**A Comparative Analysis of Programming Paradigms: Java vs C**

Since their inception, computers have played an increasingly important role in today’s society.

Advancements in technology have enabled computers to become faster and cheaper. Today, the majority of families own a home computer that is vastly more powerful than giant mainframes of years gone by. Computer hardware has been evolving rapidly with no end in sight, and with all of the advancements in computer hardware come advancements in computer software. This essay is focused on providing technical insights into differences between C programming language and Java programming language.

The first thing about these two languages is that they belong to different paradigms. C is a procedural language whereas Java is an object-oriented programming language. What does this mean and which is better? That entirely depends on how do you wish to approach the problem at hand.

**Paradigmatic Differences:**

One of the primary distinctions between C and Java lies in their respective programming paradigms. C is considered a procedural language, while Java embraces the object-oriented programming paradigm. The choice between these paradigms hinges on the approach one wishes to undertake to address a particular problem.

**1.1. Object-Oriented Programming in Java:**

Java pulls down everything into objects;everything is modeled as an object with certain qualities and capabilities.Objects are reusable and can be altered and copied-meaning you do not have to continually reinvent things when you are designing the algorithm.You can interact these objects with each other depending on their identities-jobs they are first created to execute-.For example : A schoolbag will not have a salary but it will have a shape.

**1.2. Procedural Language in C:**

While Java pulls down everything into objects,C will put data through different processes (which are called functions -or procedures- ) while the program runs.A procedural program starts at the first line and executes lines one - by – one .Code may direct to the functions at the lines depending on the program,but when it returns from the function,it will run the next line following directions depending on data .

**Comparison:**

Neither of these programming paradigms can be inherently deemed superior to the other. The choice between C's procedural approach and Java's object-oriented paradigm largely relies on the nature of the problem at hand and, perhaps more crucially, the programmer's personal preferences and problem-solving methodologies.

The choice between these paradigms is subjective and depends on the programmer's problem-solving style and the specific nature of the tasks at hand. This understanding is essential for programmers to make informed choices when selecting the right programming language for their projects.

**LANGUAGE EVALUTION CRITERIA**

There are certain elements we must keep track on while we compare the two programming languages.

Which are ;

1. – Readibility
2. – Writability
3. – Reliability
4. – Cost

# READIBILITY

One of the most important criteria when evaluating programming languages is how easy it is to read and understand the code they produce. This plays a significant role in the maintainability of software, particularly since software development teams can change over time, while the core algorithms generally remain consistent. In this essay, we will explore the significance of readability in programming languages, focusing on C and Java, highlighting their differences regarding readability, simplicity, and related factors.

**The Impact of Readability on Maintenance:**

The ease of maintaining software is greatly affected by how readable the programming language is. As development teams evolve, new members must grasp existing algorithms. Simplifying code is essential since overly complex languages can hinder both understanding and modification. Finding the right balance between complexity and simplicity is key.

**Complexity and Simplicity:**

Striking a balance between complexity and simplicity is essential for readability. A language with numerous constructs can be intimidating to understand and modify. On the other hand, an overly simple language can pose challenges even when dealing with basic programs. Achieving this balance is an art that programming languages must perform.

**The 'Goto' Dilemma:**

An example of the impact of language features on readability can be seen in the presence of 'goto' statements in C. While C supports 'goto,' Java does not. 'Goto' allows program flow to jump to a specific line, potentially creating infinite loops and reducing code comprehensibility. The program would loop forever and it would not end.There are some backways to go around this problem for sure.

A programmer can find a way to work with this and make the program run smooth.But the next programmer will have a hard time to understand what is going on and it will affect the way he / she can alter the program.

**High-Level vs. Low-Level Languages:**

C is categorized as a low-level programming language, while Java is considered a high-level language. Low-level languages are closer to machine code, making them more challenging to read, but they operate more closely with the hardware. In contrast, high-level languages are more human-readable but abstract the underlying hardware complexities, making them easier to understand.

**Data Types, Orthogonality, and Syntax:**

Readability is also influenced by data types, orthogonality, and syntax. Orthogonality refers to how easily you can combine basic language elements to create data and control structures. Total orthogonality allows for every combination of these elements, reducing exceptions in language behavior. Well-defined data types, including the Boolean type, improve readability. Syntax plays a crucial role by affecting how language elements like keywords (e.g., while, class, for) appear.

**Conclusion**:

In summary, programming language readability is a crucial aspect of software development. It influences software maintenance and the ease with which algorithms can be understood and modified. The presence of language features, like 'goto' in C, can impact readability and should be carefully considered. Additionally, the distinction between high-level and low-level languages affects readability, with trade-offs between human-readability and proximity to hardware. Elements such as data types, orthogonality, and syntax also play roles in determining language readability. While C and Java have similarities in these aspects, discrepancies like C's use of 'goto' and the structure data types differentiation affect their readability to varying degrees.

#  WRITABILITY

Writability is a measure of how easily a language can be used to create programs for a chosen problem domain. Most of the language characteristics that affect readability also affect writability. This follows directly from the fact that the process of writing a program requires the programmer frequently to reread the part of the program that is already written.

# Orthogonality and simplicity

For orthogonality and simplicity, the same considerations made for readability have to be done. Too many constructs and too many ways of combining the constructs of a language, can lead to writing errors, ignored by both the programmer and the compiler. Furthermore, too much freedom of use, especially with complex constructs, can lead to difficulties in use. This often support poor elegantness and difficulties to modify implemented solutions.

## Expressivity

In writing codes,programmer must express what he intends to do ( or make it clear that it is easy to understand ).

For ex:

count++ ; && count = count+1;

Although there is a slight difference between the two, the right-hand side is clearly expressing what process must be done,while the left-hand side is a little bit more complex.

**Java vs C on writability**

Java is a little easier to write than C(Due to low-level , high-level differences).

Both share common writability features. For example, both allow abstraction through functions. One need only write a function once and then call it multiple times.

**Similiarities** :

Both allow expressivity by conveniently allowing multiple ways to do things. “The notation count ++ is more convenient and shorter than count = count +1.” .

Sometimes “for” loops are more convenient than “while” loops even though both can be written to do the same thing.

**Oppositions :**

Java does not have the confusing pointer syntax that C has. Also in C, one must allocate and deallocate memory with \_alloc() functions that really make no sense. C++ remedied this with the “new” and “delete” keywords, but Java is easier to write than either C or C++ due to garbage collection.

 RELIABILITY

A program is deemed reliable when it consistently performs according to its specifications under all conditions. To ensure reliability, several key aspects come into play, including type checking, exception handling, and aliasing.

**Type Checking:**

Type checking involves verifying the absence of type errors in a program, either during compilation or execution. It plays a crucial role in the overall reliability of a programming language. Detecting errors at compile-time is more desirable than at run-time, as it allows for early error identification, reducing computational costs and making necessary corrections less expensive.

**Exception** **Handling**:

Exception handling is another vital component for reliability. It empowers a program to intercept runtime errors and address unusual conditions, ensuring the program can continue functioning effectively. The ability to handle exceptions and address them appropriately is a clear advantage in enhancing program reliability.

**Aliasing**:

Aliasing, defined as having multiple distinct names in a program that can access the same memory cell, is a feature that needs careful consideration. While many programming languages allow some form of aliasing, it's generally acknowledged as a risky feature. For example, in C, having two pointers or references pointing to the same variable can lead to aliasing. This introduces the challenge of keeping track of which name is accessing the memory, making it a potential source of errors that can compromise reliability.

## Readability and Writability:

## Both readability and writability play a crucial role in determining a program's reliability. A programming language that lacks natural ways to express required algorithms may force the use of convoluted and unnatural approaches, which are more prone to errors. The ease of writing code directly influences its correctness. Readability affects reliability not only during code creation but also in the maintenance phase of the software's life cycle. Programs that are hard to read are challenging to write and modify, introducing the potential for errors.

## Java vs. C on Reliability:

## In the context of reliability, Java holds an advantage over C. Java offers robust type checking, identifying type-related issues before they result in logical errors. Java's exception handling is also comprehensive, offering a safety net for handling runtime errors. In contrast, C lacks built-in exception handling mechanisms, which can make it less reliable in the face of unexpected issues.

## Moreover, C's use of pointers provides direct memory access and opens the door to aliasing problems. In C, multiple pointers can access the same memory, introducing aliasing complexities that can compromise program reliability. Java, with its higher-level abstractions and stricter type system, mitigates some of the aliasing risks encountered in C.

## In conclusion, Java's strong type checking, robust exception handling, and reduced potential for aliasing make it a more reliable choice compared to C in these important aspects. This can have a significant impact on the reliability of software developed in these languages, making Java a preferred option for many developers when it comes to building reliable and maintainable software systems.

##  COST

The total cost associated with a programming language encompasses multiple factors, each of which plays a crucial role in determining the overall expenses involved. Let's dig into these factors and then compare Java and C concerning their costs:

**Instructing Programmers:**

The cost of guiding programmers to use a specific language depends on its simplicity and orthogonality.

A more powerful language often tends to be more complex, which can increase the effort required to instruct programmers.

**Writing and Development Environment:**

The cost of writing in a particular language and setting up the development environment is another significant aspect.

A good programming environment can substantially reduce the expenses associated with training programmers and writing programs.

**Compilation Costs:**

Compiling programs in the language incurs a cost, which can vary depending on the language and the quality of the compiler.

**Execution Efficiency:**

The design of a language significantly impacts the cost of executing programs written in that language.A language that necessitates numerous run-time type checks may impede fast code execution, irrespective of the compiler's quality.

**Language Implementation System:**

The cost of implementing the language itself and the tools associated with it is another element to consider.

**Reliability Costs:**

Poor reliability can be costly, particularly if software fails in a critical system. Ensuring reliability is essential to mitigate these costs.

**Maintenance Costs:**

Maintenance costs, encompassing both modification and correction, rely heavily on the readability of the code.

Maintenance is usually carried out by individuals who did not originally write the code.

**Availability:**

A programming language's availability is determined by the presence of appropriate tools and features.

Access to the right tools and features can expedite production, streamline the development process, and contribute to the sustainability of the code.

**Java vs. C on Costs:**

In the context of cost considerations, Java surpasses C in most aspects. Java offers simplicity, ease of writing, and greater reliability. However, C holds an advantage in terms of one particular cost factor:

Compilation and Execution Time: Java incurs a noticeable cost due to its relatively longer compilation, debugging, and execution times, primarily stemming from the use of the Java Virtual Machine (JVM).

Java boasts excellent type checking, garbage collection, and object-oriented design, which enhance the language's quality but come at the cost of execution time.

In contrast, C does not employ extensive index checking and is generally considered more efficient in terms of execution speed.

In summary, Java excels in terms of simplicity, reliability, and overall ease of use, making it a favorable choice for many software development scenarios. However, it's important to acknowledge that C holds an advantage in terms of execution speed. The choice between the two languages should consider the specific needs of the project and the trade-offs between development speed and execution efficiency.

**Comparing the Learning Curve Between C and Java:**

**Learning C:**

Learning C can be likened to acquiring knowledge of Latin in the programming world. C serves as the foundation for many other programming languages, which can make it easier to grasp and work with them later. It is used in various contexts, including scripting and low-level programming tasks. Learning C provides a deep understanding of the inner workings of computer systems.

**Advantages of Learning C:**

Strong foundational knowledge that applies to many programming languages.

Ideal for tasks requiring low-level system control and optimization.

Provides a deep understanding of computer memory and hardware.

**Challenges of Learning C:**

Can be more challenging for beginners due to its lower-level nature.

Involves manual memory management, which can lead to errors.

**Learning Java:**

Learning Java is generally more approachable, especially for beginners, as it is a higher-level language compared to C. While Java offers simplicity and readability, it might come at the cost of not fully understanding all the intricacies of a program. Java is widely used in application programming and for creating public-facing software, making it a valuable skill in the software development industry.

**Advantages of Learning Java:**

Beginner-friendly and easier to understand for newcomers to programming.

Extensively used in application development and web-based software.

Strong support for object-oriented programming concepts.

**Challenges of Learning Java:**

May not provide the same low-level system insights as C.

Learning purely in Java might not lead to a deep understanding of system-level programming.

**Syntax Differences :**

**C vs. Java Syntax:**

C and Java each have their own set of rules, or syntax, for writing code. In C, the syntax is known for being more compact and can sometimes be a bit tricky, especially when managing computer memory manually. In contrast, Java has a more straightforward syntax, making it easier for beginners to get started. It's like choosing between writing a compact but complex letter or a more detailed and clear one.

**Tokens in Syntax:**

Tokens in syntax can be thought of as the building blocks of code. In both C and Java, you'll use words like if, else, and int, along with punctuation marks like {, }, and ;. These tokens play specific roles in defining the structure and logic of your program. It's similar to using words and punctuation to craft a sentence or paragraph in a language.

**Exception Handling Syntax:**

In Java, handling errors, or exceptions, is well-structured and straightforward. The language provides a set of rules and keywords that make it clear how to handle different types of issues that can come up in your code. In C, the approach to dealing with errors is less organized. It often relies on manual checks and custom code for managing unexpected situations, which can be more error-prone.

**Type System and Syntax Clarity:**

Java has a strong type system, which means it is very particular about the kinds of data you can use and where you can use it. This strictness ensures that you catch mistakes early. In C, the type system is less strict, allowing for more flexibility but at the cost of potentially introducing confusion, as it does not provide as much guidance about data types.

**Semantic Differences:**

Variable Declaration:

In C, you need to declare variable types explicitly (e.g., int num), while in Java, you declare variables with types (e.g., int num) but also specify objects' types (e.g., String text = "Hello"). Java provides stronger type checking.

Example:

int num = 5;

num = 3.14; // No compilation error in C, but it truncates the value to 3

In java,it needs explicit type casting such as -> num = (int) 3.14;

**Memory Management:**

C requires manual memory management, where you need to allocate and deallocate memory. In Java, the language handles memory automatically (garbage collection), making it easier to avoid memory-related errors.

Conclusion:

The choice between learning C and Java depends on your specific goals and interests. Learning C equips you with a strong foundation and in-depth knowledge of system-level programming, which is valuable for certain tasks. On the other hand, learning Java provides an easier entry into programming and is highly relevant for application development and web-based software. Ultimately, both languages have their unique advantages, and the choice should align with your career aspirations and project requirements.

Source book : Concepts Of Programming Languages Eleventh Edition by Robert W. Sebesta

**Hüseyin Emre Sekanlı**

**200104004049**