***Writing & Rhetoric in Computer Programming***

John Bui

*Written in Megan Lambert’s Spring 2020 ENC 1102*

# **1. Introduction**

**T**he role of programming is fundamentally important in modern society due to its role in the operation of technology. From keeping websites such as Facebook and Twitter in working condition, to operating systems found on phones, and the development of artificial intelligence, programming plays a crucial role in the devices around us. In many cases, programming allows computers to do what humans simply cannot do. Such as adding thousands, perhaps millions or billions, of numbers together into a singular number. Or virtually managing a worldwide stock market 24 hours 7 days a week and even allow for machine learning systems to automatically and independently learn and improve themselves. As a result of such powerful applications of programming, many researchers have particularly found great inspiration in examining how programing is conducted, how it can convey meaning, and how it can get things done.

## **1.1 Analysis of Previous Research**

Specifically, research was conducted on how programming methods could be integrated in academic curriculum (Hsu, Ting-Chia, et al.). It was found that computational thinking, a method widely used in computer science programming, was extensively becoming adopted in curriculums that were in fields of study other than computer science and in places around the world such as Australia (Hsu, Ting-Chia, et al. Pg. 297). Other researchers had a similar discovery. Namely, a research was undertaken to find whether the use of technology programmed for simulations were viable and effective as a means of attaining knowledge of science and engineering concepts (Falloon, Garry). The data gathered as consequence of the study indicated that the programmed simulations were effective for supporting students' knowledge (Falloon, Garry, pg. 149). The role of programming is noted by researchers as both beneficial and advantageous and therefore well known among academic communities.

Furthermore, as many computer scientists can attest, programming serves as a basis and means for society to progress into modern, lifesaving, innovative, and extraordinary technology. However, as our society advances and develops into a technology driven time period, the demand for the change in pace of knowledge accrual is apparent (Mora, Higinio, et al. pg. 141). One way to learn essential and critical knowledge for the real world for important concepts such as programming is through collaboration, as researched by Mora, Higinio, and others. Research was done to examine the role in collaboration motivated by the drive to learn computer science programming. It was discovered that when students conducted assessments on one another, as examinations progressed, the students made better comments, that is, not just superficial comments, but structural criticism supported by academic documents (Mora, Higinio, et al. pg. 147). This shows a positive correlation in the use of communication and collaboration when learning difficult subjects such as programming. And apart from the noticeable improvement due to the collaboration and communication, there was also a perceptible enhancement in the success rate in which students answered questions correctly. This success rate can be attributed by the triggering of thinking processes fostered by students when inquired by questions in teaching scenarios that demanded critical thinking (Shilo, Gila, and Noa Ragonis. pg. 335). It was ascertained by the research of Shilo, Gila, and Noa Ragonis that questions promoting critical thinking or high-order cognition often encourages growth in the overall thinking and writing process of the individual, which, in part, is observed by another researcher named Perl Sondra.

***Namely, to program means to deliberately write in a programming language with a purpose.***

Sondra investigated that while there have been studies on the average or skilled writer, there has not been an examination of unskilled writers and their writing process. A major finding of her study is that all of the students she studied displayed consistent writing processes and utilized the common elements of said writing processes, contrary to popular belief that unskilled writers did not have a writing procedure (Perl, Sondra pg. 758). Her study is important in our evaluation of the programming process due to the fact that all writers have a procedure in which they utilize to get things done, regardless of skill level or knowledge. It is also notable that her study was not conducted in the field of computer science, or programming for that matter, but yet, we can draw similarities in the wisdom learned in the data that she gathered. This is because the use of literacy, rhetoric, and genre conventions are not strictly contained into one field of study or area of interest. Rather, it is universal and can be found in programming especially. More precisely, in programming, computer scientists use specific genre conventions to communicate to machine and potentially other human beings. Without the rules, statements, and format that the programming language recognizes, meaning cannot be transmitted or received without errors or misinterpretation. With similar ideals as Sondra Perl, a researcher by the name of Ivanic expressed that writing is a form of social action in human discourse and that written text is deliberate, potentially permanent, and used as a statement for many social purposes (Marinkovich, Juana, et al. pg. 196). This particular description seems to be the case for programming as writing code is deliberate and often used as a statement for many purposes including social interactions and task driven procedures.

Indeed, programming is an intentional, purposeful, and precise method of discourse between machines and humans. Namely, to program means to deliberately write in a programming language with a purpose. This purpose can vary extremely depending on what the rhetor wants to accomplish given that programming itself has many applications and can be used to achieve a countless number of things. Including, but not limited to, the operating systems that smartphones function on, the systems that autonomous self-driving cars use to maintain safety and adhere to traffic rules, and virtual reality. Therefore, it is expected that many researchers would conduct their studies on how programming works on a computer level in addition to investigating how it can get things done by directing technology to do certain tasks.

## **1.2 Deficiency of Literary Analysis in Programming**

There is definite evidence to suggest that there has been substantial amount of work and research done in the field of computer science. In spite of this, lesser is known about the human aspect of coding and how computer scientists use communication and writing to utilize programming languages such as C, C++, python, and others to their fullest potential. Certainly, programming has been an integral objective that has brought together a myriad of collaboration, group work, and even academic curriculums that are not inherently based on computer science in numerous researches (Mora, Higinio, et al; Falloon Garry; Hsu, Ting-Chia, et al; Shilo, Gila, and Noa Ragonis). Nevertheless, they do not focus on the impact that writing has on programming, as well as the effects of writing in the development of programs utilizing C, C++, Python, and the hundreds of other programing languages that are used within the world of coding. Point of fact, previous research has not expressly considered the genre conventions, rhetorical situations, and other elements of writing in programming languages.

***However, how can writing be used in programming to develop these meaningful things?***

Though research has been done on the writing process in general (Perl Sondra) in addition to the comprehension of concepts as a result of computer science orientated curriculums (Mora, Higinio, et al; Marinkovich, Juana, et al), we do not know the full extent that rhetoric, writing, and literacy all correlate with modern technology and programming. Indeed, remarkable programs are written to develop powerful things such as supercomputers and smartphones. However, how can writing be used in programming to develop these meaningful things? Specifically, how can writing & rhetoric impact programming in such a way that meaning is produced? Such a longstanding, yet unfamiliar, issue must be addressed and solved in order to further develop a better understanding of writing and rhetoric in programming as a whole.

## **1.3 Solution to the Gap**

The principal objective of my research is to discover the correlation of writing that can be found in programming with respect to the writing elements such as genre, rhetorical situation, invention, planning and revision, and other components of writing typically found in what we consider as “normal language”. Definitively, the research aims to identify a critical threshold concept of writing: writing helps people make meaning and get things done, but there are always constraints. This objective can be accomplished in a number of ways. Methodically, one can begin by examining the process of programming directly and drawing parallels or similarities between coding and other forms of writing such as essays and short responses. For instance, an individual could seek for paradigms of code. These instances of code can be found to be purposed for many things including games, arithmetic calculators, and countless other things. Despite the codes having many purposes, one could recognize that all have similar genre due to being written in programming languages which are also consequently similar to each other. The similarity in genre they inherent may include the syntax in which the programs are written, the format and indentation, structure, functions, and other aspects which are conveniently akin to the normal language that we speak and write in. Once similarities and patterns are formed amongst examples of programs, a conjecture can be composed that can potentially answer the research question and be supported by information gathered further into the research. Ultimately, the leading approach to researching programming languages and their correlation with writing is to directly compare them with the conventions that are used in typical human discourse such as writing and conversations. Namely, the primary items to identify in programming with respect to English writing conventions are rhetorical situation, genre, constraints, invention, and other elements of interests. Thereafter, there must be a close examination of how these conventions all contribute to produce meaning.

# **2. Methodology**

## **2.1 Data/Material used**

Several rhetorics were collected in order to analyze the extent of Writing & Rhetoric within programming. Namely, I chose to start with homework assignments that were given in weeks 2 – 5 from a course called Engineering Analysis and Computation, abbreviated as EGN 3211. These homework assignments are programs that students, like myself, must write code for on a website called [repl.it](https://repl.it/). The homework assignments that will undergo analysis in this research includes an arithmetic calculator, a restaurant tipping calculator, gross pay calculator, and a questionnaire. These programs are written in a programming language called C and is written by me. Thus, we will be primarily examining the programming language C and how Writing and Rhetoric influences code written in it from the genre conventions to the rhetorical situations. We will investigate the inputs (written text) of the program, as well as the output (outcome produced) of the program as a result of these conventions. Additionally, another programming language will be examined in a much lesser scope. That is to say Arduino. Material from this programming language will also be from myself and will include programs that result in prompts (computer generated text). This alternate analysis will help in seeing the differences and similarities between languages when we compare both the C language and Arduino.

## **2.2 Procedures of Analysis**

The principle objective for every individual rhetoric in observation is to discover the writing conventions behind them and understand line by line what is going on from both an operational and Writing & Rhetoric standpoint. Specifically, one must identify the environment of the materials as well as the rhetorical situation, genre conventions, and all other elements found in the writing. After the identification step, translation will soon follow. Translation involves correlating or comparing the writing elements found in the rhetoric, that is to say the code, to the outcome that has resulted due to the program. In other words, we will observe what line of text or code produces what outcome. After translation will be comparison. Namely, the rhetoric that is studied will be compared to other or previous material observed to see whether there is a pattern emerging. This pattern will serve as the link between all the materials that are being examined. In short, the procedure of analysis will have 3 steps: identification, translation, and comparison.

# **3. Main Findings**

## **3.1 Overview**

In an effort to be as organized as possible, I have separated the research into 4 different sections: introduction, analysis, results, and discussion. The first section, that is to say the introduction, will provide background information that is fundamental in understanding the philosophy of programming languages such as what the rhetor should expect when coding and their limitations. The information provided will prove to be necessary as I will build onto these fundamentals. Moreover, the second section, analysis, will observe 4 programs along with the environments they are in using the methods from 2.2 Procedures of Analysis; identification, translation, and comparison. This section will investigate the elements of Writing & Rhetoric in programming. The third section or the results section entails the correlation of what I have found by associating the writing conventions discovered with the outcomes of each material. Specifically, I will delve into what writing conventions were present, an explanation of why these writing conventions are necessary, and how they all combine to create something meaningful. The fourth and last step will be the discussion, where I will consider all my research and what enigmas are still present.

### **Introduction**

1. Provide objectives of programing

2. Identify limitations of computers

3. Identify overall scope of programming

4. Present research purpose

### **Analysis**

1. The environment of C

2. Material 1: Less than, greater than, or equal to

3. Material 2: Addition calculator

4. The environment of Arduino

6. Material 3: Arduino Serial.print();

7. Material 4: Arduino addition

### **Results**

1. Compilation, Correlation, Discovery

### **Discussion**

1. Discussion of results

2. Discrepancies of results

3. Conclusion

# **4. Introduction to Computer Programming**

The rhetorical function of this section is to present and identify certain key aspects of programming in order to provide fundamental information used in my research. Specifically, this section will delve into the objectives of programming (1), the flaws of computers (2), the overall scope of programming (3), and present the purpose of my research (4).

## **4.1 Provide objectives of programming**

Programming languages are developed by humans in order to communicate to countless different devices such as laptops, smartphones, machine systems, desktop computers, and others of the like. As such, there are hundreds of programming languages around the world solely purposed to provide discourse between humans and technology. Some of these programming languages may serve different purposes. For instance, a programming language that is hardware orientated can direct a specialized computer to move a certain item to a certain position at a certain time. A programming language that is virtually orientated can produce video games, websites, applications, and numerous other virtual things. Some programming languages even overlap in purpose and can work with both hardware and virtual applications. As a result, there are many specialized fields in programming since it has so many different uses.

However, in this research, I will closely examine the programming languages of Arduino and C. To be precise, although the programming language C is considered both hardware and software (virtual) orientated, the environment in which I will be using the programming language C enables it to be entirely virtual based. That is, the version of C I will be using can only develop virtual applications instead of controlling physical or real-life objects. On the other hand, Arduino, the second programming language I will use, is specialized entirely in hardware. This programming language is only used with particular hardware and nothing else. Namely, Arduino interacts with a piece of hardware called the Arduino Uno R3. In turn, the Arduino Uno R3, as a result of the interaction between itself and the programming language Arduino, powers and controls robots, LED lights, circuits, and other types of systems.

## **4.2 Identify limitations of computers**

There are substantial numbers of capabilities when programmers use programming languages to interact with computers. For one, there is computing power. A single computer can calculate about 2 million procedures per second. Secondly, a computer can store more information than humans can. A computer, for example, that can store 1 gigabyte (1 GB) means it can store about 1 billion bytes, or an average of 200 and a half million words. With that being said, although computers have a considerable amount of storage and can process information at lightning speeds, they lack a crucial aspect that humans have. That is, the ability to be aware of itself, or self-consciousness. Computers are limited to the instructions that are written to them.

## **4.3 Identify overall scope of programming**

Due to the limitations of metacognition and self-consciousness, the scope of programming is limited. There are some situations where the need for computer programming arises and other situations where it is ineffective. For instance, programming would be used in a task where millions of calculations are required. In contrast, programming is not typically used when observing humor or poetry. Ultimately, as a result of the limitations that programming has, the rhetor will need to be mindful of such obstacles. By way of example, not having self-consciousness or metacognition is the reason why most random number generator programs do not truly produce random numbers. If one uses a computer to analyze the random numbers produced, a sequence or pattern can sometimes be derived. Simply put, a computer itself cannot think of a random number on a whim. Rather, it follows an algorithm that attempts to produce a random number. Whether this algorithm is successful depends on the code written by the rhetor(s).

## **4.4 Present research purpose**

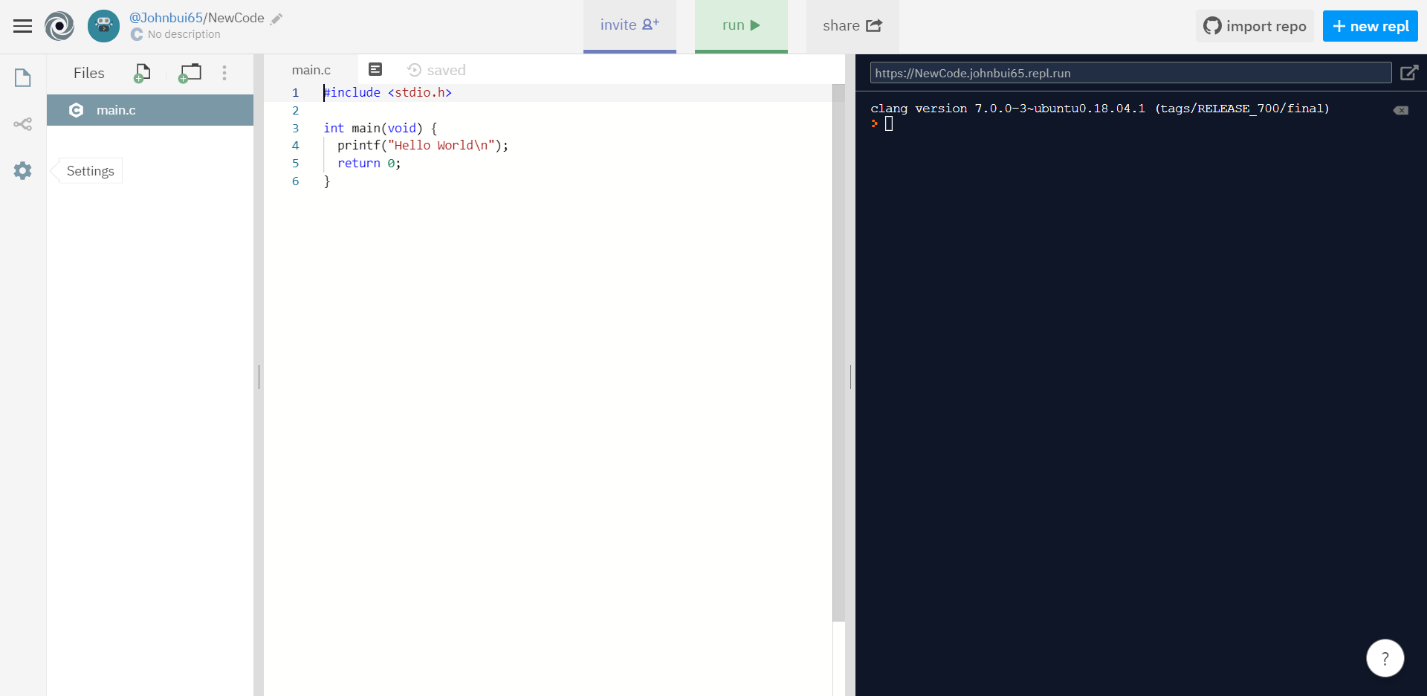
The principle objective of my research is to answer the question of how writing & rhetoric impacts programming in such a way that meaning is produced. And in doing so, correlate programming with a vital threshold concept of writing. That is to say writing helps people make meaning and get things done, but there are always constraints.

# **5. Analysis**

The rhetorical function that the Analysis section seeks to accomplish is the analytical exploration of the 5 different materials gathered along with their environments in order to investigate the effects that writing & rhetoric has on computer programs. This is accomplished by utilizing the 3 steps noted in 2.2 Procedures of Analysis. Namely, identification, translation, and comparison. Each material will have their own section and will be individually analyzed.

## **5.1 The environment of C**

The environment that hosts the materials in observation is a website called [repl.it](https://repl.it/) (see fig. 1).



**Figure 1**: Depiction of website [repl.it](https://repl.it/)

From figure 1, we can take a glimpse at 2 main sections: The white colored section containing the code and the black colored section to the right of it that will contain the outputs or results of the code if the program is ran. Additionally, there are buttons on the bar above the 2 sections that read from left to right, “Invite, Run, Share, import repo, new repl”.

// For this one, you need to include stuff from environment, stuff from hello world program

// Picture of error fig 2, hello world for fig 3

// For Arduino, stuff from environment, stuff from program.

### **5.1.1 Identification step**

* Genre – Every instance of a new program starts identically to the screen depicted in Figure 1. Namely, the rhetor or programmer will always start at the same screen with the same code.
* Contingency – All programs written in the coding section must be written in the programming language C. If something is written in the coding section that the website does not recognize, an error is given (see fig. 2).

### **5.1.2 Translation step**

* As a result of the same screen and code appearing when programmers (rhetors) choose to create a new program, a genre is formed
* Consequently, from the code produced by the genre, we see that in figure 3, when the run button on the top bar is clicked, the output section reads, “Hello Word”.