TILE and MASS, a retrospective

Steffen Dick[1], Teresa Dreyer[2], Christoph Bockisch[3]

Abstract: In conjunction with the QPED project, we have developed two teaching tools, MASS, an automated feedback tool for code, and TILE, a test-driven exercise paradigm. Over the course of three different iterations of the same university courses, we have collected data to see what effect MASS and TILE have had on the students who were confronted with both. For this we used a survey in a later module and a diagnostic assessment within the final exam in the module where TILE and MASS are introduced. We found a substantial and statistically significant positive effect in our exam data.

Keywords: survey, diagnostics, analysis, teaching testing, teaching software quality

1 Introduction

In an attempt to improve our teaching, in particular in early programming-related courses, at Phillipps University Marburg and within the QPED-project we have developed tools to help early learners achieve a better understanding of core concepts of object-oriented programming. Furthermore, those very same tools are supposed to sharpen their knowledge of software quality, especially principles of testing programming code with unit-tests. For those purposes, we've developed two major tools: An auto-assessment tool called MASS and an approach for creating tasks for students called TILE.

The *Marburg university auto ASsess System* (MASS)[4] was developed to give students early feedback on their written JAVA-code. Internally, MASS uses established software (like PMD) to generate a proto-feedback. This proto-feedback can then be used to generate a better understandable feedback for early learners. For example, MASS uses the JAVA compiler to check for any syntax-errors and uses the, sometimes hard to understand, output to generate a much easier to understand feedback.

The other tool, the *Test Informed Learning with Examples* approach [VDM22], or TILE for short, suggests that testing can be taught without detracting any valuable time from other important principles that need to be taught to early learners. TILE uses different domains of testing-tasks to accomplish this. For example, instead of using random texts for tasks, messages that insinuate the importance of testing can be used.

[1] Phillipps Universität Marburg, Germany dickst@informatik.uni-marburg.de
[2] Phillipps Universität Marburg, Germany tdreyer@informatik.uni-marburg.de
[3] Phillipps Universität Marburg, Germany bockisch@informatik.uni-marburg.de
[4] For more information on MASS visit `https://qped-eu.github.io/mass/`

doi:10.18420/seuh2024_05

To evaluate these teaching tools, we devised a survey and a diagnostic assessment. The survey was completed by multiple cohorts over the course of three years to get the perspective of the learners themselves. The diagnostic assessment was added to the final exam of the first-year object-oriented programming course, for an objective perspective.

Lastly, we performed several statistical analyses in order to examine the effect of the implementation of TILE and MASS. We analyzed the effect on the survey data as well as on the exam data. We found a substantial effect on the diagnostic assessment as well as limited effects on the scores in the self-assessment data. Therefore, we conclude that TILE and MASS added value to our lecture and were beneficial to our students.

2 Data

This section consists of four main parts. As we will be doing four simple linear regressions, the first part of this section will be about the prerequisites to make a linear regression possible. In the second part we will be discussing the results of a survey we gave to students participating in the Programmierpraktikum at Phillipps University Marburg which usually takes place in the second semester of their plan of studies. The third part consists of a diagnostic assessment we added to the exam of the first-semester lecture Object oriented Programming (OoP) that consists of an assessment of the students ability to spot quality flaws within a code snippet and writing unit-tests to check this very same code snippet. Both, the survey and the assessment, were continued over the course of three years and the first iteration of these was discussed in [DSB22]. Finally, the fourth part we will be comparing the subjective results of the study to the objective results of the assessment.

2.1 Part I: Prerequisites

For a simple linear regression, four assumptions [EGS13, chapter 18.13] should be considered. These assumptions can be formally tested and/or assessed by visually inspecting a relevant plot.

The first assumption, **Linear Effect**, specifies that the data should contain a linear effect. Additional non-linear effects can also be present. In practice, this assumption often is directly derived from the hypothesis and not formally tested.

The property **Independence of Regression Residuals** or more concretely the assumption **(No) Autocorrelation** specifies that after taking the model into account, no association is left between the regression residuals. It is often violated if the model is incomplete, i.e. not all factors that influence the outcome have been taken into account or if there are other unexplained dependencies between the measurement points.

Homoscedasticity means that the variance of the regression residuals is not different between different levels of the independent variable, i.e. between the different years.

The fourth and last assumption is the **Normal Distribution of Regression Residuals**. It means that the values differ between observed value and estimated value according to the regression follow a normal distribution. If this assumption is violated, it can sometimes be restored by linearly transforming the data. It is also possible to perform the Linear Regression using bootstrapping instead of the classical variant. Bootstrapped Linear Regression is robust against the normal distribution violation but has slightly varying results between instances.

assumption	Relevant Test	Criterion	Relevant Plot	Criterion
Linear Effect	None	None	Scatter-plot	Close to linear shape of point cloud
Autocorrelation	Durbin-Watson	not significant and statistic DW close to 2 (can range from 0 to 4)	Fitted-vs.-residuals plot	No trend visible in regression residuals
Homoscedasticity	Breusch-Pagan	not significant	Fitted-vs.-residuals plot	At each fitted value, the variance of the residuals looks similar to their shape at other fitted values.
Normal Distribution	Kolmogorov-Smirnov	not significant	Q-Q plot	Most residual values are on straight line, particularly in the middle of the theoretical distributions between -2 and +2 SD on the x-axis

Tab. 1: Overview over chosen assumption tests and relevant plots

Table 1 shows an overview over the assumptions and the assessment methods we have chosen. The choice and interpretation of the relevant plots for the visual inspection and of the formal test for autocorrelation is based on [FMF12, chapter 7.9] and [Co23, chapter 3.1]. For the choice of the test for the homoscedasticity assumption as well as for its assessment, we followed [FW11, chapter 6.5]. We will formally test the normal distribution assumption with the Kolmogorov-Smirnov test based on [EGS13, chapter 10.6.1].

When performing inferential statistics it must be taken into account that the sample size is an important factor for the size of the p-value[EGS13, chapter 8.2 and chapter 10.6]. Because of this, very small effects can become statistically significant if the number of participants is large. This phenomenon is relevant for the interpretation of the results of the linear regression and for the interpretation of the formal tests regarding the assumptions: The tests for formal assumptions test whether there is a significant deviation from the relevant assumption. Therefore, with an increasing sample size, even very small deviations can become statistically significant. Conversely, if the effects in the linear regression are small, very large sample sizes are needed in order for it to become statistically significant. This phenomenon needs to be kept in mind when interpreting the results[EGS13, chapter 8.2 and chapter 10.6]; the p-value should only be considered one part of the picture.

For the statistical evaluation we used the software R (version 4.3.1) using the libraries readxl (version 1.4.3), car (version 3.1-2), lmtest (version 0.9-40), ggplot2 (version 3.4.4)

and ggplotr (version 0.0.6). For aggregation, data rearrangement and manual data cleaning we used Microsoft Excel (Microsoft Office Professional Plus 2019).

2.2 Part II: Self-Assessment

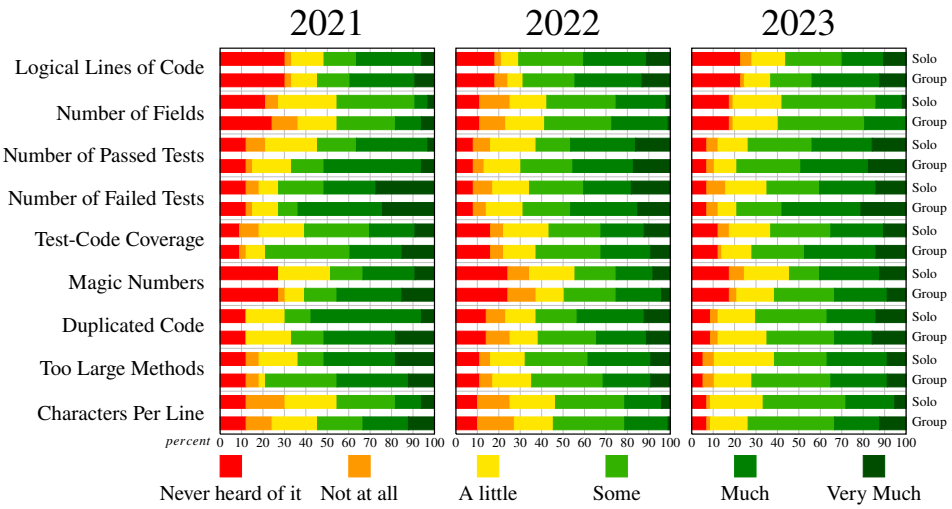

Fig. 1: Participants rating how much attention concepts receive when coding in 2021, 2022 and 2023.

Our first data collection action is based on a survey in which students self-assess how relevant that writing good quality software is for them. We asked the students to rate how much attention they pay to a selection of 9 different software metrics when producing software code in two different contexts: working alone and working within a group. The rating was done on a Likert-scale between one through six, with 1 being the most negative and 6 being the most positive, or from "never heard of it" to "very much" in words. The element "never heard of it" was interpreted as the lowest element on the scale. As described in [DSB22], a selection of software metrics had been made based on easiness of understanding and relevancy to the curriculum of the university of Marburg. This survey was done in the Programmierpraktikum which is a module that takes place in the semester after OoP, so participating students would have been introduced to TILE and MASS. The result of this part of the survey is depicted in Figure 1 sorted by year.

Figure 1 shows the 9 selected software metrics on the left hand side. Every software metric has two lines per year, the upper one for the solo context and the lower one for the group context. Their answers are portrayed within each line as a color coded bar. The negative answers, "never heard of it", "not at all" and "a little", are portrayed in red, orange and yellow respectively. The more positive answers, "some", "much" and "very much" are depicted in darkening shades of green, where "some" has the lightest shade and "very much" the darkest. The percentages are portrayed at the bottom of each individual chart.

As a preliminary examination, we can identify a positive trend: In every succeeding year, the number of students who selected "not at all" for some of the software metrics decreased with Magic Numbers, Logical Lines of Code and Duplicated Code being the exception. Some of the metrics have a little spike in 2022 but are still lower than they were at the start. The number of students who selected "never heard of it" also decreased overall.

Generally, as we can see in Figure 1, the assumption that students pay more attention to software quality metrics in a group context seems to be true. However, Duplicated Code does not follow this trend. In 2021, it can be observed that "a little" has been answered more often in the group context. This continues in 2022 with the "not at all" and "a little" being chosen more often. Even in 2023, "not at all" stays the same but "a little" was answered more often in the group context. At this point in time and with only this much data gathered, an explanation for such a behaviour can only be speculative without further research.

Since we introduced the test-focused TILE concept into the first-semester lecture OoP in the Winter Semester (WS) 21/22, we focus on the software metrics associated with testing. Furthermore, we introduced MASS into the OoP course of WS 22/23 which is also able to provide more individualized feedback on tests written by the students. With both of these tools integrated into the course, we expect to see a higher assessed importance of all software metrics, especially with the testing ones.

	2021		**2022**		**2023**	
	Solo	*Group*	*Solo*	*Group*	*Solo*	*Group*
Overall	57	64	60	62	63	69
Overall without testing	47	60	59	60	61	66
Only Testing	63	73	62	67	67	77

Tab. 2: Aggregation of green bars by context and year for the three testing metrics

For our general overview, we averaged the percentages of the positive elements on the Likert-scale per year per context to simplify the data and to make it easier to identify trends. The results of this average can be seen in Table 2. For the solo context, we can observe that the average steadily climbs by 3 percentage points each year from 57% to 60% to 63%. This shows a slight improvement in the perception of their importance. However, the group context starts at an average of 64% in 2021, then drops to 62% in 2022 and climbs to 69% in 2023 again. Even though the average dips by 2%pt. in 2022, this development is still positive.

Additionally, we averaged the software metrics excluding the testing related ones. Without these, we saw an improvement of 14%pt. between 2021 and 2023 in the solo context. The group context also shows a general, albeit less pronounced, improvement of 6%pt. between 2021 and 2023. Without the testing metrics, we do not see the previously observed small dip in 2022 and have a purely positive development in the three years.

Looking at the average of only the testing metrics, we can see the average improving from 63% to 67% in the solo context from 2021 to 2023. That same 4%pt. improvement from

73% to 77% can be seen in the group context as well, albeit with a higher starting number. Again, we have a small dip of 1%pt. and 6%pt. in 2022 for the solo and group contexts respectively.

For the statistical evaluation of the self-assessment data, we tested the hypothesis that the inclusion of TILE would improve the students' quality mindedness. To achieve this, we performed two simple linear regressions, both using the year as an independent (or predictor) variable. For the first linear regression, we looked at the effect of the year on the overall sumscore of the self assessment (dependent variable or outcome variable), "sumscore" for short. For the second linear regression, we focused on the six questions that we determined to be particularly closely related to testing: We examined the effect of the year on the sumscore of these questions. This variable we call "testing". If our hypothesis is correct and the effect is visible in the self-assessment, we would expect a positive slope in both simple regressions. For the simple linear regression, we followed the procedure detailed in [FMF12, chapter 7.4-5].

In a second step, we also explored whether the solo context vs. group context made a difference, either on its own or in interaction with the intervention, by using a multiple regression. We did not find that taking up these additional factors made a substantial contribution and are focusing on the simple linear regressions for the report.

		Estimate	Std. Error	*t* value	p-value (two-sided)	p-value (one-sided)
sumscore	(Intercept)	-3137.331	3919.282	-0.8	p=0.424	p=0.212
	year	1.585	1.938	0.818	p=0.415	p=0.207
testing	(Intercept)	-886.748	1611.891	-0.55	p=0.583	p=0.291
	year	0.450	0.797	0.565	p=0.573	p=0.286

Tab. 3: Results of the simple regressions

Table 3 contains an overview over the results of both of the simple linear regressions. The model predicting the overall sum score had a slope of 1.581: The overall score in the self assessment has grown by 1.59 points every year. The model predicting the testing related software metrics had a slope of 0.45: The score in the software metrics particularly related to testing has grown by 0.45 points every year. Because we had a directed hypothesis regarding the direction of the slope, i.e., that it should be positive because we expected an improvement over the years, the inferential statistical assessment can be based on the one-sided p-values. The effects of the year in both models were not statistically significant with one-sided p-values of p=0.21 and p=0.29, respectively. However, as these are small effects, more participants would have likely been needed in order for them to become statistically significant (see [EGS13, p. 8.2]).

Table 4 shows an overview over the results regarding the assumption tests and plot[5] inspection. While some of the formal tests had p-values below the significance level of

[5] Find relevant plots over at `https://github.com/Alucard2112/TILE_and_MASS_Plots`

	sumscore				testing			
assumption	statistic	p-value	visual	overall	statistic	p-value	visual	overall
autocorrelation	DW=1.717	p=0.043	+	+	DW=1.752	p=0.074	+	+
homoscedasticity	χ^2=3.592	p=0.058	+	+	χ^2=0.356	p=0.550	+	+
normal distribution	D=0.142	p=0.001	0	0	D=0.105	p=0.031	0	0

Tab. 4: Overview over assumptions for the simple linear regressions for both dependent variables

α=0.05, inspecting the relevant plots generally indicated an acceptable level of assumption violations. Additionally, while the Durbin-Watson (DW) test for possible autocorrelation violations also had *p-levels* below the significance level, the DW *test statistic* had levels close to the best possible score.

Regarding the assumption of normally distributed regression residuals, the picture is less clear. The Kolmogorov-Smirnov test was significant for both dependent variables. Inspecting the Q-Q plot showed that while most of the residual values are on a relatively straight line concordant with a normal distribution, in the tails there are some substantial deviations with some falling into the more relevant range between -2 and 2. Common linear transformations that can be used to mitigate the problem include taking the logarithm, the squareroot or the inverse of the dependent variable [FMF12, chapter 5.8]. For our case, neither of these transformations resulted in an improvement regarding the normal distribution assumption.

As the deviation in the untransformed data regarding the normal distribution assumption was mostly confined to the tails and the conclusions we are drawing from the data are limited, we concluded that the extent of the assumption violations was small enough to be acceptable.

In conclusion, the development in this part of the data is mostly positive. We have seen a positive development in the general importance of all nine software metrics in both contexts. Furthermore, the average of the six software metrics, those without the three pertaining to testing, is also positive. Excluding the six software metrics and only looking at the three that pertain to testing, we also saw a positive development, albeit with a small dip in the data set of 2022. To sum this development up, in all three aggregations (*overall*, *testing only* and *overall without testing*), we saw an improvement in our data.

2.3 Part III: Exam Data

For our second data collection action, we included a task in the final exams of the OoP lecture over the course of three years, which was also already outlined in the beginning of our study [DSB22]. In WiSe 2020 we've had 200 students taking the exam, 227 in WiSe 2021 and 185 in WiSe 2022. This task consisted of two sub-tasks: One focused on software

quality and the other on testing. In the first sub-task, the students were asked to fix a flaw within a given code-snippet. This ranged from missing documentation to bad performance or a missing input check. Within the second sub-task, the students were supposed to write a sufficient number of sufficient JUnit-tests to test the aforementioned code-snippet. The only thing we changed over the course of the three years was the code-snippet. We made sure that the snippet itself provided the same amount of fixable flaws.

	WS 20/21			**WS 21/22**			**WS 22/23**		
	Combined	Quality	Tests	Combined	Quality	Tests	Combined	Quality	Tests
Points	34%	41%	27%	38%	36%	40%	64%	59%	69%
ITC	r=0.59	r=0.44	r=0.52	r=0.69	r=0.31	r=0.71	r=0.63	r=0.36	r=0.64
Corrected ITC	r=0.53	r=0.41	r=0.48	r=0.64	r=0.27	r=0.63	r=0.58	r=0.33	r=0.6

Tab. 5: Item-Test-Correlation (ITC) and achieved points sorted by semester

Table 5 shows the achieved points, Item-Test-Correlation (ITC) and the Corrected Item-Test-Correlation (CITC) for the task and both sub-tasks. CITC is distinguished from ITC by subtracting the task itself from the overall achieved points in the correlation which makes it more accurate. Before the introduction of TILE and MASS, we saw a total of only 34% of achieved points in the task as a whole in WS 20/21. We also observed that students achieved more points in the quality focused task than in the testing one. This indicates that students had more trouble with testing than with quality in general.

Over the three years, we saw an improvement in the achieved points in both sub-tasks. Even though the quality task decreased by 5%pt. in WS 21/22, the testing task increased by a whopping 13%pt. which brings the overall score up to 38%. This is an increase of 4%pt. in comparison to WS 20/21 for the combined task. WS 22/23 saw another increase in both sub-tasks, landing at 64% for both. Especially the testing sub-task saw a huge increase from 27% to 69% over the three years. Because of TILE's focus on teaching testing, this positive trend is to be expected. The quality task also increased from 41% to 59% which is also a positive trend. The TILE approach should be the reason for this increase as it is based on test-driven development which increases general quality awareness in students [DJS08].

Both, ITC and CITC, saw a slight increase from 0.59 to 0.63 and 0.53 to 0.58 respectively over the three years with a small dip in WS 21/22. This can be explained in the achieved points for the combined task as they are stable between WS 20/21 and WS 21/22 and almost double in WS 22/23. If more points are achieved in this task and the overall points of students remain stable, then the correlation should worsen a little.

		Estimate	Std. Error	*t* value	p-value (two-sided)	p-value (one-sided)
sumscore	(Intercept)	13310.081	1780.333	7.476	p<0.001	p<0.001
	year	-6.563	0.881	-7.451	p<0.001	p<0.001
testing	(Intercept)	-33129.382	3025.488	-10.95	p<0.001	p<0.001
	year	16.42	1.497	10.97	p<0.001	p<0.001

Tab. 6: Results of the simple regressions for the exam results

For the statistical evaluation of the exam data, we looked at the effect of the year on the overall exam score as well as on the result in the diagnostic task. Like for the self-assessment data, we did so by performing two simple linear regressions using the year as independent or predictive variable and using the overall sumscore ("sumscore") and the score in the diagnostic tasks ("testing") as dependent variables, respectively. We transformed the variables "sumscore" and "testing" into percentages in relation to the best possible total score in order to compensate for differences between the exams regarding total scores.

Over the three observed years, we found a clear negative trend in the overall exam score: As Table 6 shows, with every year, sumscores were around 6.6 percentage points lower. This effect was statistically significant with a very low p-value at $p<0.001$.

In contrast, for the diagnostic task, we found a clear positive trend. With every year, the score in the diagnostic task was around 16.4 percent higher. This effect was statistically significant as well with a very low p-value at $p<0.001$.

For both simple linear regressions, we did not find any evidence for assumption violations with exception of the normal distribution assumption for the dependent variable "testing": The Kolmogorov-Smirnov test was significant with $D = 0.103$ and $p<0.001$ and the QQ-plot gave indication for some relevant deviation from the normal distribution hypothesis as well. For this reason, we performed a bootstrapped version of the relevant simple linear regression following the procedure detailed in [FMF12, chapter 7.10]. The bootstrapped variant has the advantage of being robust against distribution assumption violations. The difference between the results for the classical and the bootstrapped version of the linear regression were negligible so we are focusing on the classical variant.

In conclusion, we saw an increase in student-proficiency of fixing quality flaws and writing tests. Furthermore, this did not bring down the ITC or CITC in any major way. Instead, we observed an overall ITC and CITC of above 0.4 which means that a good correlation exists [MK12, Chapter 4]. The statistical evaluation of the data also showed a strong significance within our data.

2.4 Part IV: Comparing Results

We were able to identify a positive trend in the subjective data concerning the overall attention paid. This specific positive trend is mimicked in the general achieved points in the combined task. Though it is a steady but slow increase in the subjective data, the change in the objective data was a steep jump. The only divergence we can find is the steepness of the improvement and not in the trend itself, which underlines the plausibility of our results.

Beginning with the software quality sub-task, we saw a slight decrease from 41% of achieved points to 36% from WS 20/21 to WS 21/22 before rising again to 59% in WS 22/23. Since the sub-task concerned itself with the quality of the code-snippet, specifically excluding tests, we use the average of the metrics without testing for a comparison. In that we saw the

average rising from 47 to 61 in the solo context and 60 to 66 in the group context. Both, the subjective and the objective results, show a positive trend. However, they behave differently over the three years as the subjective results do not mimic the dip in 2022 that we saw in the objective results.

Lastly, the sub-task that concerns itself with writing meaningful tests for the code snippet saw a positive trend from 27% in WS 20/21 69% in WS 22/23. Software metrics focused on testing also saw an increase from 63% in WS 20/21 to 67% in the solo context and 73% to 77% in WS 21/23. However, we observed the dip in 2022 again, albeit by only 1%pt. and 6%pt. respectively. Once more, both results share a positive trend, with the subjective results depicting that dip again.

Looking at the statistical evaluation, the analysed data tentatively indicates that the inclusion of the TILE approach did have a positive effect on the students' competence in testing specific tasks but not on the overall object-oriented programming competence. The self assessment data shows a small trend but as the effects are small and not statistically significant the self-assessment data alone is not sufficient for drawing conclusions regarding the hypotheses. In combination with the exam data, however, the trend becomes clearer. Particularly, the contrast between diagnostic task and overall score in the exam data is interesting: While the overall scores became statistically significantly worse over time, there was still a substantial and statistically significant positive trend for the testing-related task. At this point in time, we can not completely rule out that the, at the time, ongoing pandemic has had an effect on the students' performance as the later cohorts, specifically those in WS 21/22, were more or less affected by ongoing school closures and the online approach of the university.

3 Threats to Validity

First of all, because of the nature of this research, we were not able to completely control our experiment. We've had outer influences that may have tainted our data like, of course, the pandemic. This affected all of our data collection.

Over the three years, the questions within the survey remained unchanged other than the inclusion of a comment field from year 2 onward. The only difference between the survey data collections for the three years is that we changed the participation in the survey from being optional in the first iteration to mandatory in the second and third iterations. This change caused the number of participants to skyrocket but might have led to students giving nonsensical answers. We adjusted conflicting answers by setting both of them to "did not know the metric". This interpretation of the data is quite conservative. This only applied to only 0.5%, 1.3% and 0.6% of the answers in the first, second and third year, respectively. As this discrepancy is not present in every metric and the percentage of answers affected is small enough, we attribute this to accidentally clicking on the wrong answer and not malicious intent.

There are a few data points of students who answered "Never heard of it" for each one of the self-assessment questions. It is possible that these results were produced by students who just wanted to quickly complete the survey and did not genuinely answer the questions. When excluding these data points, we found that the discovered effect almost vanishes. In retrospect, we've also identified a potential problem with the phrasing of the Likert-scale data points: "Never heard of it" can also be interpreted as a neutral answer instead of the worst possible one. This can have skewed the Likert-scale into a negative direction.

Regarding the statistical evaluation, the possibly violated normal distribution assumption is another threat to validity. While we considered the extent of the assumption violations for the self-assessment data acceptable, other more robust analyses could be explored for this part of the data in the future. Also some students repeated the course and participated in the survey more than once in self-assessment or/and exam, generating some overlap and probably some autocorrelation within the data. This could be examined with analyses that take into account the hierarchical structure of the data.

In addition, the assessment task was new in WS 20/21 whereas from WS 21/22 onward students may have practiced it when using old exams in their preparation.

4 Related Work

In our previous work [DSB22] we have already described the set-up of our experiment and the first set of data. But of course, only now we are able to analyze the development over the years. Another precursor to this paper is a joint publication on the TILE approach by Doorn et al. [Do23]. There we show a preliminary result of the TILE-approach within university teaching. In that paper we only looked at the diagnostic test over two years and did not analyze the data in a statistically rigorous way as in this paper.

Desai et al. [DJS08] conducted a meta study of experiments done at universities with teaching test-driven development (TDD) which TILE is based upon. They found that most studies found that TDD improves the students' quality awareness and increases their productivity. However, they found that an optimal point of introducing students to the concept has not yet been found as of the publication of their research.

5 Conclusion and Future Work

Over the three years that we have collected data, we found that including TILE and MASS has had a positive effect on our students' comprehending of software quality in both, their subjective view and in a more objective view. Specifically, their skill in writing correct and comprehensive unit-tests for a given software increased. Even though we could not find a statistically significant trend in the survey data (subjective view), we were able to find that the substantial effect in the diagnostic assessment *was* statistically significant. Therefore,

we conclude that the introduction of TILE and MASS has positively impacted the students' understanding of software quality and the importance of writing unit-tests for their code, even though they may underestimate their own quality understanding as is shown in the subjective data.

As for the future, we would need to continue the data-collection on TILE and MASS and improve upon their integration into the lecture. It would also be beneficial to add a field to the survey where students are able to share their own interpretation of the most negative point on the Likert-scale so it would be easier for future analyses to interpret their data.

Acknowledgements

This work is partly funded by the Erasmus+ project *Quality-focussed Programming Education (QPED)*, 2020-1-NL01-KA203-064626.

References

[Co23] Cornillon, P.-A.; Hengartner, N.; Matzner-Løber, E.; Rouvière, L.: Régression avec R. EDP Sciences, Les Ulis Cedex A, 2023.

[DJS08] Desai, C.; Janzen, D.; Savage, K.: A Survey of Evidence for Test-Driven Development in Academia. SIGCSE Bull. 40 (2), pp. 97–101, 2008.

[Do23] Doorn, N.; Vos, T.; Marín, B.; Bockisch, C.; Dick, S.; Barendsen, E.: Domain TILEs: Test Informed Learning with Examples from the Testing Domain. In: Research Challenges in Information Science: Information Science and the Connected World. Springer Nature Switzerland, Cham, pp. 501–508, 2023.

[DSB22] Dick, S.; Schulz, S.; Bockisch, C.: A study on the quality mindedness of students, Software Engineering im Unterricht der Hochschulen (SEUH 2022), 2022.

[EGS13] Eid, M.; Gollwitzer, M.; Schmitt, M.: Statistik und Forschungsmethoden. Beltz, Weinheim, 2013.

[FMF12] Field, A.; Miles, J.; Field, Z.: Discovering Statistics Using R. SAGE publications, London, 2012.

[FW11] Fox, J.; Weisberg, S.: An R Companion to Applied Regression. SAGE publications, Thousand Oaks, 2011.

[MK12] Moosbrugger, H.; Kelava, A.: Testtheorie und Fragebogenkonstruktion. Springer-Verlag, Heidelberg, 2012.

[VDM22] Vos, T.; Doorn, N.; Marín, B.: Test Informed Learning with Examples. In: CSERC '21. ACM Digital Library, pp. 1–2, 2022, ISBN: 978-1-4503-8576-3.