MOOC Quality Design Criteria for Programming and Non-Programming Students

Aniza Sabjan[1*], Alawiyah Abd Wahab[2*], Azizah Ahmad[2],
Rahayu Ahmad[2], Syahida Hassan[2], Juliana Wahid[2]

[1]Research and Innovation Management Centre, Universiti Utara Malaysia,
06010 Sintok, Kedah, Malaysia
aniza@uum.edu.my
[2]School of Computing, Universiti Utara Malaysia,
06010 Sintok, Kedah, Malaysia
alawiyah@uum.edu.my
azie@uum.edu.my
rahayu@uum.edu.my
syahida@uum.edu.my
w.juliana@uum.edu.my

[*]Corresponding Author

https://doi.org/10.24191/ajue.v16i4.11941

Received: 15 November 2020
Accepted: 11 December 2020
Date Published Online: 24 January 2021
Published: 25 January 2021

Abstract: The purpose of this study is to investigate the quality design criteria for developing a Massive Open Online Course (MOOC). Currently, there are limited studies that highlight the required design criteria for the MOOC programming courses. A descriptive analysis was conducted to examine the characteristics of the three important quality design criteria which are (i) Instructional Design Criteria involving Lecture Organization and Culture; (ii) Technical Criteria involving User Interface, Video Content, Learning and Social Tools, and Learning Analytics; and (iii) E-Assessment. The data were collected from 306 respondents, representing the UUM MOOC students of 2018 class, were further analyzed using the T-Test hypothesis testing to determine whether both the programming and non-programming students require the same quality design criteria. The questionnaire used in this study consists of 46 items related to the MOOC quality design criteria that were adapted from previous studies. The results indicate that out of the nine constructs, four have obtained significant differences in the mean scores, namely the Video Content, Instructional Design, Culture, and E-assessment. This signifies that different quality design criteria are needed for both the programming and non-programming students. The outcome of this study may assist the developers in designing the MOOC by providing the required criteria according to its importance.

Keywords: Instructional Design, MOOC, non-Programming, Programming, Quality Design

1. Introduction

Recently, MOOC has become one of the most prominent trends in open education as it allows a large number of global learners to attend respective online courses by engaging with a community of instructors from elite universities through videos and online presentations. A statistic published by the Class Central MOOC Report (2019) reveals the involvement of 110 million students, more than 900 universities and 13.5 thousand courses in December 2019. The number of participation indicates that

the MOOC implementation provides quality and best practices in meeting students' intentions and needs.

1.1 Important Quality Design Criteria in Developing a MOOC

According to Creelman, Ehlers, & Ossiannilsson (2014), the quality of MOOC is a prerequisite for an effective and successful learning in the MOOC environment. To deliver a quality MOOC course, a competent-based design approach is used to ensure active participation among the learners. Most importantly, the approach is appropriate in attracting (Wanli, 2019) the involvement of more diverse students as it focuses on the learning outcomes and addresses the tasks to be performed by learners. This eventually helps to empower the students by providing strategies that will change the perception of learners towards achieving their learning goals. It also provides clear learning orientation plans, joint learning design to include teamwork and discussion forums, social networks, peer-to-peer assistance and support for creating and generating students' knowledge. Other essential features include opportunities for having small groups' discussion and information exchange, offering assessment and colleagues' feedback, as well as using the enhanced learning technology media (Patru & Balaji, 2016).

Yousef, Chatti, Shroeder & Wosnitza (2014) present specific criteria that need to be considered in designing and implementing MOOCs due to their unique features. The quality criteria are classified into two dimensions and six categories. The pedagogical dimension includes the Instructional Design criteria (Lecture Organization and Culture) and the Assessment criteria (E-Assessment and Peer Assessment). On the other hand, the Technical dimension includes the User Interface, Video Content, Learning and Social Tools and Learning Analytics criteria. Their results show that assessment and learning analytics earn the highest average mean score. However, usability, content, collaboration, and instructional design, which play the key role in achieving effective MOOC, are identified as less important as compared to the learning analytics and assessment (Yousef et al., 2014).

According to Wetzinger, Standl, B, & Futschek (2018), there are three main elements that need to be focused on when developing a computer programming MOOC. First, the learning platform element should provide a well-maintained technical infrastructure and support. Second, the video lectures should be short and cover live programming examples while the handouts could consist of reading materials in PDF format that explains the content of the videos and additional programming examples. Third, course materials and activities could consist of exercises and quizzes, programming examples, glossaries, graphical visualizations, reference of common processing error messages, discussion forums, online office hours, and course emails.

Some studies on programming MOOCs (Vihavainen, Luukkainen, & Kurhila 2012; Ruiz, 2015; Dale & Singer, 2019; Abeer & Miri, 2014) have started to look at instructional design features in-depth. For example, Ruiz (2015) conducted a case study on six MOOCs for Introduction to Programming using Python subject. He concludes that instructional strategies associated with teaching presence were leveraged more than instructional strategies for cognitive and social presences. A survey done by Dale & Singer (2019) reveals that students mostly valued the programming exercises, quizzes and instructional videos, while the follow-up focus group highlighted the flexibility of the MOOC, usefulness of the videos, drop-in sessions and programming exercises. The overall mix of activities was regarded as particularly useful. A study by Abeer & Miri (2014) presents that the learner shows the motivation of engagement when the MOOC courses focus on presenting multimedia features, pictures, animations and simulations as part of the learning materials. It is crucial that the course content is technically organized, the videos are rich and the lecturer uses many 3D visualizations, pictures, as well as animations. This is to ensure that the students feel like they are experiencing the real thing from the learning materials.

On the other hand, some studies focus more on the assessment aspect (Ullah, 2018; Lepp, Palts, Luik, Papli, Suviste, Säde, & Tõnisson, 2018) of programming MOOCs. According to Ullah (2018), a student needs continuous feedback on their progress, which can obviously be provided through assessment. Novice students have to learn new syntax and semantics by practicing more programming exercises. Hence, feedback from educators on students' errors is essential to enhance their knowledge. Lepp et al. (2018) developed an integrated help system called Troubleshooters (similar to IT support services) that provides clues and coding examples to help students with their programming tasks. To

evaluate Troubleshooter, they collected feedback from 792 students attending About Programming MOOC. They found that 40.8% of those indicated that they used Troubleshooter, claiming that it was very useful.

While there is a growing literature on MOOC quality, existing studies suffer from some limitations. In particular, there is a paucity of research that looks into the differences in requirements of MOOC quality design criteria from the perspective of programming and non-programming students. Therefore, the main aim of this study is to explore whether these two categories of students require similar quality design criteria to effectively engage in MOOCs.

2. Research Methodology

A self-administered questionnaire was distributed online to the 1,183 students enrolled on various MOOCs offered in UUM. Of 1,183 students enrolled, 306 students responded (a response rate of 26%) with no cases of missing data. The low response rate can be partially explained by our usage of UUM email to distribute the questionnaires. We later learned that the students did not use UUM emails as their primary email. The MOOC, which was categorized under the Programming subject refers to the Introduction to Object-Oriented Programming, while the MOOCs categorized under non-programming subjects are Islamic Banking Management, Export Management, International Business, Accounting System Analysis and Design, Technology Planning and Management in Education, Islamic Bank Operation, Human Lifespan Development, Foundations of Banking and Fundamental of Entrepreneurship. The information regarding the student enrollment was provided by the UUM University Teaching and Learning Centre (UTLC).

Of 74 criteria, 46 were selected from Yousef et al. (2014) because the rest of the features were not implemented in the UUM MOOC. For example, the peer assessment was omitted since it was not included in the UUM MOOC. The final selected criteria to be adapted in this study are (i) the Lecture Organization (5 items) and Culture (5 items) that represents the Instructional Design Criteria; (ii) User Interface (8 items), Video Content (9 items), Learning and Social Tools (8 items) and Learning Analytics (5 items) criteria of the Technical Criteria and finally (iii) the E-Assessment criteria (6 items). The lecture organization construct addresses the way in which MOOC courses are organized in terms of the course objectives, course outline and course timeline among others. Meanwhile, the culture construct evaluates how cultural issues are being implemented in MOOC courses. Furthermore, the user interface construct covers layout issues such as availability of help system and features to control the lecture videos. On the other hand, the video content concerns the quality of video lectures that are produced, such as video sound and video length. Moreover, the learning and social tools construct refers to the availability of academic session and work deadline notifications, while the learning analytic construct measures the importance of performance reports and course activity statistics. Lastly, the e-assessment construct looks at how assessment features such as types of questions, feedback on assessments are implemented in MOOC courses. The questionnaire for this research is available upon request.

Initially, at the early stage of the study, a descriptive analysis was performed on all the 306 respondents based on the assumptions made from the findings of Yousef et al. (2014). Then, to further investigate whether the students are in favor of the same design quality criteria for their MOOCs, another descriptive analysis was run according to two categories, namely programming (155 respondents) and non-programming (151 respondents) students. From the second analysis, the results reveal different mean scores for some of the constructs which led to the development of null hypothesis for each of the pre-defined criteria adapted from Yousef et al. (2014). The hypotheses are listed as follows:

H1. There is no difference in importance between non-programming (NP) and programming (P) students category towards e-assessment (EA) criteria.
H2. There is no difference in importance between non-programming and programming students category towards instructional design (ID) criteria.
H3. There is no difference in importance between non-programming and programming students category towards technical design (TD) criteria.

H4. There is no difference in importance between non-programming and programming students category towards user interface (UI) criteria.
H5. There is no difference in importance between non-programming and programming students category towards video content (VC) criteria.
H6. There is no difference in importance between non-programming and programming students category towards learning and social tools (LST) criteria.
H7. There is no difference in importance between non-programming and programming students category towards learning analytics (LA) criteria.
H8. There is no difference in importance between non-programming and programming students category towards lecture organization (LO) criteria.
H9. There is no difference in importance between non-programming and programming students category towards culture (C) criteria.

3. Results

3.1 Descriptive Analysis Results

3.1.1 Instructional Design Criteria

The Instructional Design criteria are presented as a set of learning design principles. Table 1 summarizes the comparison of the statistical results for the Instructional Design which has been analyzed three times involving all respondents, non-programming and programming students to investigate whether they share the same mean score.

The Lecture Organization criteria is rated as more important than the Culture by both the programming and non-programming students. However, there are differences in the mean score between the programming and non programming students. For example, the programming students achieve high mean score for the LO1 (have clear objectives defined at the beginning of each lecture), LO3 (offer course outline that contains objective, subject list and time schedule) and LO5 (offer the course progress timeline in visualization graphs) items with values of 4.34, 4.12 and 3.99, whilst the non-programming students achieve the values of 4.15, 4.13 and 4.15 for item LO1, LO3 and LO4 (provide opportunities for learners to become more self-organized). This implies that all students agree that their MOOCs should have clear objectives at the beginning of each lecture, offer course outline, provide opportunities for learners to become more self-organized and offer the course progress timeline. For the culture criteria, both the programming and non-programming students denote the importance of providing samples that can be accepted by everyone regardless of their diverse cultural backgrounds (C1). Moreover, they also suggested having at least two distinctive times for learners to join video conference meetings (C2). It is also noted that programming students favour the use of English language (C3), while the non-programming students prefer to consider cultural diversity values in the lecture videos (C4).

Table 1. Descriptive Results for the Lecture Organization and Culture Criteria

Instructional Design		Items	*m* (All) (N=306)	*SD* (All) (N=306)	*m* (P) (N=155)	*SD* (P) (N=155)	*m* (NP) (N=151)	*SD* (NP) (N=151)
	Lecture Organization Criteria	LO1	4.25	0.70	4.34	0.59	4.15	0.80
		LO2	3.87	0.82	3.68	0.84	4.05	0.77
		LO3	4.12	0.67	4.12	0.57	4.13	0.76
		LO4	4.06	0.69	3.97	0.60	4.15	0.76
		LO5	4.05	0.68	3.99	0.55	4.11	0.79
		Average	**4.07**	**0.71**	**4.02**	**0.63**	**4.12**	**0.78**
		C1	4.25	0.72	4.25	0.66	4.24	0.77

n Criteria	**Culture Criteria**	C2	3.95	0.79	3.90	0.76	4.00	0.82
		C3	4.06	0.84	4.09	0.77	4.03	0.90
		C4	4.00	0.81	3.85	0.85	4.16	0.72
		C5	3.76	0.89	3.53	0.89	4.00	0.83
		Average	**4.00**	**0.81**	**3.92**	**0.79**	**4.08**	**0.81**

m mean, *SD* standard deviation, N sample size, P programming, NP non-programming students

3.1.2 E-Assessment Criteria

Table 2 summarizes the comparison of the statistical results for the e-assessment, which was first analyzed on all participants, followed by the non-programming students, and finally the programming students to investigate whether they share the same mean score.

The analysis results show that items EA1 (provide feedback or show the correct answers for each quiz), EA4 (identify the maximum number of marks for a question) and EA2 (use different types of questions) obtain a high mean score of 4.42, 4.10 and 4.05 for the programming students, whilst the non-programming students scores high mean value of 4.40, 4.29 and 4.26 for items EA1, EA4 and EA6 (have hints for each assignment) respectively. This indicates that the assessment and feedback features are important to ensure that the students are aware of their performances.

Table 2. Descriptive Results for the E-Assessment Criteria

E-Assessment Criteria	**Items**	***m* (All) (N=306)**	***SD* (All) (N=306)**	***m* (P) (N=155)**	***SD* (P) (N=155)**	***m* (NP) (N=151)**	***SD* (NP) (N=151)**
	EA1	4.41	0.67	4.42	0.57	4.40	0.76
	EA2	4.13	0.66	4.05	0.57	4.21	0.73
	EA3	4.07	0.83	3.99	0.70	4.15	0.94
	EA4	4.20	0.66	4.10	0.56	4.29	0.74
	EA5	3.86	0.84	3.72	0.84	3.99	0.83
	EA6	4.10	0.83	3.95	0.79	4.26	0.85
	Average	**4.13**	**0.75**	**4.04**	**0.67**	**4.22**	**0.81**

m mean, *SD* standard deviation, N sample size, P programming, NP non-programming students

3.1.3 Technical Criteria

Table 3 summarizes the comparison of the statistical results for the Technical category. In this category, the programming students rate the User Interface as the most important, followed by the Video Content, Learning Analytics, and Learning and Social Tools. As for the non-programming students, Video Content is rated as the most important followed by User Interface, Learning Analytics, and Learning and Social Tools. It is also found that there are differences in the mean score between the programming and non-programming students for the Technical criteria.

For the User Interface criteria, both the programming and non-programming students are in consensus that it is vital to provide a search box function to help learners find different learning materials (UI2) and control features for video clips when appropriate. The programming students prefer to have a help system that should focus on user errors (UI4) while the non-programming students opt for the ability to download the lecture videos to their devices (UI3). For the video content criteria, items VC1 (provide clear sound), VC6 (synchronize of video and lecture notes and programming examples)

and VC2 (offer references for facts) are rated as the most significant by the programming students with the mean values of 4.38, 4.28 and 4.26 respectively, whilst for the non-programming students, items VC1, VC2 and VC4 (use short video clips, no more than 20-minute clips) are rated as the most important. This signifies that both the programming and non-programming students choose the video contents with clear sounds that offer references for facts and information in the lectures.

For the Learning and Social Tools criteria, both student categories agree that it is important to offer notification tools for important news and deadlines (LST3). The programming students prefer MOOC to provide email notifications (LST2) and use video-conference tools to allow students from various locations to communicate with the teachers (LST4). On the other hand, the non-programming students prefer collaborative discussion tools (LST1) and online participants list to help in synchronous discussions (LST6).

As for the Learning Analytics criteria, all students rate providing recommendations and feedback to improve their performance (LA1) as the most important item. The programming students prefer to be provided with analytic tools for self-reflection (LA3) and performance reports (LA2) whilst the non-programming opt for the statistics on the course activities (LA4) and performance predictions (LA5).

Table 3. Descriptive Results for the Technical Criteria

		Items	***m* (All) (N=306)**	***SD* (All) (N=306)**	***m* (P) (N=155)**	***SD* (P) (N=155)**	***m* (NP) (N=151)**	***SD* (NP) (N=151)**
Technical Criteria	**User Interface Criteria**	UI1	4.30	0.72	4.29	0.69	4.31	0.75
		UI2	4.37	0.69	4.39	0.64	4.35	0.74
		UI3	4.22	0.71	4.13	0.66	4.32	0.75
		UI4	4.24	0.75	4.21	0.75	4.28	0.75
		UI5	4.19	0.79	4.13	0.84	4.26	0.73
		UI6	4.21	0.76	4.17	0.77	4.25	0.75
		UI7	4.10	0.75	4.10	0.75	4.11	0.76
		UI8	3.90	0.80	3.77	0.81	4.03	0.80
		Average	**4.19**	**0.75**	**4.15**	**0.67**	**4.24**	**0.81**
	Video Content Criteria	VC1	4.35	0.68	4.38	0.62	4.32	0.74
		VC2	4.28	0.68	4.26	0.64	4.31	0.72
		VC3	4.25	0.68	4.22	0.63	4.28	0.73
		VC4	4.18	0.73	4.06	0.64	4.29	0.80
		VC5	4.15	0.72	4.10	0.71	4.21	0.73
		VC6	4.27	0.65	4.28	0.62	4.26	0.69
		VC7	4.16	0.65	4.06	0.52	4.26	0.75
		VC8	3.95	0.84	3.75	0.84	4.15	0.79
		VC9	4.07	0.66	3.99	0.58	4.16	0.73
		Average	**4.18**	**0.70**	**4.12**	**0.64**	**4.25**	**0.74**
	Learning and Social Tool Criteria	LST1	4.08	0.61	4.01	0.46	4.16	0.73
		LST2	4.04	0.77	4.14	0.56	3.95	0.94
		LST3	4.24	0.69	4.20	0.59	4.27	0.78
		LST4	4.07	0.73	4.03	0.66	4.10	0.79
		LST5	4.01	0.65	3.98	0.43	4.04	0.81
		LST6	4.08	0.60	4.01	0.43	4.15	0.72
		LST7	3.86	0.78	3.75	0.62	3.97	0.90
		LST8	3.85	0.79	3.73	0.65	3.97	0.90
		Average	**4.03**	**0.70**	**3.98**	**0.55**	**4.07**	**0.82**
	Learning Anal	LA1	4.12	0.67	4.12	0.50	4.13	0.81
		LA2	4.08	0.71	4.05	0.50	4.11	0.88
		LA3	4.08	0.65	4.06	0.46	4.09	0.80

ytics	LA4	4.04	0.70	3.95	0.54	4.13	0.83
Crite	LA5	4.05	0.74	3.97	0.61	4.13	0.85
ria	**Average**	**4.07**	**0.70**	**4.03**	**0.52**	**4.12**	**0.84**

m mean, *SD* standard deviation, N sample size, P programming, NP non-programming students

The order of the criteria according to their importance as reflected by the mean score of the programming students begins with User Interface, followed by Video Content, Technical criteria, E-assessment, Learning Analytic, Lecture Organization, Learning and Social Tool, Instructional Design and Culture. The positioning of the criteria is slightly different for the non-programming students, which indicate the following; Video Content, User Interface, E-assessment, Technical criteria, Lecture Organization, Learning Analytics, Instructional Design, Culture, and Learning and Social Tool. The ordering is shown in Table 4.

Table 4. Mean Score Value According to the Importance of Constructs

Constructs	Programming		Non-Programming		Construct
	m	*SD*	*m*	*SD*	
User interface	4.15	0.56	4.25	0.61	Video content
Video content	4.12	0.47	4.24	0.61	User interface
Technical criteria	4.07	0.32	4.22	0.61	E-Assessment
E-Assessment	4.04	0.45	4.17	0.60	Technical criteria
Learning analytics	4.03	0.41	4.12	0.67	Lecture organization
Lecture organization	4.02	0.46	4.12	0.77	Learning analytics
Learning & social tool	3.98	0.34	4.10	0.60	Instructional design
Instructional design	3.97	0.45	4.08	0.62	Culture
Culture	3.92	0.60	4.07	0.68	Learning & social tool

m mean, *SD* standard deviation

The findings show that both the Video Content and User Interface scored the highest mean values for both the programming and non-programming students. The results differ to those of the previous survey done by Yousef et al. (2014) where Learning Analytic and Assessment scored the highest mean score of 4.25 and 4.21 respectively.

3.2 Hypothesis Testing Results

The T-Test was utilized for testing the hypothesis of all the nine constructs. According to Pallant (2007), T-tests are used when there are two groups or sets of data. For the context of this study, the students are categorized into two categories representing the programming and non-programming students. The T-test enables the estimation regarding the means differences between the two categories (Lavrakas, 2008). Results show that H1, H2, H5 and H7 were rejected meanwhile H3, H4, H6, H8 and H9 were accepted. The details of the results are presented in Table 5.

Table 5. Hypothesis Testing Results

Constructs	**Sig. (2-tailed)** ***p* value**	**Results**
H1: P & NP ID	0.033***	Not supported
H2: P & NP EA	0.003***	Not supported
H3: P & NP TD	0.068	Supported
H4: P & NP LO	0.145	Supported
H5: P & NP C	0.021***	Not supported
H6: P & NP UI	0.175	Supported
H7: P & NP VC	0.045***	Not supported
H8: P & NP LST	0.122	Supported
H9: P & NP LA	0.211	Supported

***p <0.05, significant, (two-tailed test)

The results indicate that there are differences in terms of the required criteria in developing an effective MOOC for the programming and non-programming students pertaining to the video content, instructional design, e-assessment and culture. In contrast, similar criteria relating to the technical, lecture organization, user interface, learning and social tools and learning analytics are needed in developing MOOC for all the students.

For the programming students, the developer may focus more on synchronizing the video, lecture notes, and programming examples as well as embedding an integrated development environment (IDE). IDE is a graphical software application that combines all of the features and tools needed by a software developer. The application uses windows and controls such as buttons to display information and accept input from the users. Among the examples of the tools are source code editor, project editor, toolbar and output viewer or debugger (IDE in Software: Definition & Examples, 2019). The programming students may prefer their Lecture Organization to be in the form of exercises as these exercises could help them to become familiar with programming languages syntax and programming errors. The relevance of exercises has had some support in previous programming MOOCs literature, for example, Vihavainen et al. (2012) found that rigorous exercises MOOC could enhance learners' programming expertise. Pertaining to the cultural perspective, the developer should ensure that the overall programming course to be taught in the most understandable form for all, regardless of the cultural background. This is important as the programming language itself is a new form of language for every programming student. In the assessment category, the programming students should be given the opportunity of getting feedback on the attempted quizzes, particularly on the correct answers. Furthermore, there should be a mechanism in MOOC to assist learners by providing hints to assigned tasks. Previous work in programming MOOCs similarly found the importance of providing a self-assessment feature that could provide sample answers and clues to common programming errors (Lepp et al., 2018; Vihavainen et al., 2012). The current work, therefore, supports previous MOOC research, reinforcing the importance of exercises and feedback on assessments in programming MOOCs.

The non-programming students, on the other hand, prefer a short lecture with not more than 20 clips of video content. In addition, they would like to have references and facts to be included in the video lectures so that they can discover further if needed. In terms of the Lecture Organization for the Instructional Design criteria, the focus should be given more on providing the chance for the non-programming students to be more self-coordinated by clearly stating the course guideline to promote engagement. On the cultural perspective of the Instructional Design category, the developers may consider the diversity of the cultural values in the content of the video lectures. For the assessment category, the non-programming students may prefer each test or quiz to have hints as their lectures may focus on theories or facts.

4. Conclusion

The purpose of this study is to identify whether the programming and non-programming students require different quality design criteria to maintain engagement in MOOC. The results of this study are based on the small scope of the UUM MOOC students, class of 2018. The students were divided into two categories of programming and non-programming students. The results show that the Video Content, E-assessment, Instructional Design and Culture criteria have significantly different mean scores for both categories of students. This indicates that the programming and non-programming students may require different features implementations regarding the quality design of MOOC related to the four criteria mentioned above. Furthermore, future MOOC developers could focus more on User Interface, Video Content, Technical criteria and E-assessment criteria to design MOOC courses that could engage and offer meaningful learning experiences (Norliza, Mohd Sahari, Arnida & Ahmad Fauzi, 2020). Future studies could further explore other factors such as the respondents' demographic that could contribute to the differences in preference towards MOOC quality design criteria. Furthermore, a bigger scope is particularly recommended to provide more holistic and comprehensive results. The quality design of MOOC is an important research agenda that needs to be further researched as MOOC has the potential to change and improve future learning experiences.

5. Acknowledgements

This research was partially supported by the School Centre of Excellence (SCoE) research grant, Universiti Utara Malaysia (SO CODE: 13722).

6. References

Abeer, W., Miri, B. (2014). Students' preferences and views about learning in a MOOC, *Procedia – Social and Behavioral Sciences 152*, 318 – 323.

Dhawal, S., (2019). *By the numbers: MOOC in 2019*. [Website]. Retrieved from https://www.classcentral.com/report/mooc-stats-2019/, 2019.

Creelman, A., Ehlers, U., & Ossiannilsson, E. (2014). Perspectives on MOOC quality-an account of the EFQUEL MOOC quality project. *International Journal for Innovation and Quality in Learning, 2*(3), 78–87.

Dale, V.H.M, , Singers, J. (2019). *Learner experiences of a blended course incorporating a MOOC on Haskell functional programming*, Smart Learning Environments.

IDE in Software: definition & examples. (2019). Retrieved from https://study.com/academy/ lesson/ide-in-software-definition- examples.html.

Lavrakas, P.J. (2008). *Encyclopedia of survey research methods* (Vols. 1-0). Thousand Oaks, CA: Sage Publications, Inc. doi: 10.4135/9781412963947

Lepp, M., Palts, T., Luik, P., Papli, K., Suviste, R., Säde, M., & Tõnisson, E. (2018). Troubleshooters for Tasks of Introductory Programming MOOCs. *International Review of Research in Open and Distributed Learning*, *19*(4).

Norliza, G., Mohamad Sahari, N., Arnida, A., & Ahmad Fauzi, M.A. (2020). The relationship between students' MOOC-efficacy and meaningful learning. *Asian Journal of University Education*, *16*(3), 89-101.

Pallant, J.F. (2007), *SPSS Survival Manual: A step by step guide to data analysis using SPSS for Windows version 15*. [Online] Retrieved from: https://www.researchgate.net/publication/234812476_SPSS_Survival_Manual_A_Step_by_Step_Guide_to_Data_Analysis_Using_ SPSS_for_Windows_Version_15

Patru M., & Balaji V. (2016). *Making Sense of MOOCs: A guide for policy makers in developing countries*, United Nations Educational, Scientific and Cultural Organizations (UNESCO) & Commonwealth of Learning.

Ruiz, M.A.J (2015). *A case study of introductory programming with MOOCS*. Open Access Theses. 604. https://docs.lib.purdue.edu/open_access_theses/604.

Ullah., Z., Lajis, A., Jamjoom, M., Altalhi, A. (2018). *The effect of automatic assessment on novice programming: Strengths and limitations of existing systems*, Computer Applications in Engineering Education, Wiley, DOI:10.1002/cae.21974.

Wanli, X. (2019). Exploring the influences of MOOC design features on student performance and persistence, *Distance Education*, 40:1, 98-113, doi: 10.1080/01587919.2018.1553560.

Wetzinger, E., Standl, B., Futschek, G. (2018). Developing a MOOC on introductory programming as additional preparation course for CS (Computer Science) freshmen, *Proceeding of EdMedia + Innovate Learning 2018* – Amsterdam, Netherlands.

Vihavainen, A., Luukkainen, M., & Kurhila, J. (2012). Multi-faceted support for MOOC in programming. In *Proceedings of the 13th annual conference on Information technology education* (pp. 171-176).

Yousef, A.M.F., Chatti, M.A., Shroeder, U., Wosnitza, M. (2014). What drives a successful MOOC? An empirical examination of criteria to assure design quality of MOOCs. *IEEE 14th International Conference on Advanced Learning Technologies*.