Global Journal of Information Technology: Emerging Technologies

Volume 14, Issue 2, (2024) 72-78

www.gjit.eu

Hour of code and scratch programming platforms for beginners and children

Magzhan Kaulanov [a1], Karagandy University, in memory of academician E.A. Buketov, 100024, 28, Universitetskaya Street, Karaganda, Republic of Kazakhstan. travercy@gmail.com https://orcid.org/0000-0002-3077-0503
Dinara Kazimova [b], Karagandy University of the name of academician E.A. Buketov, 100024, 28, Universitetskaya street, Karaganda, Republic of Kazakhstan. https://orcid.org/0000-0001-7169-7931

Suggested Citation:
Kaulanov, M. & Kazimova, D. (2024). Hour of code and scratch programming platforms for beginners and children. *Global Journal of Information Technology: Emerging Technologies, 14*(2), 72-78. https://doi.org/10.18844/gjit.v14i2.9538

Received from; March 8, 2024, revised from; May 21, 2024, and accepted from August 21.
Selection and peer review under the responsibility of Assoc. Prof. Dr. Ezgi Pelin YILDIZ, Kafkas University, Turkey

iThenticate Similarity Rate: 6%

Abstract

This article explores coding as an essential skill in today's technology-driven world. Given the impact of technology on almost every aspect of daily life, understanding coding is increasingly valuable. The study examines the fundamentals of coding, emphasizing its societal importance and highlighting key benefits, such as enhanced problem-solving skills and increased earning potential. Using a descriptive approach, the article outlines essential steps for beginners, including selecting appropriate learning resources and practicing consistently. Findings underscore coding's role in empowering individuals to navigate and contribute to the digital world more effectively. The article recommends aspiring coders adopt a structured learning path and engage with practical exercises to build proficiency, positioning themselves in the evolving job market.

Keywords: Coding; computer programming; learning resources; problem-solving; technology skills

* ADDRESS FOR CORRESPONDENCE: Magzhan Kaulanov, Karagandy University of the name of academician E.A. Buketov, 100024, 28, Universitetskaya street, Karaganda, Republic of Kazakhstan. *E-mail address*: travercy@gmail.com

1. INTRODUCTION

In the world of technological change where patterns tend to be replaced by new ones swiftly, it is almost impossible to remain up to date with the latest developments in Educational Technology. The aspiration to combine the achievements of the education system with the latest innovative technologies is driven by the desire to form a competitive young generation with 21st-century skills and high digital literacy (Timotheou & Ioannou 2021).

In the modern Education system where we smoothly without realizing transiting to the Internet of Things, programmers play a key position. Robins et al., (2003) advance that coding is one of the most needed skills in the 21st century and there is a huge demand for the programmers and students' passion to learn programming is sharply increasing. However, learning programming is an arduous process. Robins et al., (2003) indicate that programming courses are tough and it has high withdrawal rates. Generally, it takes approximately a decade of programming to be an expert in this field (Robins et al., 2003) or 10.000 hours of experience (Gladwell, 2008).

1.1. Purpose of study

This paper provides an overview of programs like Hour of Code and Scratch, examining how they positively influence the learning process and shift students' attitudes toward coding. Additionally, the discussion explores block-based coding as an accessible entry point for children and beginners, effectively engaging them in foundational programming concepts.

2. METHOD AND MATERIALS

This study adopts a descriptive, qualitative methodology to explore the fundamentals, societal relevance, and educational implications of teaching coding to children and beginners. The research focuses on analyzing current educational programs, such as Scratch and Hour of Code, which utilize block-based programming to engage young learners in foundational coding concepts. To assess the influence of these platforms, secondary data was gathered from recent studies, and academic literature on educational technology and computer science education. This approach enabled an in-depth examination of coding as a skill essential to modern education, with particular emphasis on its role in developing problem-solving abilities and digital literacy in children.

3. RESULTS

3.1. Programming/coding

"Learning to write programs stretches your mind, and helps you think better, creates a way of thinking about things that I think is helpful in all domains"

Bill Gates,

The term "programming" encompasses a wide range of interpretations, often regarded among the most significant discoveries of the twentieth century. Within the sphere of groundbreaking advancements, few achievements rival the discovery that an organism's development follows a genetic program encoded within its gene set. This chapter specifically addresses programming in the context of computers, wherein instructions or "programs" are embedded in machines to guide their functions a process commonly referred to as computer programming.

Programming is the process and the art of creating computer tasks using programming languages and it combines elements of art, science, mathematics, and engineering. In the narrow sense of the word, programming is considered as coding; the implementation of one or more interrelated algorithms in some programming language. Programming, in simple terms, is the process of creating software through code, effectively guiding a computer to perform specific tasks by "speaking" in its language. Kelleher and Pausch (2005) from Carnegie Mellon University describe programming as a system of symbols representing calculative tasks, which a computer interprets through various programming languages. Since the 1960s, numerous engineers, mathematicians, and researchers have developed diverse programming languages and platforms

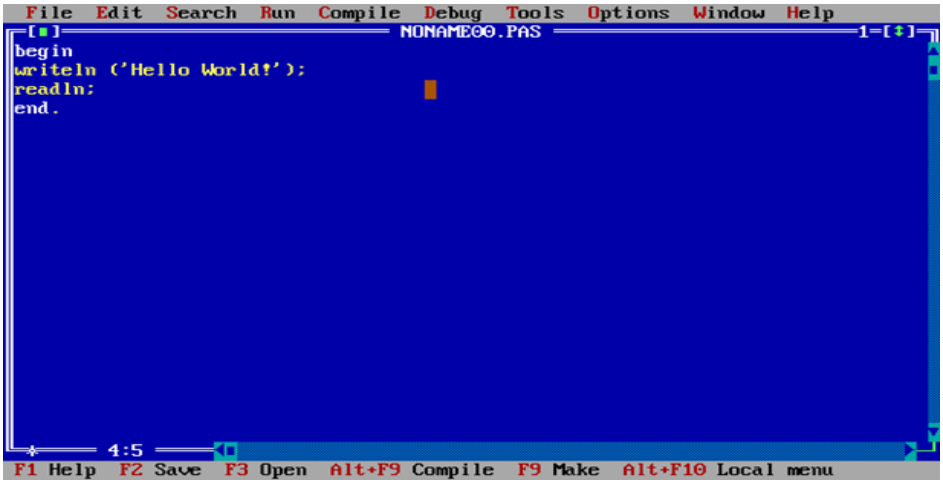
aimed at expanding accessibility to programming. Today, a wide range of platforms exists for beginners to learn foundational programming skills and principles.

3.2. Role of coding in education

Teaching Coding education has become increasingly popular and integral in school computer science curricula, with global interest in teaching coding to young students on the rise (Benton et al., 2017; Wu et al., 2024). However, implementing programming in educational settings presents significant challenges. Introductory programming courses, often referred to as Children Introduction Programming (ChIP) or CS1, require careful consideration of the programming environment and language selection to suit beginners' needs (Ivanovic et al., 2017). One effective approach for introducing programming to children and novices is visual, block-based programming (Hao et al., 2023). In this block-oriented model, students begin by learning foundational concepts and principles of object-oriented programming (Benton et al., 2017). This method, which allows learners to see the direct outcomes of their code, supports the transition "from knowledge to product." The following Figure 1 illustrates key distinctions between traditional programming and block-based coding, which has proven especially beneficial in engaging younger students.

Figure 1

Interface of Pascal programming language



source: borlandpascal.wikia.com

Figure 2

The online interface of Scratch

source: wired.com

From the figures 1 and 2 provided, there is a colossal difference between the two programming approaches. The latter one is block coding and it is designed to make programming accessible to children and make the process enjoyable. This block coding approach is a great way to teach problem-solving, and creativity skills to

children and is much easier for beginners. Bau et al., (2017) state *"Learning a programming vocabulary is difficult. Blocks simplify this problem because picking a block from a palette is far easier than remembering a word".*

Teaching computer programming in primary and secondary schools has a long history in the United Kingdom; schools began incorporating programming languages such as LOGO and BASIC in the late 20th century (Benton et al., 2017). One of the early pioneers in bridging education and technology was MIT professor Dr. Seymour Papert, who introduced the theory of constructionism. According to Papert, constructionism "shares constructivism's connotation of learning," emphasizing the idea that students acquire knowledge through tangible, imaginative engagement with materials rather than solely through abstract concepts (Rifkin, 2016). LOGO, one of the earliest programming languages used in schools, embodies this approach by allowing students to actively interact with programming concepts visually and intuitively, as illustrated in Figure 3.

Figure 3

A student learning via LOGO Turtle

source: cyberneticzoo.com

At first sight, it might seem like a game, however, it is a programmable LOGO Turtle for exploring coding with children. People learn by playing as we used to explore the world by playing counting games to memorize numbers. Both learning and playing are integral to children's everyday lives, and learning engages them in a funny, enjoyable environment (Sorrentino et al., 2017; Roussou, 2004; Tisza et al., 2023). However, Brull (2016) claims that gamification is not an effective way of teaching

3.3. Hour of code culture

Since 2012, every year, the non-profit organization Code.org has provided one hour of coding across more than 180 countries, to engage students in programming. Over 100 million students tried Hour of Code and this year; more than 150000 coding events were registered globally (Code.org, 2017). Annually, events take place on the Computer Science week between 4-10 December each year. Learners build their codes in a drag-and-drop interface to create blocks of code (Wang et al., 2017), (see Figure 2). What makes Hour of Code attractive and engaging to children is that most tasks are relevant to specific topics either to cartoons or famous games. Perhaps we can already safely proclaim that this has already become a culture. Their contribution to raising computer literacy, and attracting young people to programming is colossal.

Apart from the programming skills, Hour of Code tries to emphasize diversity and equity in learning programming. To create a comfortable study environment for all students, computer science must be in classrooms, where classrooms are distributed equally with the participation of all sectors of the community (Wilson 2015). People tend to think that computer science, especially programming is for males, however, Code.org is trying to diversify the situation. For instance, according to their site, 49% of more than 505.000.000 served students are girls which is almost half of users.

One of the disadvantages is that the content and scenarios of the coding are for young generations and it can be challenging to learn for the elderly generation, simply because they have not watched or played the cartoons, or games respectively.

3.4. Scratch

Scratch is a new programming environment launched less than 15 years ago by MIT that allows children to create their own animated interactive stories, games, and models. Students' completed projects can be shared with the community by the authors of Scratch projects around the world (Coravu et al., 2015). In Scratch, one can play with various objects, modify their appearance, move them around the screen, and establish forms of interaction between objects. It is an object-oriented environment in which program blocks are assembled from multi-colored code bricks.

One can start using the language from scratch, without any prior knowledge of programming. It is important to note that Scratch comes with other important pedagogical initiatives. It allows a One Child – One Computer (OneOne) learning approach. The focus for children's learning programming should extend beyond merely using applications; it should encompass a variety of activities that encourage creativity and critical thinking. Children should engage in creating their own stories, inventing games, and developing computer models. Language developers focus on how children in Scratch's environment can independently learn modern culture, and play with images, sounds, and animation. In this platform, students do not use ready-made computer games, although construct their own. The latter idea is at the heart of the learning approach called constructivism. *"*In constructivist learning approach, the student is an actor who has a major and active contribution in his process of learning*,"* according to Coravu et al., (2015).

Learning programming via Scratch is one of the effective and progressive ways. Apart from programming skills, Coravu et al., (2015) identified some features of Scratch in the educational process. When students have initial coding skills in Scratch they can start to strengthen and reinforce their knowledge by doing such activities as learning foreign languages, presentation of the town or school, creating animated postcards, creating quizzes, creating games, learning Mathematics with Scratch (Coravu et al., 2015).

To sum up, programming platforms such as Scratch, and Code.org are powerful tools to enhance learning and develop computer skills. Such platforms create a graphical, colorful, and playful environment, which contributes to attracting learners (Coravu et al., 2015). Such programs change traditional learning of programming and give positive results in changing students' attitudes toward coding. This constructionist learning approach, also known as the active learning procedure, will not allow the child to be biased from learning programming but rather motivate them to learn and to program in the future (Sarkar, 2016). One notable limitation observed through teaching experience is that beginner-focused programming environments lack some capabilities found in more advanced programming languages. However, given that these platforms are designed specifically for children and beginners, this limitation can be seen as an intentional and beneficial exception, providing an accessible entry point to programming fundamentals.

4. CONCLUSION

Shortly, programming will not be only a technical tool of the educational process. It will lead to the formation of a new intellectual background, and a new operating environment, organically and naturally used by the child in his development at school and home. The opportunities provided by programming, and the new tasks of education will inevitably have a significant impact on the main provisions of developmental psychology, the established didactic principles, and forms of education. Their implementation will accelerate the intellectual maturation of the child, increase his activity, and make him better prepared for professional activities.

This paper outlines the advantages and societal importance of programming, particularly for the younger generation. While acknowledging certain limitations, the focus remains on supporting active learning through accessible programming approaches, aiming to foster a constructive and engaging environment. From an educational perspective, the emphasis is placed on promoting programming as a valuable skill that empowers students to actively participate in a technology-driven world.

Kaulanov, M. & Kazimova, D. (2024). Hour of code and scratch programming platforms for beginners and children. *Global Journal of Information Technology: Emerging Technologies, 14*(2), 72-78. https://doi.org/10.18844/gjit.v14i2.9538

REFERENCES

Bau, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: blocks and beyond. *Communications of the ACM*, *60*(6), 72-80. https://dl.acm.org/doi/abs/10.1145/3015455

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging primary programming and mathematics: Some findings of design research in England. *Digital Experiences in Mathematics Education*, *3*, 115-138. https://link.springer.com/article/10.1007/s40751-017-0028-x

Brull, S., & Finlayson, S. (2016). Importance of gamification in increasing learning. *The Journal of Continuing Education in Nursing*, *47*(8), 372-375. https://journals.healio.com/doi/abs/10.3928/00220124-20160715-09

Code.org (2017). The Hour of Code Leaderboards. [online] https://code.org/

Coravu, L., Marian, M., & Ganea, E. (2015). Scratch and recreational coding for kids. In *2015 14th RoEduNet International Conference-Networking in Education and Research (RoEduNet NER)* (pp. 85-89). IEEE. https://ieeexplore.ieee.org/abstract/document/7311973/

Gladwell, M. (2008). *Outliers: The story of success*. Little, Brown.

Hao, X., Xu, Z., Guo, M., Hu, Y., & Geng, F. (2023). The effect of embedded structures on cognitive load for novice learners during block-based code comprehension. *International Journal of STEM Education*, *10*(1), 42. https://link.springer.com/article/10.1186/s40594-023-00432-9

Ivanović, M., Xinogalos, S., Pitner, T., & Savić, M. (2017). Technology-enhanced learning in programming courses–an international perspective. *Education and Information Technologies*, *22*, 2981-3003. https://link.springer.com/article/10.1007/s10639-016-9565-y

Kelleher, C., & Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM computing surveys (CSUR)*, *37*(2), 83-137. https://dl.acm.org/doi/abs/10.1145/1089733.1089734

Murcia, K., Cross, E., & Lowe, G. (2024). Young children's computational thinking: educator pedagogy fostering children's play and learning with a tangible coding device. *The Australian Educational Researcher*, 1-19. https://link.springer.com/article/10.1007/s13384-024-00762-9

Riffkin G., (2016). Seymour Papert, 88, Dies; Saw Education's Future in Computers. *NYTimes.com Technology blog.* https://www.nytimes.com/2016/08/02/technology/seymour-papert-88-dies-saw-educations-future-in-computers.html

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer science education*, *13*(2), 137-172. https://www.tandfonline.com/doi/abs/10.1076/csed.13.2.137.14200

Roussou, M. (2004). Learning by doing and learning through play: an exploration of interactivity in virtual environments for children. *Computers in Entertainment (CIE)*, *2*(1), 10-10. https://dl.acm.org/doi/abs/10.1145/973801.973818

Sarkar, A. (2016). Constructivist design for interactive machine learning. In *Proceedings of the 2016 CHI conference extended abstracts on human factors in computing systems,* 1467-1475. https://dl.acm.org/doi/abs/10.1145/2851581.2892547

Sorrentino, F., Spano, L. D., Casti, S., Carcangiu, A., Corda, F., Cherchi, G., ... & Scateni, R. (2017). ChIP: teaching coding in primary schools. In *CEUR Workshop Proceedings, 1910*, 106-110. CEUR-WS. https://iris.unica.it/handle/11584/222752

Timotheou, S., & Ioannou, A. (2021). Learning and innovation skills in making contexts: a comprehensive analytical framework and coding scheme. *Educational Technology Research and Development*, *69*(6), 3179-3207. https://link.springer.com/article/10.1007/s11423-021-10067-8

Tisza, G., Markopoulos, P., & Bekker, T. (2023). Learning to code: interplay of attitude, emotions, and fun. *Humanities and Social Sciences Communications*, *10*(1), 1-11. https://www.nature.com/articles/s41599-023-02235-3

Wang, L., Geng, F., Hao, X., Shi, D., Wang, T., & Li, Y. (2021). Measuring coding ability in young children: relations to computational thinking, creative thinking, and working memory. *Current Psychology*, 1-12. https://link.springer.com/article/10.1007/s12144-021-02085-9

Kaulanov, M. & Kazimova, D. (2024). Hour of code and scratch programming platforms for beginners and children. *Global Journal of Information Technology: Emerging Technologies, 14*(2), 72-78. https://doi.org/10.18844/gjit.v14i2.9538

Wang, L., Sy, A., Liu, L., & Piech, C. (2017). Deep knowledge tracing on programming exercises. In *Proceedings of the fourth (2017) ACM conference on learning@ scale* (pp. 201-204). https://dl.acm.org/doi/abs/10.1145/3051457.3053985

Wilson C., (2015). Hour of Code - Bridging Research to Scale, *First Bytes.*

Wu, Z., Zheng, L., & Huang, L. A. (2024). Learning to code and coding to learn: A robotics curriculum integrating tangible programming and road safety education for young children. *Education and Information Technologies*, 1-39. https://link.springer.com/article/10.1007/s10639-024-12757-1