decade-long history of the Project Jupyter the investigated university put it into operation experimentally just 2 years ago. JupyterLab and Jupyter Notebook solutions were familiar, but the on-premises feature of the JupyterHub made it favourable versus the simpler solutions (e.g.: installing a local JupyterLab client on every classroom computer). A similar online solution worthy of mentioning could be the Colaboratory by Google. However, the list of supported programming languages is narrowly limited and focuses mainly on Python. Our goal was to customise the platform in such a way that best serves the institute. The main reference used to accomplish the designated objectives was the Applied Informatics study programme, since it overlaps the rest of the programming-related training. At our university teaching takes place in many programming languages, such as Python, C/C++, C#, Assembly, MATLAB, etc. With JupyterHub the implementation of all the mentioned languages is solved in one place. By expanding JupyterHub with additional kernels we managed to offer a central platform capable of running the required languages, store files and enable the possibility of writing organised notes for programming. To further increase the convenience the users can login with their credentials provided by the university. Last, but not least it is important to point out that the long-term storage of files, lesson notes and source codes can be solved through this platform. Previously used solutions for file management by students included private cloud drives, self-addressed emails, external data storage devices, and other.

4 Results

The survey was concluded with the participation of 105 university students who attend either applied informatics or pedagogy programmes with an informatics major. The distribution of genders shows that these programmes appeal more to male students. The exact results are 89% in favour of male students. Although the questionnaire was shared with all the students of the mentioned studies, freshmen proved to be the keenest while a similar number of answers were collected from second and third years (Figure 1).

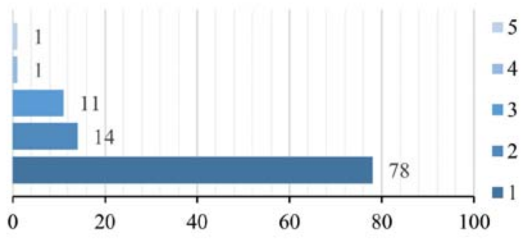

Figure 1 - Distribution of respondents by study level

The questionnaire focused on acquiring data regarding students' notetaking habits and the usage of the JupyterHub platform. At the time of the survey the platform was available for a year. Higher classes with more demanding, practical tasks could benefit less from the functionality of Jupyter. This caused a visible division regarding the platform's assessment. Next, we would like to highlight the cross results of two questions, *"Do you take notes during practical programming lessons?"* and *"Do you recommend Jupyter?"*. Figure 2 can be interpreted as follows, the left side houses the negative answers regarding the first question, right the affirmation while the top side holds the positive answers for the second. Since the adoption of the platform is considered recent, we first grouped the respondents if they had used JupyterHub beforehand. The orange-coloured bubbles on the horizontal (X) axis show the respondents who did not use JupyterHub at all, therefore their notetaking habits were evaluated during the survey. Nearly 45% of the students stated that they had no previous experience with the program itself. However, the dangerous part appears on the left side of the figure, which represents the students who do not take notes by any means. As Figure 2 shows, more than half of the respondents stated that they do not take notes during programming themed lectures and neither take notes during learning programming.

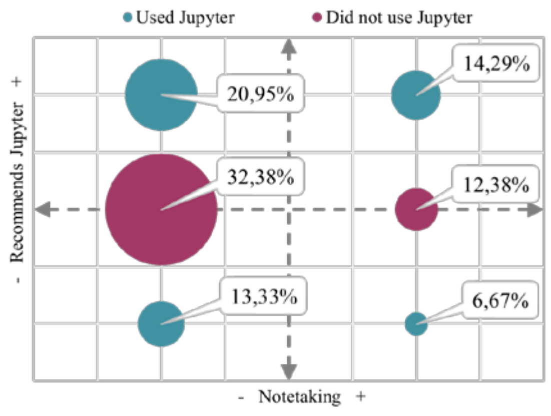

Figure 2 - Notetaking habits & Jupyter experience

We found that it is worthy to mention that 33% of our students started their university studies in the field of informatics without prior programming experience. This partially explains the high number of attritions and also shows that many students struggle to develop the necessary skillset to effectively learn the subject. The data supports our assumption that notetaking, as a valued student skill is not promoted and expected as before. The below figure (Figure 3) shows the ranking of 8 different knowledge sources based on individual preferences. The collected responses highlight the fact that students ranked professional literature and online courses frequently as the last options to study from. However, the top half of the ranking shows rather uniform distribution among sample tasks, teacher provided materials and online video sharing platforms. This arrangement of priorities suggests that students are mainly looking for quick, easy to understand and direct solutions for concrete problems. However, this attitude may obstruct the deep aspect of the learning process itself.

The sources can be further classified as internal and external sources. In this case we label learner and teacher created materials as internal (marked with a dotted pattern) and the rest as external. The data translated in accordance with the previous description supports (Rank 1 – 32%, Rank 2 – 39%, Rank 3 – 35%) our assumption that internally created teaching and learning materials are prioritised and valued.

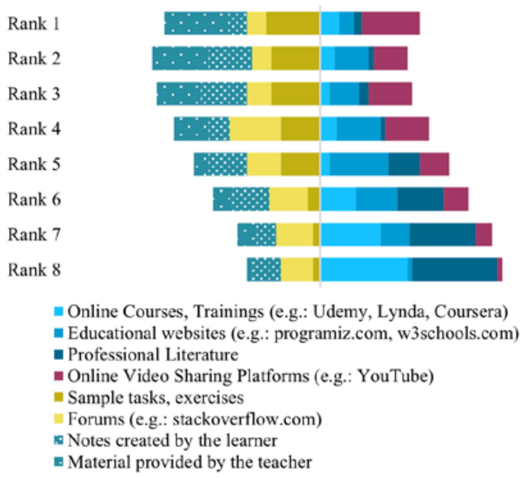

Figure 3 - Ranking of various knowledge sources

Despite the fact that programming language documentation and professional literature are among the most reliable foundations of programming these options are mostly represented in the bottom rankings. To add another perspective to the results we can group courses, literature, and educational websites as demanding sources of deep knowledge (marked with a blue colour). Ironically the mentioned trio is underrepresented in the top half of rankings.
Each section in the questionnaire featured a 4-level Likert scale with multiple statements. In most cases, students were asked about their level of agreement with the proposed statements: "Please indicate your level of agreement with the following statements", in which case the scale ranged from 1 = strongly disagree to 4 = strongly agree.

This first such question contained statements regarding learning habits. By applying clustering to the data 3 groups of learners were identified. With more than 50% the first cluster contains the most members who mainly prefer to learn alone and rather reject the group learning opportunities. The other two clusters are more open towards learning in small and large groups. The most divisive statement of the section was the last one, "I prefer explanations from similar aged". Although the majority of respondents answered with "likely" or "more likely", there is a clear separation among the created clusters.

The second Likert scale dealt with the students' notetaking habits (see Figure 4). With no significant difference between 3 and 4 clusters the presented results display the former. The gathered data suggests that about half of the respondents do not possess the necessary skill to create appropriate notes, however, majority of them are aware of the beneficial effects of notetaking.

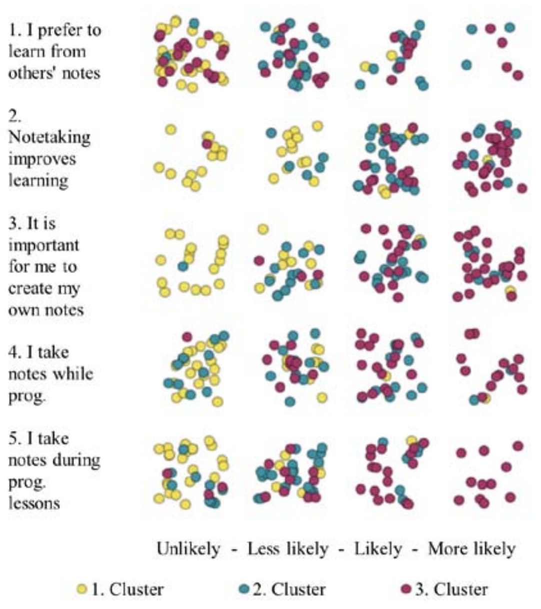

Figure 4 - Students' attitude towards notes and notetaking

The different learning approaches can also be clearly observed in Figure 4. The first cluster contains most of the surface learners, since they are the ones who do not take notes in class, not recognise its benefits. This cluster relies on third party notes, which in this case refers to notes provided by teachers and other students. The representatives of the third cluster we consider deep learners. They take notes in class and prefer to rely on them. Finally, the second cluster (green colour) collects the students who use notes during their studies, but do not invest time in creating their own. Some causes of this attitude surfaced during the interview.

The following question inquired details about preferred primary notetaking methods. The results showed surprising popularity of the less effective methods (marked with shades of blue on Figure 5). The results showed a tie between effective and less effective methods. No respondent marked simple text editors (Notepad, WordPad) as their primary choice.

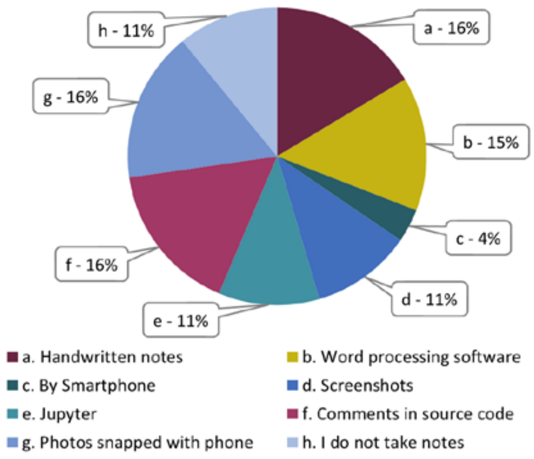

Figure 5 - Students' primary notetaking methods

Next, the frequency of preparation was asked where the respondents could choose their answer from “Regularly”, “Sometimes”, “Rarely”, Before exams” and “Never”. The results show that only 16% of the respondents prepare regularly for classes and most of them occasionally (Sometimes – 37%). Nearly one fifth of the surveyed students stated that they prepare only for examinations. These results combined with Figure 6 show that a bad habit emerged among students which combines ineffective notetaking with irregular preparation patterns.

The last section of the questionnaire dealt with the JupyterHub user experience. Despite the service being available for a year at the time of the survey only 55% of the respondents reported that they are familiar with the platform. Despite the relatively small user base the respondents expressed positive opinions and only 15% were dissatisfied with the platform and its functions.

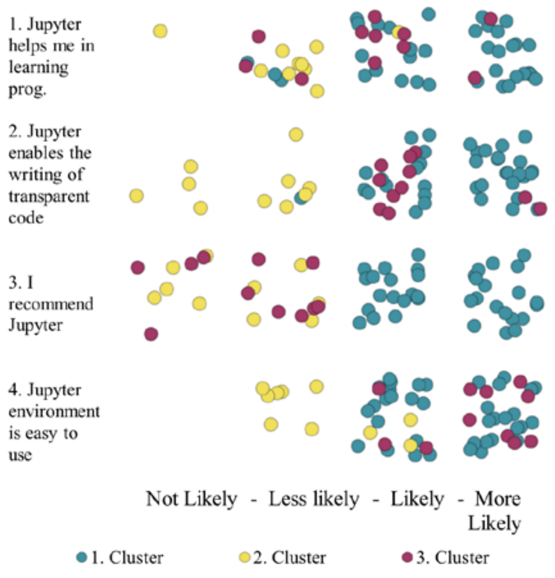

Figure 6 - Students' review of JupyterHub

Figure 6 displays the two main and lesser groups of Jupyter users. The representatives of the first cluster consider JupyterHub a useful addition to their toolset. The second cluster includes students who responded negatively and believe they will not find it useful throughout their studies. The third cluster turned out to be the most interesting. These students mostly share the opinion of the first cluster but in a more restrained way. However, they do not recommend the Jupyter to other students. Our basic assumption was that those who have a positive opinion of the platform will also recommend it. However, as the results show, for some reason the matter is not so clear. We found two possible explanations for the results partially supported by the interview. The first is that members of the third cluster have a positive attitude towards the platform but know or use a service that is more convenient for them. The other reason could be simply the nature of their personality. The available data was not enough to further determine the reasons behind the given answers.

4.1 Interview results

The interview part strongly connected to the questionnaire results and gave the asked opportunity to express their opinion in more details. The participation was voluntary. The first set of questions was related to programming experience gathered prior university studies. Most student responses were negative, mainly reporting the short amount of time secured to programming. However, they acknowledged high school as the main place for gaining programming experience and practising. The authors of the paper suspected that students do not prepare regularly for lessons. Sadly, this idea found support both in the questionnaire and during the interview. As a result of this attitude the learning process shortened and wedged in right before the exam. The reasons included two main factors, which were repeated about notetaking again: time and energy consuming. Another reason some respondents mentioned was that they considered notes provided by the teacher primary and unalterable, therefore they did not see any reason to write down the exact same note again. Finally, some students voluntarily admitted that they did not think they had the necessary skills to point out the key information and take notes on their own.

In order to better understand the outcome of the data displayed on Figure 4 we attempted to further uncover the details. We asked the subjects to describe characteristics of good learning material. The most frequent answer was the time factor and quick success (solving a concrete exercise, e.g.: generate random numbers between 1 and 10) was preferred over comprehensive knowledge (understand how a function works and how to parametrize it, e.g.: generate random numbers on any given interval) of the given topic. Although this attitude is reprehensible, especially in higher educational environments, we think it is also a virtue of the current generation. The last questions focused on the shortcomings of the university JupyterHub server. Some students missed the high-level code completion features present in modern IDEs. While others missed the shared library and integrated file sharing functionality of Colaboratory.

Finally, the interview section ended with 3 teachers who are actively using JupyterHub during their lessons. We asked the participants open questions and let them express their opinion. First, we wanted to know their point of view on the importance of notetaking. The answers were divisive, since according to two respondents making notes was important. On the contrary, the third person stated the opposite and considered source codes appropriate. On the other hand, they all agreed that the majority of students have bad notetaking methods and skills. They pointed out the uselessness of students taking photographs of the projected material with phones. Respondents considered this frequent phenomenon useless and a waste of time, not to mention that most materials (presentations, notes, source codes, examples) are available for the students online. As a supplement to this problem, it was pointed out that the access of information and technology went through a huge development over the last decades. Some years ago, the best and only sources of information for students were the textbooks, professional literature and the teachers themselves. Nowadays every student owns devices capable of snapping high quality pictures and has access to the Internet. Although the necessary information came within reach Figure 4 shows that students prefer to digest specific forms and dosage.

The next topic was to evaluate the JupyterHub platform, list arguments for and against it. Overall, these types of interfaces are seen as a good option. However, one of the interviewees did not choose the mentioned platform because, "Many free and maintenance-free programming-oriented document editors are available, such as Colaboratory". Disadvantages were mentioned such as, “Using the platform deprives students from real IDE experience” or “It lacks some core features such as code completion”. The harm the former statement outlines can be overcome by teaching the students to use IDEs and Jupyter in parallel and draw attention to the differences of the environments. The lack of auto completion was considered by the respondents as useful during the early stages of programming, but uncomfortable while dealing with advanced topics where students already have the necessary skillset to write code.

5 Conclusion

IT earned its place at all levels of education. The interdisciplinary usability further ensures that such skills are not going to be redundant in the near future. This field of science has its place in most industry sectors and students must be equipped with suitable skillset to be successful. We believe that programming in education has come a long way but teaching and learning processes are far less refined compared to traditional core subjects like mathematics or languages. The questionnaire pointed out that some differences between the current and older generations must be taken into account if one wants to improve and adapt the learning process. Actual students were born into rapid development of technology and a fast-paced lifestyle. As a result, they tend to cut down on learning time which may end in superficial learning. We plan to repeat and further refine the survey and observe the operation of JupyterHub more closely.

Literature:

1. Paksi, D., Csóka M.: *JupyterHub as a Higher Education Teaching Platform*. Valencia: IATED Academy, 2022. 2157-2163 p. ISBN 978-84-09-37758-9.
2. Csóka M.: *Notebook Interfaces as Teaching Aids in Programming Education*. Palma: IATED ACADEMY, 2021. 9248-9252 p. ISBN 978-84-09-31267-2.
3. Végh, L., Takáč, O.: *Possible Negative Impacts of Utilizing ICT for Educational and Non-educational Purposes*. In. Proceedings of the 10th International Scientific Conference: “eLearning and Software for Education”. Vol 4. Bucharest: “CAROL I” National Defence University Editura, Universitara, 2014. 485-490 p. ISSN 2066-026X.
4. Paksi, D., Csóka M., Annuš N.: *An Overview of Modern Methodological Approaches of IT Education*. Palma: IATED Academy, 2022. 5812-5817 p. ISBN: 978-84-09-42484-9.
5. Czakóová, K.: *Developing algorithmic thinking by educational computer games*. In. Proceedings of the 16th International Scientific Conference: “eLearning and Software for Education”. Vol 1. Bucharest: “CAROL I” National Defence University Editura, Universitara, 2020. 26-33 p. ISSN 2066-026X.
6. Végh, L., Takáč, O.: *Using Interactive Card Animations for Understanding of the Essential Aspects of Non-recursive Sorting Algorithms*. In: Proceedings of the 2015 Federated Conference on Software Development and Object Technologies. Cham: Springer, 2016. 336-347 p. ISBN 978-3-319-46534-0.
7. Willis, A., Charlton, P., Hirst, T.: *Developing Students’ Written Communication Skills with Jupyter Notebooks*. New York: Association for Computing Machinery, 2020. 1089–1095 p. ISBN 978-14-50-36793-6.
8. Czakóová, K., Stoffová, V.: *Training teachers of computer science for teaching algorithmization and programming.* In: The 14th International Multi-conference on Society, Cybernetics and Informatics: Proceedings (Post-Conference Edition). Winter Garden: International Institute of Informatics and Systemics, 2020. 231-235 p. ISBN 978-1-950492-40-4.
9. Dong, H.: *A Qualitative Study of Cleaning in Jupyter Notebooks*. New York: Association for Computing Machinery, 2021. 1663–1665 p. ISBN 978-1-4503-8562-6.
10. Choithram, S., Suwimon W., Nonglak W.: *Deep Learning and Its Effects on Achievement*. Elsevier Ltd., 2014. 3313-3316 p. ISSN 1877-0428.
11. Marton F., Säljö R.: *On Qualitative Differences in Learning - II Outcome as a Function of the Learner's Conception of the Task*. British Journal of Educational Psychology, 1976. 115-127 p. DOI https://doi.org/10.1111/j.2044-8279.1976.tb02304.x
12. Marton F., Säljö R.: *On Qualitative Differences in Learning: I - Outcome and Process*. British Journal of Educational Psychology, 1976. 4-11 p. DOI https://doi.org/10.1111/j.2044-8279.1976.tb02980.x
13. Asikainen, H., Gijbels, D.: *Do Students Develop Towards More Deep Approaches to Learning During Studies? A Systematic Review on the Development of Students’ Deep and Surface Approaches to Learning in Higher Education*. New York: S pringer Science+Business Media, 2017. 205-234 p. ISSN 1573-336X.
14. Czakóová, K., Udvaros J.: *Applications and games for the development of algorithmic thinking in favor of experiential learning*. In. EDULEARN21: Proceedings of the 13th International Conference on Education and New Learning Technologies. Valencia: IATED Academy, 2021. 6873-6879 p. ISBN 978-84-09-31267-2.
15. Bruner, S. J.: *The Process of Education*. Cambridge, Harvard University Press, 1960.
16. Sáiz-Manzanares, M., Consuelo O., García O., César I., Díez-Pastor, J., Martín A., Luis J.: *Will personalized e-Learning increase deep learning in higher education?* I. issue. Emerald Publishing Limited, 2019. 53-63 p. ISSN 2398-6247.
17. Siegel, J.: *Did you take “good” notes?: On methods for evaluating student notetaking performance*. 2018. 85-92 p. ISSN 1475-1585.

18. Ilter, I.: *Notetaking Skills Instruction for Development of Middle School Students' Notetaking Performance*. VI. issue. 2017. 596-611 p. DOI https://doi.org/10.1002/pits.22021
19. Zhong, L., Wei, Y., Yao, H., Deng, W., Wang, Z., Tong, M.: *Review of Deep Learning-Based Personalized Learning Recommendation*. New York: Association for Computing Machinery, 2020. 145–149 p. ISBN 978-14-50-37294-7.
20. Lee, P.-L., Lan, W., Hamman, D., Hendricks, B.: *The effects of teaching notetaking strategies on elementary students' science learning*. III. issue. 2008. 191–201 p. DOI https://doi.org/10.1007/s11251-007-9027-4
21. Svitek, Sz., Annuš, N., Filip, F.: Math Can Be Visual - *Teaching and Understanding Arithmetical Functions through Visualization*. Mathematics. 10. 2656. 2020. DOI 10.3390/mat h10152656
22. Czakóová, K.: *Interaktív modellek és szimulációk az oktatásban*. In. XXXII. Didmattech 2019 - Proceedings – New Methods and Technologies in Education and Practice: III New Methods and Tools in Education. Trnava: Trnavská univerzita v Trnave, 2019. ISBN 978 80 568 0398 1.
23. Manzoor, H., Naik, A., Shaffer, C.A., North, C., Edwards, S.H.: *Auto-Grading Jupyter Notebooks*. New York: Association for Computing Machinery, 2020. 1139–1144 p. ISBN 978-14-50-36793-6.
González-Carrillo, C.D., Restrepo-Calle, F., Ramírez-Echeverry, J.J., González, F.A. *Automatic Grading Tool for Jupyter Notebooks in Artificial Intelligence Courses*. Sustainability. XXI. issue. 2021. ISSN 2071-1050.

Primary Paper Section: I

Secondary Paper Section: AM, IN