

Submission Date: 30/09/2017

# Evaluating the use of EiPE and the Development of a Domain- Specific Language for the Novice Programmer

Supervisor: Associate Professor Raymond Lister

Thesis submitted in fulfilment of the requirements for the  
degree of:

Bachelor of Science (Honours) in Information  
Technology.

At the:  
**University of Technology Sydney.**

Author: Thomas A. Pelchen

## Abstract

The purpose of this paper is to identify whether the language used by a novice when responding to an Explain in Plain English question can be used to identify their current transition to the expert. This paper will involve the analysis of the results and responses from a cohort that undertook their final examination for an introductory programming subject (CS1).

When the responses to the Explain in Plain English questions have been categorised to the SOLO taxonomy and given a mark accordingly, prior research has shown that the transition of a novice to the expert is evident by said mark. This paper presents an alternate way the Explain in Plain English questions can be used to identify the transition of the novice: that the transition is evident through the language used by the novice in their response.

This paper also addresses the concerns educators may have over the suitability of the Explain in Plain English question as an examination method. Firstly, by showing that as the marks received for the Explain in Plain English will correlate to marks for the traditional examination methods; that when included, the Explain in Plain English questions will not skew the marks of the cohort. Secondly, by showing that the ability to answer an Explain in Plain English question is not dependent on the English language proficiency of the student; educators can be assured that level of English language proficiency required to answer these questions is no greater than what was required for admission into their course.

This paper confirms the findings of previous research regarding the relationship between tracing, reading and writing code. That a skill in tracing is a pre-requisite to the abilities of reading and writing code, a relationship that is more apparent through nonparametric tests. It also extends these findings by using both a larger test population (in a single institution) and providing a Phi-coefficient value for identifying the direction of this relationship, a statistic missing in the previous tests.

To identify whether the language used within the answers to the Explain in Plain English questions can be used to identify a novice's current transition to the expert programmer, this paper considered two possible indicators: (1) The first unsuccessfully looked at direct and indirect tautological reference of the question in the words of the answers, finding that a tautological response is not apparent in the responses of this cohort; (2) The second found strong evidence within the responses for a domain-specific language. The use of this domain-specific language correlated strongly with the total mark received for the Explain in Plain English questions; a mark which prior research has shown to be an indication of the current transition of a novice.

The paper concludes with the discussion of ways to further explore the findings of this paper, noting the possible benefits that the analysis of the domain-specific language used by a novice may bring.

## CERTIFICATE OF ORIGINAL AUTHORSHIP

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Submission of Final Thesis:

Signature: \_\_\_\_\_  
Production Note:  
Signature removed  
prior to publication.

Date: 30/09/2017

## Ethics Clearance

Human Negligible Low Risk Ethical clearance was granted for this project by the University of Technology Sydney Human Research Ethics Committee under approval numbers ETH16-0340.

## Acknowledgements

After such a time of my life spent on the development of this thesis, I have a number of people I wish to thank dearly.

I wish to thank my family for putting up with me while I have been sequestered away in my room for weeks at a time, dedicating such a large proportion of my time and effort into this course and thesis.

I wish to thank my friends for sticking with me while I've become (more of) a social recluse throughout this year and a half. I also wish to thank them all for putting up with me when I've almost bored them to tears, talking about that "interesting" observation I've just made in my data set.

I also wish to thank Donna Teague, a previous PhD student of my supervisor, who has helped considerably with the structure of my thesis. Although we have never spoken to each other, the review of her thesis's structure has been an inspiration when considering the order of my own.

Last, but very much not least, I wish to give my heartfelt thanks to my supervisor Raymond Lister. Who, for some reason, decided to give this student with a half-baked idea a chance. Throughout hours of meetings involving: scanning documents, head-scratching assignments and far too many document revisions, he has provided me with immense support.

As with most fine things in life, the few words that I have written here for all of you cannot fully express the depth of the thankfulness I feel. Once again, thank you all.

~ Thomas Albert Pelchen

## Table of Contents

Abstract.....	i
CERTIFICATE OF ORIGINAL AUTHORSHIP .....	ii
Ethics Clearance.....	iii
Acknowledgements.....	iv
Chapter 1. Introduction.....	1
1.1. Purpose of This Paper .....	1
1.2. The Background .....	2
1.3. Who Will Make use of this Paper?.....	4
1.4. Research Questions .....	4
1.5. Significance.....	5
1.6. Structure of this Thesis.....	5
Chapter 2. Literature Review.....	7
2.1. Summary of the Teaching and Assessment of Programming .....	7
2.2. The Current Research.....	12
2.3. Conclusion: The Gap – Unexplored Territory .....	15
Chapter 3. Approach.....	19
3.1. The Exam .....	19
3.2. Retrieval and Storage of the Data .....	20
3.3. Conducting Word-Based Tests.....	21
Chapter 4. Results .....	26
4.1. Introduction .....	26
4.2. The Suitability of Explain in Plain English Questions.....	28
4.3. Tracing as a Pre-Requisite Skill .....	36
4.4. Measuring Tautology: An Indication of SOLO Level .....	42
4.5. Domain-Specific Language.....	49
Chapter 5. Conclusion.....	56
5.1. The Research Questions .....	56
5.2. The Findings of this Paper .....	57
5.3. Significance and Future Research.....	63
Chapter 6. Appendix .....	70
6.1. Rules for Data Input .....	70
6.2. The Program: Class Diagram .....	71
6.3. The Database .....	72
6.4. Indirect Reference: Distinct Words Test.....	77
6.5. Repeated Words in Answers to Q32 .....	79
Reference List .....	82

## Chapter 1. Introduction

### Keywords

Introductory programming, novice programmer, learning programming, abstract reasoning, Explain in Plain English, tautology, Domain-Specific Language.

### 1.1. Purpose of This Paper

It is the goal of any educator to ensure that their students are educated in the domain that they are teaching, but how can we determine the student has been educated adequately? Educators will often first look at the abilities and knowledge of an expert, deciding what knowledge and abilities are required for the novice to be labelled an expert. It can, therefore, be restated that the goal of any educator is to assist the transition of their student from novice to expert. However, a similar question is needed to be asked, how can the educator determine the distance their student has transitioned from novice to expert?

Educators will often use a number of examination methods, throughout a course, which is intended to determine if their student exhibits the expected abilities and knowledge of an expert. This is no different to the domain of programming education, where a final examination is often used to determine this transition. Determining if the student is on the way to becoming an expert is particularly important in the field of programming, as the time spent in transition from novice to expert programmer has been argued to exceed the bounds of a regular four-year computer science course (Winslow 1996).

It is, therefore, crucial that the examination methods used within a final exam can easily and consistently determine the transition from novice to expert. Originally developed by Whalley et al. (2006) to determine a student's ability to read and explain code, subsequent research has shown that responses to the 'Explain in Plain English' question can consistently determine the transition of the student from novice to expert (Lister et al. 2006; Murphy, McCauley & Fitzgerald 2012).

Prior research regarding the analysis of the 'Explain in Plain English' question's responses involved categorising these responses to the levels of the SOLO taxonomy; these categories were then used to determine the transition of the tested students. However, there has yet to be an analysis of the words used in these responses. If the marks assigned to these questions are determined by what is read and categorised in the response, then it can be logically assumed that there is some observable structure required in the language of the response for it to be marked as correct.

*This paper proposes that the transition from novice to expert programmer can be gauged by analysing the responses to the 'Explain in Plain English' question within a final examination. That not only do the marks received to these questions will indicate their transition but so does the language used in their answers.*

This paper concerns the answers (or responses) from the 'Explain in Plain English' examination method, a type of question given to novice programmers in their final examination for their introductory programming subject. An overview of this examination method will be provided in section 1.2.2.

For the sake of clarity, several words will need to be identified for the scope of this paper. The following paragraphs will define these words:

The term *introductory programming* refers to the first programming subject undertaken at a university. Sometimes referred to as a CS1 subject, the goal of this subject is to teach the basics of programming. Exactly what is defined to be ‘the basics’ of programming is still a matter of debate. This paper views ‘the basics’ as an ability for a novice to understand and explain a given piece of code, this may not equate to the ability to abstract a solution to different problems or programming languages. The novice should be able to write simple code at the same level of complexity that is asked for them to understand and explain.

The *novice programmer* is to be defined as an undergraduate student who has little to no experience with any programming language. This paper will be looking at students within a single institution and cohort whose age is 17 years or above.

The term *tracing* refers to the hand (or manual) execution of programming code. A tracing question requires a student to execute code manually before a conclusion can be met, this answer cannot be derived from simply looking at the code.

The abbreviation *EiPE* is short for ‘Explain in Plain English’ and refers to an examination method that first appeared in a paper by Whalley et al. (2006). For the sake of brevity, the term ‘Explain in Plain English’ will be referred to as EiPE when appropriate.

## 1.2. The Background

The following section will provide the background for this paper; it will be separated into two sections of which understanding is required to interpret the purpose of this paper. These sections are: (1) a brief outline of the history regarding the education of programming leading to the development of the ‘Explain in Plain English’ question; (2) an overview of what the ‘Explain in Plain English’ question involves and a summary of the research surrounding it.

### 1.2.1. Brief History of Research in the Educational Programming Domain

Although being a relative newcomer to the education domain, there has been extensive research conducted on the field of educational programming, especially at the introductory level. However, after the advent of the McCracken et al. study in 2001, the direction of research shifted. Before this point, the research had been primarily focused on the differences found when comparing the model of a novice to the mental model of an expert. After all the research and teaching based on mental modelling, the McCracken et al. study determined that the majority of novices performed quite poorly. Due to the multi-national and multi-institutional nature of the test, this has important implications. They asserted that the poor performance of the novices was, in fact, systemic and that poor performance of the novice was a direct result of their inability to problem solve.

In response, Lister et al. (2004) investigated the cause of the problems identified but not explained in the McCracken et al. study. Lister et al. argued that before a novice could begin to problem solve, it is required that they have a basic handle on some pre-requisite skills. However, as noted by the Whalley et al. (2006) paper, the questions in the Lister et al. paper were not developed using a framework intended for determining competency in these skills. Whalley et al. developed a number of questions to determine competency in these skills, one of which was an entirely new examination method and is the focus of this paper: the ‘Explain in Plain English’ question.



A full evaluation of the literature surrounding the events listed above is provided in section 2.1 of the Literature Review Chapter.

### 1.2.2. Overview of the 'Explain in Plain English' (EiPE) Question and its Research

As noted earlier, the Whalley et al. (2006) study included questions specifically designed to test novices for competency in skills which are a pre-requisite for the ability to problem solve. Of particular importance was to test for a novice's ability to read and explain code, however, constrained to the setting of a final exam, an examination method had yet to be developed that could test for these abilities. Called the 'Explain in Plain English' question, this question requires a novice to read a given piece of code and explain its purpose in simple English. Previously, examining the ability for a student to read and understand the purpose of code often required completion of take-home assignments, something unsuitable within the time constraints of a final exam. Figure 1 below provides an example of an EiPE question:

```
In plain English, explain what the following
segment of code does:

    bool bValid = true;
    for (int = 0; i <iMAX-1; i++)
    {
        if (iNumbers[i] > iNumbers[i+1])
        {
            bValid = false;
        }
    }
}
```

Figure 1 - Background: 'Explain in Plain English' Question Recreated from Source (Whalley et al. 2006, p.5).

The correct solution to the above question requires an answer that describes the purpose of the given code simply; providing a line by line read-through of the code would be marked as incorrect. When categorising the response to the levels of the SOLO taxonomy, Whalley et al. were able to determine the novice's ability to read and explain code. Both Whalley et al. (2006) and Lister et al. (2006) note a relationship between the ability to read and write code, arguing that novices who were unable to '...read a short piece of code and describe it in relational terms [were] not intellectually well equipped to write similar code' (Lister et al. 2006, p. 5).

Studies subsequent from the Whalley et al. study found further benefits of the EiPE examination method, noting that the responses from the EiPE questions could be mapped consistently to the levels of the SOLO taxonomy (Lister et al. 2006). This enables educators to determine the current ability of the novice, showing how far (or little) they have transitioned to the expert (Murphy, McCauley & Fitzgerald 2012).

A full analysis of the papers relevant to the EiPE question, as well as other areas of current research that relevant to the research questions of this paper, are detailed in section 2.2 of the Literature Review Chapter.

### 1.3. Who Will Make use of this Paper?

The main stakeholder of this paper would be educators who are interested in the development and learning of novice programmers.

These educators would be interested in both the use of the ‘Explain in Plain English’ question and the information that can be found in the language of their student’s answers to the said questions. By using the findings of this paper, educators can both be confident in the use of the EiPE question as an examination method and use the responses to the EiPE questions to identify the current transition of their students from novice to expert programmer.

### 1.4. Research Questions

It was stated earlier that this paper proposes that *the transition from novice to expert programmer can be gauged by analysing the responses to the ‘Explain in Plain English’ question within a final examination. That not only do the marks received to these questions will indicate their transition but so does the language used in their answers.*

In order to explore whether the above is the case, this proposition will be broken down into research questions. Those questions which require more definition or measurement will be operationalised further.

- R1** Are the Explain in Plain English questions a suitable examination method?
  - R1.1** Are the results given for the Explain in Plain English questions comparable to the results from traditional examination methods (multiple choice and writing code)?
  - R1.2** Is the ability to answer the Explain in Plain English questions dependent on the student’s English language proficiency?
  
- R2** Is the ability to trace a pre-requisite skill for the ability to read or write code?
  
- R3** Is the language used in the answers to the Explain in Plain English questions indicative of the transition from novice to expert programmer?
  - R3.1** Is a tautological response evidence for this transition?
  - R3.2** Do students within a certain mark range show shared language within their answers, could this be used to show the transition?

It must be stated that research question **R2** evolved from the exploration of the data, as prior research has argued that the ability to trace will precede the ability to read or write code (Lister, Fidge & Teague 2009; Venables, Tan & Lister 2009).

A complete review of the gap in the current literature, of which this paper will fill by exploring the above research points, can be found under section 2.3 of the Literature Review Chapter.

### 1.5. Significance

The significance of this paper lies in the analysis of written responses collected from novices when answering a series of EiPE questions within a final exam. Prior research regarding the EiPE question have been quantitative, that is they have been marked according to criterion, where the mark received for the questions have been analysed. The research of this paper concerns the language used in the responses to the EiPE question, confirming that there is a relationship between the language that is used within these responses and the marks that the novice receives.

This paper also explores the suitability of the EiPE question as an examination method, identifying the examination method both correlates with traditional examination methods and requires an English language proficiency from the student that is no greater than what was needed for admission into their course.

In terms of its findings regarding the relationship between tracing, reading and writing code, this paper supports the findings of previous research in noting that the ability to trace is a necessary but not sufficient skill. At 334 students, this paper will be the largest study of tracing as a pre-requisite skill within a single cohort of the same university. It is also the first to use Phi-coefficient to quantify the direction of this nonparametric relationship.

Figure 2 below, graphically illustrates the gaps of which the research of this paper will fill:

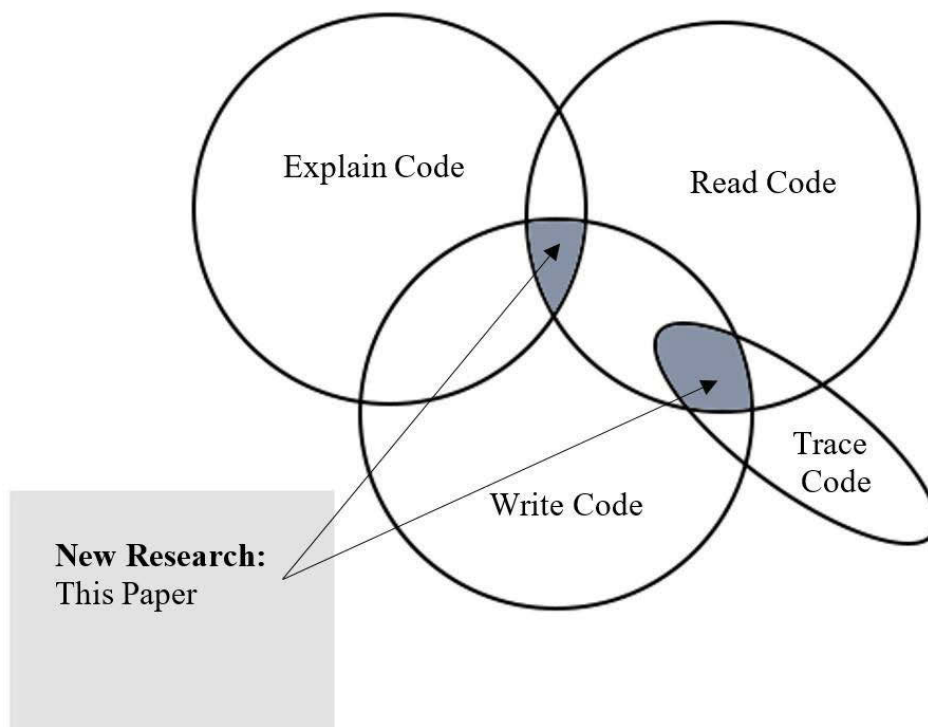


Figure 2 - Significance: Gap in the Current Research

### 1.6. Structure of this Thesis

This thesis will be structured into a number of main chapters, with each point within it referred to as a section. A brief description of each chapter (excluding this one) and its sections are as follows:

**Chapter 2. Literature Review:**

This chapter will be split into three sections, the first providing an overview of the literature that precedes the current research, detailing the prior and current schools of thought in the novice programming educational domain (2.1.). The second will review the current research that is pertinent to the research questions of this paper (2.2.). The third will identify gaps in the current research, of which have prompted the research questions of this paper to fill said gaps (2.3.).

**Chapter 3. Approach:**

This chapter will be split into three sections, each detailing a separate component of this paper's tests. The first will provide an overview of the final exam, outlining the parts and content of the exam that the novices undertook (3.1.). The second outlines the process of which the novice's responses and marks were retrieved from their written form and organised in a manner that was testable (3.2.). The third will outline the tests themselves, explaining the logic and process undertaken to reach the results described in Chapter 4 (3.3.).

**Chapter 4. Results:**

The largest component of this paper, this chapter will be split into five sections. The first will introduce the chapter, reviewing the potential bias that missed answers could have had on the results and concluding the effects to be minimal at best (4.1.). Sections 2-5 each review part of the research questions outlined previously; each section will begin with a restatement of the research question guiding it (4.2.-4.5.).

**Chapter 5. Conclusion:**

This chapter will contain three sections. The first will summarise the research questions of this paper (5.1.). The second will outline the findings of this paper, structured by the research questions (5.2.). The third and last will detail the significance of this paper's findings, concluding with the identification and discussion of future research this paper has prompted (5.3.).

## Chapter 2. Literature Review

For the novice programmer, learning how to program can be a daunting task. It is the goal of the educator in the domain of programming to accommodate the transition of their student from novice to expert. Consequently, it is crucial that an educator can assess a novice's progress during this transition. The research into the teaching and assessment of this field has been varied and, although well debated (Schulte et al. 2010), a solid consensus of how to teach programming has yet to be reached.

It is not the purpose of this chapter to provide a complete history of the teaching and assessment of programming; there are existing resources which do this in detail. The purpose of this chapter is three-fold: (1) to provide historical context into the development of the EiPE question; (2) to provide an overview of existing research into the EiPE question; (3) to highlight where the current research falls short in answering the research questions of this paper.

### 2.1. Summary of the Teaching and Assessment of Programming

To illustrate the need for the EiPE question, a brief history of the teaching and assessment of programming needs to be understood; of which the history can be separated into two parts. The first will review the old schools of thought, of which were which were pioneered in the days of punch cards and refined before the turn of the century. The second occurred after this, spurred by the consequences of the McCracken et al. (2001) study.

#### 2.1.1. Prior School of Thoughts

In the 20<sup>th</sup> century, owing to its rapid development as a field, the teaching and assessment of programming was often an afterthought by educators. For the educator, the methods of the teaching and assessment of programming were often to be decided in a manner that was trial and error. Educators would come to conclusions based on their previous cohort, making adjustments to the teaching and assessment for the next. Studies in the education of programming were often made by a single educator, reflecting upon their experiences with students. This trial and error teaching style was often reflected within their studies, leading to some fruitless research. The Bug Catalogue: I by Johnson et al. (1983) is a prime example of this, it was an attempt to classify all known programming bugs: they stopped at the fourth volume.

#### Mental Modelling

One concept that is still researched in the domain of programming is that of a mental model. Although the definition of a mental model is still argued over today, most of the proposed models share common elements (Schulte et al. 2010).

To give a broad definition, a mental model is a thought process used by a programmer to develop a solution to a given problem. Schulte et al. (2010) assert that it is a combination of the following elements that make a mental model: understanding the problem domain area; recalling and combining the correct structures of code; having a well-designed programming method; correctly understanding how to write and execute the code on the system.

Expert programmers have the ability to store and retrieve learnt programming structures into memory. These structures are ready to be reused, repurposed, combined and applied to the right problem.

#### Mental Modelling: Developing the Model

Soloway et al. (1982) provides one of the first examples of mental models used by expert programmers. It was argued that novice programmers lack sufficiently developed models which, as a result, caused most of the bugs within their code. It was later determined by Spohrer & Soloway (1989) that these bugs were of similar origin and could be grouped into a small number of types. These types could account for the majority of bugs within a novice's program.

#### Mental Modelling: Cause of the Bugs

It was asserted by Spohrer & Soloway that that poor plan composition was the cause of bugs in novice code. Poor plan composition resulted from difficulty merging pre-learnt programming plans into a single coherent structure (1989). However, poor plan composition was not the only explanation to be had for the origin of these bugs; several others were brought forward.

Pea (1986) in his paper 'Language-Independent Conceptual "Bugs" in Novice Programming', provided the first true study into the issues of a novice programmer that was independent of a programming language. Pea argued that the bugs a novice will face result more from the interaction between humans and computers, than the semantics of a programming language. Essentially, their mental model is incorrectly developed upon a novice's flawed understanding of how the computer will execute the code.

The idea of a disconnect between the mental model and the physical computer is shared by Du Boulay (1989), a contemporary of Pea. Opinions differ on whether the cause of the bugs is language independent.

In the paper 'Some Difficulties of Learning to Program', Du Boulay (1989) provides further evidence of bugs resulting from poor mental models. Du Boulay argues that an incorrect understanding of the 'notional machine', how a computer system internally works, will lead to a flawed foundation for a novice to learn. It is argued that this flawed understanding is further re-enforced by the poor syntax of a programming language, inhibiting the learning of structures and comprehension of programming code.

Du Boulay further argues that an incorrect understanding of the notational machine can stem from poor analogies used by the educator, a point echoed by Winslow (1996).

In his paper, 'Programming Pedagogy -- A Psychological Overview', Winslow (1996) provides several notable observations and arguments. One of the main assertions is that the transition between novice and expert programmer, within a four-year course, is impossible. He explains that by the teaching of correct and unambiguous mental models, a novice programmer could at least achieve competency. Otherwise, in the absence of properly taught models, the novice will construct their own (and often flimsy) mental model.

Winslow's paper was one of the first to identify the area of teaching programming as one of psychological interest; something expanded upon later when applying the concept of abstract

reasoning. Winslow asserted that process of writing code is the process of problem-solving, a point later identified and expanded upon by the McCracken et al. (2001) study.

#### Mental Modelling: Teaching of the Modelling

One of the last papers on mental modelling before the McCracken et al. study, the paper ‘Novice comprehension of small programs written in the procedural and object-oriented styles’ (Wiedenbeck & Ramalingam 1999), argued that the concept of a mental model can be separated into a domain and information model.

Wiedenbeck & Ramalingam, the authors of the paper, reviewed how the ‘style’ of a programming language would affect the development of the domain or information model. To clarify, the ‘style’ of language refers to whether it is Object-Oriented or procedural: the paradigm of the language. This is separate to the effect of language independence mentioned earlier by Pea (1986) and Du Boulay (1989), which discussed a difference based on the language used (both languages were procedural).

Wiedenbeck & Ramalingam, assert that the style a novice learns will affect the development of either the domain or information model. Each style will alternately negatively affect one model while supporting the other. It is asserted that regardless of the style taught; effort should be made to support the deficient model.

#### Mental Modelling: What is its Relevance to this Paper?

Although mental modelling is not the focus of this paper, it is important nonetheless for supplemental reasons. Within the literature, there are several important issues which discount the use of Mental Modelling as the sole explanation of how the novice learns.

From its foundation, the concept of a mental model was developed from the study of expert programmers. Knowing this, all that a mental model can inform the educator is how unlike a novice is to an expert. Willingham (2009), a cognitive scientist, provides insight into why this might be the case: “Cognition early in training is fundamentally different from cognition late in training”(p. 127) and later asserts that you shouldn’t “... Expect Novices to Learn by Doing What Experts Do”(p. 143).

The development of some of the mental models are also suspect; Spohrer & Soloway (1989) provide findings based on a selection of the available results. They purposefully ignore programs that have syntax based errors to focus on what they want to examine. In doing so, they willfully ignore information that may have provided a different explanation to what was observed.

A parallel can be seen within the framework presented by Schulte et al. (2010), a framework based on various mental models. As the framework was the combination of selected elements of several models, it logically stands that elements were discounted. Without an explanation for these discounted elements, a question arises: how can the educator be assured that the remaining elements are correct?

It is not the intention of this section to discount the concept of mental modelling, as the use of remembered ‘structures’ is a fundamental component of neo-Piagetian learning often labelled as ‘chunking’. The development of a mental model is a consequence of learning, not a framework for the teaching and assessment of programming to be developed. This is akin to

the view that a car moves because its wheels are rolling, instead of the combustion of fuel within the engine.

#### End of an Era: the McCracken Study

The early studies of this era are open to critique for their worth; was the observation correct for all students or are these results a consequence of the institution or teaching style of an educator? The solution to this dilemma was to undertake a study that is multinational and multi-institutional. Such a study could corroborate the results found by the same tests in other institutions, to provide a result independent of differences between teacher and institution.

The first multinational and multi-institutional evaluation of introductory programming was undertaken in a paper by the McCracken et al. (2001) group. The paper: 'A multinational, multi-institutional study of assessment of programming skills of first-year CS students' raises three main points, each providing insight based on testing that had yet to occur before it.

The first is that code writing is an exercise in problem-solving, this can be broken down into five steps. Across the board, it was noted that novices were unable to complete all of these steps. This point was originally proposed by Winslow (1996) but is expanded upon within this study.

The second point is that novice programmers, regardless of nationality, share common issues when attempting to abstract from given questions. This is similar to what was noted in an earlier study by Pea (1986).

The third and final point is that problems originate from a lack of knowledge rather than a lack of skill, an issue that seemed to affect most of the novice programmers tested.

Several points have been raised in contention to the results of this study, most importantly from the McCracken et al. group themselves. The McCracken et al. group noted issues regarding the assessment and conduct of the study. The collaboration and the keeping of identical testing conditions was a difficult process between the universities. These deficiencies lead to a lack of definite answers offered by the study, something that Whalley et al. (2006) noted "[that] It was not clear why the novices struggled to write the required programs"(p. 1).

For what it is worth, the paper itself is still useful. As the first multinational study, it was one of the first papers that provided clear evidence for programming issues to be independent of an institute's teaching. It also showed, that regardless of the decades of research and teaching of mental models, that the average introductory programming student failed to program at the end of their introductory subject competently. It spurred further study into the field and shifted the focus from studies based on mental modelling to the application of learnings from modern educational psychology.

#### 2.1.2. Modern Educational Psychology

Before the advent of the McCracken Study, research that viewed the education of novice programmers as an application of cognitive psychology was limited at best. Studies often focused on how different the novices were to experts, leaving educators knowing little more than how different the novice is to the expert. The paper 'Mental Representations of Programs by Novices and Experts' by Fix, Wiedenbeck & Scholtz (1993), provides an example of such a study. They assert that the mental models of expert programmers display



five common abstract characteristics, some of which (or all of which) are absent from the novice programmer. It is explained that novice programmers lack the basic skills which are required for these characteristics to develop.

It was not until the McCracken study, the first study that was multinational and multi-institutional, that it was understood just how lacking the novices were: to the point where a majority of students had not reached the required educational objectives that were set by the educator. Perkins & Martin (1986) describe the majority of a novice's knowledge as 'fragile'. Perkins & Martin also advocated the existence of a solid pedagogy, decrying the lack of its existence to be the cause for this 'fragile' knowledge.

Papers focusing on the differences between novices and experts fail to establish a focus for the educator in developing their curriculum to accommodate this transition. Neither do they provide how these differences can determine the extent of which these novices have transitioned to the expert. It was in the application of the Bloom taxonomy and the SOLO taxonomy where these issues were accommodated.

### Applications of the Bloom Taxonomy

The modern educational system is designed around the teaching of educational objectives and expected behaviours from the student. Before the introduction of what is now referred to as the Bloom's taxonomy in 1956, there was no widely used theoretical underpinning for how these objectives and intended behaviours interrelated or could be developed. With this in mind, Bloom et al. set out to construct a taxonomy based on modern psychological findings.

Before its introduction, it was the widely held belief that students who struggled with the content were not deserving of the attention of the educator, as they could never understand the content as well as those who could. As Bloom described:

In the late 1940's ...the purpose of education was to determine which students should be dropped at each stage of the education process and which merited... the rigours of more advanced education. (Bloom 1994, p. 7)

The taxonomy showed that any student has the ability learn, as long as the student receives structured support and can learn from a well-developed curriculum.

It is the unfortunate situation where this old belief is still present in the field of computer science education, where even recently it has been advocated that admission into a computer science course should be determined by an assessment of the student's abilities prior to their study in the field (Kramer 2007).

As a pedagogical framework, Bloom's taxonomy can be used as to both structure the curriculum of the subject and to develop examination questions; as this paper is primarily focused on the examination of novice programmers, this paper will focus on the latter.

The first true application of the Bloom's taxonomy to the examination of novice programmers was by Whalley et al. (2006), where a modified version of the taxonomy was used in developing the questions for the examination. The paper notes that in a prior study (Lister et al. 2004) the choice of examination questions was not guided by any framework, leading to ambiguity to what skills the novice programmer was deficient in. Upon applying the taxonomy when developing the questions, Whalley et al. identified several new question types that could assess the student's competency in different ways. One of these new types of questions was the 'Explain in Plain English' (or EiPE) question, its use of which is the focus

of this paper. The Whalley et al. group then applied the SOLO taxonomy to review the responses from this question, a brief history of the application of the SOLO taxonomy to the domain of novice programming will be reviewed in the next section.

These new question types were a predicted benefit by Bloom, noting that the application of the taxonomy could enable educators to identify more ways of determining (and therefore) examining educational objectives (Anderson & Sosniak 1994).

There are further papers that examine the use of the Bloom's taxonomy to novice programming, but as this paper focuses on the response to the EiPE questions, the brief history of its development is all that is relevant for the comprehension of this paper.

### Applications of the SOLO Taxonomy

Bloom's taxonomy is a framework where the application of it when developing a curriculum is intended to develop educational objectives, which when learnt, will match the novice's behaviour to that of an expert. Unfortunately, Bloom's taxonomy does not provide an assessment to see if a novice's actual behaviours meet their expected behaviours, noting this to be "...a matter of grading or evaluating the goodness of performance" (Bloom et al., cited in Anderson & Sosniak 1994, p. 14). The solution to this was provided in a separate taxonomy, one that reviewed the student's responses to these questions; this was called the 'Structure of Observed Learning Outcomes' or SOLO taxonomy.

It is important to stress that the SOLO taxonomy is not intended to be used by the educator for designing the curriculum or examination questions, its use lies in assessing a student's competency through the answering of well-designed questions. Thompson (2004) provides one of the first applications of the SOLO taxonomy to the domain of novice programming, in his paper the SOLO taxonomy was used to create marking criteria for a take-home programming assignment. It was found that the marking criteria returned a better idea of the student's competency when applied to the program as a whole rather than the previous criteria that were applied to the program's components in isolation.

When applying the SOLO taxonomy to the final examination of novice programmers, there is little choice for questions of which the taxonomy can be applied. Traditional examination methods of multiple choice and short code writing are unsuited as the taxonomy assesses written communication from the student: the traditional examination methods provide information that is essentially binary (correct or incorrect). Although assessable to the taxonomy, take home programming assignments make for poor questions in a final examination, where time is limited. Developing their own question type, Whalley et al. (2006) created a question that was both short enough to include in a final examination and had a response that could be measured by the taxonomy; the prior referred to 'Explain in Plain English' question. The EiPE question will be both described and have its use reviewed in section 2.2.1.

## 2.2. The Current Research

To understand the impetus behind the research goals of this paper, the prior research within this field needs to be explained. This section will be looking at the findings of this research whose results are interrelated and relevant to the proposed research questions of this paper. The three main areas of research that will be reviewed include: (1) research into the use of

EiPE questions; (2) research related to tracing as a pre-requisite skill and the relationship between tracing and both reading and writing code; (3) research into think aloud studies.

### 2.2.1. EiPE Questions (and the Relationship Between Reading and Writing)

As stated earlier, the EiPE question was first described and applied in the 2006 paper by Whalley et al. The EiPE question asks the reader to explain a piece of given code and describe its purpose. The question from the paper is depicted again in Figure 3 below:

Question 10

In plain English, explain what the following segment of code does:

```

bool bValid = true;
for (int i = 0; i < iMAX-1; i++)
{
    if (iNumbers[i] > iNumbers[i+1])
    {
        bValid = false;
    }
}

```

*Figure 3 - EiPE Question taken from Whalley et al. (2006, p. 6)*

By categorising the results of the EiPE question using the SOLO taxonomy, Whalley et al. determined that the ability to read programming code and describe it in a relational manner is a pre-requisite for the ability to write code; a point echoed in further studies (Lister 2011; Lister et al. 2006; Murphy et al. 2012). More importantly, the results from the above paper were repeatable; Lister et al. (2006) note in their paper ‘Not Seeing the Forest for the Trees: Novice Programmers and the SOLO taxonomy’ that when categorising the responses of both novices and experts the SOLO Levels categorised consistently.

The above findings have important implications for the teaching of novice programmers, in particular regarding how they should be taught. Lister (2011) in his paper, ‘Concrete and Other Neo-Piagetian Forms of Reasoning in the Novice Programmer’ assert that a novice programmer who is given the task to study by writing code, but is yet at a stage to truly understand what they are writing, will receive no benefit from the task. A later study by Corney et al. (2014) found that teaching only the ability to write code and neglecting the ability to read code, may hinder their ability to program.

However, most importantly, the EiPE question can be used for determining a student’s ability to think abstractly. Murphy, McCauley & Fitzgerald (2012) note in their paper ‘Explain in Plain English’ Questions: Implications for Teaching’ that introductory programming courses will cover topics ‘loosely’. If a misconception is developed in an early topic, the student may continue to with this misconception and make further misconceptions when applying it to later topics: Murphy, McCauley & Fitzgerald refer to this effect as ‘cumulative’. It is asserted that by determining a student’s level of abstract reasoning through the use of EiPE questions, educators can identify those students who have these misconceptions and therefore preventing consequences which could be ‘disastrous’.

### 2.2.2. Tracing as a Pre-requisite Skill

The 2001 McCracken et al. study determined that a majority of the students within an introductory programming course lack the ability to problem solve. The Lister et al. (2004) group extended this research, concluding that novices lack certain basic skills that precede the ability to problem solve. They identify these basic skills as the ability to read and trace code. These skills are interrelated; it is only when a novice is sufficient in them can they begin to comprehend the program code and apply their knowledge to the problem.

Conducting further research into the ability to trace code, Lopez et al. (2008) observed that there was a correlation between a student's ability to trace and their ability to write code. Noting that they also found a relationship between the ability to read code and the ability to write code, they proposed that all three abilities were related and suggested the possibility of a hierarchy of related tasks. Such a hierarchy was also supported by the findings of Venables, Tan & Lister (2009), but cautioning that these skills must be developed in parallel as it is intended to support code writing and not to replace it.

However as identified by Lister, Fidge & Teague (2009), the correlations within the previous papers were relatively weak. It was considered that the relationship between tracing, writing and explaining (therefore also testing reading) code might not correlate further than the pass point (where they receive enough marks to pass the exam), this would explain both why there was a correlation and why the correlated values were so weak. By using a nonparametric test Lister, Fidge & Teague were able to confirm this relationship, providing a  $\chi^2$  value that was greater than the required critical value. They concluded that students who were unable to trace code are usually unable to write code. Or, as stated by Venables, Tan & Lister (2009), the ability to trace code is a skill necessary for code writing but is not sufficient to enable code writing.

### 2.2.3. Think Aloud Studies

Regarding the analysis of responses to questions by novices, most research has been focused on the thought processes of the novice, information for which has been collected by transcribing the novice's utterances when prompted to 'think aloud' when completing a question. By analysing these responses, educators are provided with a 'snapshot' of the novice's knowledge and thought pattern. Think aloud studies are very different in execution from the EiPE questions; but as both types of studies concern the analysis of responses given by novices to identify their misconceptions, their findings are peripherally relevant to this paper. It is in the previously mentioned differences which will highlight the need for analysis of the EiPE responses; this will be reviewed in section 2.3.3.

In execution, think aloud studies usually involve an educator sitting with a novice and observing their efforts when answering a given question. While answering, the novice will verbally describe their thought process when answering the question. What varies between studies is the interactions (or lack of) between the educator and the novice and at what point within their subject (or course) that the novice is being tested. The following papers are not the only examples of think aloud studies but are perhaps the most relevant to the research of this paper.

Perkins & Martin (1986) provide in their paper 'Fragile knowledge and neglected strategies in novice programmers', an early and well-known example of what could be labelled a 'think aloud study'. Although not officially labelled as such, nor following a framework for conducting them, this think aloud study provides an early example of research into the

teaching of programming by asking novices to explain what they are doing. From assessing the novice's responses Perkins & Martin determined that the majority of the knowledge that a novice programmer learns is 'fragile' and that this fragility can lead to a partial or incorrect solution of a problem. Such fragility was said to manifest in the novice either not knowing the knowledge to answer, failing to recall the knowledge or being unable to apply or construct the knowledge correctly.

More recently, Teague & Lister (2014) in their paper 'Longitudinal Think Aloud Study of a Novice Programmer' provide a more modern application of a think aloud study. Their study consisted of a number of think aloud sessions with a single student (referred to as 'Donald') throughout a semester, with another think aloud session three semesters later. It was asserted that their study was the first direct observation of a novice transitioning through the different neo-Piagetian stages. Teague & Lister discount the classical Piagetian approach: one that makes the assumption novices are pre-disposed to being successful at programming. If a novice is struggling to learn, then directed help targeted at their current level of knowledge should be provided to help their transition through their stages of development.

Sudol-DeLyser (2015) provides in her paper, 'Expression of Abstraction: Self Explanation in Code Production', an application of the SOLO taxonomy to the responses of novices in a think aloud study. A group of 24 novices were asked to 'think aloud' when answering a set of questions. Statements given by the novices when thinking aloud were categorised into different types; types of which were mapped to different SOLO categories. Sudol-DeLyser found that novices who gain greater proficiency with code writing will generally provide responses that use multiple levels of abstraction more than those with a weaker ability. A finding that aligns with the previously discussed notion that the ability to explain code is a pre-requisite to the ability to write code.

### 2.3. Conclusion: The Gap – Unexplored Territory

The prior two parts of this literature established both the history behind and the prior research that is relevant to the research questions of this paper; the following section will provide the basis for which these questions were determined, demonstrating how they fill a gap that, as of this paper, had yet to be reviewed. There will be three points within this section, each corresponding to the research question they support.

#### 2.3.1. Suitability of the EiPE Question as an Examination Method

As an examination method, there are several papers that explore the EiPE question's usefulness in determining a novice's ability to abstract their knowledge to a problem. It is an unfortunate situation that the majority of authors of papers that support its use (including this one) are related in some manner, be it supervisor or the same person(s), to the researchers that developed the question in the first place. There are two possible reasons for why the research and therefore uptake, of the EiPE question, is limited. The first could be from a lack of interest from computing educators of whom, as discussed by Lister (2012) in his paper "Rare research: why is research uncommon in the computing education universe?", rarely reconsider their pedagogy. The second explanation may lie in the confidence of its use; educators may have yet to be convinced of the suitability of the EiPE question as an examination method. Of the two explanations, this paper will address the latter. If considering the former to be only a lack of 'marketing' of the EiPE questions, then the marketing of the questions without addressing the concerns of its suitability will limit its uptake. By assuring

the suitability of the EiPE question to the educators first, there is a chance that the educators may share the question with others: therefore, addressing both reasons at the same time.

To address the concern of whether the EiPE question as an examination method is suitable, this paper will explore this issue as the following research question (**R1**): “*Are the Explain in Plain English questions a suitable examination method?*”

For something to be labelled ‘suitable’, the subject must meet certain criteria to enable its application. Research question **R1** has been decomposed into two separate and examinable criteria. These criteria have been derived from both perceived notions and criticism of the EiPE questions in regards to its use as an examination method. These points are: (1) that the EiPE question does not compare to traditional examination methods; (2) that the ability to answer an EiPE question depends more on English language proficiency than the ability to comprehend code.

### Comparison to Traditional Examination Methods

Prior papers in analysing the EiPE questions do well to establish the relationship between reading and writing code; they note the EiPE questions are a useful tool for assessing the ability of a novice to read and explain code. However, a low uptake of this examination method suggests that evidence of this relationship does not translate into confidence in its suitability. The EiPE question’s value, as an examination method, has already been proven by its use in assessing the novice’s ability to read and explain code. However, its comparable worth against the marking schemes of other examination methods have yet to be examined.

This paper seeks to prove the EiPE examination method’s comparable worth by exploring the question (**R1.1**): “*Are the results given for the Explain in Plain English questions comparable to the results from traditional examination methods (multiple choice and writing code)?*”. In doing so, the educator can be confident that the results of the EiPE examination method will correlate with the results of traditional examination methods.

### English Language Proficiency

The main form of criticism laid against the use of the EiPE questions as an examination method has been regarding the level of English proficiency required when answering the question. This criticism was summarised best by Corney et al. (2012) noting controversy around whether EiPE questions were assessing the novice’s ability to either read and understand code or the ability for the novice to express themselves.

Simon et al. (2009) in their paper ‘Surely we must learn to read before we learn to write!’, assert that for the novice programmer, the ability to write code is not entirely reliant on their ability to read and comprehend code. This is later refuted by Simon & Snowdon (2011) who conclude after a study of students at both a local English speaking campus and an international campus, that English language proficiency does not affect performance for the EiPE questions anymore so than the other examination type tested, a result also shared by Corney et al. (2012). This paper seeks to extend these findings by exploring the following question (**R1.2**): “*Is the ability to answer the Explain in Plain English questions dependent on the student’s English language proficiency?*”. Prior papers, however, have only compared against one other examination method: code writing questions. This paper will differ in two main ways: (1) it will compare EiPE questions against both code writing and multiple choice

questions; (2) in examining 334 students, this paper will also test against a much larger cohort than any of the previous studies.

### 2.3.2. Tracing as a Pre-Requisite skill

There has been extensive research regarding the ability to trace as a pre-requisite skill to explain or write code. This paper seeks to confirm previous research by examining the following question (**R2**): “*Is the ability to trace a pre-requisite skill for the ability to read or write code?*”.

This paper will extend the previous research in two ways: (1) with a test population of 334 students, this study will be larger than any previously tested at a single institution; (2) this paper will use phi-coefficient to identify the direction of this relationship that is apparent by a nonparametric test, a direction previously commented upon through visual observation but not proven statistically. Confirmation of this research question would re-affirm and strengthen the notion that the ability to trace is a pre-requisite skill for the reading and writing of code.

### 2.3.3. Looking at the EiPE Responses

It is important to note that although analysis of the responses from a novice can be achieved from both EiPE questions and think aloud studies, it is in the execution of these testing methods that vastly differ. In execution, think aloud studies involve sitting down with a single novice and recording their thought process: this is both impractical and implausible to undertake on a large scale. Equally implausible would be its inclusion as an examination method in a final exam.

Prior examinations into the use of EiPE questions have been quantitative based; that is, all responses from the EiPE questions have been categorised, it was these categories that were then used to determine their findings. If a pattern has been found when the responses have been categorised, then it would stand to reason that patterns should be apparent in the language used when responding. This paper will explore the following question (**R3**): “*Is the language used in the answers to the Explain in Plain English questions indicative of the transition from novice to expert programmer?*”. The prior question will be sub-divided in two further questions in order to pinpoint aspects of the language which can be used to identify this transition, these questions are: (1) does the repetition of the question in an answer (a tautological response) correspond to the mark received by the novice? (2) is the language used in the responses of the novice is shared between those with similar marks?

#### Looking at the EiPE Responses: Tautological Response Rate

Tautology, as defined by Biggs & Collis, is a “restatement of the question”(1982, p. 27). Simply put, it is a direct reference to the question in a novice’s response. In the case of this paper, the ‘question’ refers to the programming code given to the novice to explain its purpose. Within the SOLO taxonomy, a response that is tautologous is deemed to be at the pre-structural level, the lowest level of the taxonomy. If the responses from the EiPE questions are consistently classifiable to the SOLO taxonomy as prior research has shown (Lister et al. 2006), then a higher repetition of words should be observable in the answers of the lower performance ranges. Prior papers have not addressed (or mentioned) tautology as a condition for a pre-structural response when analysing the novice’s responses. This paper will explore this gap by answering the following question (**R3.1**): “*Is a tautological response evidence for this transition [from novice to expert programmer]?*”.

### Looking at the EiPE Responses: A Shared Language

As stated earlier, there has yet to be a study regarding the language used by the novice when responding to an EiPE question. Due to the very nature of its question type, when giving a response to an EiPE question, there are only so many ways a given piece of code can be explained. Previous research has shown that categorising the response to the SOLO taxonomy can be done consistently; logically it would be expected that the language should also be similar. What has yet to be explored is to what degree this language is shared between similar marks and, if it is shared, the parts of the language are shared. This paper will explore the following question to determine this (**R3.2**): “*Do students within a certain mark range show shared language within their answers, could this be used to show the transition [from novice to expert programmer]?*”.



## Chapter 3. Approach

To completely document the approach taken with this paper, the following chapter will be split into three sections. Each of these sections will describe a key component of this paper, these sections are; (1) a description of the exam and cohort of which the data for this paper is collected from; (2) an explanation of the process of which the data have been extracted; (3) an explanation of how this system would be used to conduct word based tests.

### 3.1. The Exam

This paper examines the answers of a final exam given to a cohort of introductory programming students. In total, 334 students undertook the 2015 first semester exam. The exam is 20% of their final mark and participation of the exam is not compulsory to pass the subject. No supplementary material is allowed to be brought into the examination room. The subject teaches the programming language Java.

The exam comprises of 3 separate sections:

- **Section A:** 23 multiple choice questions (Q1 – Q23) worth one mark each, for a total of 23 marks. Each multiple choice question had four or five choices. Note: Q24 was an optional question, used by the subject coordinator to gauge the time a student took to complete the section. The results of Q24 will not be considered for this paper.
- **Section B:** 12 EiPE questions (Q25 – Q36) worth one mark each, for a total of 12 marks. A correct answer is one that replies in a relational manner, describing the purpose of the code: a line by line readout would be marked as incorrect. For an overview of EiPE questions, see section 2.2.1.
- **Section C:** 6 writing code questions (Q37 – Q42) with a total of 15 marks. The marks for these vary; their worth is as follows:
  - Q37 – Q39 is worth one mark each.
  - Q40 is worth three marks.
  - Q41 is worth four marks.
  - Q42 is worth five marks

To summarise the above in a table:

*Table 1 - Approach: Question/Mark Summarisation of the Sections in the Exam*

	Question Type	Total questions	Total Marks
<i>Section A</i>	Multiple Choice	23	23
<i>Section B</i>	EiPE	12	12
<i>Section C</i>	Writing Code	6	15

The only exposure the novices had to the EiPE questions prior to the final examination was provided once in a practice exam given shortly before their final exam.

### 3.2. Retrieval and Storage of the Data

For the cohort's responses to the EiPE questions to be analysed, the responses had to be converted from physical, written text into digital, editable text. For privacy, each exam paper was stripped of their identifiable information and given a unique identification number. These papers were then scanned and saved onto disk.

#### 3.2.1. Transcribing to Spreadsheet

Once the exam papers were scanned to PDF format, the cohort's EiPE answers were transcribed into a large spreadsheet. Each student's paper was read, their handwriting interpreted and typed into the said spreadsheet. In total 4008 answers to the EiPE questions were read, interpreted and transcribed.

#### Justification for Typing

Both voice-to-text and image-to-text were considered to expedite the process. After consideration, it was decided that both methods would have introduced problems that would have delayed the process far longer than any time saved in not typing the answers.

As a tool, Voice-to-text traditionally works by training the tool to the user's speech pattern. In reading out the answers from 334 different students, the tool would be presented with 334 different speech patterns. Eventually, the tool would be trained in a manner that would increasingly produce words that are different to what is being read.

Image-to-text would scan the written answers and attempt to generate a digital text output based on the handwriting in the image. This idea was quickly discarded due to the poor quality of handwriting observed in the cohort's exam papers.

In contrast, the transcribing of the text by typing allowed changes to the text to be made simultaneously when recording. The changes were dictated by a set of rules making the data easier to store and search. Particular focus was on the distinction between a variable's name and the actual word the variable is named after, for example, a variable called "number" or the noun "number". Variables in the text will be denoted by surrounding them with single quotation marks. In doing so, this will clearly identify the letter or word as a variable, making it easier to search.

The exam used in this paper contains 12 'Explain in Plain English' questions, not all of these are attempted by a student. In cases where the student has missed, crossed out or not attempted an answer, '#NA' will be entered into its place.

Spelling mistakes made within answers will be corrected when transcribing, this is to reduce error rates when matching for words in the word-based tests. In the case where a word cannot be determined by context, or if there are a number of possible words that will each change the intention of the answer, '<illegible>' will be entered in its place. Replacing these words with '<illegible>', rather than guessing the word, will prevent the chance of false positives or false negatives which would skew the results.

A full listing of these rules can be found in section 6.1 of the Appendix Chapter.

### 3.2.2. The Program and Database

A program, called Exam Reader, was written to import the information from the typed spreadsheet into a MySQL database. Exam Reader imports the spreadsheet of the cohort's answers, separating each student into an object and dividing the text of their answer into individual words. After applying this to every student and all of their answers, a separate spreadsheet containing the exam marks of the cohort will be read. These marks are then attached to the appropriate student within the cohort. The class diagram of this program can be found in section 6.2 of the Appendix Chapter.

In order to test for a tautological response, the text from the questions is also typed and imported into the database in a manner similar to what is noted above for the responses.

Exam Reader then inserts each student of the cohort into the database. Information regarding the database, including both a database diagram and a data dictionary, can be found in section 6.3 of the Appendix Chapter. The data in the database can now be queried for the purpose of testing the answers to the EiPE questions.

### 3.3. Conducting Word-Based Tests

A major component of the tests conducted within this paper deals with the analysis of the words used in the answers to EiPE questions. Unlike the other tests within this paper, the word-based tests have no previous tests that are similar in design. Therefore, as it is not a standard test, the following section will provide a description of how these word-based tests have been conducted.

For this section, one example answer to a single EiPE question is provided; it must be noted that this answer would have 333 other responses for it to be compared with. The sample question is provided in Figure 4 below:

Q29. In one sentence that you should write in the empty box below, explain in plain English what this code does.

In each of the three boxes containing sentences beginning "Swap the values in ..." assume that appropriate code is provided – do NOT write that code.

```
if (y1 < y2)
```

Swap the values in y1 and y2.

```
if (y2 < y3)
```

Swap the values in y2 and y3.

```
if (y1 < y2)
```

Swap the values in y1 and y2.

Figure 4 - 'Explain in Plain English' Question 29.

Through the process described in section 3.2.2 above, the transcribed digital text of the novice's answer will be imported and stored in the MySQL database system. Instead of repeatedly entering the same words into the database for each response, there will be a central dictionary table. This table will contain each unique word used in both the text of the questions and answers. The words will be stored in upper case, to both prevent duplication

and preserve space. As each word in the dictionary table is paired with a unique number, words used in the question can easily be matched in a novice's answer.

The tables and figures below provide a brief example of how this imported text will appear in the tables of the database: Table 2 below shows the main dictionary table where each word is unique; Table 3 shows the questions table, where the words are stored by their associated Word ID (listed in the dictionary table); Figure 5 shows a sample answer to question 29 (depicted in Figure 4 above), with their words recorded with the associated Word ID in Table 4.

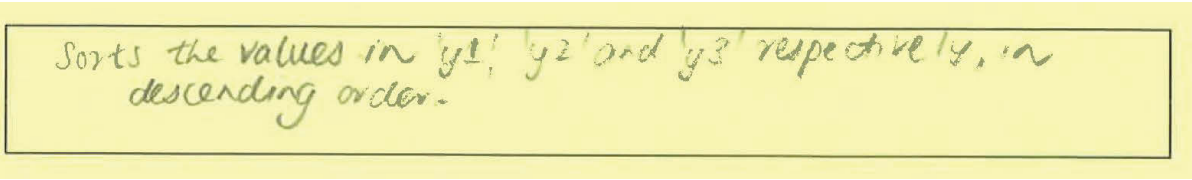
Table 2 - Dictionary Table

<b>Word ID</b>	<b>Word</b>
1	
2	IF
3	'Y1'
4	(
5	<
6	)
7	'Y2'
8	SWAP
9	THE
10	VALUES
11	IN
12	AND
13	.
14	'Y3'
15	\n
16	SORTS
17	,
18	RESPECTIVELY
19	DESCENDING
20	ORDER

Note: the word with Word ID '1' in Table 2 above is a space.

Table 3 - Question Table: Based on Q29 in Figure 4.

<b>Question Number</b>	<b>Words (Word ID)</b>
29	2, 1, 4, 3, 1, 5, 1, 7, 6, 15, 8, 1, 9, 1, 10, 1, 11, 1, 3, 1, 12, 1, 7, 13, 15, 2, 1, 4, 7, 1, 5, 1, 14, 6, 15, 8, 1, 9, 1, 10, 1, 11, 1, 7, 1, 12, 1, 14, 13, 15, 2, 1, 4, 3, 1, 5, 1, 7, 6, 15, 8, 1, 9, 1, 10, 1, 11, 1, 3, 1, 12, 1, 7, 13



Sorts the values in 'y1', 'y2' and 'y3' respectively, in descending order.

Figure 5 - Sample Answer to EiPE Q29

Table 4 - Answer Table: Based on Sample Answer in Figure 5

Answer Number	Words (Word ID)
1	16, 1, 9, 1, 10, 1, 3, 17, 1, 7, 12, 1, 14, 1, 18, 17, 1, 11, 15, 19, 1, 20, 13

By organising the data this way, the system can take advantage of the SQL query optimisation. At this stage, the data can be searched through quite easily; the words of this novice's answer can now be compared against either the words of the question or the words of answers from other novices with a similar mark. Within both tests, only words that are either greater than three characters in length or is the name of a variable will be considered.

### 3.3.1. Tautological Response Rate

By matching the Word ID numbers assigned to a novice's answer with the Word ID numbers assigned to the question, the total number of matching words can be compared against the total words in the answers as a percentage.

The percentage of a student's answers that were tautologous can be calculated as: the per question summation of the distinct words in an attempted answer divided by the summated count of distinct words that match the words from the question asked. To put this into an equation:

$$\forall s \in C, s_p = \sum_{i=1}^t a_i / \sum_{i=1}^t q_i$$

Where:

#### Symbol Meaning

$C$	The cohort of introductory programming students undertaking the final exam. The $C$ has a size of 334 students.
$s$	A student within $C$ .
$p$	The percentage of words in the answers given by $s$ that were tautological.
$a$	The total distinct words used in an answer by $s$ .
$q$	The number of words in $a$ that matches the words in the question
$t$	The number of attempted EiPE questions by the student: maximum of 12.
$i$	The current question number (treating the first question as 1).

To count the words correctly, two conditions will be set: (1) the words will distinct: if a word is repeated more than once in the same answer (or question) it will only be counted once; (2) words will only be counted if they are greater than three characters in length or they are the

names of a variable. These conditions ensure simple words like ‘it’, ‘or’ and ‘a’ are not counted, but words such as ‘array’, ‘loop’ and the variable named ‘a’ are.

In Table 5 below, both the words from the question and the words from the answer are outlined. Numbers that are crossed out do not meet the conditions set above, numbers who are **bold** match between the two and those who are *italicised* meet the conditions but do not match.

Table 5 - Matching Responses to the Question (Matches are Highlighted in Red)

<b>Question Number</b>	<b>Words (Word ID)</b>
29	2, 1, 4, <b>3</b> , 1, 5, 1, <b>7</b> , 6, 15, 8, 1, 9, 1, <b>10</b> , 1, 11, 1, 3, 1, 12, 1, 7, 13, 15, 2, 1, 4, 7, 1, 5, 1, <b>14</b> , 6, 15, 8, 1, 9, 1, 10, 1, 11, 1, 7, 1, 12, 1, 14, 13, 15, 2, 1, 4, 3, 1, 5, 1, 7, 6, 15, 8, 1, 9, 1, 10, 1, 11, 1, 3, 1, 12, 1, 7, 13
<b>Answer Number</b>	<b>Words (Word ID)</b>
1	16, 1, 9, 1, <b>10</b> , 1, <b>3</b> , 17, 1, <b>7</b> , 12, 1, <b>14</b> , 1, 18, 17, 1, 11, 15, 19, 1, 20, 13

The above table shows that ignoring spaces, out of the possible 23 words in the answer, eight words meet the two conditions of: (1) being either greater than three characters or are the name of a variable; (2) appearing only once. Only four of these words match with the question. Referring back to the equation listed above,  $a_5 = 8$   $q_5 = 4$ , these would then be summated with the rest of the set and the percentage would be the sum of  $a$  divided by the sum of  $q$ .

### 3.3.2. Domain-Specific Language

In the tautological tests, the tests were conducted on a student level: the tautological rate belonged to the student and was compared to the mark. The domain-specific language tests mark ranges, that is all students within the low middle or high mark range. Both will still only consider words that have a character size greater than three characters or is the name of a variable.

The tests are initially run on a per question basis; each question will have a list of repeating words belonging to the low, middle or high mark range. As the purpose of the domain-specific language tests is to search for shared, common words across the novice’s answers, words will only be considered if they are used five or more times in either a correct or incorrect answer. Those words which are, in context, the same word will be counted as the same word. For example: ‘reverse’, ‘reversed’ and ‘reversing’ will all count as the same word. Repeated words which meet the above conditions can safely be considered to be common to that mark range.

For each question, the following metrics are collected: the number of unique words, the number of words shared between mark ranges and the number of words that are unique to that mark range. These metrics are then summated across all of the questions to be analysed.

To illustrate what data are collected from the questions of the paper, the metrics for question 32 is provided as an example which, due to the size of the tables, can be found in section 6.5 of the Appendix Chapter.

## Chapter 4. Results

The following chapter will detail the results of the research conducted on the data collected and retrieved through the processes outlined in the Approach Chapter (section 3.2 above). A number of tests were conducted in order to explore the research questions of this paper. The following section will explain the structure of this chapter and connect the further sections to their relevant research questions.

### 4.1. Introduction

Section 1.4 of the Introduction Chapter outlines the research questions of this paper. These questions decompose the core proposition of this paper into testable components. Each section of this chapter (apart from this one) will correspond to a research question; the following list will label the section's (and sub-sections) corresponding research question:

Section 4.2 will explore research question **R1**

Sub-section 4.2.1 will explore research question **R1.1**

Sub-section 4.2.2 will explore research question **R1.2**

Section 4.3 will explore research question **R2**

Research question **R3** will be explored by its operationalised questions

Section 4.4 will explore research question **R3.1**

Section 4.5 will explore research question **R3.2**

Each of the above research questions will be repeated within each their relevant sections for the sake of context.

#### 4.1.1. The Impact of Missed Answers

Many of the following tests involve analysis of the answers given by the cohort to the EiPE section, as such the validity of this data must be considered. Of particular interest, in regards to the validity of the data, are students who have failed the EiPE section. It would stand to reason that those who have failed may not have attempted enough questions to pass in the first place, this, however, does not seem to be the case. Only 11 out of the 121 students who have failed the EiPE section attempted less than six questions: only 9.1%. Figure 6 below plots the marks students received for the EiPE section (limited by those who have failed) to the EiPE questions that they have attempted.



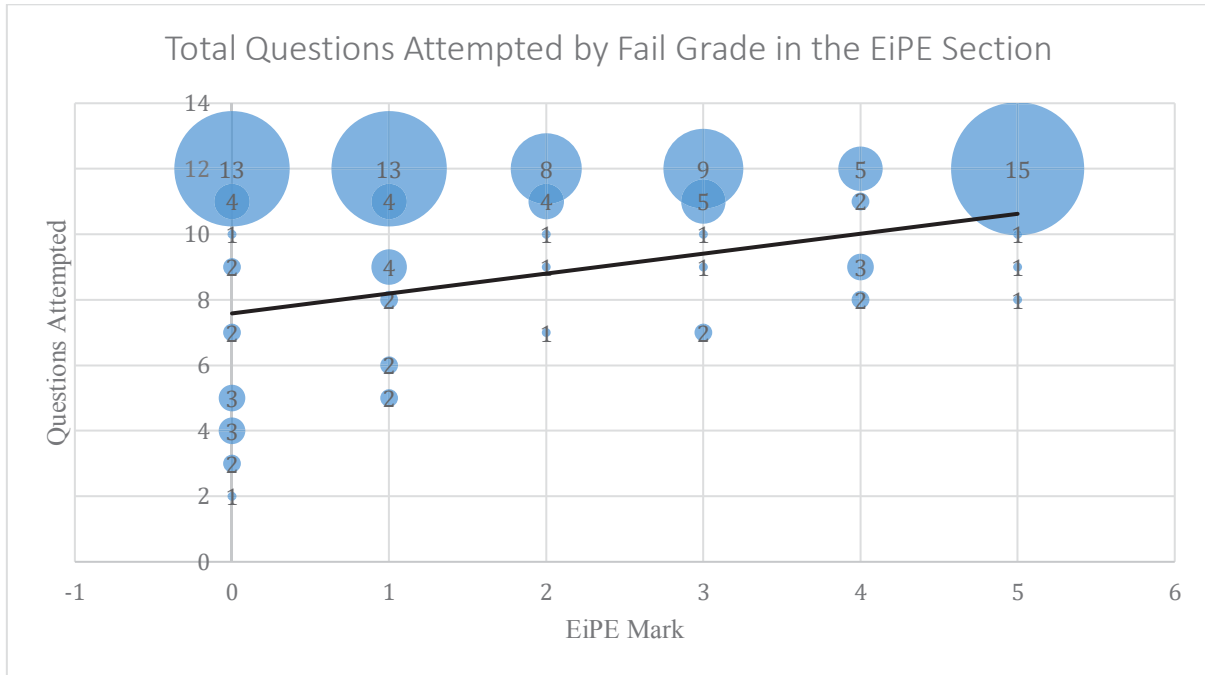


Figure 6 - Missed Answers: Total Questions Attempted by Fail Grade in the EiPE Section

With an  $R^2$  value of 0.0963 it can be assured that, at the lower end of the mark scale, the number of questions attempted by a student seems to have no correlation to the mark a student receives. In fact, 86 out of the 121 students (71.1%) depicted in Figure 6 above attempt 10 or more questions.

When comparing attempted against those who have passed the EiPE section, Figure 7 below shows even less of correlation with a recorded  $R^2$  value of 0.0583.

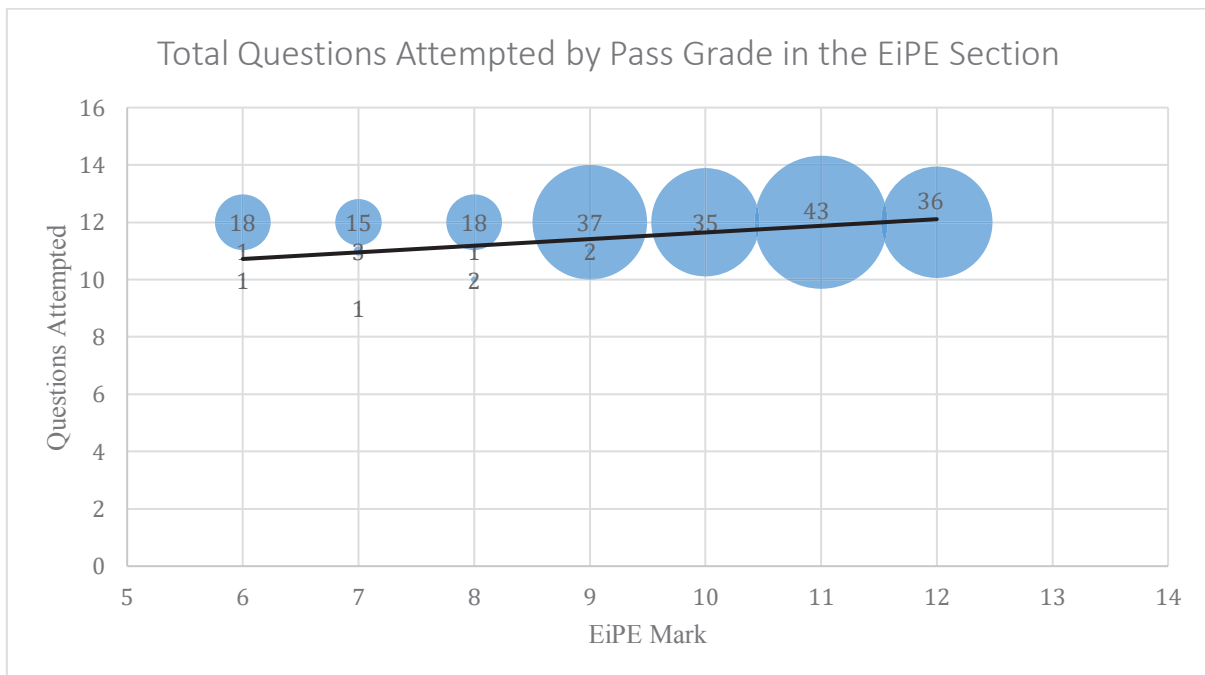


Figure 7 - Missed Questions: Total Questions Attempted by Pass Grade in the EiPE Section

By percentage of the cohort against the number of EiPE questions attempted, 79.34% of the cohort attempted 12 questions (as seen in Table 6 below). It can safely be said that any EiPE questions left unanswered, would have a negligible effect on the upcoming tests at best.

Table 6 - Missed Answers: Percentage of Cohort by Number of EiPE Questions Attempted

<i>Number of EiPE Questions Attempted</i>	<i>Percentage of Cohort</i>
0	0.00%
1	0.00%
2	0.30%
3	0.60%
4	0.90%
5	1.50%
6	0.60%
7	1.50%
8	1.50%
9	3.89%
10	2.10%
11	7.78%
12	79.34%

#### 4.2. The Suitability of Explain in Plain English Questions

For the novice programmer, the development of skill in reading code is a must. Prior research has already confirmed this; identifying that to write code sufficiently, the ability to read code is a pre-requisite skill (see section 2.2.1 of the Literature Review Chapter). Within a final exam, researchers have had to develop a new examination method for testing a student's ability to read code in a relational manner: the 'Explain in Plain English' question.

Before the development of the EiPE question, the skill of reading code has always been indirectly but never directly tested in a final exam. To test the skill in a relational manner, traditional examination methods are either insufficient in determining a student's ability to read code or would take too long to complete when used in a final exam (see section 2.2.1 of the Literature Review Chapter).

Being a relatively new examination method, there has been some hesitance to the adoption of the EiPE as a tool for assessment. As noted in section 2.3.1 of the Literature Review Chapter, the lack of uptake could be the result of a lack of confidence in the EiPE question as an examination method. To explore this point, this paper is pursuing the issue as the following research question (**R1**): *Are the Explain in Plain English questions a suitable examination method?*

Two key issues have been raised regarding the suitability of the EiPE questions: (1) that the marking of the questions may not correlate with traditional examination methods; (2) that to answer the questions, for some students, the required level of English language proficiency may be too great. These are explored as research questions **R1.1** and **R1.2** respectively.

For the sake of brevity, the term ‘traditional examination methods’ (or traditional for short) will refer to the sum of the marks received for the multiple choice questions (MCQ) and writing code questions within the context of these tests.

#### 4.2.1. Comparison Between Traditional Examination Methods

To instil confidence in the use of the EiPE question as a tool for examination, its usage has to be comparable to the existing traditional exam methods. Research has already established that answers given to EiPE questions can be consistently categorised to the SOLO level of the student (see section 2.2.1 of the Literature Review Chapter). However, there has yet to be a direct assessment to determine if the results given to EiPE questions share a linear relationship with the results from traditional examination methods. To that end, the following tests seek to answer the following research question (**R1.1**): *Are the results given for the Explain in Plain English questions comparable to the results from traditional examination methods (multiple choice and writing code)?*

It would be expected that in a final exam, regardless of the examination method, those who score poorly with one method would also score poorly with the rest; the opposite of this should also be the case.

#### The Tests

Three scatterplots will be modelled comparing the cohort’s EiPE marks to their MCQ marks, writing code marks and traditional marks (sum of MCQ and writing code).  $R^2$  and Pearson’s  $r$  will both be calculated. Pearson’s  $r$  will provide how closely the marks are correlated, while  $R^2$  will show how the variance of the marks received for the EiPE questions explains the spread of the examination method marks tested against it.

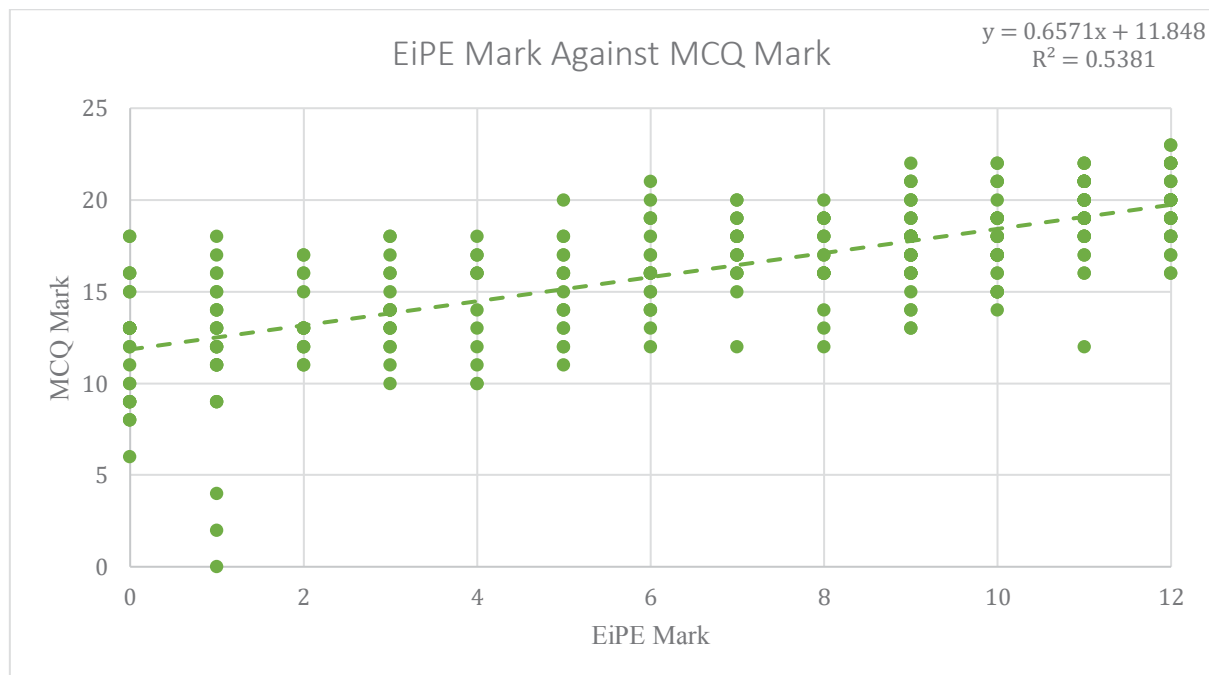


Figure 8 - Examination Method Comparison: EiPE Against MCQ

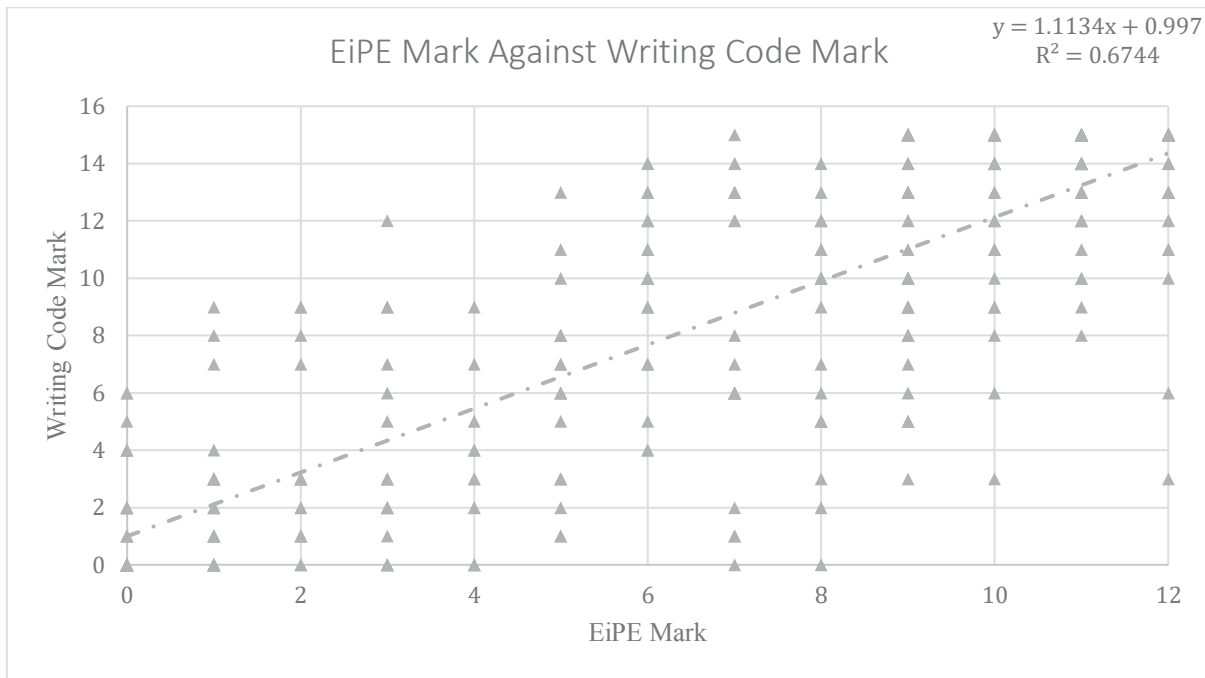


Figure 9 - Examination Method Comparison: EiPE Against Writing Code

Both Figures 8 and 9 above show a strong correlation between the EiPE marks and their tested examination method. Individually, the EiPE mark is correlated weakest against the MCQ examination method with an  $R^2$  value of 0.5381. In comparison, the correlation between the EiPE questions and the writing code questions is stronger with an  $R^2$  value of 0.6744. The difference between the correlations may be explained by the format of the MCQ questions: where a student’s mark can vary due to guessing.

Interestingly, it can be observed that the correlation of the marks received for the MCQ and writing code questions is stronger when combined than when tested separately ( $R^2$  value of 0.7181). Figure 10 below models this as the ‘traditional’ mark:

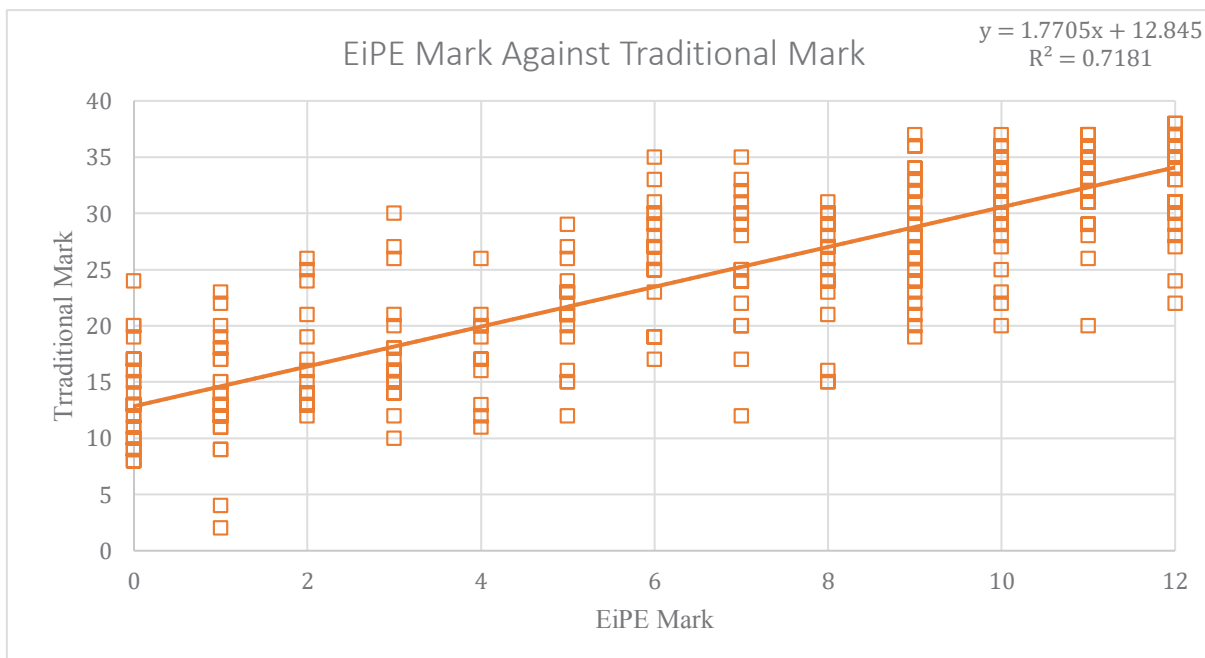


Figure 10 - Examination Method Comparison: EiPE Against Traditional

The statistical findings of the examination method comparison tests (Figures 8, 9 and 10) are summarised in Table 7 below. It can safely be said that these findings show a strong correlation between the EiPE questions and the traditional examination methods.

Table 7 - Examination Method Comparison: Statistical Results Summary

<b>Examination Methods Compared</b>	<b>R<sup>2</sup></b>	<b>Pearson's r</b>
<i>EiPE Against MCQ</i>	0.5381	0.7335
<i>EiPE Against Writing Code</i>	0.6744	0.8212
<i>EiPE Against Traditional</i>	0.7181	0.8474

**Stronger Combined?**

It was observed in the comparison tests above that the marks received for the MCQ and writing code questions have a stronger correlation together than if modelled separately. The traditional examination methods must have more in common with the EiPE questions when combined than if separate. As the writing code method has the highest R<sup>2</sup> value, it must be the addition of the MCQ marks that is the cause of the closer correlation with the EiPE marks.

To visualise this change, the writing code and MCQ marks are plotted together (scaled to their combined mark of 38) in Figure 11 below. The change in marks, caused by the MCQ questions, is displayed as a curve.

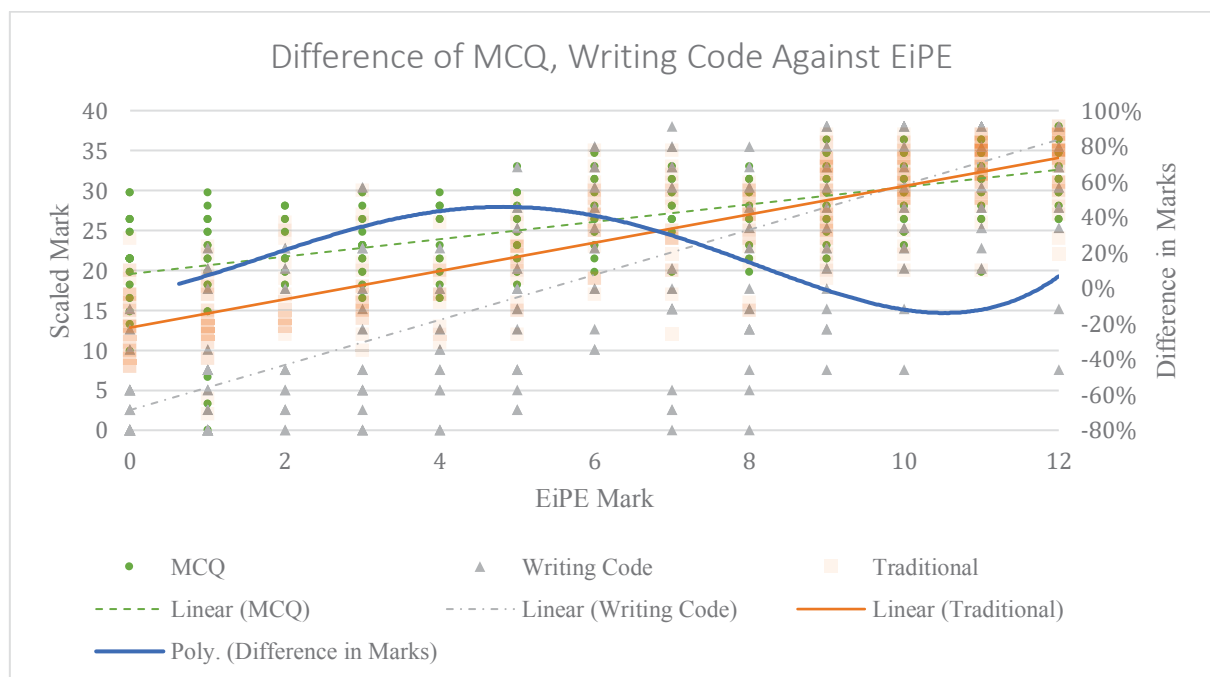


Figure 11 - Examination Method Comparison: Difference of MCQ, Writing Code Against EiPE

It can be seen that the addition of the MCQ marks does two things: (1) it pushes the writing code marks higher at the lower end of the EiPE mark range; (2) it slightly drops the marks at the higher range.

This could be a result of what knowledge that EiPE questions tests for, compared with the knowledge tested by MCQ and writing code questions. The traditional examination methods, when combined, may only examine some (not all) of the same knowledge as the EiPE questions. To loosely illustrate, consider that EiPE questions tests for factors 'a', 'b' and 'c': both the MCQ and writing code methods test for 'c', while 'a' is only tested by MCQ and 'b' is only tested by writing code. Combined, the traditional methods tests for 'a', 'b' and 'c', resulting in a higher correlation to EiPE.

What of the remaining 0.2819 of the  $R^2$  value? This is to be expected, as the point of the EiPE questions is to test for knowledge that was, previously, never directly tested by the traditional examination methods. If the values correlated completely, then it could be said that spread of the marks for the traditional examination methods could be completely explained by the marks for the EiPE questions. This would mean that the EiPE questions would test for the same knowledge as the traditional examination methods, defeating the point of which they were created for.

#### Use of the EiPE Question Will Not Skew the Results

It can be clearly seen that the results found in the above tests show that the EiPE questions correlate quite closely with traditional examination methods. Educators can be assured that when combined with the traditional examination methods, the usage of the EiPE questions will not skew the results of a cohort. Its usage provides a wealth of information regarding both a student's ability to read code and provides information of their cohort that the traditional examination cannot provide.

#### 4.2.2. Dependency on English Language Proficiency

Originally raised as a potential issue in the paper that proposed the EiPE question (Lister et al. 2004), a student's success in interpreting and answering the EiPE questions may be dependent on their English language proficiency. It has been suggested that the EiPE questions are more of an English comprehension question than a code comprehension question. The EiPE questions would require a higher English language proficiency to answer correctly than the traditional examination methods of multiple choice and code writing (see the heading 'English Language Proficiency' under section 2.3.1. of the Literature Review Chapter).

To determine if English proficiency does play a factor, this paper has addressed the following research question (**R1.2**): *Is the ability to answer the Explain in Plain English questions dependent on the student's English language proficiency?*

#### Conducting the Tests

To test if the ability to answer EiPE questions is dependent on a student's English language proficiency, the marks the cohort receives for said questions can be plotted against the marks received for the traditional examination methods. It would stand to reason that if the level of English language proficiency needed to interpret and answer the EiPE questions is too advanced, then a portion of the cohort with a poorer grasp of the English language should be observable in the plotted marks. These students will achieve a fail mark in the EiPE questions and a pass mark for the traditional examination methods.

Although the above will test for the dependency of English language proficiency, it shows nothing regarding the language used by the cohort when answering the EiPE questions. The effects of the level of English language proficiency may also be evident in words used by the cohort in their answers: particularly the words that repeat across them.

When answering an EiPE question correctly, certain words will have to be used when answering. To demonstrate, Figure 12 below depicts an EiPE question used in the exam:

Q27. In one sentence that you should write in the empty box below, explain in plain English what this code does.

Assume that the “x” is an array of five integers.

```
temp = x[4];
x[4] = x[0];
x[0] = temp;

temp = x[3];
x[3] = x[1];
x[1] = temp;
```

Figure 12 - Example EiPE Question

Three sample answers for the depicted EiPE question is provided in Table 8 below:

Table 8 - Sample Answers for EiPE Question

<i>Answer Number</i>	<i>Sample Answers</i>
A1	“It reverses the array x”.
A2	“It reverses the order of the elements inside array x”.
A3	“Using 'temp' to swap the first and last array and also the second and third.”

Considering only words that are greater than three characters in length or are the name of a variable, the answers A1, A2 and A3 repeat the word: ‘array’. A1 and A2 also repeat the words ‘x’ and ‘reverses’ in addition to ‘array’. The words ‘array’ and ‘x’ describe the object the code is manipulating, while ‘reverses’ describes the manipulation itself. Regardless of the flexibility of the English language, there are only so many ways to interpret the question correctly. As displayed in answers A1, A2 and A3, this leads to words that repeat across the cohort’s answers.

It is the correct combination of these repeating words which will lead to a correct answer. If the marks a student receives for the EiPE questions depend on their English language proficiency, then the number of repeating words should increase as the EiPE marks increase. By analysing the count of repeated words to a student’s mark, we can determine whether the mark a student receives is due to knowledge of the repeated words or rather the arrangement and omission of repeated words. The latter would suggest that the lower marks are due to a lack of knowledge in the domain, rather than former which suggests a lack of understanding the question or not knowing what words to use.

To test using this repetition, the repeated words will be categorised by the EiPE mark ranges. There will be three different ranges: 0-3 (low), 4-9 (middle) and 10-12 (high).

The low mark range is based on the linear regression line as observed in Figure 10 of section 4.2.1, where an EiPE mark of 4 is the point the linear regression line crosses the pass mark

for the combined MCQ and writing code questions. The 4-9 and 10-12 mark ranges are broken up roughly by student size, the student count for these ranges is outlined in Table 9 below:

Table 9 - English Proficiency: Student Count by Range

<b>Mark Range</b>	<b>Student Count</b>
Low (0-3)	91
Middle (4-9)	129
High (10-12)	114

As noted earlier in section 3.3.2 of the Approach Chapter, a repeating word is defined as a word that is either greater than three characters in length or is the name of a variable and appears more than five times in either a correct or incorrect answer. Those words which are, in context, the same word will be counted as the same word. For example: ‘reverse’, ‘reversed’ and ‘reversing’ will all count as the same word. The total count of repeating words will be the sum of distinct words that repeat within the mark ranges.

The impact of a lack of words caused by a student who misses a question(s) is negligible; this has been discussed earlier in section 4.1.1.

Correlation Tests

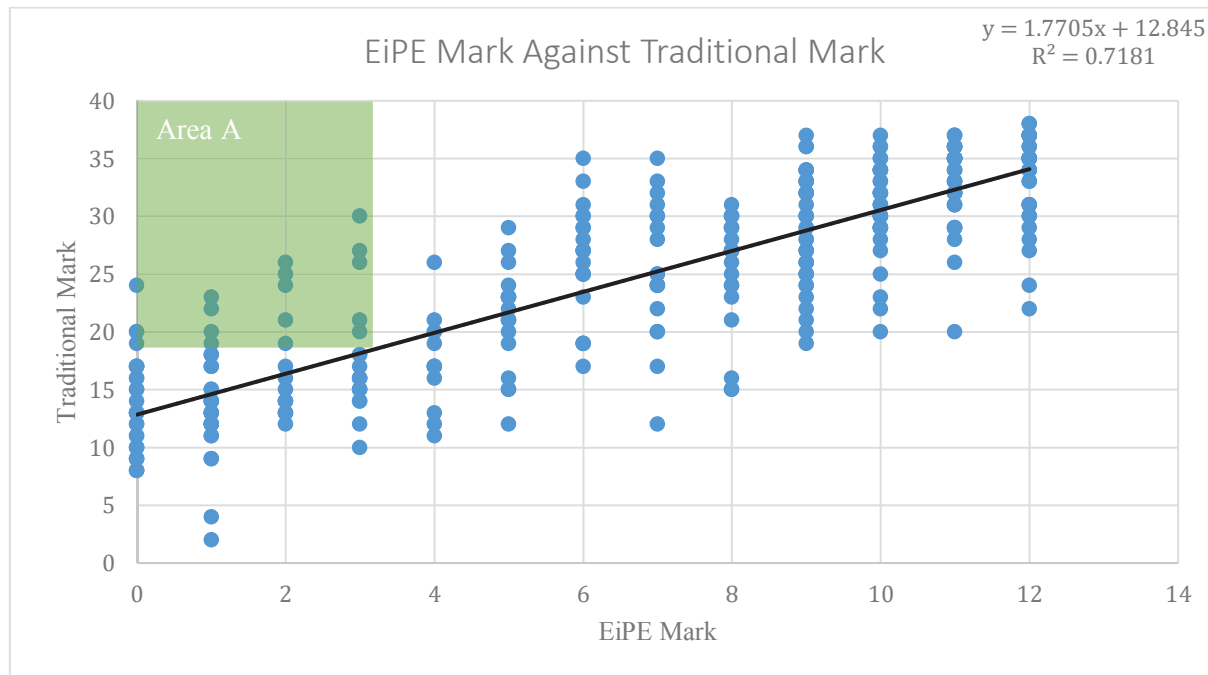


Figure 13 - English Proficiency: Mark Correlation

In Figure 13 above it can be seen, when plotting the traditional marks to the EiPE marks, that there are little to no students in the top left area of the plot. Labelled ‘Area A’, this would be the area on the plot where students that receive a fail mark in the EiPE questions, pass the



traditional examination questions. The above diagram depicts frequency of its data poorly, Figure 14 below depicts the same data but displays the frequency as the bubble's size.

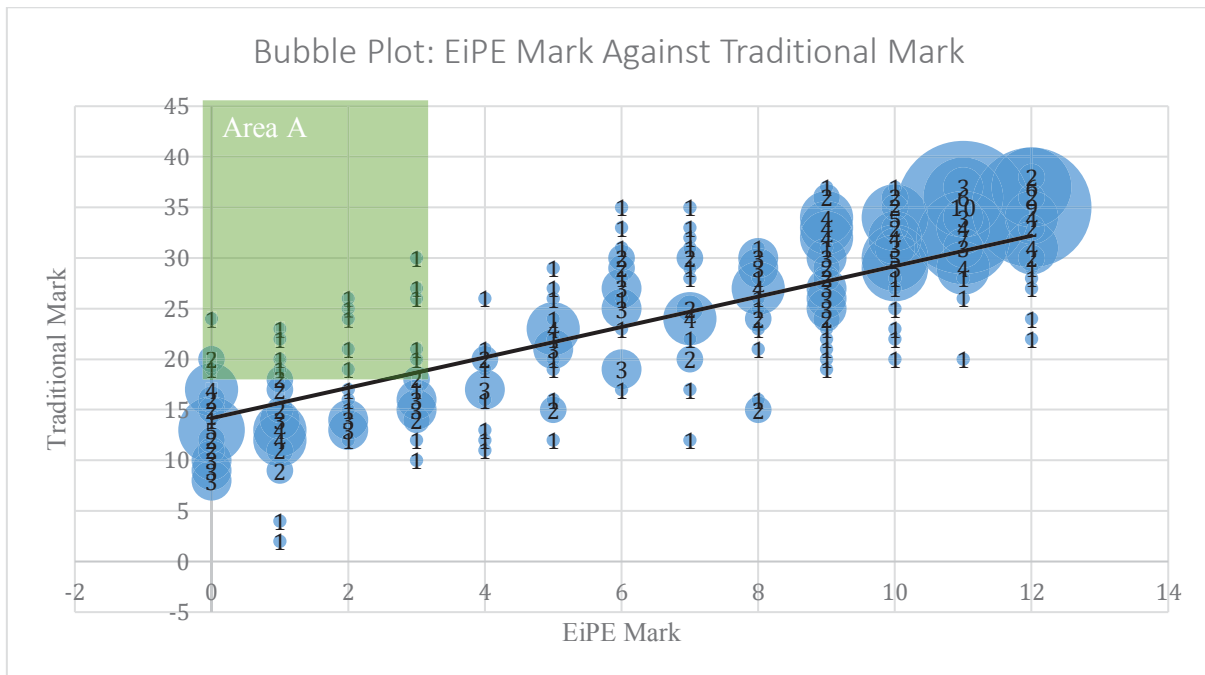


Figure 14 - English Proficiency: Bubble Mark Correlation

With the frequency now easily depicted, it shows that the number of students under Area A is quite small. Only 18 students fall under Area A, comprising 5.4 % of the cohort. If the argument that a student with a weak grasp of the English language performs poorly on the EiPE questions but not the traditional examination methods, then this argument is not reflected in the above figures.

With an  $R^2$  value of 0.7181, the correlation of marks is evident. A student's level of English language proficiency does not seem to affect their ability to undertake or interpret the EiPE questions. The level of English language proficiency required for the admission into the university is sufficient enough to enable a student to interpret and answer the EiPE questions. If a student does not have the required level of English language proficiency to answer the EiPE questions correctly, then they also cannot correctly answer the other question types.

#### Repeating Words Across the Mark Ranges

Earlier in this section, reference was made to three sample answers given to an EiPE question (Q27 previously depicted in Figure 12). Answers A1 and A2 are samples of correct answers given by students to this question; A3 is a sample of an incorrect answer.

In the case of A3, only the word 'reversed' is shared with both A1 and A2. Note that while the student observes the correct manipulation, they fail to describe that the array is being manipulated. It is missing the descriptive words of 'array' or 'x' that is repeated in answers A1 and A2. Figure 15 below depicts the repeating word count across the EiPE mark ranges; it shows that at the low mark range the count of repeating words is much lower than at the middle or high mark ranges.

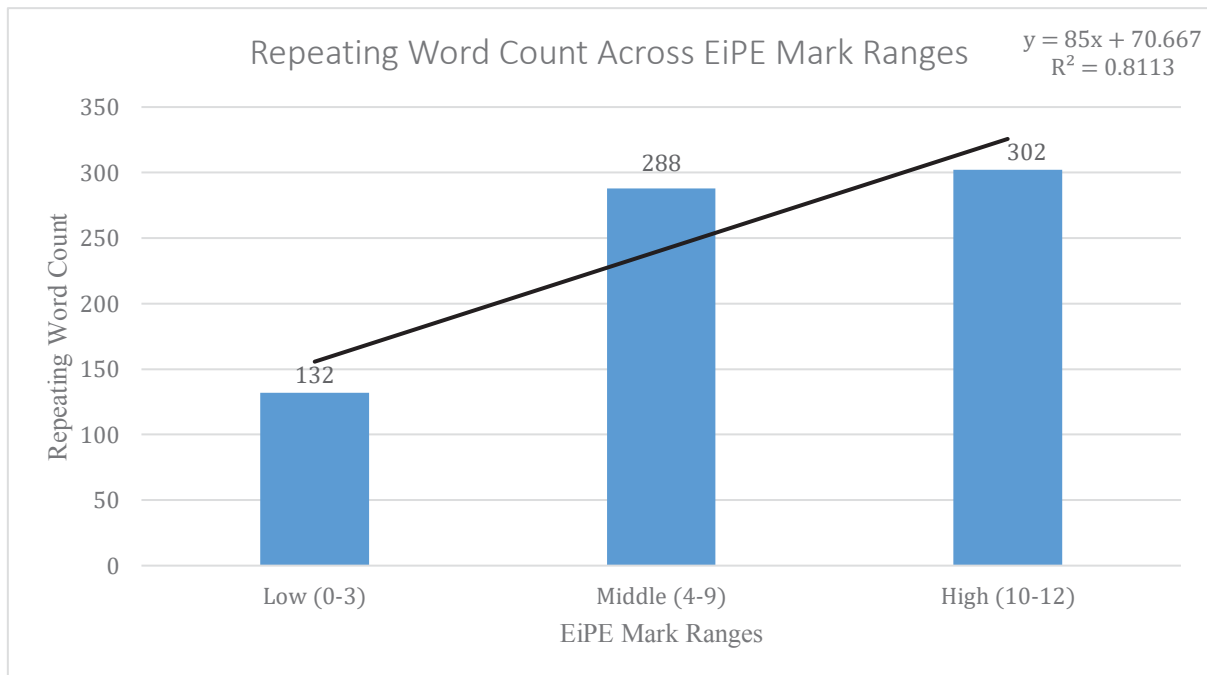


Figure 15 - English Proficiency: Repeating Word Count Across EiPE Mark Ranges

There is a sharp 156 repeated word difference between the low and middle ranges; this is more than double the number of the actual repeated words in the low range. However, between the middle and high ranges, there are only 14 words of difference. Within the context of the findings of the correlation test, these observations make sense.

With a difference of only 14 words, it cannot be a lack of knowing the words that are the difference between the middle and high ranges: the difference must be in the arrangement (or omission) of repeating words. This demonstrates that it is the absence of domain knowledge (or its lack of application) that is the difference between the middle and high mark ranges, rather than an absence of the ability to describe the situation in the English language.

For the difference between the low and middle ranges, a lack of English language proficiency may indeed be the reason for this. As stated earlier, the low mark range was decided by the crossing of the regression line between the marks of the EiPE questions and the traditional examination methods: a mark of four for the EiPE questions is the point on the line where a student will pass the traditional examination methods. The prior correlation test showed that if poor English language proficiency is inhibiting a student's ability to answer EiPE questions, then it must inhibit their ability to answer other question types. As such, what is seen in the difference in words between the low and middle mark ranges may be an example of this.

#### 4.3. Tracing as a Pre-Requisite Skill

In the domain of introductory novice programming examination, one of the points raised in prior research by Lister, Fidge & Teague (2009) was the notion that in order to read and write code the ability to trace is a pre-requisite skill. Lister, Fidge & Teague note that in parametric tests, the relationship between tracing, reading and writing is quite weak. However, when conducting a nonparametric test, there is a strong relationship. The research in the following section seeks to extend that point by further exploring the relationship

between said skills. This paper explores this research as the following research question **(R.2)**: *Is the ability to trace a pre-requisite skill for the ability to read or write code?*

For the following section, each of the skills mentioned above will be represented by a related task. The tracing task will consist of 14 multiple choice questions, each question requiring a student to hand execute the code of the question before a correct answer can be given. The reading task will consist of every EiPE question from section B of the exam paper. Finally, the writing task will comprise of every short writing code question from section C of the exam paper.

There will be two types of tests undertaken to explore the relationship of tracing, reading and writing code: (1) the parametric test, retrieving the  $R^2$  value of the correlation between the marks of the tracing, reading and writing tasks; (2) the nonparametric test, comparing the task results as categorical variables, categorised by median, using a Chi-Squared test.

The tests mentioned above will be conducted three times, each focusing on a different relationship: (1) tracing and reading; (2) tracing and writing; (3) tracing and the sum of reading and writing.

It must be noted that the following tests focus solely on the relationships of the tracing task, not the relationship between reading and writing code. In addition to already being well-established in research (see section 2.2.1 of the Literature Review Chapter), the relationship between reading and writing code for this dataset is already established in the EiPE to Writing Code test in section 4.2.1 above.

#### 4.3.1. Parametric Test

As described by Venables, Tan & Lister (2009), the initial research of the relationship between the abilities of tracing, reading and writing code have all considered parametric tests as their first option. In these examinations, the parametric tests have not provided a strong correlation between the abilities to trace, read and write code. The following parametric tests seek to see if this trend continues.

#### Conducting the Tests

To test by parametric values, the tracing task will be plotted three times: (1) against the reading task; (2) against the writing task; (3) against the combined reading and writing code tasks. A scatterplot will be generated; its  $R^2$  value will be the indicator of any correlation to the regression line.

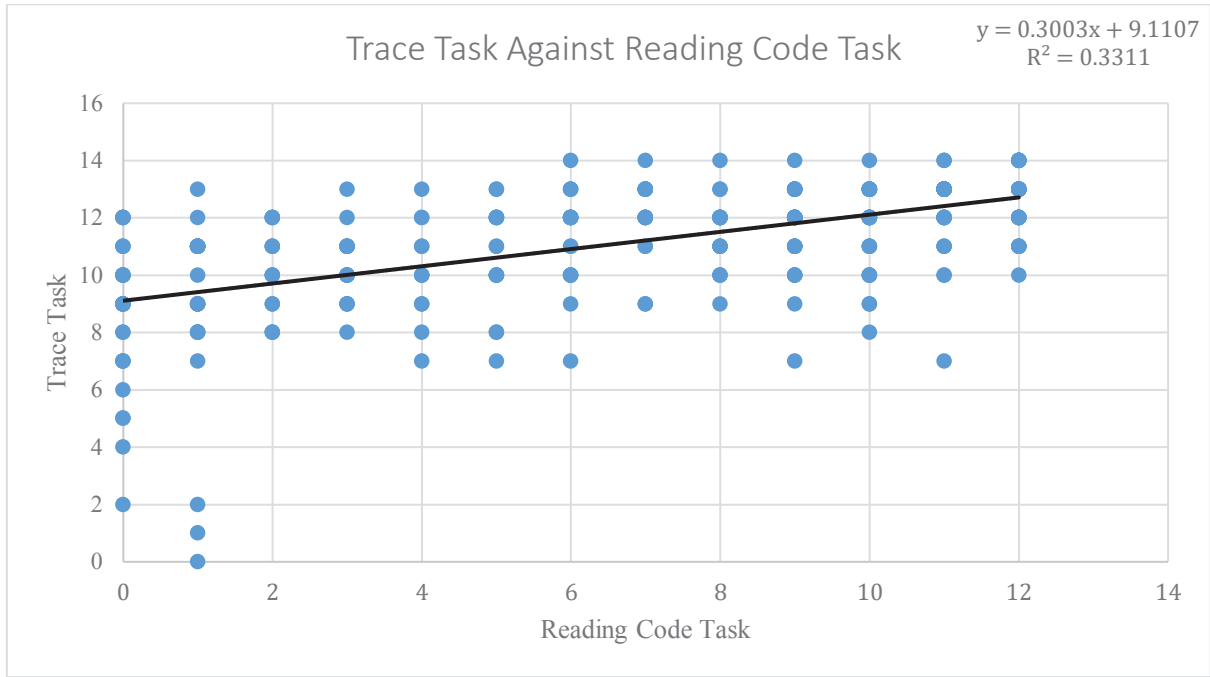


Figure 16 - Parametric Test: Trace Task Against Reading Code Task

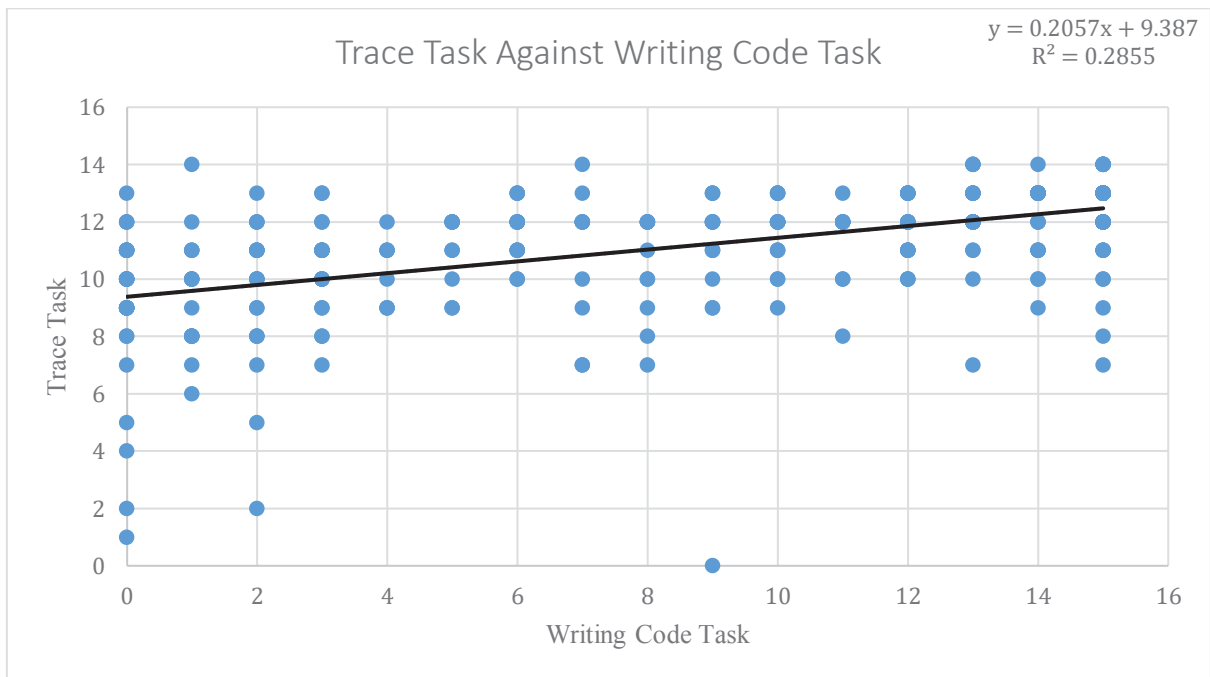


Figure 17 - Parametric Test: Trace Task Against Writing Code Task

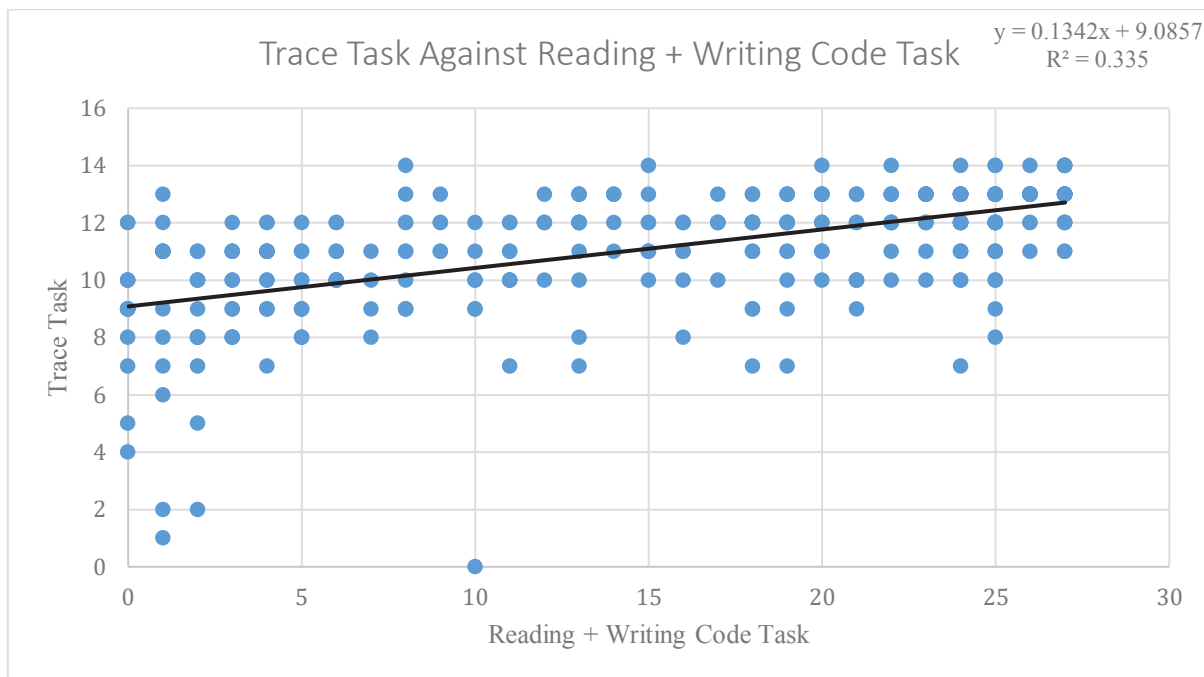


Figure 18 - Parametric Test: Trace Task Against Reading + Writing Code Task

Table 10 - Parametric Tests: Summary of  $R^2$  Results for Parametric Tests

<b>Tasks Compared</b>	<b><math>R^2</math> value</b>
Trace against Reading Code	0.3311
Trace against Writing Code	0.2855
Trace against Reading + Writing Code	0.335

As shown in Table 10 above, the tests have provided a low correlation between the tracing, reading and writing code tasks (the highest proving an  $R^2$  of 0.335). Even on the basis that these are human trials, the  $R^2$  values are notably weak.

However, a low  $R^2$  does not imply a lack of relationship between tracing, reading and writing. There is an observable trend when it comes to values on or above the pass mark for reading code and writing code; that those who fail the tracing tasks will also fail the reading and writing code tasks. This trend is true for all the above tests\*.

\*Note: the outlier observed in Figure 17 for code writing consists of only one student. It can be seen that when combined with the reading code task, the observation remains true as their combined mark of 10 is below the level of pass (13.5).

#### Not Correlated but Not Independent

The above results for the tests are similar to prior research conducted regarding parametric-based testing for the correlation between tracing, reading and writing code. An  $R^2$  value describes how close the data are plotted on the regression line; it describes how the variability of the tracing mark will explain the entire variability for the reading or writing marks. It does not explain the above-observed trend, where those who fail the tracing task will also fail the reading and writing tasks.

A low mark in the tracing task will lead to a low mark in the reading and writing code tasks; they fail in all the tasks. This trend is not reversible: a high mark in the tracing task will not lead to a high mark in the reading or writing tasks.

Such an observation lends support to what was reported by Venables, Tan & Lister (2009), that knowledge of tracing is a necessary but not sufficient skill. To further explore this trend a nonparametric test, categorised by the median, will identify if such a relationship exists.

#### 4.3.2. Nonparametric Test

The parametric tests in section 4.3.1 showed an observable trend: that those who fail the tracing task will also fail the reading and writing code tasks. This trend was not linear: those who achieved a high mark for the tracing task did not necessarily achieve a high mark for the reading and writing code tasks.

Venables, Tan & Lister (2009) noted a comparable situation in their experimentation, where a parametric test only hinted at a relationship. After conducting a nonparametric test, they concluded that passing the tracing task was necessary to pass the code writing task, but is not sufficient enough to enable code writing at an operational level.

The following nonparametric tests seek to extend this prior observation to both reading code in addition to writing code. It also seeks to plot the direction of this relationship, as the previous tests observed only that the binary variables were not independent of each other: this will statistically prove the existence of a relationship in the identified direction.

#### Conducting the Tests

To test the data in a nonparametric form, a student's tracing task mark will be categorised as either above (1) or below (0) the median: a binary variable. The same categorisation will be applied to the student's mark for the reading and writing code tasks. The results will be tabulated in contingency tables, a Chi-squared test and phi-coefficient will be calculated for each table.

The Chi-squared test will determine whether the binary variables are independent or non-independent (a relationship), where (at  $DF = 1$ ) a  $\chi^2$  value  $> 3.841$  shows a relationship. Phi-coefficient will determine whether there is an association between the binary variables, in the direction of the findings.

Table 11 - Contingency Table: Trace Task Against Reading Task

	<b>Below Reading Code Median</b>	<b>Above Reading Code Median</b>	<b>Total</b>
<i>Below Trace Median</i>	109	43	152
<i>Above Trace Median</i>	51	131	182
<b>Total</b>	<b>160</b>	<b>174</b>	<b>334</b>

Table 12 - Contingency Table: Trace Task Against Writing Code Task

	<b>Below Writing Code Median</b>	<b>Above Writing Code Median</b>	<b>Total</b>
<i>Below Trace Median</i>	115	37	152
<i>Above Trace Median</i>	50	132	182
<b>Total</b>	<b>165</b>	<b>169</b>	<b>334</b>

Table 13 - Contingency Table: Trace Task Against Reading + Writing Code Task

	<b>Below Reading + Writing Code Median</b>	<b>Above Reading + Writing Code Median</b>	<b>Total</b>
<i>Below Trace Median</i>	114	38	152
<i>Above Trace Median</i>	50	132	182
<b>Total</b>	<b>164</b>	<b>170</b>	<b>334</b>

Table 14 - Nonparametric Tests: Summary of Statistics for Contingency Tables

<b>Tasks Compared</b>	<b>P-Value</b>	<b>Chi-Squared (<math>\chi^2</math>)</b>	<b>Phi-Coefficient (<math>\phi</math>)</b>
<i>Trace against Reading Code</i>	$1.73 \times 10^{-15}$	63.35	0.4355
<i>Trace against Writing Code</i>	$1.7672 \times 10^{-18}$	76.93	0.4799
<i>Trace against Reading + Writing Code</i>	$5.094 \times 10^{-18}$	74.86	0.4734

Table 14 above summarises the findings of the nonparametric tests conducted on the contingency tables. With the p-values much lower than the significance level of 0.05 ( $1.73 \times 10^{-15}$  the highest), it can be assured that observed values are not to random chance.

Regarding independence of the variables, Table 14 records  $\chi^2$  values much higher than the required  $\chi^2$ -critical level of 3.841 (63.35 the lowest). This shows that the binary variables of the contingency tables are not independent. However,  $\chi^2$  does not identify the relationship between the binary variables. Phi-coefficient ( $\phi$ ) provides insight into the direction of a relationship within a  $2 \times 2$  contingency table. The positive numbers listed for  $\phi$  in Table 14, describe a positive relationship in the direction of top left (both tasks below the median) to bottom right (both tasks above the median).

When looking at the positive relationship in the trace task against the reading + writing code task (Table 13), 73.65% of students are contained within this relationship. A similar percentage of students is evident in Tables 11 and 12. The percent of students in the binary variables for the trace task against the reading + writing code can be seen in Table 15 below:

Table 15 - Percent of Students in Binary Variables for Trace Task Against Reading + Writing Code Task

	<b>Below Other Tasks Median</b>	<b>Above Other Tasks Median</b>	<b>Total</b>
<i>Below Trace Median</i>	34.13%	11.38%	45.51%
<i>Above Trace Median</i>	14.97%	39.52%	54.49%
<b>Total</b>	<b>49.10%</b>	<b>50.90%</b>	<b>N = 334</b>

#### 4.3.3. Tracing: A Necessary but Not Sufficient Skill

The findings in both the parametric and nonparametric tests support the notion by Lister, Fidge & Teague (2009), where knowledge of tracing is a necessary but not sufficient skill.

This notion can be divided into two points: (1) to be able to read or write code; the ability to trace is a pre-requisite skill; (2) the relationship between tracing, reading and writing code is not linear.

Parametric tests have provided evidence of point 2, where a high ability to trace does not translate to a high ability to read or write code. When plotting the tracing, reading and writing code tasks, it is not surprising that a weak correlation was recorded between the tasks.

The parametric tests alone could not support point 1: only a suggestion of it was observed as a trend. This observation found that those who failed the tracing task also failed the reading and writing code tasks. Confirmation of point 1 was made when testing this observation in a nonparametric manner. Categorized by whether their mark was above or below the median, the results were tabulated into contingency tables. Analysis of these tables showed a strong relationship: the positive relationship contained most of the students tested (average of 73.15%). Ultimately it is not surprising that this relationship is nonparametric as the ability to trace is a necessary but not sufficient skill.

#### 4.4. Measuring Tautology: An Indication of SOLO Level

As part of the SOLO taxonomy Biggs & Collis (1982) discuss a ‘Relating Operation’, this refers to how an asked question and a given answer interrelate. At the lowest level of reasoning, labelled pre-structural, a student struggles to decouple the question from the answer. One indication of this struggle is a tautological response, an answer which restates the question and provides no further information or observation from the student.

If as stated by Murphy, McCauley & Fitzgerald (2012) that the marking of EiPE questions is a good gauge for the current transition of a novice to the expert, then reference to the question may be an indication of how little they have progressed towards the expert. This paper seeks to explore this as the research question (**R3.1**): *Is a tautological response evidence for this transition [between novice and expert]?*

Biggs & Collis (1982) provide an example of a tautological response when applying their taxonomy to Geography; an earlier geographical education study is reclassified using the SOLO taxonomy. In this study, students are asked a question after reading a passage from a textbook (Rhys, cited in Biggs & Collis 1982, pp. 130-2). Biggs & Collis consider the passage (or text) to be part of the question as it provides context essential to answer the question.

The passage reads: “Diamond Brook, which had once been brimming and crystal-clear through the year, had... vanished [leaving] only its stoney channel... left”. With this in mind, the following question was asked: “Why did the deep fertile soil cover disappear and make farming impossible?”.

One answer was classified at the pre-structural level: “Because Diamond Brook was drying up” (Rhys, cited in Biggs & Collis 1982, p. 131). It was observed that the response was essentially tautological, that the answer was given without consideration to what was being asked by the question: a restatement of information read from the passage of text.

It would stand to reason that such responses would also be evident in the answers for the EiPE questions, as the responses to these questions have proven to be consistently categorizable to the levels of the SOLO taxonomy (Lister et al. 2006). The EiPE questions are, disregarding differences in the domain, similar to the above geography question. Instead of a passage from a textbook, a segment of code is provided for the student to read. The



question is also similar, asking for knowledge that cannot be obtained by a line-by-line read through of the text or code. To answer either question correctly requires a deeper knowledge of the domain; an answer is not obtainable by direct reference to the text or code.

The manner of which a student can provide a tautological response is two-fold: either by direct or indirect reference to the question. A direct reference would involve answering the question by partly (or wholly) quoting words used in the question verbatim. An indirect reference would involve padding the answer with unrelated information and restate the words from the question in a different manner (e.g. using the word 'text' instead of the word 'string' from the question).

As the questions and answers to the EiPE questions are stored in a format that can be queried, tests can be conducted to determine if the direct or indirect reference to the questions occurs and, if so, at what range of marks. The following two sub-sections will cover the tests which attempt to measure the direct and indirect referencing respectively. For the rest of this section (4.4), a reference to the 'question' will refer to the code in addition to the question itself.

#### 4.4.1. Direct Reference

To determine if the direct reference to the questions is evident in the answers given to the EiPE questions, two methods will be tested. The first will compare the percentage of words in the student's answers that match words from the question. It will detect students who use the words from the question as a crutch. However, this method can be sensitive to either a lack of words used in the question or an abundance of other words used in the answer. The second method is completely resistant to this; it will count only the number of matching words used by the student.

Both methods will be compared to the mark received for the EiPE section and the total mark.

#### Conducting the Tests

These four tests will be plotted on scatter plots; this will give a visual indication of the spread. Both  $R^2$  and Pearson's  $r$  will be calculated: with  $R^2$  denoting how the percentage or count of the repeated words influence the tested mark, while Pearson's  $r$  denotes how closely (or sparsely) they are correlated.

The below is a brief restatement of the process for this test, it is explained in greater detail in section 3.3.1 of the Approach Chapter.

The percentage of a student's answers that were tautologous can be calculated as: the per question summation of the distinct words in an attempted answer divided by the summated count of distinct words that match the words from the question asked.

To put this into an equation:

$$\forall s \in C, s_p = \frac{\sum_{i=1}^t a_i}{\sum_{i=1}^t q_i}$$

Where:

**Symbol Meaning**

$C$	The cohort of introductory programming students undertaking the final exam. The $C$ has a size of 334 students.
$s$	A student within $C$ .
$p$	The percentage of words in the answers given by $s$ that were tautological.
$a$	The total distinct words used in an answer by $s$ .
$q$	The number of words in $a$ that matches the words in the question
$t$	The number of attempted EiPE questions by the student: maximum of 12.
$i$	The current question number (treating the first question as 1).

To count the words correctly, two conditions will be set: (1) the words will distinct: if a word is repeated more than once in the same answer (or question) it will only be counted once; (2) words will only be counted if they are greater than three characters in length or they are the names of a variable. These conditions ensure simple words like 'it', 'or' and 'a' are not counted, but words such as 'array', 'loop' and the variable named 'a' are.

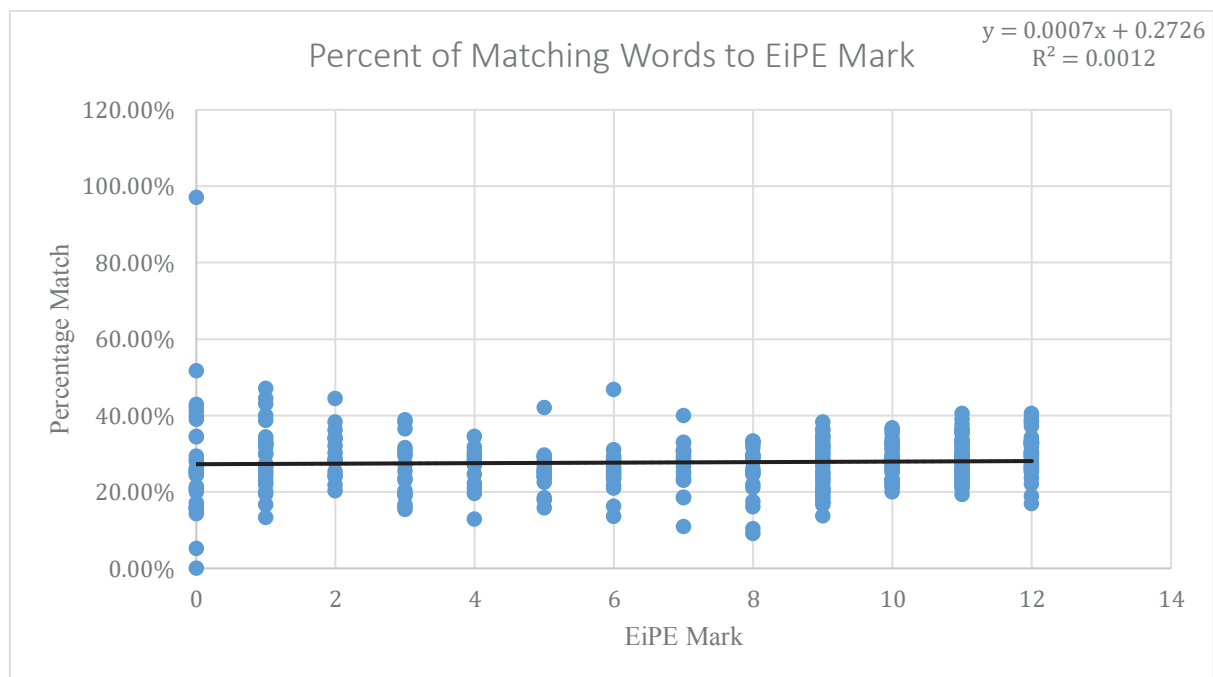


Figure 19 - Direct Reference: Percent of Matching Words to EiPE Score

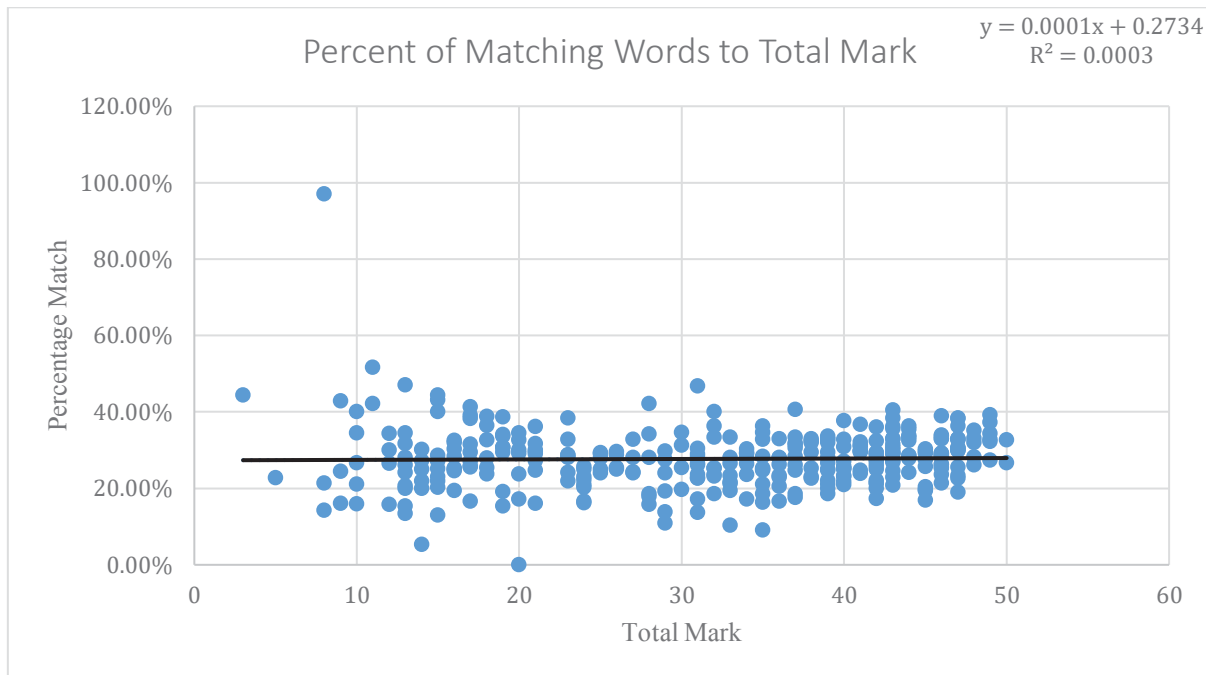


Figure 20 - Direct Reference: Percent of Matching Words to Total Mark

When comparing the percentage of matching words to the cohort’s total mark, we can observe in Figures 19 and 20 that the spread of the models is very flat. Table 16 below summarises the statistical findings of the above figures. With the highest reported  $R^2$  out of the above figures reading only 0.0012, it can be confidently said that there is not enough evidence to prove a relationship between the percent of matching words and the marks for the EiPE section or the total mark of the exam.

Table 16 - Direct Reference: Percentage of Matching Words Statistics Summary Table

<b>Direct Reference Test</b>	<b>R<sup>2</sup></b>	<b>Pearson’s r</b>
<i>Percent of Matching Words to EiPE Mark</i>	0.0012	0.0344
<i>Percent of Matching Words to Total Mark</i>	0.0003	0.0179

This result was somewhat predicted, it was noted earlier that the test for percentage is sensitive to the number of distinct words in both the question and answer. The fact that, apart from one outlier, the percentage of repeating words never exceeds roughly 50% may be an indication of this. This percentage may have called into question the validity of the above tests, if not for the results shown in the following tests for the count of matching words.

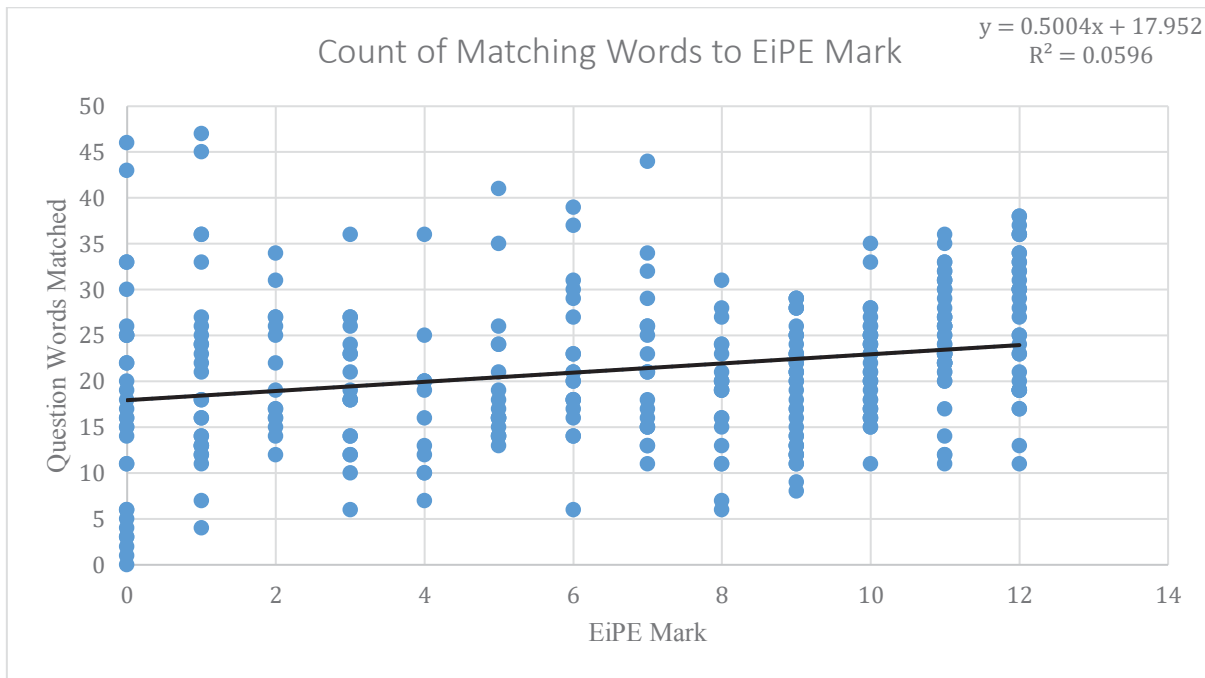


Figure 21 - Direct Reference: Count of Matching Words to EiPE Mark

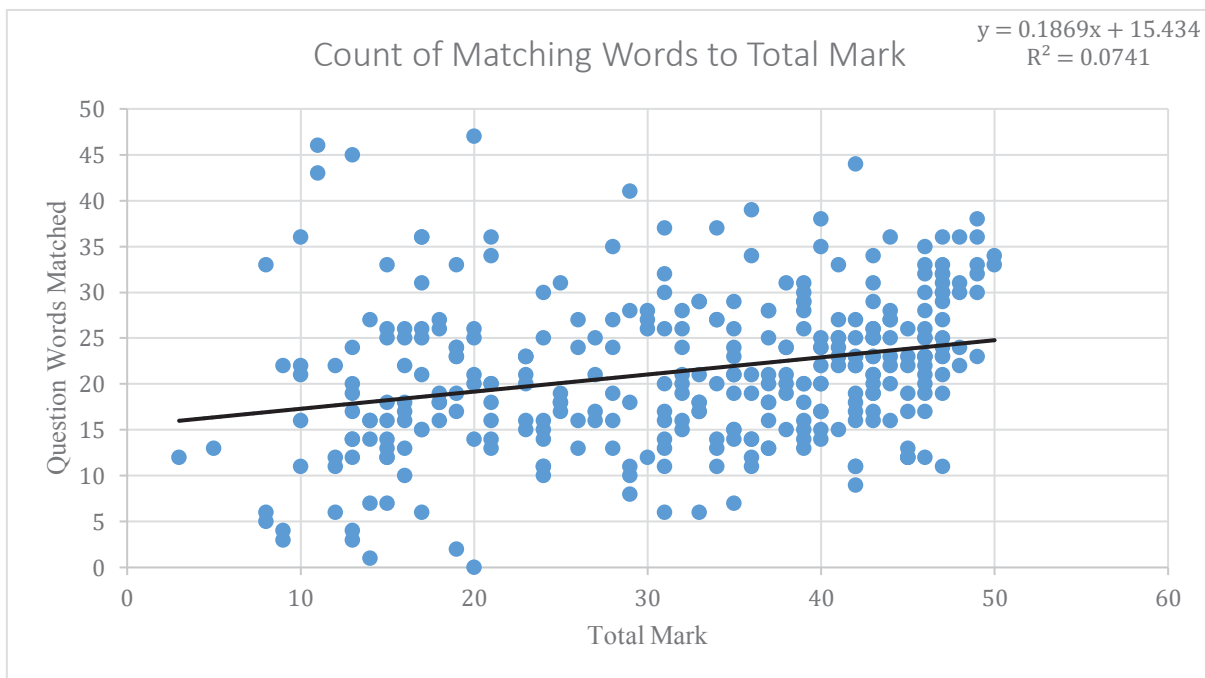


Figure 22 - Direct Reference: Count of Matching Words to Total Mark

The count of matching words is, as opposed to the percentage of matching words, completely resistant to any influence that a lack of distinct words in the question or an abundance of words in a student’s answer may have. In saying this, however, Figures 21 and 22 describe practically the same situation that was observed in the matching percentage tests (of Figures 19 and 20): although not neatly as grouped. Table 17 below shows that the highest  $R^2$  value recorded was 0.0741 which, while larger than the  $R^2$  values for the previous percentage tests (highest being 0.0012), provides no evidence for a relationship between the count of matching words to the EiPE mark or total exam mark.

Table 17 - Direct Reference: Count of Matching Words Statistics Summary Table

<b>Direct Reference Test</b>	<b>R<sup>2</sup></b>	<b>Pearson's r</b>
Count of Matching Words to EiPE Mark	0.0596	0.2442
Count of Matching Words to Total Mark	0.0741	0.2721

#### 4.4.2. Indirect Reference

Instead of directly referencing the question with exact words, a student may be indirectly referencing the question with generalised words (e.g. the general word 'list' instead of the exact word 'array' from the question). If that is the case, it begs to reason that the student who uses more words in their answer would be giving a more line-by-line readout of the question as an answer (an incorrect answer).

#### Conducting the Tests

To test if an indirect reference can be used as a measure of a tautological response, two tests will be undertaken. The first will check for an overall word count against the total mark; this will determine whether the verbosity (or conciseness) of a student's answer is an indication of indirect reference to the question. As it is a count, the first test is sensitive to students who have not attempted answers (missed answers are discussed further in section 4.1.1): There will be fewer words to test than those who have completed more answers.

The second test is completely resistant to that issue, taking the word count divided by the questions a student has attempted to answer. This returns a per attempted question average of words used by a student in their answer. To put this into an equation:

$$\forall s \in C, s_a = \sum_{i=1}^t w / t$$

Where:

#### **Symbol**   **Meaning**

<i>C</i>	The cohort of introductory programming students undertaking the final exam. The <i>C</i> has a size of 334 students.
<i>s</i>	A student within <i>C</i> .
<i>a</i>	The average words used in the attempted answers given by <i>s</i> .
<i>w</i>	The words used in an answer by <i>s</i> .
<i>t</i>	The number of EiPE questions attempted by <i>s</i> .

Unlike the direct response tests, the words used in the indirect response tests are not needed to be distinct to have a valid test. For comparison's sake, the distinct word tests are reported in section 6.4 of the Appendix Chapter.

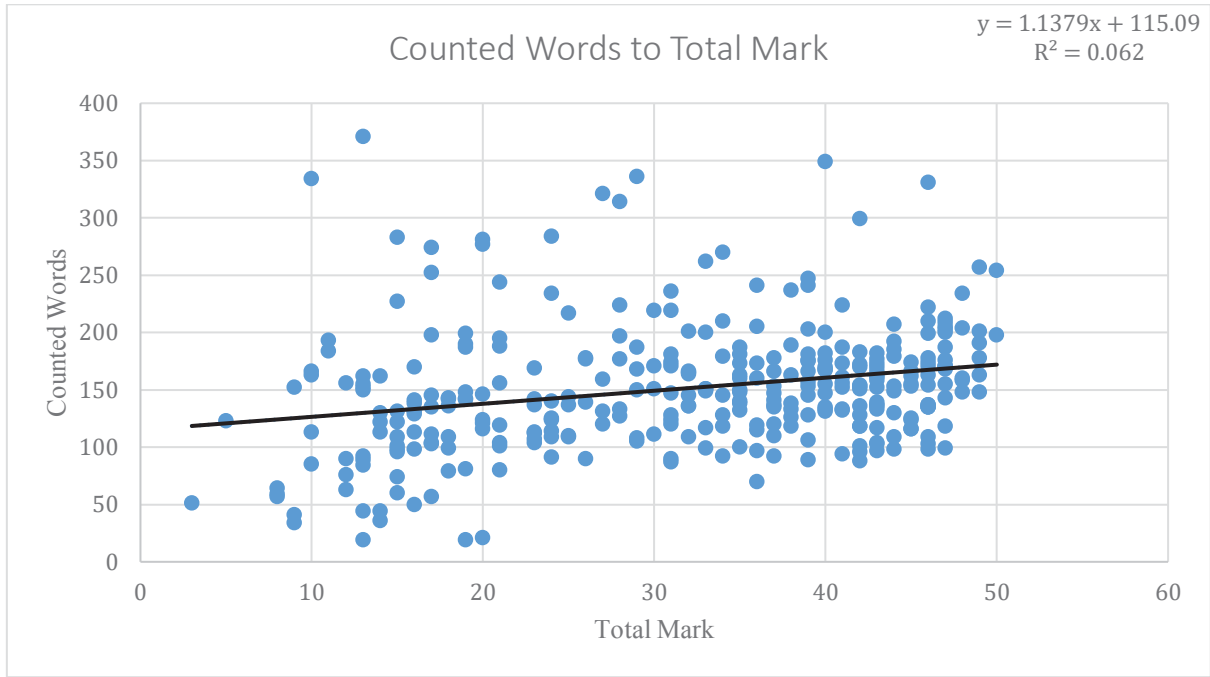


Figure 23 - Indirect Reference: Counted Words to Total Mark

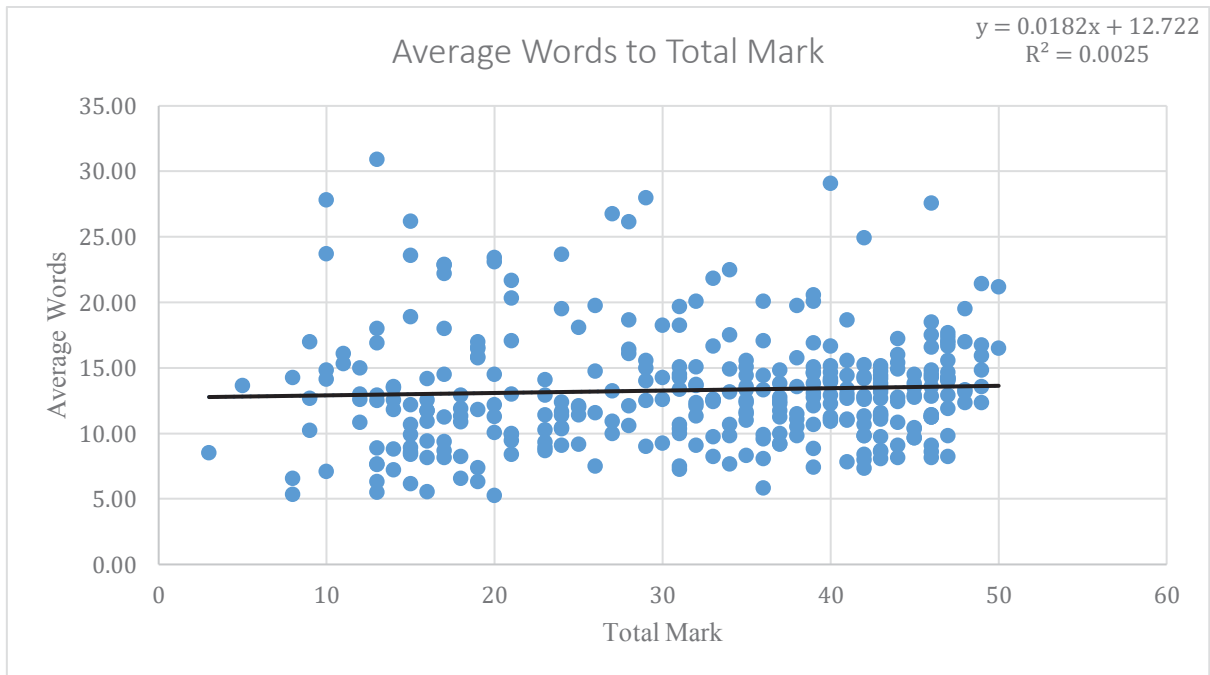


Figure 24 - Indirect Reference: Average Words to Total Mark

When comparing both the counted and average words to the total mark, we can see in Figures 23 and 24 above that the regression line is quite flat. The recorded results are summarised in Table 18:

Table 18 - Indirect Reference: Summary Statistics of Indirect Reference Tests

<i>Indirect Reference Tests</i>	<b>R<sup>2</sup></b>	<b>Pearson's r</b>
<i>Counted Words to Total Mark</i>	0.062	0.249
<i>Average Words to Total Mark</i>	0.0025	0.05

With the highest reported R<sup>2</sup> value for the above tests being 0.062, it can be determined that neither the count of words nor the average of words in a student's attempted answers correlates to the total mark received by the said student.

#### 4.4.3. Tautology is not Measured in this Cohort

Biggs & Collis (1982) assert that a tautologous response is an indication of the pre-structural level SOLO level. It was seen in the tests conducted by Sheard et al. (2008) that the responses to EiPE questions could be categorised consistently to the same level of the SOLO taxonomy: the responses were multi-structural. If this holds true for all EiPE responses, then it is not surprising the results are uncorrelated. An increase or decrease in tautological response may only occur when students within the cohort are at varying levels of the SOLO taxonomy (novice to expert). Alternatively, as this exam is not compulsory, this could be an indication that the students who are at the pre-structural level did not attempt the exam.

Regardless of speculation, it can be concluded that the above tests found no evidence of direct or indirect referencing in the cohort's answers. A tautological response may indeed be an indication of the lowest SOLO level, direct or indirect reference to the question could also be the measure for determining this. It is clear however that the tests recorded within this section are not ways of determining such measurements for this data set.

#### 4.5. Domain-Specific Language

When testing for the answering of EiPE questions and its dependency on English language proficiency (section 4.4.2), the test for indirect reference looked for repetition of words in the cohort's answers. Said test depicted a repeating word count (repeated within their range) which, categorised by mark range, more than doubled between the low (0-3) and middle (4-9) ranges but only increased 4.6% between the middle and high (10-12) ranges.

English language proficiency may explain the difference between the low and middle ranges, but neither proficiency nor correlation to marks can explain the difference between the middle and high ranges. An explanation must lie in how the repeating words are used by the cohort in their answers. To that end, this paper will conduct tests on these repeating words to explore the following research question (**R3.2**): *Do students within a certain mark range show shared language within their answers, could this be used to show the transition?*

Referred to prior in section 4.2.2, Figure 25 below depicts an EiPE question taken from the exam, this is followed by Table 19 which provides three sample answers for the said question:

Q27. In one sentence that you should write in the empty box below, explain in plain English what this code does.

Assume that the “x” is an array of five integers.

```
temp = x[4];
x[4] = x[0];
x[0] = temp;

temp = x[3];
x[3] = x[1];
x[1] = temp;
```

Figure 25 - Example EiPE Question

Table 19 - Sample Answers for Example EiPE Question

<b>Answer Number</b>	<b>Student answer</b>
A1	“It reverses the array x”.
A2	“It reverses the order of the elements inside array x”.
A3	“Using 'temp' to swap the first and last array and also the second and third.”

Regarding the repetition of words, Answers A1 and A2 repeat the words ‘array’, ‘x’ and ‘reverses’ while A3 repeats only the word ‘array’. When providing an answer to an EiPE question, there are two types of words that need to be used: one that describes the manipulation of a variable and the other which describes the variable itself (in this case an array). Without using both a student has failed to describe what the piece of code does (its purpose). Both A1 and A2 are samples of correct answers given by students; they identify the variable and how it is being manipulated. A3 lacks a description of how the variable is being manipulated. Therefore, A3 is a sample of an incorrect answer.

When constrained to a question seeking knowledge of a specific domain, regardless of the flexibility of the English language, there are only a limited number of ways to properly answer the question; this is especially evident when applied to the domain of programming. Consider the concept of mental modelling (which is arguably is an application of ‘chunking’); skilled programmers will use previously learned structures of code and combine them appropriately to create a solution to a given problem. This is what can be seen with the EiPE responses. However, instead of code structures, students are using words to describe the purpose of the code. These words are taken from a domain-specific language, of which students must make use of to answer the question correctly: this language is developed throughout a student’s study on the subject.

Like learning any language, a competent student will come to know new words to expand their vocabulary. They will also learn the grammar of the language, increasing their ability to use these new words correctly while communicating. If a developing language is the cause of the observed shift between the mark ranges, then the use of repeating words across the EiPE answers will identify the development of both the domain-specific vocabulary and grammar.

#### 4.5.1. Words Used Correctly?

The following tests for vocabulary and grammar will consider the correct usage of the repeated words. The words will be put into three categories based on the rate the words are used correctly. This rate is determined by taking the number of times a word was used in a



correct answer divided by the number of times the word was used in an incorrect answer. Based on this rate, a value that is higher than 1.2 is considered ‘mostly correct’, lower than 0.8 to be ‘mostly incorrect’ and a value between these (0.8-1.2) to be ‘mostly equal’. By categorising the repeated words, the count of correct word usage can be used in nonparametric tests against their mark range.

#### 4.5.2. Domain-Specific Vocabulary: Unique Words

As stated before, the difference between the EiPE mark ranges may be caused by the development of a domain-specific vocabulary. For this to hold true, there must be words that are used at specific stages of their development, only to be replaced when they learn new and better words to communicate the answer.

For example, a piece of code that uses a well-known algorithm to sort an array may be described in different words unique to a mark range. At the low range, it may be described by the action alone. At the high range, it may use the name of the algorithm to describe the action performed. The high range answer would not need as many words to describe the action and will miss some of the words used in the low range answer. It will be the repetition of these words, unique to their range, that will show the development of the cohort’s domain-specific vocabulary.

#### Conducting the Tests

For a word to be unique, it must only be used within its range and not in the others. By using both the count of the unique words to a mark range and the correctness of the usage of these words, the existence of an evolving vocabulary may be apparent. A bar graph will depict the count of repeating unique words to their mark range,  $R^2$  will show how closely the data are grouped to the linear regression line. A frequency table will depict the correct usage of the repeated words,  $\chi^2$  will show if the variables are not independent.

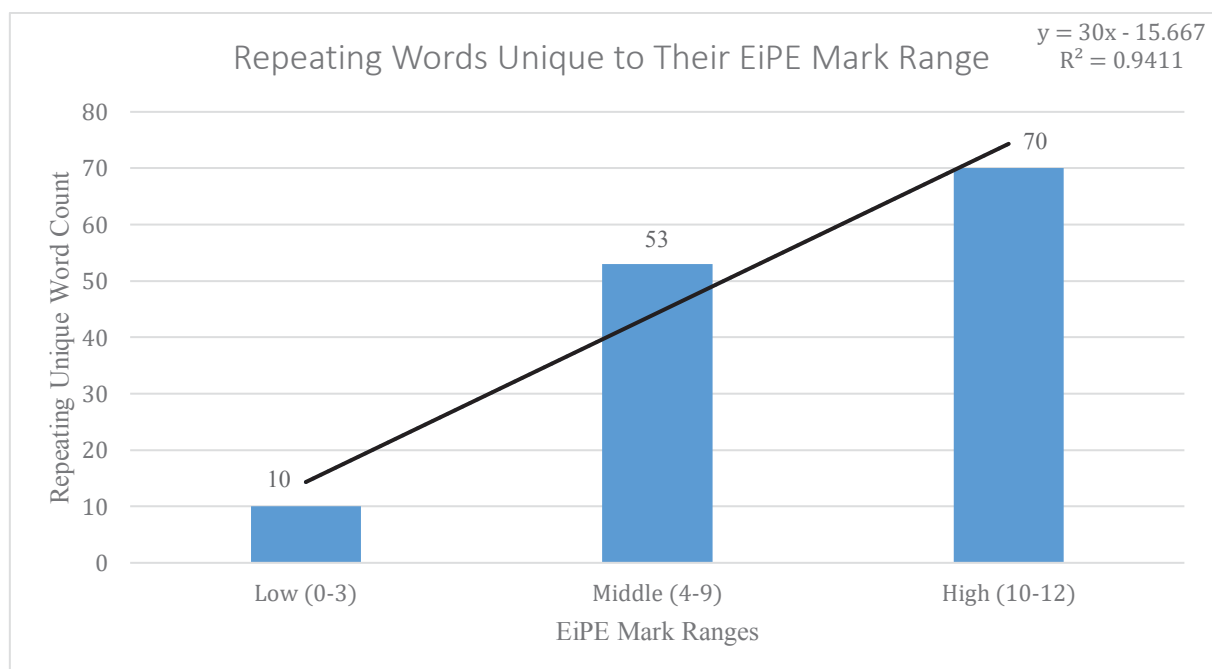


Figure 26 - Unique Repeating Words: Count in Their Mark Range

With an  $R^2$  value of 0.9411, Figure 26 above shows that there is a strong correlation between the count of unique repeating words and the mark range. As the above data are only three points, it is not completely surprising that the  $R^2$  value is that high. Table 20 below provides more evidence to support what is shown above; it depicts that as the mark ranges increase, so does the usage of these unique words in the correct answers.

Table 20 - Unique Repeating Words: Correctness in Mark Range

<b>Word Usage</b>	<b>Low Range (0-3)</b>	<b>Middle Range (4-9)</b>	<b>High Range (10-12)</b>
<i>Mostly Incorrect</i>	10	31	4
<i>Mostly Equal</i>	0	4	0
<i>Mostly Correct</i>	0	18	66

<b>Results</b>	<b>Value</b>
$\chi^2$	69.067
<i>p-value</i>	$3.5711 \times 10^{-14}$

With an  $\chi^2$ -critical value greater than 9.4877, the results outlined in Table 20 above show that the variables are not independent of each other.

It can be clearly seen that not only does the number of unique words increase to the mark range, but so does a student's ability to use these words correctly. Showing that a more competent student will add new words to their vocabulary as they learn in the domain. These words will better answer the question in more descriptive ways, words of which the students in the lower mark ranges have yet to learn of or understand.

#### 4.5.3. Domain-Specific Grammar: Shared Across Mark Ranges

An essential part of learning a language lies with understanding the grammar of the language: how one orders words in ways that convey meaning to an intended audience. Programming is no different to this, with structures strung together to form instructions that a machine can interpret and execute correctly. It is the correct order of these structures that is the grammar of programming.

EiPE questions require a student to read pieces of code and explain the purpose of them; this explanation requires correct interpretation of the code's grammar. The ability to use and understand the domain-specific grammar evolves as a student becomes more knowledgeable in the domain. An indication of this evolving use of grammar may be observable in the repeating words shared between mark ranges.

Consider the following three hypothetical answers in Table 21 below to a hypothetical EiPE question:

Table 21 - Domain-Specific Grammar: Hypothetical EiPE Answers

<b>Answer Number</b>	<b>Mark Range</b>	<b>Example Answers</b>
A1	Low (0-3)	Number 'x' is sorted in the array ascendingly.
A2	Middle (4-9)	The number elements in array 'x' are bubbled ascendingly.
A3	Middle (4-9)	The array is bubbled.
A4	High (10-12)	Using bubble sort, the elements in array 'x' are sorted ascendingly.

This hypothetical question uses a bubble sort algorithm to sort an array of numbers ('x') into ascending order. Note that the answers A1 and A2, which are of low to middle mark range, use practically the same words. Knowing the context of this question highlights the importance of using correct grammar: the wrong use of grammar has led to answer A1 incorrectly identifying what is being manipulated.

Answers A3 and A4 share similar words. However, the use of the word 'bubble' in answer A3 is grammatically incorrect: the student cannot use the word on its own, reference to the order of the sort has to be made. Note that the middle mark range answers A2 and A3 both use the word 'bubble' with different levels of success; the high range answer A4 uses it correctly.

It will be the recording of the correct use of these shared words within a mark range, which when compared to the use of the same word in the other mark ranges (or range), that will determine if a student's evolving ability to correctly use domain-specific grammar relates to their mark.

### Conducting the Tests

To identify if the use of domain-specific grammar evolves as the student progresses in the domain, two tests looking at different types of shared words will be undertaken. One will look at repeating words that appear in all the mark ranges, while the other will look at repeating words between just the middle and high mark range.

These tests will be nonparametric, looking at the correct usage of the shared words in the mark ranges. Tabulated in frequency tables,  $\chi^2$  will show if the correctness of the usage of repeated words are not independent of the mark range that is being examined.

Table 22 - Domain-Specific Grammar: Shared in all Ranges

<b>Word Usage</b>	<b>Low Range (0-3)</b>	<b>Middle Range (4-9)</b>	<b>High Range (10-12)</b>
<i>Mostly Incorrect</i>	97	26	0
<i>Mostly Equal</i>	3	10	0
<i>Mostly Correct</i>	7	71	107

<b>Results</b>	<b>Value</b>
$\chi^2$	218.3295
<i>p-value</i>	$4.2897 \times 10^{-46}$

Table 22 above shows a sharp decline in incorrect usage of shared words advancing through the mark ranges: a decrease of 71 between the low and middle ranges. For the high range, there was no case of a shared word being used in more incorrect answers than correct.

With a  $\chi^2$  value of 218.3295, much higher than the critical value of 9.4877, the values are not independent of each other. It can be said that as the student increases through the mark ranges, so increases their ability to use words in a grammatically correct manner: words that are common to all mark ranges.

Shared words amongst all the mark ranges are not the only way to identify a transition by the ability to use domain-specific grammar. It is in words shared between these ranges where this transition becomes more apparent, especially past the point where English language proficiency may not be a factor (the middle to high mark ranges). Table 23 below depicts this relationship:

Table 23 - Domain-Specific Grammar: Shared Between Middle and High Ranges

<b>Word Usage</b>	<b>Middle Range (4-9)</b>	<b>High Range (10-12)</b>
<i>Mostly Incorrect</i>	22	1
<i>Mostly Equal</i>	12	0
<i>Mostly Correct</i>	85	118

<b>Results</b>	<b>Value</b>
$\chi^2$	36.5384
<i>p-value</i>	$1.6353 \times 10^{-8}$

Table 23 above shows that students in the middle range are repeating words to a lower rate of success than those in the high range. 18.6% of repeating words used by the middle range was used mostly incorrectly, while 10% of the words were used mostly equally. This trend is similar to the one observed in the shared words across all ranges test: that as the student progresses between the mark ranges, their ability to use shared words in a grammatically correct manner increases. With a  $\chi^2$  value larger than the required critical value of 5.9915, the correctness of words shared between the middle and high ranges are not independent of each other.

#### 4.5.4. The Presence of a Domain-Specific Language

Essential to understanding any language lies within learning its grammar and vocabulary, a competent student can understand more of the language as they learn new words and ways of using them. With evolving understanding students can communicate increasingly complex meaning and thought to an audience; a programming language is no different to this.

It is this evolving understanding that will develop as a novice programmer becomes more competent with a programming language: being able to combine learned structures (the vocabulary) in a specific order (the grammar) which a machine (the audience) can then interpret and execute. Tests conducted within this section has shown that evidence for a domain-specific language can be found in the words of the cohort's answers to the EiPE questions. Where analysis of the repeated words in these answers, when compared to an increasing mark range, corresponds positively to an ability to use domain-specific grammar and vocabulary.

It is not surprising that this is the case; as shown in a test conducted in section 4.2.1, marks from the EiPE questions strongly correlate to the marks for the writing code questions. Section 2.2.1 of the Literature Review Chapter refers to this as the relationship between the ability to read and write code: it is not too far of a reach to extend an indication of this relationship to include the use of domain-specific language when explaining code.

## Chapter 5. Conclusion

This paper explores the proposition that *the transition from novice to expert programmer can be gauged by analysing the responses to the 'Explain in Plain English' question within a final examination. That not only do the marks received to these questions will indicate their transition but so does the language used in their answers.*

In order to answer the above proposition, the proposition was broken down into a number of testable research questions. This chapter will be separated into the following three sections: (1) a restatement of the research questions identified in section 1.4 of the Introduction Chapter; (2) a summary of the findings in the Results Chapter by the research question; (3) a review of the significance of this paper's findings, concluding with the identification and discussion of future research.

### 5.1. The Research Questions

Below is a restatement of the research questions detailed in section 1.4 of the Introduction Chapter:

- R1** Are the Explain in Plain English questions a suitable examination method?
  - R1.1** Are the results given for the Explain in Plain English questions comparable to the results from traditional examination methods (multiple choice and writing code)?
  - R1.2** Is the ability to answer the Explain in Plain English questions dependent on the student's English language proficiency?
  
- R2** Is the ability to trace a pre-requisite skill for the ability to read or write code?
  
- R3** Is the language used in the answers to the Explain in Plain English questions indicative of the transition from novice to expert programmer?
  - R3.1** Is a tautological response evidence for this transition?
  - R3.2** Do students within a certain mark range show shared language within their answers, could this be used to show the transition?

Note: Research question **R2** was determined after exploration of the data set. The opportunity arose to extend previous research regarding the ability to trace as a pre-requisite for the ability to read and write code, without the need for additional data. The justification for the inclusion of this research question originates from the prior research into tracing, where it is indicated that the ability to trace is linked with the ability to read, and therefore explain code.

## 5.2. The Findings of this Paper

As stated prior, the following sections will be ordered by the research questions of this paper. Each section will summarise the relevant, to their research question, findings in the Results Chapter. Each sub-section will begin with their corresponding research question.

### 5.2.1. Suitability of EiPE as an Examination Method

**R1** *Are the Explain in Plain English questions a suitable examination method?*

Despite the extensive research regarding the benefits of the EiPE question as an examination method, the low uptake of the EiPE question may be due to lack of the confidence in the question as a suitable examination method. Section 2.3.1 of the Literature Review Chapter discusses this low uptake in further detail. The term ‘suitable’ is a label, one that is applied when the subject meets particular criteria to enable its application. Taking into consideration the criticism surrounding the EiPE questions, the research questions **R1.1** and **R1.2** were developed as criteria for the EiPE question to meet.

### Comparison Against Traditional Examination Methods

**R1.1** *Are the results given for the Explain in Plain English questions comparable to the results from traditional examination methods (multiple choice and writing code)?*

Prior research regarding EiPE questions has been centred around the benefits that analysing the ability to read and explain code brings, abilities that were not previously tested to the same extent by traditional examination methods. Prior research has also indicated there to be a strong relationship between the abilities of reading, explaining and writing code (see section 2.2.1 of the Literature Review Chapter for more on this), but there has yet to be research into the degree of which this relationship affects the marks of the examination method.

It would be expected that, regardless of the examination method, those who perform poorly with one method would also perform poorly with the others; the reverse of this should also be the case. To determine if this is the case with the EiPE questions, the cohort’s EiPE marks were compared against the cohort’s marks for the traditional examination methods.

Comparing the correlations between the EiPE questions and the other examination methods revealed an interesting observation. As noted in Table 24 below, the weakest correlation was with the MCQ (multiple choice questions). A stronger correlation was identified with the writing code method.

Table 24 - Examination Method Comparison: Statistical Results Summary (From section 4.2.1 of the Results Chapter)

<i>Examination Methods Compared</i>	<b>R<sup>2</sup></b>	<b>Pearson’s r</b>
<i>EiPE Against MCQ</i>	0.5381	0.7335
<i>EiPE Against Writing Code</i>	0.6744	0.8212
<i>EiPE Against Traditional</i>	0.7181	0.8474

However, strongest out of all was when the EiPE questions were compared to the combined mark of MCQ and writing code methods (called ‘Traditional’ in the table). To identify the cause of the stronger correlation, the results of all three examination methods were plotted together.

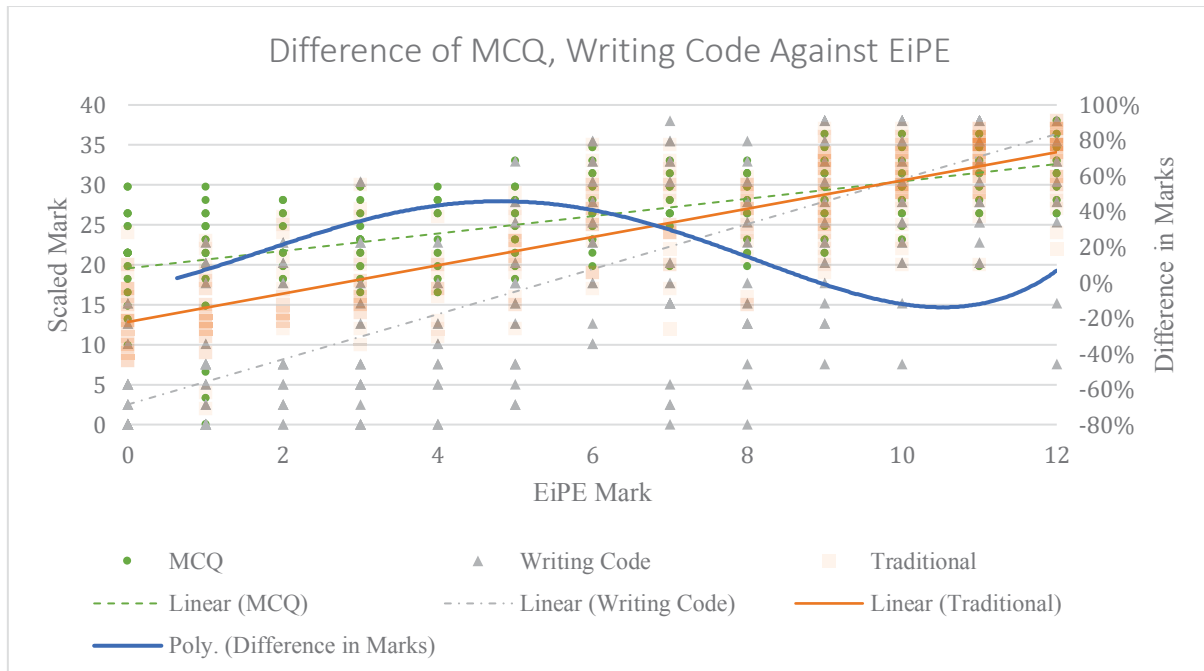


Figure 27 - Examination Method Comparison: Difference of MCQ, Writing Code Against EiPE (from section 4.2.1 of the Results Chapter)

Considering the writing code mark to be the base mark, as it correlated the closest, and the MCQ mark to be adjusting it, Figure 27 graphically depicts the change in the regression line. For comparison, the marks of the writing code and MCQ were scaled to their maximum combined mark. Calculated as a percentage, the difference between these methods was plotted as a polynomial, where a positive or negative value (shown on the right vertical axis) indicated the MCQ’s impact on the ‘Traditional’ regression line.

Cause for the closer correlation when the traditional examination methods are combined may lie in the abilities and knowledge that they are testing for. Considering the situation in an abstract manner: if MCQ tests for ‘a’, code writing tests for ‘b’ and both MCQ and code writing test for ‘c’, that when combined, the traditional examination methods must test for ‘a’, ‘b’ and ‘c’ something that the EiPE questions also test for. However, stopping the analogy here is not a complete description of the situation, as 0.2819 of the spread of the traditional marks is still yet to be explained by the EiPE marks. To accommodate this, it must be said that in addition to ‘a’, ‘b’ and ‘c’, the EiPE questions must also test for ‘d’.

However, this is not unexpected. If the EiPE questions correlated completely with the traditional examination methods, then they must be testing for the same things. This would, therefore, defeat the purpose of the EiPE questions. Speculating, ‘d’ is most likely to be the ability to read and explain code. However, to what degree in testing for ‘d’ is testing for the ability to read or write code, is a matter for future research.



## English Language Proficiency

### **R1.2** *Is the ability to answer the Explain in Plain English questions dependent on the student's English language proficiency?*

The most commonly published criticism regarding the use of the EiPE questions lies in the English language proficiency that is needed to answer the question. This has caused some controversy regarding the EiPE question, leading it to be labelled by some as more of an exercise in English comprehension than code comprehension.

To investigate the above research question, two tests were undertaken: one to look at the spread of marks when plotting the EiPE questions against the other questions and the other to look at the language used in the answers of the EiPE questions.

If the EiPE questions require a level of English language proficiency higher than traditional examination methods, then it stands to reason that this would be reflected in the marks. There would, therefore, be many cases of students who receive low marks for the EiPE questions but high marks for the other examination methods. However, plotting these marks (in section 4.2.2 of the Results Chapter) showed this to be not so. With only 18 students (5.4% of the cohort) meeting the above case, the above argument is not supported by what is seen in the data.

The above test, however, proves nothing regarding the effects of English language proficiency in the answers themselves. There are only so many ways the same piece of code can be explained; it would be expected that correct answers would share the same words in their answers. If English proficiency limits the correct answers, then incorrect answers must be lacking in repeated words. If plotted, it would be expected that as a student's marks increase, so does the count of repeated words.

For the test, three different mark ranges (based on the EiPE marks) were chosen. The first, low (0-3), was determined by the prior English proficiency test, as a mark of four was the point where the regression line crossed the pass point for the traditional marks. The other two, middle and high (4-9 and 10-12), were chosen based on the number of students they contained (about half of the remaining cohort each).

Observing the repeated words, between the low and middle ranges there is a sharp difference of 156 repeated words. However, this sharp difference does not extend past these ranges; between the middle and high ranges, there is only a difference of 14 repeated words. The sharp difference between the low to middle ranges makes sense; considering that the low range contains mostly students who have failed all the examination methods, the sharp difference of repeated words may be the result of a poor grasp of the English language.

Poor proficiency with the English language may indeed affect the ability to answer the EiPE questions. The above tests, however, show that this effect does not extend beyond the point where the students pass the examination methods. It is clear that if the poor English language proficiency is impairing the ability to answer the EiPE questions, then it is also impairing the ability to answer the other examination methods.

Consequently, it can be said that the level of English language required to answer EiPE questions is no greater than the level of English language required for admission into the university.

## 5.2.2. Tracing as a Pre-Requisite Skill

**R2** *Is the ability to trace a pre-requisite skill for the ability to read or write code?*

This paper explored the above research question by extending previous research regarding the ability to trace code. Prior research has shown that the ability to trace is a pre-requisite to the ability of reading and writing code; this relationship is only strongly apparent when conducting nonparametric tests. See section 2.3.2 of the Literature Review Chapter for a discussion on the previous research regarding tracing a pre-requisite skill.

To discover if the same pattern is apparent in this data set, a parametric and nonparametric test was undertaken. 14 multiple choice questions were identified to require tracing (hand execution) of the code before a correct answer could be given.

Similar to the previous studies, plotting the tracing score against all other examination methods (EiPE and writing code) provided weak correlation. However, a visual observation of the plots revealed an observable trend: those that failed the tracing tasks also failed the reading (EiPE) and writing tasks. This was further corroborated by the results of the nonparametric tests.

By categorising the tracing and other task marks by being either above or below the median, the binary variables were then tabulated into contingency tables. The Chi-squared tests for the contingency tables showed values well above the required critical level, as such the non-binary variables were not independent of each other. Phi-coefficient provided a positive direction to these relationships, showing that those who score below the median for trace will also score below the median for the other tasks; the opposite of which is also the case.

Table 25 - Percent of Students in Binary Variables for Trace Task Against Reading + Writing Code Task (from section 4.3 of the Results Chapter)

	<b>Below Other Tasks Median</b>	<b>Above Other Tasks Median</b>	<b>Total</b>
<i>Below Trace Median</i>	34.13%	11.38%	45.51%
<i>Above Trace Median</i>	14.97%	39.52%	54.49%
<b>Total</b>	<b>49.10%</b>	<b>50.90%</b>	<b>N = 334</b>

Table 25 above shows the percentage of the cohort in either of the four options. Within the positive relationship identified by phi-coefficient, 73.65% of the cohort falls within the relationship. The above results re-affirm the previously identified notion that the ability to trace is one that is necessary but not sufficient to enable the reading and writing of code.

## 5.2.3. Language as an Indication of Transition from Novice to Expert

**R3** *Is the language used in the answers to the Explain in Plain English questions indicative of the transition from novice to expert programmer?*

To operationalise the above question, two further research questions were identified, each exploring a different avenue to which the language in the answers of the novice can reveal the

extent of their transition to expert programmer. The corresponding results for research questions **R3.1** and **R3.2** will be summarised in the following sub-sections.

### Tautological Response

#### **R3.1** *Is a tautological response evidence for this transition?*

Referenced within the SOLO taxonomy itself, Biggs & Collis (1982) label a tautological response to be an indication of a pre-structural response. The pre-structural response is one that provides no more information than what is given in the question; it is a restatement of the question. Prior research has shown that responses to the EiPE questions are consistently categorizable to SOLO levels (Lister et al. 2006). Therefore, it would be expected that tautological answers should be identifiable at the lower mark ranges of the EiPE questions.

In the case of the EiPE questions, a tautological response would be one that made reference to the code, providing a line-by-line description of the code. To answer an EiPE question correctly, a responder is required to explain the purpose of the code; a line-by-line answer would be incorrect. To ensure that all types of tautological responses are covered, two separate tests were conducted. The first looked at the direct references to the question, while the second looked at the indirect references to the question.

To test for direct reference, a tautological response rate was calculated for each student. To summarise, for every student sum all the distinct words\* in an answer for each question attempted, this is divided by the sum of the distinct words\* in the answer which matched the words from the question. If a tautological response is an indicator of a lower ability, an anti-correlation should be apparent when comparing this percentage against the EiPE mark.

\*Greater than three characters in length (or the name of a variable) and is not repeated.

Reference to the question may not be made by direct reference; a student may be referring to the code by other words. If this is the case, a line-by-line readout of the code would still require more words than an answer which simply explains the code's purpose. Similar to the direct reference, if a tautological response is an indicator of a lower ability, an anti-correlation should be seen between the number of words in their answers and the mark received for the EiPE questions.

After conducting the above tests, it was found that neither direct nor indirect reference showed any correlation with the marks received for the EiPE section. A number of reasons could be the cause for this. The first lies in the type of student who would provide a pre-structural response; as attendance of the exam is not compulsory, a student who would provide this level of response may not have attempted the exam. The second could be the usual level of responses provided to these questions by the cohort; prior research has shown that responses to the EiPE question are reliably categorised to the multi-structural level (Sheard et al. 2008). If this is to be the case, then tautological responses may not be seen within the answers.

Regardless of speculation, what can be determined is that the above tests are not ways for determining tautological responses within the answers of this cohort.

### Domain-Specific Language

#### **R3.2** *Do students within a certain mark range show shared language within their answers, could this be used to show the transition?*

When explaining the purpose of a given piece of code, there are only so many ways to give a coherent answer. It would, therefore, be logical to assume that the language used in a single correct answer is similar (or the same) to the language used in other correct answers. A previous test in this paper, regarding English language proficiency, concerned the repeated words used within all answers across the mark ranges (see section 4.2.2 of the Results Chapter for this test).

It was observed that the difference between the repeated words of the low (0-3) and middle (4-9) ranges was more than double the repeated words for the low range. A lack of English proficiency may be the cause for the difference in marks between the low and middle mark ranges. However, with only a slight increase of 4.6% in the repeated words between the middle and high (10-12) ranges, the count of the repeated words cannot explain the difference in marks between the two ranges. The difference must be in the way the words were being used; further testing showed that it was an evolving ability to use the language, specific to this domain, that could account for the difference in the marks between the ranges.

Put simplistically, the learning of a language is comprised of two parts: learning the vocabulary of the language and learning the grammar of the language. Developing a vocabulary enables the student to use increasingly complex words to explain a subject, the understanding and order of when to use these words comes with a better grasp of the language's grammar. When it comes to explaining the purpose of a piece of code, learning how to programming is no different to the learning of a language. Evidence from the tests conducted in this paper shows that, through testing for both a developing vocabulary and developing the use of grammar, an evolving domain-specific language is apparent in the responses of the EiPE questions according to the mark range.

To identify the presence of a domain-specific vocabulary, two main tests regarding the use of unique words (used only within a single mark range) were undertaken. The first test involved looking at the count of the unique words. If an evolving domain-specific vocabulary is apparent across the mark ranges, simple words which are learned at the lower mark ranges are later replaced with more complicated words at the higher mark ranges. Those of the higher mark ranges would not need to make use of these simple words, as the words they currently know are more suitable in explaining the purpose of the code. Conversely, those at the lower levels will not have the knowledge to know the words of the higher ranges.

The second test involved the correct usage of these repeated words. By categorising a unique word by the frequency of how often it was used in either a correct or incorrect answer, it could also be determined the degree of which these unique words are being used correctly.

The results from both tests showed that not only does the number of unique words increase according to the mark range, but so does their ability to use these unique words. Therefore, indicating the presence of an evolving domain-specific vocabulary.

To identify the presence of an evolving grasp of domain-specific grammar, two tests regarding repeated words shared across the mark ranges were conducted. Both are similar in that they look at the correct usage of these repeated words, what differs are in the mark ranges these words are shared across. The first test involves repeated words that are shared

across all mark ranges, while the second involves repeated words that are shared only between the middle and high ranges.

Of the words that are shared between all ranges, there is a clear trend that the higher mark ranges will use the shared words correctly at a rate greater than the mark ranges before it. However, this is also evident past the point where English language proficiency may not be a factor. Testing also showed that of the words shared between the middle and high ranges, the high range used these words at a rate which was 28% better than the middle range. The above tests combined show that not only is an evolving grasp of domain-specific grammar evident across all of the mark ranges, but it is also evident between the mark ranges.

As a domain-specific language consists of both a domain-specific vocabulary and a domain-specific grammar, what is found to apply to the vocabulary and grammar will also apply to the language. The identification that an evolution of both vocabulary and grammar is evident in the progression of the mark ranges will also apply to the domain-specific language. Considered together, the results of the above tests show that a novice's grasp of domain-specific language correlates to their EiPE mark. Prior research has shown that an EiPE mark can indicate the current transition from a novice to the expert programmer (Murphy, McCauley & Fitzgerald 2012). Therefore, as the EiPE marks correlate with the domain-specific language, the domain-specific language found within the EiPE responses can be used to identify the novice's current transition to the expert programmer.

### 5.3. Significance and Future Research

In conclusion, to the educator who is interested in the development of their introductory programming students, the answers to this paper's research questions present a number of significant findings.

The educator can be assured that the EiPE question is a suitable examination method to use within a final exam; that the marks given to the EiPE questions are comparable to marks given to traditional examination methods. The educator can also be reassured that the English language proficiency required for the student to answer the EiPE questions is no greater than the English language proficiency that was required for their admission into the university.

Regarding the ability to trace, the findings of this paper support and extend the findings made in previous research; that the ability to trace is a pre-requisite skill for the ability to read and write code. Educators who do not invest part of their curriculum to the teaching or assessment of this ability cannot expect their students to be able to read, explain or write code at any sufficient level.

In regards to identifying a student's current transition from novice to expert programmer through the response to the EiPE questions, this paper provides two findings. The first is that there is no evidence in this cohort that either direct or indirect reference to the question in a student's answer, a tautological response, is indicative of the student's transition from novice to expert. That, possibly, the students who attempt a final exam are at a level that is beyond giving a tautological response. The second and most important finding of this paper provides a new way to identify a student's current transition from novice to expert programmer. That a student's transition from novice to expert programmer is evident in the language in which they use to explain the purpose of a piece of code; a language of which is domain-specific.

### 5.3.1. Future Research

The findings of the research questions within this paper provide multiple opportunities for future research. The following sub-sections will identify this potential research and discuss potential methods for testing them.

#### Quantifying the Remaining EiPE Knowledge

Testing within this paper identified that the EiPE questions evaluate an additional ability that is different to the abilities tested by the traditional examination methods. To put into abstract terms, EiPE tests for the abilities ‘a’, ‘b’, ‘c’ and ‘d’ while traditional examination methods test for only ‘a’ ‘b’ and ‘c’.

Most likely, ‘d’ is testing for the ability to read and explain the purpose of the code. What has yet to be determined is the extent of which ‘d’ is the ability to read and explain the purpose of the code. In conjunction with a correlation test, similar to what was conducted in section 4.2.1 of the Results Chapter, providing an explanation for ‘d’ would give a quantitative measure of all the abilities that are tested by the EiPE questions.

#### English Language Proficiency: A Direct Observation

Regarding the tests conducted for English language proficiency (in section 4.2.2 of the Results Chapter), it was concluded that at the cohort or macro level, the student’s level of English language proficiency was no restriction for completing the EiPE questions. However, these results were determined without consideration given to neither the percentage of the cohort that spoke English as their second language nor the student’s actual level of English language proficiency.

The next step in testing would be to reproduce the macro level outcome on a per student or micro level. Testing at the micro level would involve using a ‘think aloud’ study, similar to what was conducted in a study by Teague & Lister (2014). However, several changes to the Teague & Lister study would need to be made to test for the dependency of English language proficiency. Instead of only testing for one student, a sample of students at multiple levels of English language proficiency would be needed. In regards to the testing of these students, only their final exam would need to be reviewed; as the final exam is the point where it can be assumed the students have the most knowledge of the subject.

It would be the comparison between the observations made while answering EiPE questions, similar to the ones in this paper, and their level of English language proficiency that will provide a result comparable to the macro level test in this study.

The recording of a student’s level of English language proficiency may prove to be difficult. The following are three potential sources of information of student’s English language proficiency.

The first would be information recorded prior to the commencement of the course; retrieving this information would be a matter of internal approval of the institution these students belong with. However, as the proficiency of the student may develop during their attempt of their computer science course, this might not be a true indication of their current English language proficiency.

The second source would be from a survey, asking the student to rate their level of English language proficiency. This is open to the student providing information that is not accurate;

the student may be hesitant to reveal their actual level of English language proficiency, believing that their answer may influence the mark they receive.

The third source would be to have the sample of students sit an English language proficiency test, prior to (or straight after) conducting the think aloud study. Correlating this score to their EiPE marks would provide the most accurate indication of whether the student's marks are directly related to their English language proficiency. However, this type of test has three main drawbacks: (1) there would be a large expense in running an English proficiency exam for each student; (2) without incentives, there would be a low participation rate of the study. It would be difficult to encourage students to be involved with a study asking them to undertake another assessment in addition to the think aloud study; (3) this type of test would not be easily repeatable past a single institution, as the standards of the available English language proficiency tests may differ.

#### Tracing as a Pre-Requisite Skill: A Wider Look

Future research regarding tracing as a pre-requisite skill would require conducting similar tests across multiple institutions. Finding the same trend as seen in this paper at multiple institutions would remove the possibility that the results of this paper are unique to this specific cohort or is due to the educator's teaching style.

#### Tautological Response: An Earlier Examination

The flat results observed within the tautological tests (refer to section 4.4 of the Results Chapter) could be from a lack of students at different SOLO levels. If prior research is correct, students who answer EiPE questions at the end of their introductory programming subject will give responses that can be consistently categorised to the same, multi-structural, level (Sheard et al. 2008).

A tautological response may yet be apparent in those who are at different stages of their education. By changing the tested population, two potential tests can be conducted to explore if a tautological response is apparent. The first involves a test population of both experts and students at varying stages of their course. This would determine if the rate of tautological response will change in the responses which are classified at different levels of the SOLO taxonomy.

The second involves looking at a cohort at an earlier stage of their introductory programming subject. As a tautological response is indicative of the earliest SOLO level, a response which is tautologous may only be evident with those who have the most basic of knowledge.

#### Domain-Specific Language

The tests for domain-specific language have provided plenty opportunity for further exploration into this area. As this is the first time such tests have been conducted, it would be prudent to see if the findings recorded, in section 4.5 of the Results Chapter, are reproducible within different cohorts and years.

Additional grounds for research can be found in the limitations of the tests used within this paper to explore the domain-specific language; the following two sub-sections will outline potential areas for future research.

### Domain-Specific Language: The High-End Problem

Looking at the spread of marks within the high EiPE mark range, Table 26 below shows the count of students within the high mark range by their corresponding mark:

Table 26 - High Range Problem: Student Count Breakdown EiPE High Mark Range (10-12)

<b>Mark</b>	<b>Count</b>
10	35
11	43
12	36

The ‘high range’ problem arises when comparing this count with the results found for the other tests of the domain-specific language. Only 36 of the 114 students in the high mark range have received full marks, the remaining 78 students have incorrectly answered one or two questions. This does not seem to be accounted for within the tests for a domain-specific language. If incorrect word usage corresponded to marks, then roughly 18 out of the 70 words would be used incorrectly. As Table 27 below shows, the observed count of words used ‘mostly incorrect’ was quite different:

Table 27 - High Range Problem: Times Words Were Used Mostly Incorrect by Test

<b>Test</b>	<b>Mostly Incorrect</b>
<i>Words Shared Across All Ranges</i>	0
<i>Words Shared Between Middle and High Ranges</i>	1
<i>Words Unique to Their Range</i>	4

When compared to the expected 18 words, the above figures fall short. If domain-specific language can account for the difference of marks within the high range, then different tests would need to be conducted. Tests which focus on the students who receive marks of 10, 11 and 12.

To this end, tests similar to the ones conducted for the domain-specific language may be sufficient to explain the problem (looking at the marks of 10, 11 and 12). However, the opportunity exists for a test to both an explanation for this problem and a deeper delve into the effects of a domain-specific language: a test that looks at the meaning of words within the answers.

### Domain-Specific Language: Meaning-Based Words

Regarding the development of a domain-specific language in the answers for all the EiPE questions, when looking at the words within these answers, it makes sense to look at the metrics of the words instead of their meaning.

If observed on the whole that the word ‘array’ was the most repeated word used in answers for the EiPE questions (it was); then this would be an interesting trivia fact. However, if the word ‘array’ was found to be the most repeated word within answers for a question that doesn’t mention an array, then this becomes more relevant in determining the aptitude of the



cohort. By looking at the meaning of words on a per question basis, a greater understanding can be found of how these the words make part of an evolving domain-specific language

To show the information that meaning based word testing could provide regarding the development of a domain-specific language, an analysis of the words used in answers to Question 32 is provided. Question 32 is outlined in Figure 28 below:

**Q32. In one sentence that you should write in the empty box below, explain in plain English what this code does:**

```
int z = 0;
for (int i=0 ; i<x.length ; ++i )
    z = z + x[i];
System.out.println(z);
```

Figure 28 - Meaning Based Words: Question 32 of the Exam

By looking at the meaning of words used in this question, evidence of the development of a domain-specific language is apparent. Observe Table 28 below, where the word ‘z’ (the name of a variable) is tabulated:

Note: the percentages of the following tables are compared to the total of the column.

Table 28 - Meaning Based Word Testing: Occurrence of ‘Z’ in Answers for Q32

<b>Mark Range</b>	<b>Correct Count</b>	<b>Incorrect Count</b>	<b>Total Count</b>
<i>Low (0-3)</i>	1 (2.17%)	83 (73.45%)	84 (52.83%)
<i>Middle (4-9)</i>	28 (60.87%)	30 (26.55%)	58 (36.48%)
<i>High (10-12)</i>	17 (36.96%)	0 (0.00%)	17 (10.69%)
<b>Total</b>	<b>46</b>	<b>113</b>	<b>159</b>

It can be seen that as the mark range ascends, there is less reference to the variable ‘z’. Inversely, the correct usage of the word rises. This is no surprise, as a reference to the variable ‘z’ does not (and arguably should not) have to be made to answer the question correctly. The point of the EiPE questions is to determine if a student can communicate and describe the purpose of the code to an audience that has no reference to the code.

Table 29 below shows that of the words used uniquely in the middle mark range, most are used in incorrect answers.

Table 29 - Meaning Based Word Testing: Unique Words in 4-9 Range

<b>Word</b>	<b>Correct Count</b>	<b>Incorrect Count</b>
-------------	----------------------	------------------------

<i>Number</i>	6	2
<i>Each</i>	5	5
<i>Until</i>	3	5
<i>I'</i>	1	5
<i>Length</i>	1	6
<i>Final</i>	1	5

The meaning of the words and their incorrect use suggest that these words (disregarding ‘Number’) appear in answers that describe the code line-by-line: answers of which that do not provide the purpose of the code. A trend that is present within, but does not extend past, the middle range. Table 30 below shows that the use of the one shared word between the low and middle ranges decreases by half between them. As with the unique words, the word ‘loop’ is used in a line-by-line answer. This word is not used at all in the high ranges.

Table 30 - Meaning Based Word Testing: Occurrence of ‘loop’ in Answers for Q32

<b>Mark Range</b>	<b>Correct Count</b>	<b>Incorrect Count</b>	<b>Correctness</b>
<i>Low (0-3)</i>	1 (16.67%)	17 (77.27%)	Mostly Incorrect
<i>Middle (4-9)</i>	4 (66.67%)	5 (22.73%)	Mostly Equal
<i>High (10-12)</i>	Word does not appear in this range.		

Such findings are consistent with the prior tests looking at the number of words used, where the evolution of a domain-specific vocabulary is present as a student rises through the mark ranges. In the above cases, the students seem to transition away from directly referencing the conditions of the loop (or the loop itself) in their answer. At the higher mark ranges, these words are replaced by words that better describe the situation.

Evidence of an evolving domain-specific vocabulary can be seen in Table 31 and Table 32 below, where words such as ‘Find’ and ‘calculate’ are used in conjunction with the word ‘total’ to describe the purpose of the code.

Table 31 - Meaning Based Word Testing: Unique Words ‘Find’ and ‘Calculate’ in 10-12 Range

<b>Word</b>	<b>Correct Count</b>	<b>Incorrect Count</b>
<i>Find</i>	6	0
<i>Calculate</i>	5	0

Table 32 - Meaning Based Word Testing: Occurrence of ‘total’ in Answers for Q32

<b>Mark Range</b>	<b>Correct Count</b>	<b>Incorrect Count</b>	<b>Correctness</b>
<i>High (10-12)</i>	12 (57.14%)	0	Correct
<i>Middle (4-9)</i>	9 (42.86%)	0	Correct
<i>Low (0-3)</i>	Word does not appear in this range.		

However, the describing of code requires two parts: the manipulation and the object being manipulated. Correctly describing the situation requires ordering these parts in a way an audience can understand; the ability to do this is developed as an understanding of domain-

specific grammar. This evolving understanding is evident in Table 33 below, where the use of the word 'array' is tabulated:

*Table 33 - Meaning Based Word Testing: Occurrence of 'array' in Answers for Q32*

<b>Mark Range</b>	<b>Correct Count</b>	<b>Incorrect Count</b>	<b>Total Count</b>
<i>Low (0-3)</i>	3 (1.56%)	22 (35.48%)	25 (9.84%)
<i>Middle (4-9)</i>	86 (44.79%)	39 (62.90%)	125 (49.21%)
<i>High (10-12)</i>	103 (53.65%)	1 (1.61%)	104 (40.94%)
<b>Total</b>	<b>192</b>	<b>62</b>	<b>254</b>

Unlike with the variable 'z', a reference to the word 'array' is perfectly acceptable when answering the question. The object being manipulated must be described in some manner, the table above shows that this ability to use words in the correct order increases throughout the mark ranges.

Examining this question does seem to support what was observed in the metric tests, but further research needs to be done. The above was found after a quick analysis of the words in the answers to question 32, further analysis of the word meaning needs to be undertaken with these answers. To determine if the results which are seen above are repeatable, and not unique to this question, a meaning based word test will need to be conducted with other questions.

A complete listing of repeated words in the answers to Question 32 is listed in section 6.5 of the Appendix Chapter.

## Chapter 6. Appendix

### 6.1. Rules for Data Input

Table 34 below outlines the rules applied to the text of the student's EiPE answers while transcribing their words from the scanned digital PDF format into the excel spreadsheet:

Table 34 - Data Input Rules

<b>DI Number</b>	<b>Data Input Rule</b>
DI-01	When directly referring to a variable, the variable name will be enclosed by single quotation marks, e.g. variable ' <b>a</b> '.
DI-02	When a spelling mistake is made, the correct spelling will be inputted.
DI-03	When a student enters a new line, if possible (due to limitations of the size allowed in an excel cell), the text input will reflect this.
DI-04	When a question has not been attempted, the text: <b>NA#</b> will be inputted.
DI-06	When a word is too muddled, or cannot be deciphered from surrounding words, the text will be replaced with <b>&lt;illegible&gt;</b> .
DI-07	When possible, arrows will be depicted by: <b>-&gt;</b> .

Note: the text in the table above, words in **bold** denotes the text that is inputted into the excel spreadsheet.

6.2. The Program: Class Diagram

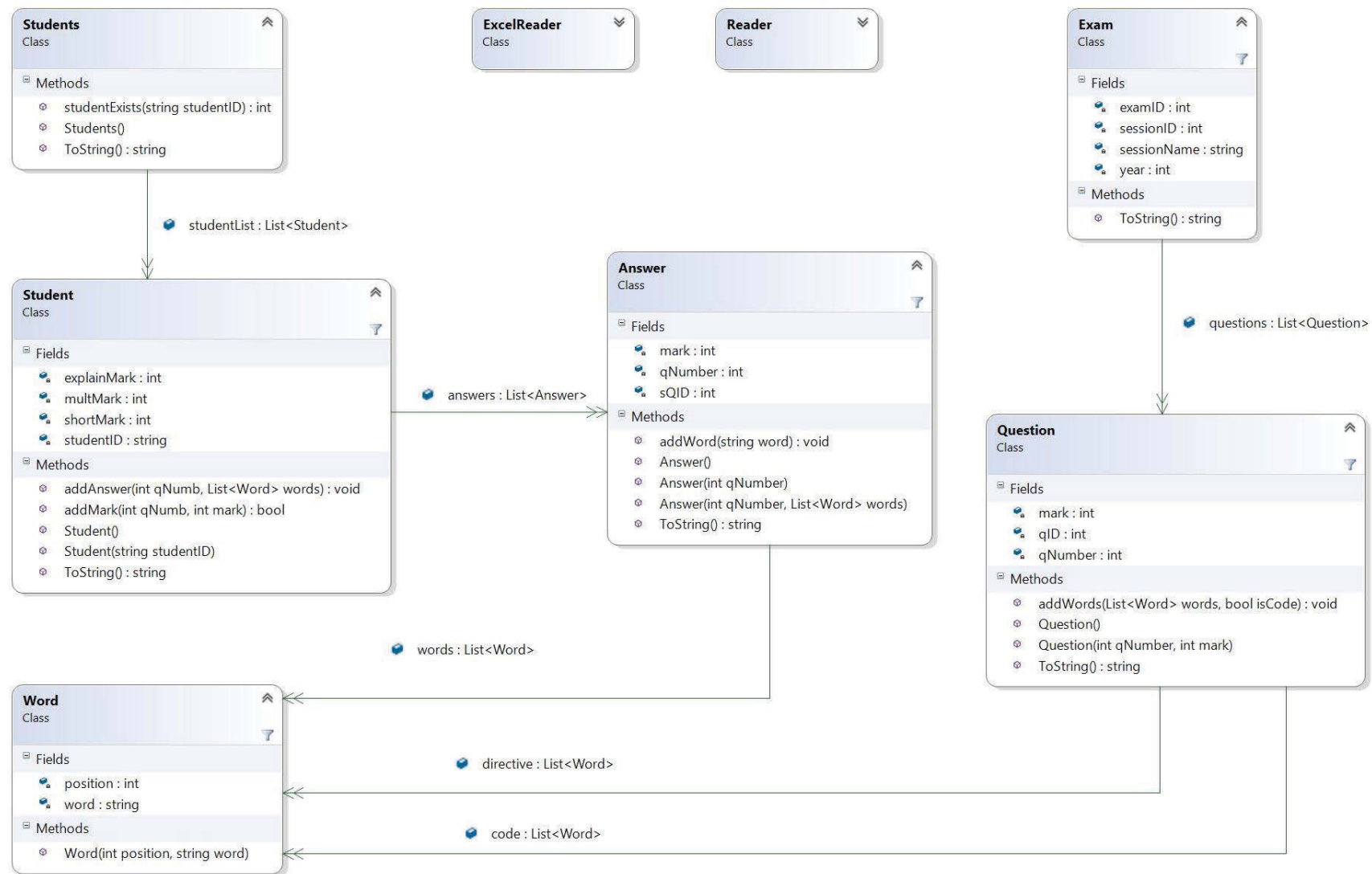


Figure 29 - The Class Diagram of the Program

### 6.3. The Database

The following database diagram and data dictionary will provide both a description of the structure of the database and a description of the data stored within it.

#### 6.3.1. Database Diagram

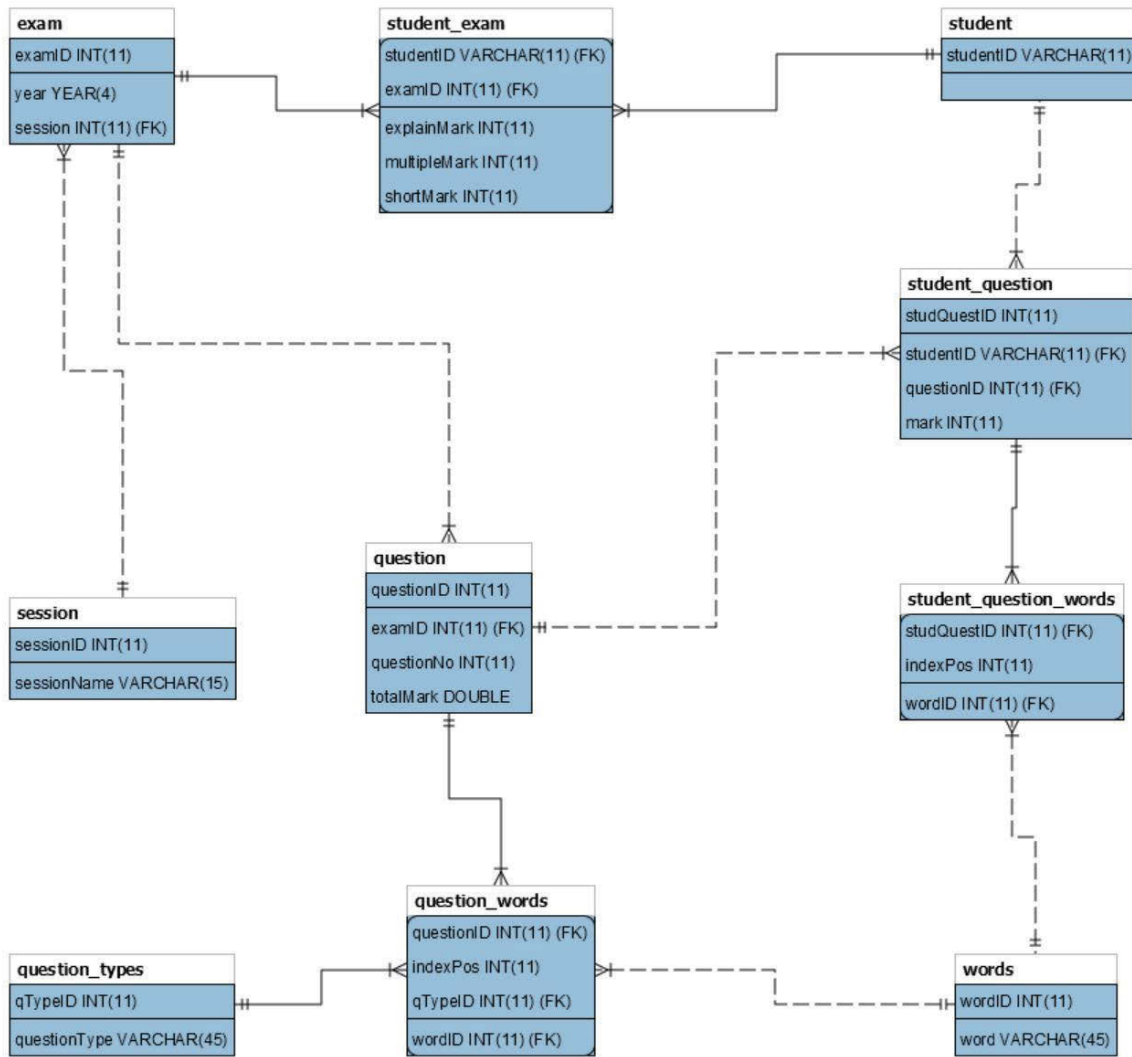


Figure 30 - Database Diagram

## 6.3.2. Data Dictionary

Table 35 - Data Dictionary: 'exam' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>examID</i>	INT	11	Primary Key	The unique ID of the exam, auto-generated.	1
<i>year</i>	YEAR	4	Not Null	The year the exam took place.	2015
<i>session</i>	INT	11	Foreign Key	The session the exam took place in, linked with 'sessionID' in the session table.	1

Table 36 - Data Dictionary: 'session' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>sessionID</i>	INT	11	Primary Key	The ID of the session, auto-generated.	1
<i>sessionName</i>	VARCHAR	15	Not Null	The teaching period the exam took place in.	Autumn

Table 37 - Data Dictionary: 'student' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>studentID</i>	VARCHAR	11	Primary Key	A unique string of characters that identifies the student. Will be numerical.	99145861

Table 38 - Data Dictionary: 'student\_exam' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>studentID</i>	VARCHAR	11	Primary Key, Foreign Key	The ID of the student who sat an exam.	99145861
<i>examID</i>	INT	11	Primary Key, Foreign Key	The ID of the exam which the student took.	1
<i>explainMark</i>	INT	11	Not Null	The mark this student has received for the EiPE section of this exam.	10
<i>multipleMark</i>	INT	11	Not Null	The mark this student has received for the MCQ section of this exam.	5
<i>shortMark</i>	INT	11	Not Null	The mark this student has received for the short code writing section of this exam.	8

Table 39 - Data Dictionary: 'question' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>questionID</i>	INT	11	Primary Key	Auto-generated, the unique ID given to a question to uniquely identify it.	1
<i>examID</i>	INT	11	Foreign Key	The ID of the exam which this question belongs to.	1
<i>questionNo</i>	INT	11	Not Null	The number of this question within the exam.	5
<i>totalMark</i>	DOUBLE	N/A	Not Null	The mark that this question is worth. The type is double in case of questions with point values.	3



Table 40 - Data Dictionary: 'student\_question' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>studQuestID</i>	INT	11	Primary Key	Auto-generated, the unique ID given to a student's answer of the question; regardless if attempted.	1
<i>studentID</i>	VARCHAR	11	Foreign Key	The ID of the student of which this answer belongs to.	1
<i>questionID</i>	INT	11	Foreign Key	The ID of the question that is being answered.	1
<i>mark</i>	INT	11	Not Null	The mark that this student has received for their answer.	3

Table 41 - Data Dictionary: 'words' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>wordID</i>	INT	11	Primary Key	Auto-generated, a unique ID assigned to a word appearing in a question or answer.	2
<i>word</i>	VARCHAR	45	Not Null	Could be a string of characters or an escape character.	Array

Table 42 - Data Dictionary: 'question\_words' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>questionID</i>	INT	11	Primary Key, Foreign Key	The ID of which the word in the question belongs to.	1
<i>indexPos</i>	INT	11	Primary Key	The position of the word within the text of the question.	2
<i>qTypeID</i>	INT	11	Primary Key, Foreign Key	Where the word is positioned in the structure of the question.	1
<i>wordID</i>	INT	11	Foreign Key	The ID of the word that is part of the question.	2

Table 43 - Data Dictionary: 'question\_types' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>qTypeID</i>	INT	11	Primary Key	Autogenerated, the unique ID that points to the specific part of the question.	2
<i>questionType</i>	VARCHAR	45	Primary Key	A label that describes the part of a question structure.	Code

Table 44 - Data Dictionary: 'student\_question\_words' Table

<b>Field Name</b>	<b>Data Type</b>	<b>Length</b>	<b>Constraint</b>	<b>Description</b>	<b>Example</b>
<i>studQuestID</i>	INT	11	Primary Key, Foreign Key	The joint foreign ID, identifying both the student and the question the word belongs to.	1
<i>indexPos</i>	INT	11	Primary Key	The position of the word within the text of the student's answer.	4
<i>wordID</i>	INT	11	Foreign Key	The word that is used in the answer text.	2

6.4. Indirect Reference: Distinct Words Test

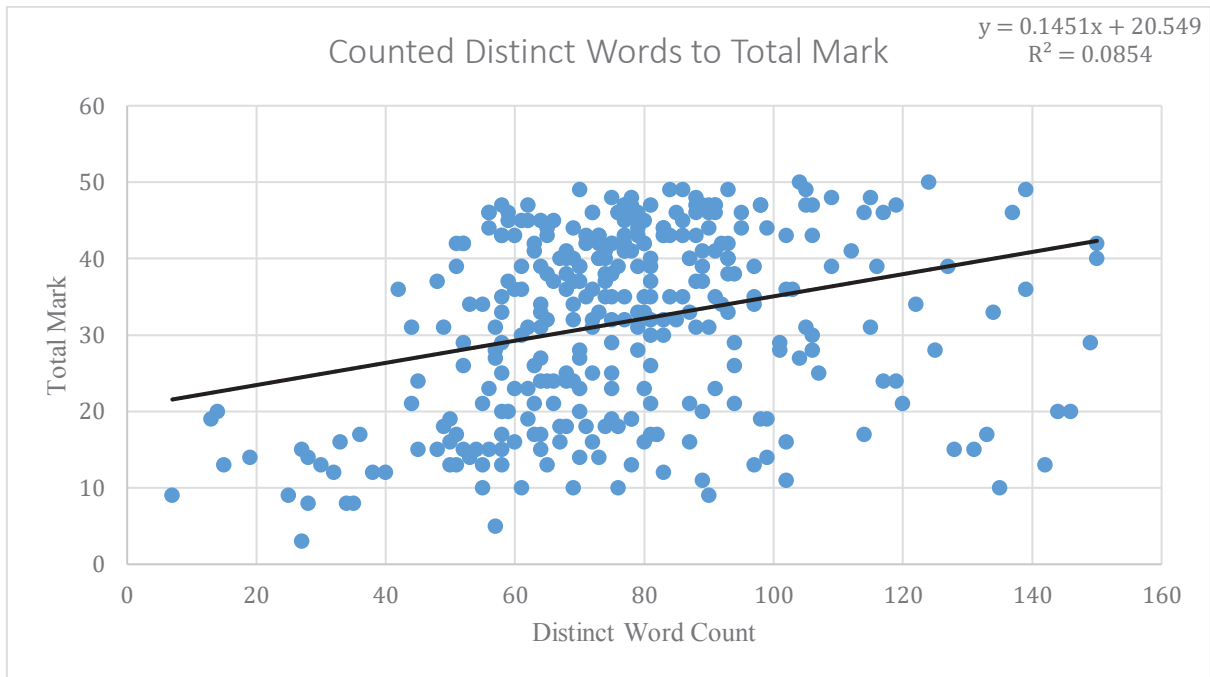


Figure 31 - Indirect Reference: Counted Distinct Words to Total Mark

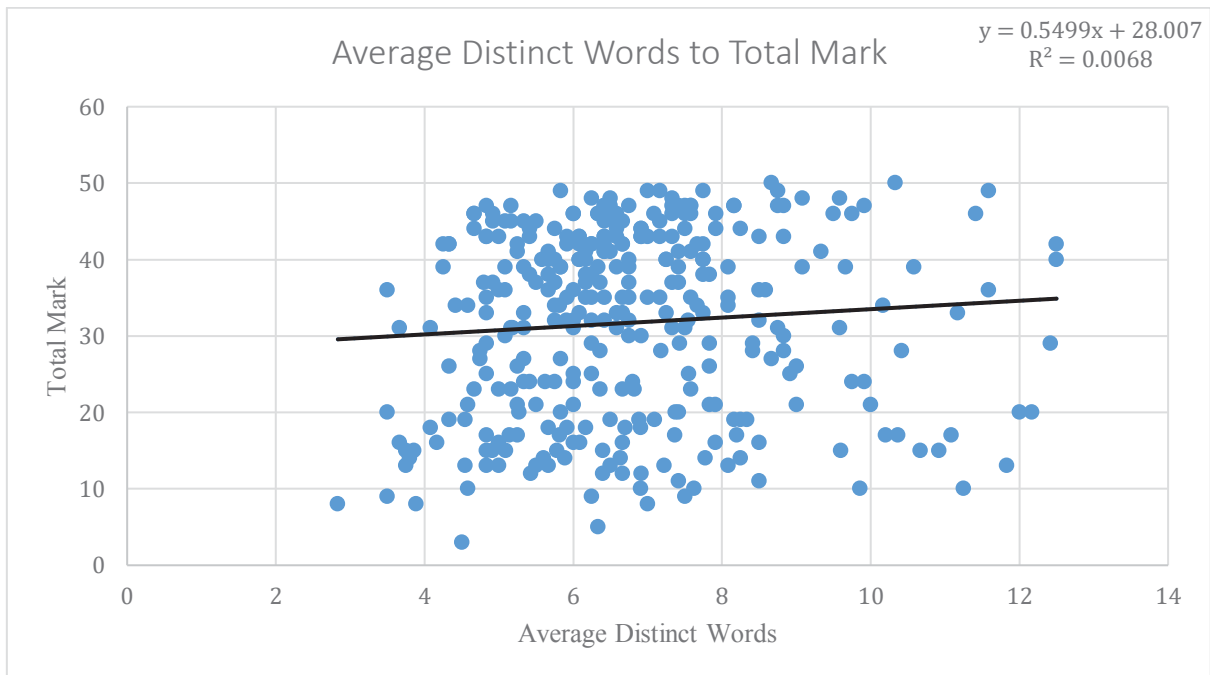


Figure 32 - Indirect Reference: Average Distinct Words to Total Mark

*Table 45 - Indirect Reference: Summary Statistics of Distinct Word Tests*

<i><b>Distinct Word Test</b></i>	<b>R<sup>2</sup></b>	<b>Pearson's r</b>
<i>Counted Distinct Words to Total Mark</i>	0.0854	0.2923
<i>Average Distinct Words to Total Mark</i>	0.0068	0.0823

## 6.5. Repeated Words in Answers to Q32

Table 46 - Repeated Words for Q32: 0-3 Mark Range

<i>Word</i>	<b>Times Word Used in a Correct Answer</b>	<b>Times Word Used in Incorrect Answer</b>	<b>Times Word Used in Correct Answer Overall</b>	<b>Times Word Used in Incorrect Answer Overall</b>	<i>Word States</i>
<i>ARRAY</i>	3	22	192	62	<i>In All</i>
<i>PRINT</i>	2	51	152	82	<i>In All</i>
<i>ADDED</i>	2	14	22	27	<i>In All</i>
<i>THEN</i>	1	11	14	12	<i>In All</i>
<i>'Z'</i>	1	83	46	113	<i>In All</i>
<i>STORE</i>	1	5	25	5	<i>In All</i>
<i>LOOP</i>	1	17	6	22	<i>Also in Middle</i>
<i>VARIABLE</i>	1	20	35	33	<i>In All</i>
<i>VALUE</i>	1	67	109	104	<i>In All</i>
<b>Students in Range:</b>		<b>91</b>	<b>Students Correct:</b>		<b>2</b>

Table 47 - Repeated Words for Q32: 4-9 Mark Range

<i>Word</i>	<b>Times Word Used in a Correct Answer</b>	<b>Times Word Used in Incorrect Answer</b>	<b>Times Word Used in Correct Answer Overall</b>	<b>Times Word Used in Incorrect Answer Overall</b>	<i>Word States</i>
<i>ARRAY</i>	86	39	192	62	<i>In All</i>
<i>PRINT</i>	59	30	152	82	<i>In All</i>
<i>VALUE</i>	56	36	109	104	<i>In All</i>
<i>'X'</i>	43	14	117	51	<i>Also in 10-12</i>
<i>'Z'</i>	28	30	46	113	<i>In All</i>

<i>VARIABLE</i>	18	13	35	33	<i>In All</i>
<i>ADDED</i>	15	13	22	27	<i>In All</i>
<i>ELEMENT</i>	13	6	49	7	<i>Also in High</i>
<i>CODE</i>	11	11	23	25	<i>Also in High</i>
<i>TOTAL</i>	9	0	21	0	<i>Also in High</i>
<i>STORE</i>	8	0	25	5	<i>In All</i>
<i>SUM'S</i>	8	1	22	1	<i>Also in High</i>
<i>THEN</i>	7	1	14	12	<i>In All</i>
<i>THIS</i>	6	5	17	14	<i>Also in High</i>
<i>INTEGER</i>	6	3	13	6	<i>Also in High</i>
<i>NUMBER</i>	6	2	8	2	<i>Unique</i>
<i>WILL</i>	6	7	13	19	<i>Also in High</i>
<i>'X[]'</i>	5	1	11	1	<i>Also in High</i>
<i>EACH</i>	5	5	7	12	<i>Unique</i>
<i>LOOP</i>	4	5	6	22	<i>Also in Low</i>
<i>UNTIL</i>	3	5	4	13	<i>Unique</i>
<i>'I'</i>	1	5	1	46	<i>Unique</i>
<i>LENGTH</i>	1	6	1	18	<i>Unique</i>
<i>FINAL</i>	1	5	3	6	<i>Unique</i>
<b>Students in Range:</b>		<b>129</b>	<b>Students Correct:</b>		<b>88</b>

Table 48 - Repeated Words for Q32: 10-12 Mark Range

<b>Word</b>	<b>Times Word Used in a Correct Answer</b>	<b>Times Word Used in Incorrect Answer</b>	<b>Times Word Used in Correct Answer Overall</b>	<b>Times Word Used in Incorrect Answer Overall</b>	<b>In Ranges?</b>
<i>ARRAY</i>	103	1	192	62	<i>In All</i>
<i>PRINT</i>	91	1	152	82	<i>In All</i>
<i>'X'</i>	74	2	117	51	<i>Also in Middle</i>
<i>VALUE</i>	52	1	109	104	<i>In All</i>
<i>ELEMENT</i>	35	0	49	7	<i>Also in Middle</i>
<i>'Z'</i>	17	0	46	113	<i>In All</i>
<i>STORE</i>	16	0	25	5	<i>In All</i>
<i>VARIABLE</i>	16	0	35	33	<i>In All</i>
<i>SUM'S</i>	14	0	22	1	<i>Also in Middle</i>
<i>TOTAL</i>	12	0	21	0	<i>Also in Middle</i>
<i>CODE</i>	12	0	23	25	<i>Also in Middle</i>
<i>THIS</i>	11	0	17	14	<i>Also in Middle</i>
<i>INTEGER</i>	7	0	13	6	<i>Also in Middle</i>
<i>WILL</i>	7	0	13	19	<i>Also in Middle</i>
<i>THEN</i>	6	0	14	12	<i>In All</i>
<i>FIND</i>	6	0	7	3	<i>Unique</i>
<i>'X[]'</i>	6	0	11	1	<i>Also in Middle</i>
<i>CALCULATE</i>	5	0	8	3	<i>Unique</i>
<i>ADDED</i>	5	0	22	27	<i>In All</i>
<b>Students in Range:</b>		114	<b>Students Correct:</b>		112

## Reference List

- Anderson, L.W. & Sosniak, L.A. (eds) 1994, *Bloom's Taxonomy: A Forty-year Retrospective*, Book, The National Society for the Study of Education, Chicago, Illinois.
- Biggs, J.B. & Collis, K.F. 1982, *Evaluating the quality of learning: The SOLO taxonomy (Structure of the Observed Learning Outcome)*, Academic Press, New York, NY.
- Corney, M., Fitzgerald, S., Hanks, B., Lister, R., McCauley, R. & Murphy, L. 2014, 'Explain in Plain English' Questions Revisited: Data Structures Problems', paper presented to the *Proceedings of the 45th ACM technical symposium on Computer science education*, Atlanta, Georgia, USA.
- Corney, M., Teague, D., Ahadi, A. & Lister, R. 2012, 'Some empirical results for neo-Piagetian reasoning in novice programmers and the relationship to code explanation questions', paper presented to the *Proceedings of the Fourteenth Australasian Computing Education Conference - Volume 123*, Melbourne, Australia.
- Du Boulay, B. 1989, 'Some Difficulties of Learning to Program', in E. Soloway & J.C. Spohrer (eds), *Studying the Novice Programmer*, Lawrence Erlbaum, New Jersey, pp. 283-300.
- Fix, V., Wiedenbeck, S. & Scholtz, J. 1993, 'Mental Representations of Programs by Novices and Experts', paper presented to the *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, Amsterdam, The Netherlands.
- Johnson, W.L., Soloway, E., Cutler, B. & Draper, S. 1983, *Bug catalogue: I*, Yale University, Department of Computer Science.
- Kramer, J. 2007, 'Is abstraction the key to computing?', *Commun. ACM*, vol. 50, no. 4, pp. 36-42.
- Lister, R. 2011, 'Concrete and Other Neo-Piagetian Forms of Reasoning in the Novice Programmer', *Conferences in Research and Practice in Information Technology Series*, vol. 114, pp. 9-18.
- Lister, R. 2012, 'Rare research: why is research uncommon in the computing education universe?', *ACM Inroads*, vol. 3, no. 4, pp. 16-7.
- Lister, R., Adams, E.S., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Mostr, J.E., Sanders, K., Sepp, O., Simon, B. & Thomas, L. 2004, 'A Multi-National Study of Reading and Tracing Skills in Novice Programmers', *SIGCSE Bull.*, vol. 36, no. 4, pp. 119-50.
- Lister, R., Fidge, C. & Teague, D. 2009, 'Further evidence of a relationship between explaining, tracing and writing skills in introductory programming', *SIGCSE Bull.*, vol. 41, no. 3, pp. 161-5.
- Lister, R., Simon, B., Thompson, E., Whalley, J.L. & Prasad, C. 2006, 'Not Seeing the Forest for the Trees: Novice Programmers and the SOLO Taxonomy', *SIGCSE Bull.*, vol. 38, no. 3, pp. 118-22.
- Lopez, M., Whalley, J., Robbins, P. & Lister, R. 2008, 'Relationships between reading, tracing and writing skills in introductory programming', paper presented to the *Proceedings of the Fourth international Workshop on Computing Education Research*, Sydney, Australia.
- McCracken, M., Almstrum, V., Diaz, D., Guzdial, M., Hagan, D., Kolikant, Y.B.-D., Laxer, C., Thomas, L., Utting, I. & Wilusz, T. 2001, 'A multi-national, multi-institutional study of assessment of programming skills of first-year CS students', *SIGCSE Bull.*, vol. 33, no. 4, pp. 125-80.



- Murphy, L., Fitzgerald, S., Lister, R. & McCauley, R. 2012, 'Ability to 'explain in plain english' linked to proficiency in computer-based programming', paper presented to the *Proceedings of the ninth annual international conference on International computing education research*, Auckland, New Zealand.
- Murphy, L., McCauley, R. & Fitzgerald, S. 2012, 'Explain in Plain English' Questions: Implications for Teaching', paper presented to the *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, Raleigh, North Carolina, USA.
- Pea, R.D. 1986, 'Language-Independent Conceptual "Bugs" in Novice Programming', *Journal of Educational Computing Research*, vol. 2, no. 1, pp. 25-36.
- Perkins, D.N. & Martin, F. 1986, 'Fragile knowledge and neglected strategies in novice programmers', in E. Soloway & S. Iyengar (eds), *Empirical Studies of Programmers*, Ablex Publishing Corporation, Norwood, New Jersey.
- Schulte, C., Clear, T., Taherkhani, A., Busjahn, T. & Paterson, J.H. 2010, 'An introduction to program comprehension for computer science educators', paper presented to the *Proceedings of the 2010 ITICSE working group reports*, Ankara, Turkey.
- Sheard, J., Carbone, A., Lister, R., Simon, B., Thompson, E. & Whalley, J.L. 2008, 'Going SOLO to assess novice programmers', paper presented to the *Proceedings of the 13th annual conference on Innovation and technology in computer science education*, Madrid, Spain.
- Simon, Lopez, M., Sutton, K. & Clear, T. 2009, 'Surely we must learn to read before we learn to write!', paper presented to the *Proceedings of the Eleventh Australasian Conference on Computing Education - Volume 95*, Wellington, New Zealand.
- Simon & Snowdon, S. 2011, 'Explaining program code: giving students the answer helps - but only just', paper presented to the *Proceedings of the seventh international workshop on Computing education research*, Providence, Rhode Island, USA.
- Soloway, E., Ehrlich, K., Bonar, J. & Greenspan, J. 1982, 'What Do Novices Know About Programming?', in A. Badre & B. Shneiderman (eds), *Directions in Human/Computer Interaction*, Ablex Publishing Corporation, Norwood, New Jersey, pp. 27-54.
- Spohrer, J.C. & Soloway, E. 1989, 'Novice Mistakes: Are The Folk Wisdoms Correct?', in J.C. Spohrer & E. Soloway (eds), *Studying The Novice Programmer*, Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 401-16.
- Sudol-DeLyser, L.A. 2015, 'Expression of Abstraction: Self Explanation in Code Production', paper presented to the *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, Kansas City, Missouri, USA.
- Teague, D. & Lister, R. 2014, 'Longitudinal Think Aloud Study of a Novice Programmer', paper presented to the *Proceedings of the Sixteenth Australasian Computing Education Conference - Volume 148*, Auckland, New Zealand.
- Thompson, E. 2004, 'Does the sum of the parts equal the whole', *Proceedings of the seventeenth annual conference of the National Advisory Committee on Computing Qualifications*, National Advisory Committee on Computing Qualifications, pp. 440-5.

- Venables, A., Tan, G. & Lister, R. 2009, 'A closer look at tracing, explaining and code writing skills in the novice programmer', paper presented to the *Proceedings of the fifth international workshop on Computing education research workshop*, Berkeley, CA, USA.
- Whalley, J.L., Lister, R., Thompson, E., Clear, T., Robbins, P., Kumar, P. & Prasad, C. 2006, 'An Australasian study of reading and comprehension skills in novice programmers, using the bloom and SOLO taxonomies', *Proceedings of the 8th Australasian Conference on Computing Education-Volume 52*, Australian Computer Society, Inc., pp. 243-52.
- Wiedenbeck, S. & Ramalingam, V. 1999, 'Novice comprehension of small programs written in the procedural and object-oriented styles', *Int. J. Hum.-Comput. Stud.*, vol. 51, no. 1, pp. 71-87.
- Willingham, D.T. 2009, *Why don't students like school?: A cognitive scientist answers questions about how the mind works and what it means for the classroom*, Jossey-Bass, San Francisco, CA.
- Winslow, L.E. 1996, 'Programming Pedagogy -- A Psychological Overview', *SIGCSE Bull.*, vol. 28, no. 3, pp. 17-22.