**Experience Report for Microsoft’s Quantum Katas without Explicit Guidance**

**Chris Mayo**

**Abstract**

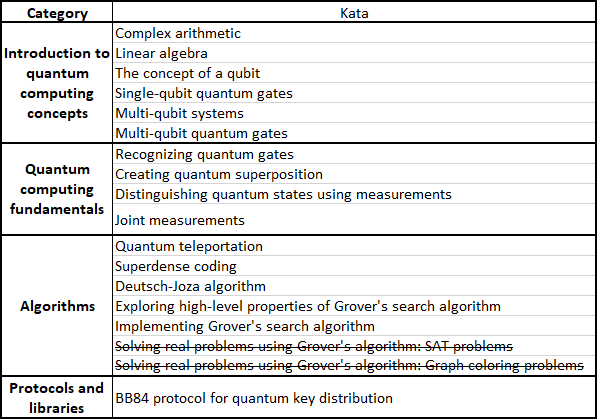
The Microsoft Quantum Katas are a set of self-paced tutorials and problems designed to teach the basics of quantum computing and Q#. While previous research has explored using these katas as an educational tool, this paper explores the value of these katas as a sole educational source. As there is concern that there may not be a sufficient workforce to continue quantum computing research, it is critical that an educational solution is found. Sixteen of Microsoft’s Quantum katas were completed for this purpose and evaluated based on time to complete, number of attempts, and any issues that arose. A lack of consistency across the katas and a lack of tutorials for crucial quantum computing concepts hinder the potential of the quantum katas. However, they still served as an effective tool for teaching quantum computing as well as Q#. The key finding of this research was that the quantum katas are very likely able to be adapted to focus very little on mathematical concepts, and instead approach quantum computing from a computer science perspective. The effectiveness of the quantum katas should be formally studied, and potential for a computer science focused adaptation should be explored.

**Introduction**

Quantum computing is an area of computer science with massive potential. It can bring significant improvements in areas such as machine learning, with a quantum binary neural network reaching 90% certainty in 6 steps as opposed to the classical implementation’s 57 steps [1]. In fact, some researchers have predicted that further development of quantum computing will result in a software engineering “golden age” [2]. Currently there are only a handful of quantum computing languages One quantum computing language of note is Microsoft’s Q#. Unlike typical quantum languages which are circuit based, Q# is an abstraction of this concept that treats quantum gates as functions and qubits as objects [3]. An open-source series of tutorials were recently developed, known as Microsoft’s quantum katas. Q#, and especially Microsoft’s quantum katas, have been used as a basis for teaching computer science students quantum computing successfully [4]. However, the only research on the effectiveness of Microsoft’s quantum katas was done by Microsoft and taught by Microsoft’s quantum researchers [4]. The objective of this research is to develop an experience report on Microsoft’s Quantum Katas from the perspective of a computer science student and without the explicit guidance of Microsoft’s quantum team. Currently, there is concern that there may not be enough researchers and programmers in the realm of quantum computing. The growth of the quantum computing workforce does not match the projected growth or budget allocated to the field [5]. If Microsoft’s quantum katas are effective, it may help to resolve this disparity. As Microsoft’s quantum katas are self-paced and self-taught tutorials, they do not require a trained instructor to facilitate education of quantum computing. There is ongoing research the suggests that self-directed learning may be as effective as instructor-based learning [6]. If Microsoft’s quantum katas could be analyzed from a perspective of self-directed learning, then it is possible that with some improvements Microsoft’s quantum katas could become the premiere way to learn quantum computing.

**Materials & Methods**

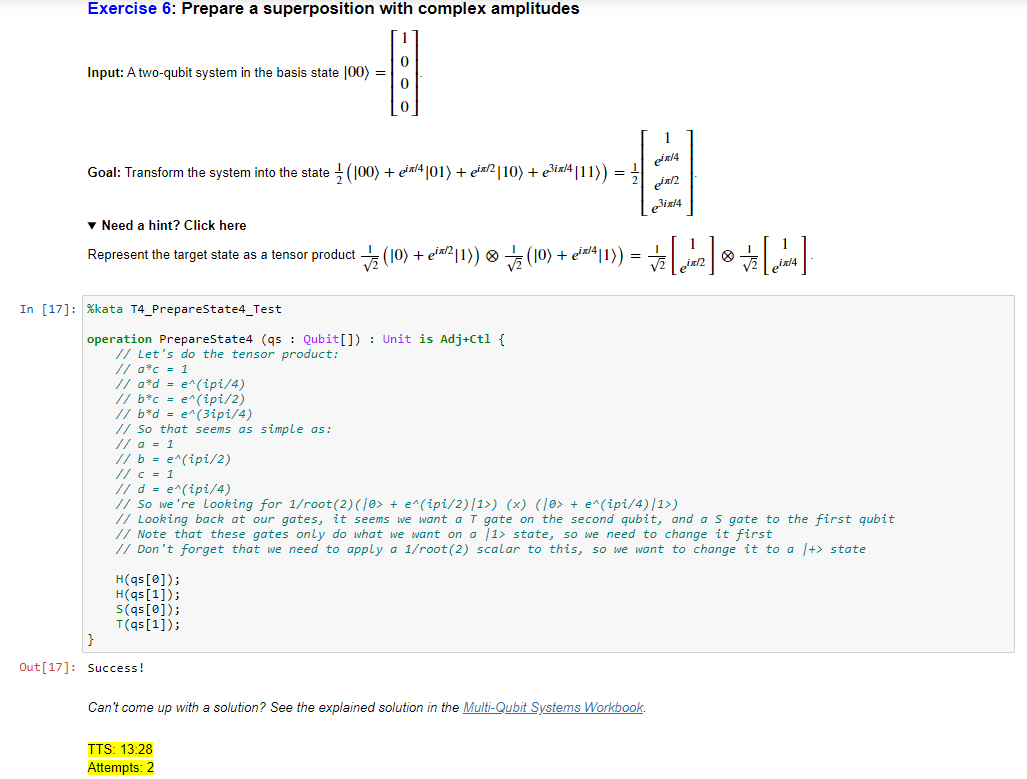
Eighteen quantum katas were selected to give a general overview of the Quantum Katas in an achievable timeframe. This amount was chosen based on Microsoft’s Introduction to the Quantum Katas page, found here (<https://docs.microsoft.com/en-us/quantum/tutorials/intro-to-katas>). However, only 16 were completed due to time constraints. However, these 16 should still give a sufficient overview of the Quantum Katas for the purpose of review. The original 18 that were selected are shown in **Table 1**. The two katas that were excluded have been struck through. There are three methods in which to run the Katas – online as a Jupyter Notebook, offline as a Jupyter Notebook, and offline using Visual Studio. For this report, the katas were ran as an offline Jupyter Notebook, as it allowed for documentation to easily be placed below the code. Each problem in each kata is attempted using only the resources given by the katas and without checking the answer workbook unless absolutely necessary. For each problem, the time to solve and number of attempts are tracked.



The time to solve is measured from when the problem was read to when it runs successfully. The only circumstance in which a timer would be paused is if there was a real life need that had to be attended to (eating, bathroom break, etc.). An “attempt” is defined as running the program with the expectation of it producing a correct result. If the program fails due to a typo or programming error, it does not count as an attempt. Logical reasoning is noted (typically in the form of code comments) for more difficult problems, and previous failed attempts may be saved if significantly different from the completed code. Any issues with the problems are noted, including but not limited to – needing outside resources, issues in wording, insufficient information. These notes, along with the time to solve and attempts, are written below each task in a kata. Additionally, the time to solve and attempts were tracked in an Excel file so that they could be analyzed statistically. Once enough katas were completed, a report was written describing the work in detail and analyzing the katas as a whole, for example what problems in each kata may need to be rewritten and how, what katas took the longest, what problems took the most attempts, etc. All data can be found at the following GitHub page: <https://github.com/MayoCh/Quantum-Katas>.

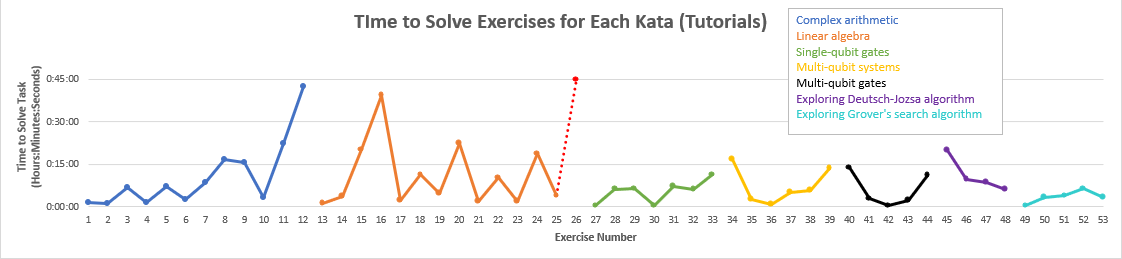
**Results**

All katas listed in **Table 1** were completed. The general outline for a completed kata is given in **Figure 1.** The data for the time to complete each task, the number of attempts taken, and the difficulty of each problem was graphed in various ways.



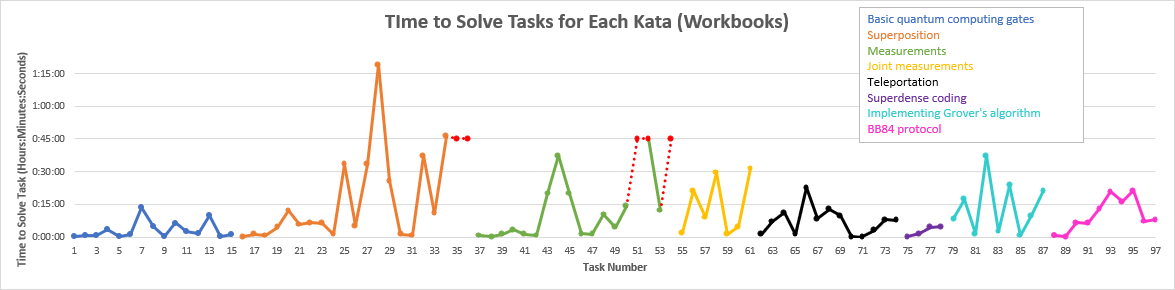
**Figure 1. An example of a completed quantum kata.** A completed kata consists of a given input, a desired output, possibly one or more hints, a code block with a predefined operation name, and a documentation block with time to solve, number of attempts, and possibly additional notes. Highlighted in yellow is the time to solve (TTS) and number of attempts.

The first way in which this data was graphed is creating a line graph for the time to complete for each task separated by kata (**Figure 2, Figure 3**). A noticeable trend in these graphs is that there exist many instances of a sharp increase then decrease in the graph. Most problems seem to stay below the 15-minute marker, but there are some katas in particular that often go above this threshold. Notably, some problems in the earlier katas took much longer than the later ones. This could be attributed to becoming more comfortable with Q# and quantum computing in general.



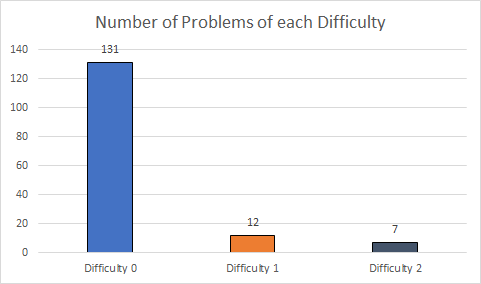
**Figure 2. Graph of the time to solve each tutorial kata.** The time to complete each task of each kata is shown, with each tick mark representing a time period of 15 minutes. Each kata is separated by color as indicated in the legend. Red dots indicate exercises not completed due to time constraints implemented later on in the project.

The most unfinished tasks were contained in two of the earliest workbooks, so this may be further evidence of this theory. However, it is also possible that this could be due to the lack of tutorials for the Superposition kata and a much too difficult set of problems at the end of the Measurements kata (**Figure 3**). While one unfinished exercise exists in the Linear algebra tutorial, nothing in the tutorial graphs goes above the 45-minute threshold (**Figure 2**). This is also the case for the workbook katas, excluding one particularly lengthy Superposition kata. Also of note is that the tutorials past the first two tend to resemble a curve more than a series of jagged edges. It is interesting that the composition of most tutorials are designed this way, possibly to encourage understanding and then eventually test said understanding.



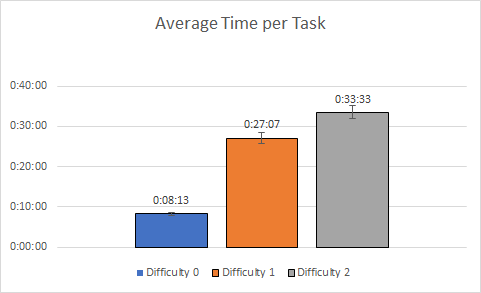
**Figure 3. Graph of the time to solve each workbook kata.** The time to complete each task of each kata is shown, with each tick mark representing a time period of 15 minutes. Each kata is separated by color as indicated in the legend. Red dots indicate tasks not completed due to time constraints implemented later on in the project.

The data was also graphed based on the stated difficulty for each problem in the katas. **Figure 4** shows the number of problems of each difficulty, while **Figure 5** and **Figure 6** compare each difficulty to the average time to solve and attempts to complete, respectively. We can see that in these graphs that difficulty markers are underutilized throughout the quantum katas. There only exist 19 problems that have difficulty markers throughout the 16 tested katas (**Figure 4**). However, of note is that in the reference implementation included with the quantum katas (essentially an undocumented workbook containing the solution to each task in the kata) there does exist difficulty markers that are not present in the actual kata. Perhaps this difficulty rating was simply forgotten about in later katas.



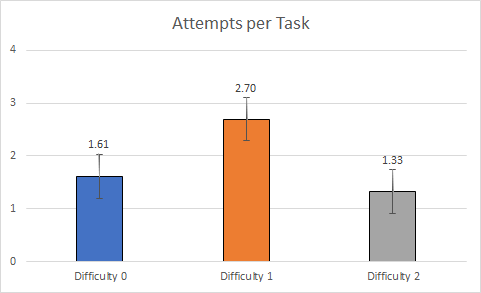
**Figure 4. The number of tasks of each difficulty as indicated in the quantum katas.**

Despite the low number of tasks marked with a difficulty, these difficulties do accurately relate to the amount of time that will be spent on them (**Figure 5**). While the measurement for tasks of difficulty 0 may be skewed due to trivial problems, the number of tasks marked with difficulty 1 and difficulty 2 are quite close. As the average time per task of difficulty 2 is higher than difficulty 1, this leads me to believe that the difficulty ratings are finely tuned and quite accurate, and it is a shame that they are not more prevalent throughout the katas.



**Figure 5. The average time to complete a task for each difficulty, with standard error bars.**

However, the number of attempts per each task is much more surprising. While once again difficulty 0 may not be an accurate representation of the difficulty level with its overabundance, it is interesting to see that problems of difficulty 2 actually had the lowest number of attempts on average. This may be an indication that these problems were not hard in the sense of not getting the output one expects, but rather hard to figure out how to solve it in the first place.



**Figure 6. The average number of attempts to complete a task for each difficulty, with standard error bars.**

All of the notes on issues in the problems were compiled and placed into **Table 2**. This table shows the areas of issues and offers a potential solution that could remedy each one. These issues should be self-explanatory, but there are two types of issues of note. The first is one of consistency. As seen above with the lack of difficulty ratings, the katas tend to showcase their open-source nature quite a bit. While minor inconstancies between katas is acceptable, the discrepancy between the katas and the website (which one could potentially do the katas on) is not. There are essential tutorials that are completely ignored by the website, which is the primary way people view the quantum katas initially. The second area of issue is lack of tutorialization for certain katas. If the quantum katas want to be a source to allow people to understand quantum computing and Q#, it is essential that every kata has an associated tutorial that requires one, especially the early katas. If potential learners struggle with early katas due to lack of tutorials, then it is possible they may quit the katas altogether and the quantum computing field may lose out on potential future workforce.

**Table 2. Tasks in each kata containing issues.** These issues may be subjective, such as poor wording, or objective, such as not enough information given. A suggested solution is given for each issue.

Table

Description automatically generated

**Discussion**

Overall, the tasks were quite effective in teaching quantum computing and Q#. I was able to keep up with the more advanced katas, and did not need to look outside of the resources linked by the katas and the Q# documentation. I now feel confident in the basic concepts of quantum computing and Q#, to the point where I was able to build a (albeit quite simple) random number generator in a task where one was meant to be imported. The concept and implementation of Grover’s algorithm in particular was very engaging, and if more time was available I would eagerly attempt the two additional katas on Grover’s algorithm and possibly create my own demonstration program. While complex mathematics is required to fully understand what exactly a quantum gate or algorithm means, I was able to complete a high number of katas with only a computer science perspective. If these katas were adapted to focus a bit more from a computer science perspective, it is absolutely possible to teach a computer science student basic quantum computing without tackling complex mathematics. In this area, I would say that the Microsoft Quantum Katas succeed in teaching Q# and quantum computing.

However, there are many noticeable flaws in the katas themselves that should be addressed before they are employed as a standalone method for learning quantum computing. As a large amount of the katas come from contributions from a variety of people that may or may not be affiliated with Microsoft Quantum, there are some minor discrepancies in formatting. The most concerning discrepancy, as mentioned before, is the one that exists between the introduction to quantum katas page and the quantum kata index. Each of these has a different name for every kata, which can lead to confusion. Furthermore, a lot of the hints in non-tutorial katas are lackluster. These hints often only give the most basic information, such as a reminder how to import the math class. All major areas of issue have been listed and discussed (**Table 2**), but it is important to reiterate how the lack of tutorialization for early katas can hinder the learning process. Despite how much I have improved in quantum computing and Q#, I would not have reached this point naturally as I would have given up on one of the early katas. The core quantum katas must contain a easy to follow tutorial with abundant hints in order to retain learners to the point where they can tackle more complex topics. Particularly, the end of the Measurements kata and the entirety of the Superposition kata may drive away potential learners due to their difficulty. These sections, however, are not even required from a computer science perspective. While the concept of Superposition is key to quantum computing, the extent which the kata tested is very intimidating to those attempting to minimize the amount of math performed to complete the katas. On the other hand, the Grover’s search algorithm tutorial perfectly allowed me to complete the much more complex implementation. It is crucial that the earlier tutorials have the same level of quality, as I feel that once one is able to explore quantum algorithms, they will be able to find something that sparks their interest to learn further. These results compare to the original results in an interesting way. While the original study suggested that more complex mathematic background is required for study of these katas, I found that very little mathematical prowess is needed more so than that of a typical computer science student [4]. Quantum computing does require advanced math to understand, yes, but it does not require such math to implement programs that may be tested and experimented with. Pattern recognition can take the place of mathematical understanding when the tasks are shaped with a focus on how a quantum algorithm works rather than why. My results concur with the study that students have an increased interest in the field after completion of the course, and that the quantum katas are very engaging [4].

While there is little research into the quantum katas or quantum education in general, in terms of self-education these results cannot really be compared to any computer science self-education research. While the katas are technically self-taught, the tutorials serve as a self-paced lecture, so very little individual learning is done. There are a number of notable limitations for this study. The most obvious is that this is a subjective experience report, not an objective study. As such, the data gathered from completing the katas are only relevant in giving a general overview on how one specific person experienced the katas. Additionally, each person who attempts the katas may have a learning style that may or may not be conducive to the style of instruction given. While the katas feature a lecture-based style of teaching, it can also be viewed from a hands-on perspective once some katas have been completed. However, if students are not able to get to the point where it can be viewed from a hands-on perspective, the katas will not be accepting to them. Possible future work includes a formal, objective study on a large number of individuals regarding the effectiveness on Microsoft’s quantum katas. Another future avenue of study may be to try presenting the information in the Microsoft quantum katas in a more automated format, such as educational software. Also, potential for a computer science focused adaptation of the quantum katas should be explored to see if there is a way to effectively teach quantum computing concepts without needing an extensive mathematical background.

In conclusion, the Microsoft Quantum Katas showcase great potential in teaching computer science students in a self-paced manner the basics of quantum computing and Q#. However, certain flaws may cause students to give up and lose interest before they reach the most engaging katas. A lack of tutorialization for some of the early katas create issues in advancing further, and a lack of consistency may cause students using the website primarily to miss important tutorials as well as causing confusion. In this experience report, it was discovered that the quantum katas may be viewed from a less mathematically rigorous computer science perspective, and thus may be able to introduce a large pool of potential learners to the field.

**References**

[1] B. Wang, “Quantum algorithms for machine learning,” *XRDS Crossroads ACM Mag. Stud.*, vol. 23, no. 1, pp. 20–24, Sep. 2016, doi: 10.1145/2983535.

[2] M. Piattini, G. Peterssen, and R. Pérez-Castillo, “Quantum Computing: A New Software Engineering Golden Age,” *ACM SIGSOFT Softw. Eng. Notes*, vol. 45, no. 3, pp. 12–14, Jul. 2020, doi: 10.1145/3402127.3402131.

[3] K. Svore *et al.*, “Q#: Enabling Scalable Quantum Computing and Development with a High-level DSL,” in *Proceedings of the Real World Domain Specific Languages Workshop 2018*, New York, NY, USA, Feb. 2018, pp. 1–10, doi: 10.1145/3183895.3183901.

[4] M. Mykhailova and K. M. Svore, “Teaching Quantum Computing through a Practical Software-driven Approach: Experience Report,” in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, New York, NY, USA, Feb. 2020, pp. 1019–1025, doi: 10.1145/3328778.3366952.

[5] G. Peterssen, “Quantum technology impact: the necessary workforce for developing quantum software,” p. 17.

[6] V. Isomöttönen and V. Tirronen, “Teaching programming by emphasizing self-direction: How did students react to the active role required of them?,” *ACM Trans. Comput. Educ.*, vol. 13, no. 2, p. 6:1–6:21, Jul. 2013, doi: 10.1145/2483710.2483711.