ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ ҒЫЛЫМ ЖӘНЕ ЖОҒАРЫ БІЛІМ МИНИСТРЛІГІ
МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РЕСПУБЛИКИ КАЗАХСТАН
MINISTRY OF SCIENCE AND HIGHER EDUCATION OF THE REPUBLIC OF KAZAKHSTAN

ХАЛЫҚАРАЛЫҚ УНИВЕРСИТЕТІ
МЕЖДУНАРОДНЫЙ УНИВЕРСИТЕТ
INTERNATIONAL UNIVERSITY

ХАЛЫҚАРАЛЫҚ АҚПАРАТТЫҚ ЖӘНЕ КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАР ЖУРНАЛЫ

МЕЖДУНАРОДНЫЙ ЖУРНАЛ ИНФОРМАЦИОННЫХ И КОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES

2022 (3) 2
Сәуір-маусым

ISSN 2708–2032 (print)
ISSN 2708–2040 (online)

Халықаралық ақпараттық және коммуникациялық технологиялар журналы
ISSN 2708–2032 (print)
ISSN 2708–2040 (online)
Меншіктенуші: «Халықаралық ақпараттық технологиялар университеті» АҚ (Алматы қ.).
Қазақстан Республикасы Ақпарат және әлеуметтік даму министрлігінің Ақпарат комитетінде – **20.02.2020** жылы берілген.
№ KZ82VPY00020475 мерзімдік басылым тіркеуіне қойылу туралы куәлік.
Тақырыптық бағыты: ақпараттық технологиялар, әлеуметтік-экономикалық жүйелерді дамытудағы цифрлық технологиялар, ақпараттық қауіпсіздік және коммуникациялық технологияларға арналған.
Мерзімділігі: жылына 4 рет.
Тиражы: 100 дана
Редакцияның мекенжайы: 050040, Алматы қ-сы, Манас к-сі, 34/1, 709-кабинет, тел: +7 (727) 244-51-09).
E-mail: ijict@iitu.edu.kz Журнал сайты: https://journal.iitu.edu.kz

Международный журнал информационных и коммуникационных технологий
ISSN 2708–2032 (print)
ISSN 2708–2040 (online)
Собственник: АО «Международный университет информационных технологий» (г. Алматы).
Свидетельство о постановке на учет периодического печатного издания в Министерство информации и общественного развития Республики Казахстан **№ KZ82VPY00020475,** выданное от **20.02.2020 г.**
Тематическая направленность: информационные технологии, информационная безопасность и коммуникационные технологии, цифровые технологии в развитии социо-экономических систем**.**
Периодичность: 4 раза в год.
Тираж: 100 экземпляров.
Адрес редакции: 050040 г. Алматы, ул. Манаса 34/1, каб. 709, тел: +7 (727) 244-51-09).
E-mail: ijict@iitu.edu.kz Сайт журнала: https://journal.iitu.edu.kz

---

«International Journal of Information and Communication Technologies»
ISSN 2708–2032 (print)
ISSN 2708–2040 (online)
Owner: International Information Technology University JSC (Almaty).
The certificate of registration of a periodical printed publication in the Ministry of Information and Social Development of the Republic of Kazakhstan, Information Committee No. KZ82VPY00020475, issued on 20.02.2020.
Thematic focus: information technology, digital technologies in the development of socio-economic systems, information security and communication technologies
Periodicity: 4 times a year.
Circulation: 100 copies.
Editorial address: 050040. Manas st. 34/1, Almaty. +7 (727) 244-51-09). E-mail: ijict@iitu.edu.kz
Journal website: https://journal.iitu.edu.kz

МАЗМҰНЫ

СОДЕРЖАНИЕ

CONTENTS

INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES
ISSN 2708–2032 (print)
ISSN 2708–2040 (online)
Vol. 3. Is. 2. Number 10 (2022). Pp. 26–36
Journal homepage: https://journal.iitu.edu.kz
https://doi.org/*10.54309/IJICT.2022.10.2.003*

УДК 004.42

APPROACHES TO AUTOMATIC CHECKING OF PRACTICAL ASSIGNMENTS IN MOOCS AND ONLINE LEARNING

A.*A. Buravov*, *N.T. Duzbayev*

Alexey A. Buravov — master student of the «Computer engineering» department, International Information Technology University
ORCID 0000-0002-6930-4219. E-mail: 24793@iitu.edu.kz;
Nurzhan T. Duzbayev — PhD, associated professor, vice-Rector for Digitalization and Innovation, International Information Technology University
ORCID 0000-0002-7989-9463. E-mail: n.duzbayev@edu.iitu.kz.

Abstract. One of the most important components of distance learning for software developers is practice. For effective learning and scaling, an instructor needs to check student's code automatically. This paper presents an actual overview of the present state of online e-learning platforms, Massive Open Online Courses (MOOCs), and other online learning services and solutions with automatic checking of students' practical assignments. It also provides an extensive theoretical framework for automatic compilation, code validation, architecture, scalability, and fault tolerance.

Keywords: online courses, MOOC, automatic code checking, automatic grading, online learning, practical code assignments

For citation: A.A. Buravov, N.T. Duzbayev. Approaches to automatic checking of practical assignments in moocs and online learning // INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES. 2022. Vol. 3. Is. 2. Number 10. Pp. 26–36 (In Russ.). DOI: *10.54309/IJICT.2022.10.2.003.*

ПРАКТИКАЛЫҚ ТАПСЫРМАЛАРДЫ АВТОМАТТЫ ТЕКСЕРУ ЖӘНЕ ОНЛАЙН ОҚЫТУ ТӘСІЛДЕРІ

А.А. Буравов*, Н.Т. Дузбаев

Буравов Алексей Александрович — «Есептеуіш техника» кафедрасының магистранты, Халықаралық ақпараттық технологиялар университеті
ORCID 0000-0002-6930-4219. E-mail: 24793@iitu.edu.kz;

Дузбаев Нуржан Токкужаевич — PhD, Халықаралық ақпараттық технологиялар университеті, байланысты профессоры, цифрландыру және инновациялар жөніндегі проректоры
ORCID 0000-0002-7989-9463. E-mail: n.duzbayev@edu.iitu.kz.

Аннотация. Бағдарламалық жасақтаманы әзірлеушілер үшін қашықтан оқытудың маңызды құрамдастарының бірі тәжірибе болып табылады. Тиімді оқу және масштабтау үшін нұсқаушы студенттің кодын автоматты түрде тексеруі керек. Бұл мақалада желідегі қазіргі жағдайдың нақты шолуы берілген электрондық оқыту платформалары, Жаппай Ашық Онлайн Курстары (MOOCs) және басқа да онлайн оқыту қызметтері мен студенттердің практикалық тапсырмаларын автоматты түрде тексеретін шешімдер. Ол сондай-ақ автоматты жинақтау, кодты тексеру, архитектура, масштабтау және ақауларға төзімділік үшін кең теориялық негізді қамтамасыз етеді.

Түйін сөздер: онлайн курстар, MOOC, кодты автоматты тексеру, автоматты бағалау, онлайн оқыту, практикалық код тапсырмалары

Дәйексөз үшін: А.А. Буравов, Н.Т. Дузбаев. Практикалық тапсырмаларды автоматты тексеру және онлайн оқыту тәсілдері // ХАЛЫҚАРАЛЫҚ АҚПАРАТТЫҚ-КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАР ЖУРНАЛЫ. 2022. Том. 3. Is. 2. Нөмірі 10. 26–36 бет (орыс тілінде). DOI: 10.54309/IJICT.2022.10.2.003.

ПОДХОДЫ К АВТОМАТИЧЕСКОЙ ПРОВЕРКЕ ПРАКТИЧЕСКИХ ЗАДАНИЙ В MOOCS И ОНЛАЙН-ОБУЧЕНИИ

А.А. Буравов*, Н.Т. Дузбаев

Буравов Алексей Александрович — магистрант кафедры «Компьютерная инженерия», Международный университет информационных технологий
ORCID 0000-0002-6930-4219. E-mail: 24793@iitu.edu.kz;
Дузбаев Нуржан Токкужаевич — PhD, ассоцированный профессор, проректор по цифровизации и инновациям, Международный университет информационных технологий
ORCID 0000-0002-7989-9463. E-mail: n.duzbayev@edu.iitu.kz.

Аннотация. Одним из важнейших компонентов дистанционного обучения для разработчиков программного обеспечения является практика. Для эффективного обучения и масштабирования преподавателю необходимо автоматически проверять код ученика. В этой статье представлен фактический обзор текущего состояния онлайн-платформы электронного обучения, Массовые Открытые Онлайн-Курсы (MOOCs) и другие сервисы и решения онлайн-обучения с автоматической проверкой практических заданий студентов. Он также предоставляет обширную теоретическую основу для автоматической компиляции, проверки кода, архитектуры, масштабируемости и отказоустойчивости.

Ключевые слова: онлайн-курсы, MOOC, автоматическая проверка кода, автоматическая оценка, онлайн-обучение, практические задания по коду.

Для цитирования: А.А. Буравов, Н.Т. Дузбаев. Подходы к автоматической проверке практических заданий в moocs и онлайн-обучении // МЕЖДУНАРОДНЫЙ ЖУРНАЛ ИНФОРМАЦИОННЫХ И КОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ. 2022. Том. 3. Is. 2. Номер 10. Стр. 26–36 (на русском языке). DOI: 10.54309/IJICT.2022.10.2.003.

Introduction

Online learning is an integral part of IT-industry as community needs more software developers, architects, analysists, etc. as the foundation for future digitalization. Online learning has gained particular relevance during the massive transition of most industries to remote work during the COVID epidemic (Patricia, 2020). Many services and platforms provide online education for IT-specialists. However, practice remains one of the most important components of teaching programming. The student must carry out and repeat all theoretical stages of learning in real programs (Keuning et al., 2016). In addition, one important aspect of the practical classes is the familiarity with similar solutions of other students, which encourages competition and improved results (Shaw, 2012). In classical education formats (schools, courses, universities), this aspect is provided by the work of teachers, manual scoring, checking and correcting the student's work. However, in the online format, this component becomes a real problem, since manually checking a student's practical assignments requires a lot of the teacher's resources and is pricy (Drummond et al., 2014: 785–790).

The lack of automated practice tests prevents the online learning platform from scaling effectively, and in some cases increases student attrition before completing the course or at its initial stage (Coussement et al., 2020). Different platforms solve this problem in different ways – for example, on Udemy there is no possibility to create practical assignments at all, so the quality of mastering materials on Udemy suffers (Wrigley et al., 2018). Other platforms provide the possibility to work on practical assignments, but with some specific types of assignments, such as JavaScript or Linux Bash. Some information systems like Moodle can be connected to an external checking system (Del Fatto et al., 2016).

Another important point is that for a better quality of learning, practical assignments should be varied and have individual components for each student (Hozak, 2020). In some areas of development, the situation is complicated by local features. For example, when testing mobile applications, one needs to run them on a real device or emulator and check the behavior logically (Bruzual et al., 2020). A similar situation occurs when checking programming assignments in distributed environments ― it is necessary to check the behavior of each node launched by students (Maicus et al., 2019).

Existing platforms and services will be analyzed and reviewed to research existing methods and tools for organizing automated practice assignments.

Contributions are:

− a review of current approaches to automated code testing and evaluation;

− a current review and analysis of learning platforms with the function of automatic checking of assignments. There is no current review or analysis of those platforms. This analysis can become a foundation for further research on this topic;

− an assessment of these platforms according to different criteria like the quality of automatic code checking, availability of manual checking, built-in correction for students and others. At the moment there is no current review considering solutions with automatic code checking in the context of these criteria. This assessment also includes a description of the features and disadvantages of the platforms in question. Further research in this area may use this analysis as a reference and a model.

The rest of the paper is organized as follows:

Section II presents the problem identification and its significance. Section III discusses the related work devoted to the existing methods. Section IV presents existing approaches to checking practical code assignments. Section V presents review of existing MOOC and open-source solutions with the feature of automatic code evaluation. Section VI presents discussion of results and Section VII is conclusion.

Materials and methods

The main problem is to identify automatic e-learning platforms. As there is a lack of a universal tools for automatically checking practical assignments in online coding educational systems.

Furthermore, there is no current detailed review of existing automatic checking methods and tools, which should also include online-services. There are many online platforms and learning services, including open-source ones (e.g., Moodle). However, only a small number of them have practical assignments, where a student can code after completing a theoretical unit. Most of them contain only lecture modules and closed multiple choice tests.

This problem has a great negative effect, which is that a student who starts learning programming on online platforms cannot immediately check and apply the knowledge he or she has acquired. Also, the platform and instructors cannot verify how well the student has mastered certain theoretical modules. The problem has great importance, because the field of education for software engineers has an accelerated tendency to move to a distance format, which accelerated even more during the COVID-19 pandemic (Patricia, 2020).

There are several possible suggestions for solving this problem. For example, simplifying manual checking of practical assignments by an instructor via email or git-hosting. It is also possible to create courses on platforms with the ability to automatically check, however each platform has its own limitations and disadvantages. Optimistic solution consists of some universal system which is open and understandable to instructors and students and allow checking.

This section provides important characteristics of the relevant methods. Staubitz et al. (Staubitz et al., 2016). discussed the practical programming exercises and automated assessment. The proposed work used some typical scenarios of user experience in online courses. The authors showed the basic low-level integrations and some parts of code checking architecture, however, they failed to provide detailed practical examples.

Glassman, et al. (Glassman et al., 2015: 1–35) reviewed different approaches of code reviewing and logical tree building. This work showed examples of detailed statistical analysis of evaluating student's code. The authors provided detailed review of visualization and code comparison practices in the field of automatic code checking. However, they didn't provide any examples of architecture of these systems.

Lowis, et al. (Löwis et al., 2015) discussed social parameters of MOOC users, statistics of continuous learning curve and support stats.

Királya, et al. (Király et al., 2017) described creating a platform for online courses with automatic Java code checking using a test framework. They reviewed methods for building an evaluation system based on test libraries and frameworks.

Nickchen and Mertsching (Nickchen et al., 2016: 482–488) analyzed tools for automatic checking of Octave3D tasks. They demonstrated the operation of automatic task checking for external software. But the authors failed to analyze the architecture of the system.

Rim (Rim, 2017) looks at a system for building online courses. He showed main features of online learning platform for children. But he failed in description of examples of custom practical assignments or a system for professional software developers.

English (English, 2004) described powerful solution for Graphical User Interface (GUI) software tasks written on Java. He showed main features of this solution: automatic checking of GUI-programs and automatic code validation. But he failed in description of tasks on other programming languages besides Java.

Derval et al. (Derval et al., 2015: 86–91) described architecture and principle of external grading system called INGInious. They provide analysis of main solution's features such as editing code in a browser, connection to edX courses as backend checking system and support of many programming languages like Python, C++, Java, Scala. But authors didn't provide any examples of checking user-interface tasks.

Pieterse (Pieterse, 2013: 45–56) provided analysis of online checking system for educational department. Author underscored important system's features like uploading directory of source files and using makefiles. But described system was developed only for C language and doesn't have any frontend editor.

Wang et al. (Wang et al., 2021) describe techniques for peer-reviewing students when solving practical programming problems, as well as the relationship between commits in the version control system (Git) and assessments of reviewers. Now cross-review is often used as a main or additional technique in some Coursera courses.

Approaches to automatic code checking

This section provides our classification of methods for testing and checking program code in general cases. All approaches below use term "grader", referring to both the testing system and its environment.

• Compilation and communication with result via standard in and out system channels or console input and output. This method is one of the oldest approaches to testing solutions written in any programming language. Advantages: simplicity. Disadvantages: the need to configure compilation and assembly separately for each programming language, the complexity of building complex scenarios, probability of

breakage of the grader logic while reading diagnostic messages or logs of the invoked code.

• File input/output. In this method grader compiles solution, uses some files as input, also checks files with predefined names as output of code. This approach could be used in combination of approach a. This approach allows to test work with larger amounts of data than when working through standard input-output streams (a).

• Web requests, API-testing. With this approach, the code under test is compiled and run as a web server. Next, the grader sequentially sends requests to this web server via HTTP/HTTPS, TCP, WebSocket or other protocols and compares the responses with the expected values. Advantages – simplicity, less dependence on the language. Disadvantages – this approach is not applicable in all cases, quite often there is a need to test the visual interface, or other code outputs. Also, student must have the skills to write a web server, and this is usually not the initial stage of training to apply this approach.

• Testing with own tools of a language. Most programming languages have their own tools for writing internal tests. For example, they are Jest for JavaScript, testify and test packages for Golang, JUnit for Java, and so on. In this approach tutor should write test cases on same programming languages as the learning language. The advantages of using it are wide support for language functions, the ability to quickly write test cases. Disadvantages — binding to a specific language, the need to write everything anew for each new programming language (Akahane et al., 2015: 1–6).

• UI/visual testing. This approach is needed when students write tasks with some graphic results as solution. For example, webpages, desktop or mobile applications. As part of this approach, an automated grader can check html tags in a rendered web page, compare pixel colors at certain coordinates, emulate keystrokes and mouse clicks. Advantages — independence from the programming language, the application is tied much to real applications. Disadvantages – the complexity of setting up test cases, many opportunities to deceive the grader, for example, by drawing pixels or tags specifically for it, but not completing the rest of the assignment (Király et al., 2017).

• Containerization. In this approach application is compiled and moved to container (for example — Docker). On next steps automated grader could launch console commands, create and read files, make web requests, check output webpages or interface, mixing previous approaches. Advantages – universality (for example, student could write Dockerfile and instructions to compilation), no need to modify the environment and test logics for each programming language, and the ability to mix other approaches. Disadvantages – increased resource consumption, and possible errors with the removal and cleaning of containers (Maicus et al., 2019).

Mooc and open-source solutions

One of the most popular Massive Open Online Courses (MOOCs) Coursera[1] is an online learning platform where the most advanced and popular universities can create their own courses. It is possible to create both open and closed tests with variants and tasks with automatic code checking. Of course, this is a proprietary closed system, but some of the grading system code is open source[2]. This system is written in Python and uses Docker containers to execute students' code. Accordingly, the course infrastructure team should write their solution in Python language and publish it using the built-in

1 https://www.coursera.org

2 https://github.com/coursera/courseraprogramming

command-line utilities. At the same time, the Coursera's frontend interface does not look so advanced and, in fact, is a normal universal interface for downloading files or archives.

Another popular platform, edX[3], also has the ability to check student's assignments automatically. To check them the course's instructors must connect an external grader through a special interface — xqueue[4]. Your grading server must accept JSON-requests and respond to them with messages of a special format. Unlike Coursera, it is entirely up to you to manage resources and provide the verification itself; the platform actually provides only an external graders interface. No interface or editor for students is provided here either.

The next platform, Hexlet[5], is a relatively young proprietary online learning platform designed primarily for the Russian-speaking segment. At the same time, it provides the same opportunities both for regular automatic checking of student assignments and for manual checking of individual project assignments by selected instructors. One of the advantages of the platform is a powerful online editor with code highlighting. A student can write code in the selected language, view the results of a web server, Linux container, edit any allowed files within his/her container, and call network commands. When checking the code, the platform shows the error's specific place and the full test code so that the student can get familiarized with it, which almost eliminates the cases of a student's misunderstanding of the checking rules and the peculiarities of the grading system (which happens on other platforms). Not the main but rather important advantage of the platform is the code style checker's functionality. For example, for practical exercises in the JavaScript language, the system, after validating the code output result, also checks the code style using the ESLint plugin. However, since the system is proprietary, all assignments and courses are posted only by the company's employees, so the external tutor cannot create custom courses.

The Codio[6] platform is specially designed for creation of tutors' and universities' own computer science courses. It provides a convenient infrastructure for students and tutors, including an online code editor. It also provides the infrastructure for creating, editing, and running coding assignments for students. Special attention should be paid to the specialized interface for editing such tasks by the tutor. This interface allows you to create standard tasks and tests for Java, Python, and any other languages by executing commands from the command line and providing output HTML code tests.

The Stepic[7] platform is also designed to host and provide courses for universities and individual instructors. Depending on the tariff plan, it provides the possibility of manual and automatic checks, including editing tests in the Java language in a local editor using a special library EduTools (written in Java). However, these possibilities are rather limited and have several disadvantages. Automatic code checking is limited

3 https://www.edx.org
4 https://github.com/edx/xqueue
5 https://hexlet.io
6 https://www.codio.com
7 https://stepik.org

to tasks for the Linux terminal and tasks in the Java language. Also, the platform does not have a convenient interface for students to edit the code.

The INGInious[8] platform is an open solution focused on self-building infrastructure and already includes all the necessary components for developing and maintaining online courses of a university, company or instructor. It is written in Python and uses Docker containers to run student's code inside a secured environment.

INGInious provides a backend system with flexible and secure architecture which manages running and interaction with Docker and checking code, and a frontend for students and tutors (Derval et al., 2015: 86–91). The frontend part allows tutors to create and modify code tasks but doesn't have a convenient code editor. Also, a serious advantage of the system is the ability to integrate it with other platforms, such as edX or Moodle. Among the shortcomings, one can single out a not very clear interface and insufficiently detailed documentation. However, these disadvantages can be eliminated as the solution code is open source.

LearnDB[9] is also an interesting example of a specialized online learning system. This platform is specially designed for the study of SQL, which determines its features. The platform has a powerful command editor with syntax highlighting in the browser, a test base, and a practical task for each passed module. The platform's key advantage is a well-thought-out organization for performing practical tasks with databases, which other online platforms do not have. Since the platform is proprietary, there is no way to integrate or create your own courses. There are also no practical assignments in other languages or frameworks.

Codecademy[10] is a popular well-automated platform for learning programming in various languages and frameworks. On this platform, special attention is paid to practical assignments. After each micromodule, the student must immediately solve several tasks with increasing difficulty. In this case, tasks have a different level of participation. In some assignments, students only have to change a certain symbol. In other more complicated tasks students should write full-fledged code that solves a specific task. The built-in editor has unique functionality for highlighting the necessary sections of the code and interactive tutorials. Simultaneously, the platform also offers projects with manual checks for training (after the student reaches a certain level).

Codegym[11] is another platform for teaching development in a specific language. Java programming courses and assignments are well designed and integrated with gamification. This platform is intended mainly for beginners, and middle developers, i.e., assignments are designed to learn the syntax and principles. Another interesting feature is the format of mini projects. The platform has mini projects where the student must solve any small practical problem (program the movement, the ATM logic, and so on).

Each of the considered solutions has its own pros and cons. However, the best

8 https://github.com/UCL-INGI/INGInious
9 https://learndb.ru
10 https://www.codecademy.com
11 https://codegym.cc

solutions based on the review results are Hexlet, Codio and INGInious. Hexlet has features built and developed with the highest quality, especially the mechanism for creating and checking containers, various types of tasks, the ability to change and launch almost all entities inside an isolated container (which contributes to the speed of learning and the ability to try new techniques even outside the task). Codio and INGInious allow the instructor to create their own courses to develop almost any practical assignment. At the same time, INGIniousis is an open-source solution, which allows you to level its shortcomings and adjust it to your infrastructure.

Based on the review and analysis, the main features of a solution with automatic code checks can be merged and described as following:

• Solution should have feature of automatic code checking (by compilation and verification). Hexlet, Codio and CodeCademy show the best support of this feature in a review.

• Solution should have possibility of human manual checking for some tasks (for example, for projects) for best flexibility of learning process. This will adapt e-learning model to all cases and tasks. Hexlet, CodeCademy and Codio show best support of human-checking feature.

• Support as many programming languages as possible. From reviewed list Coursera and edX support almost all variants of languages but that is because these systems move compilation logic to external graders or developers from tutor's side.

The solutions under consideration currently support a different number of languages, as shown at Figure 1.

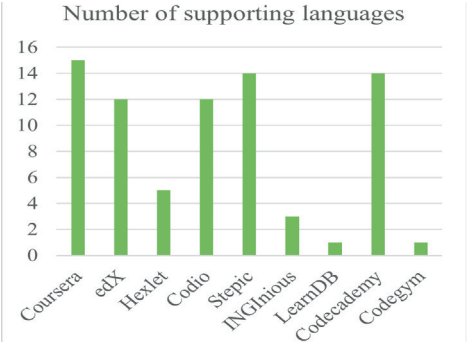

Figure 1 – Number of supporting languages for each solution

• An external editor is not such a common feature, but its presence significantly improves user experience. The student can edit the code directly in the browser and use additional features to explore the environment. Also, an external editor can be made on the base of a desktop program. This feature is best represented in the Hexlet, CodeCademy and LearnDB platforms.

• The code style checking functionality is rather optional. However, this is a good addition if the platform provides courses for novice developers and professionals. As the basic rules of syntax become more obvious with an in-depth study of technology, it

becomes more important to write code according to the standards and guidelines. This functionality is best represented on the LearnDB and Hexlet platforms.

• The backend architecture is, although externally invisible, a significant parameter of the platform as a whole. Automatic compilation and code validation are very resource-intensive tasks. The architecture must be flexible and scalable as needed. At the same time, the backend must remain well protected by the responsive core of the platform. The INGInious's backend is organized in the best way because it is divided into components and allows to scale containers quickly.

• It can also be concluded that the number of supported languages (L criteria) correlates well with the functionality of practical exercises (PT criteria).

• **Results and discussion**

• One of the main problems of online programming education is the lack of practical assignments. As a result, the quality of education suffers directly; students learn the material worse and have significant problems in applying real work skills. The review and analysis of platforms with automatic code checking function, made in this work, brings up to date information about the current situation in this area.

• The result and quality of the online programming learning platform depend not only on practical assignments' functionality. There should also be some or all of the features of manual teacher review, extended support for languages and frameworks, built-in code editor, code style checker, and the ability to scale flexibly and quickly.

• One of the main difficulties was a large number of platforms and the difficulties in assessing their components and functionality. Also, some platforms are known only locally and do not work in the global market; nevertheless, having a good implementation of automatic checks could be overlooked.

• Individual low-level components and libraries for checking and compiling the code have not been considered. The analysis did not include separate techniques and methods for the automatic compilation of code in an isolated environment. These topics will be discussed and analyzed in the future works.

Conclusion

The paper introduces an overview of existing online e-learning platforms with automatic code checking. It reviews and analyzes the existing solutions.

As a result, learning platform for programming could have one or more features like automatic code checking, manual checking, many supported languages and frameworks, and a convenient external editor for students, flexible and easy-scaling backend architecture, code-style checking, and automated peer-review.

In the future, we are interested in analyzing existing low-level libraries, tools for automatic UI tasks evaluation and other techniques for automatic code compilation.

REFERENCES

Akahane Y., Kitaya H., Inoue U. (2015). «Design and evaluation of automated scoring java programming assignments», in 2015 ieee/acis 16th international conference on software engineering, artificial intelligence, networking and parallel/distributed computing (SNPD), Takamatsu. — Jun. 2015. — Pp. 1–6. — Doi: 10.1109/snpd.2015.7176255.

Bruzual D., Montoya Freire M.L., Di Francesco M. (2020). «Automated assessment of android

exercises with cloud-native technologies», annu. Conf. Innov. Technol. Comput. Sci. Educ. Iticse. — Issue section 5. — Pp. 40–46. —2020. — Doi: 10.1145/3341525.3387430.

Coussement K., Phan M., De Caigny A., Benoit D.F., Raes A. (2020). «Predicting student dropout in subscription-based online learning environments: the beneficial impact of the logit leaf model», decis. Support syst. — V. 135. — Issue may. — Pp. 113–325. — 2020. — Doi: 10.1016/j.dss.2020.113325.

Drummond A., Lu Y., Chaudhuri S, Jermaine C., Warren J., Rixner S. (2015). «Learning to grade student programs in a massive open online course», proc. - ieee int. Conf. Data min. Icdm, n. 2015-january. — issue January. — Pp. 785–790. — 2014. — Doi: 10.1109/icdm.2014.142.

Del Fatto V., Dodero G., Gennari R. (2016). «How measuring student performances allows for measuring blended extreme apprenticeship for learning bash programming», comput. Hum. Behav. — V. 55. — Pp. 1231–1240. — 2016. — Doi: 10.1016/j.chb.2015.04.007.

Derval G., Gego A., Reinbold P., Frantzen B, Van Roy P., De Louvain U. (2015). «Automatic grading of programming exercises in a mooc using the inginious platform», proc. Third eur. Moocs stakehold. Summit. — V. 1. — issue 12. — Pp. 86–91. — 2015.

English J. (2004). «Automated assessment of gui programs using jewl», sigcse bull. Assoc. Comput. Mach. Spec. Interest group comput. Sci. Educ. — V. 36. — Issue 3. — Pp. 137–141. — 2004. — Doi: 10.1145/1026487.1008033.

Glassman E.L., Scott J., Singh R., Guo P.J., Miller R.C. (2015). «Overcode: visualizing variation in student solutions to programming problems at scale», acm trans. Comput.-hum. Interact. — V. 22. — Issue 2. — Pp. 1–35. —2015. — Doi: 10.1145/2699751.

Hozak K. (2020). «online software using randomization and automated grading to help teach linear programming with excel and solver in business courses», j. Educ. Bus. — Pp. 1–8. — 2020. — Doi: 10.1080/08832323.2020.1821345.

Keuning H., Jeuring J., Heeren B. (2016). «Towards a systematic review of automated feedback generation for programming exercises», annu. Conf. Innov. Technol. Comput. Sci. Educ. Iticse. — № 11–13. — July. — 2016. — Pp. 41–46. — 2016. — Doi: 10.1145/2899415.2899422.

Király S., Nehéz K., Hornyák O. (2017). «Some aspects of grading java code submissions in moocs», res. Learn. Technol. — V. 25. — Issue 1063519. — Pp. 1–16. — 2017. — Doi: 10.25304/rlt.v25.1945.

Maicus E., Patterson S., Peveler M., Cutler B. (2019). «Autograding distributed algorithms in networked containers», sigcse 2019 - proc. 50th acm tech. Symp. Comput. Sci. Educ. — Pp. 133–138. — 2019. — Doi: 10.1145/3287324.3287505.

Staubitz T., Klement H, Renz J., Teusner R., Meinel D.C. (2016). «Towards practical programming exercises and automated assessment in massive open online courses», proc. 2015 ieee int. Conf. Teach. Assess. Learn. Eng. Tale 2015. — Issue December. — Pp. 23–30. — 2016. — Doi: 10.1109/tale.2015.7386010.

Shaw S.R. (2012). «A study of the relationships among learning styles, participation types, and performance in programming language learning supported by online forums», comput. Educ. — V. 58. — Issue 1. — Pp. 111–120. — 2012. — Doi: 10.1016/j.compedu.2011.08.013.

Löwis M., Staubitz T., Teusner R., Renz J., Meinel C., Tannert S. (2015). «Scaling youth development training in it using an xmooc platform», proc. - front. Educ. Conf. Fie. — 2015. — Doi: 10.1109/fie.2015.7344145.

Nickchen D., Mertsching B. (2016). «Combining mathematical revision courses with hands-on approaches for engineering education using web-based interactive multimedia applications», procedia - soc. Behav. Sci. — V. 228. — Issue June. — Pp. 482–488. — 2016. — Doi: 10.1016/j.sbspro.2016.07.074.

Rim H. (2017). «A study on teaching using website „code.org“ in programming education based on computational thinking», j. Korea multimed. Soc. — V. 20. — Issue 2. — Pp. 382–395. — feb. —2017. — Doi: 10.9717/kmms.2017.20.2.382.

Pieterse V. (2013). «Automated assessment of programming assignments», 3rd comput. Sci. Educ. Res. Conf. Comput. Sci. Educ. Res. — V. 3. — Issue April. —Pp. 45–56. — 2013.

Patricia A. (2020). «College students' use and acceptance of emergency online learning due to covid-19», int. J. Educ. Res. Open. — Pp. 100011. — 2020. — Doi: 10.1016/j.ijedro.2020.100011.

Wang S., Jager L.R., Kammers K., Hadavand A., Leek J.T. (2021). «Linking open-source code commits and mooc grades to evaluate massive online open peer review», arxiv210412555 cs stat. — apr. — 2021. — viewed: february 27. — 2022 [online]. Available at: http://arxiv.org/abs/2104.12555

Wrigley C., Mosely G., Tomitsch M. (2018). «Design thinking education: a comparison of massive open online courses», she ji, — V. 4. — Issue 3. — Pp. 275–292. — 2018. — Doi: 10.1016/j.sheji.2018.06.002.

ХАЛЫҚАРАЛЫҚ АҚПАРАТТЫҚ ЖӘНЕ КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАР ЖУРНАЛЫ

МЕЖДУНАРОДНЫЙ ЖУРНАЛ ИНФОРМАЦИОННЫХ И КОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

INTERNATIONAL JOURNAL OF INFORMATION AND COMMUNICATION TECHNOLOGIES

Правила оформления статьи для публикации в журнале на сайте:
https://journal.iitu.edu.kz
ISSN 2708–2032 (print)
ISSN 2708–2040 (online)
Собственник: АО «Международный университет информационных технологий» (Казахстан, Алматы)

ОТВЕТСТВЕННЫЙ РЕДАКТОР
Ералы Диана Русланқызы

КОМПЬЮТЕРНАЯ ВЕРСТКА
Жадыранова Гульнур Даутбековна

Подписано в печать 15.06.2022.
Формат 60x881/8. Бумага офсетная. Печать - ризограф.7,0 п.л. Тираж 100
050040 г. Алматы, ул. Манаса 34/1, каб. 709, тел: +7 (727) 244-51-09.

Издание Международного университета информационных технологий
Издательский центр КБТУ, Алматы, ул. Толе би, 59