Comparing C And JAVA

Programming language is a system used to write programs for the computer. Most programming languages are text-based formal languages, but some can also be graphical. They are kind of computer language. In this essay we will compare programming languages with two different philosophies. These languages are C and Java. The reason why they have two different philosophies is that while the C language is procedural, the Java language is an object-oriented language.

C is a general-purpose, procedural, high level programming language used in the development of computer software and applications, system programming, games, web development, and more. C language was developed by Dennis MacAlistair Ritchie at the Bell Telephone Laboratories in 1972. It is powerful and flexible language which was first developed for the programming of the UNIX operating system.

Java language was invented in 1991 by James Gosling of Sun Microsystems (later acquired by Oracle). Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers write once, run anywhere. This means that compiled java code can run on all platforms that support Java without the need to recompile.

Procedural programming can be defined as a programming model which is derived from structured programming, based upon the concept of calling procedure. Procedures, also known as routines, subroutines or functions, simply consist of a series of computational steps to be carried out. During a program’s execution, any given procedure might be called at any point, including by other procedures or itself.

Object-oriented programming can be defined as a programming model which is based upon the concept of objects. Objects contain data in the form of attributes and code in the form of methods. In the object-oriented programming, computer programs are designed using the concept of objects that interact with the real world.

Keywords

Keywords are words that have a predefined meaning in a programming language. Keywords are reserved words that developers or programmers cannot use as the name of say a variable, constants, and functions in most programming languages. Java has a relatively large and complex set of keywords. C has a smaller set of keywords compared to Java and uses a more minimalist language structure. This is because Java is an object-oriented language. It contains many keywords brought by object-oriented.

Identifier

In computer programming languages, an identifier is a lexical token that names the language's entities. Some of the kinds of entities an identifier might denote include variables, data types, labels, modules. Java and C languages are a case-sensitive languages. Their identifiers must begin with letter or an underscore. Following the first character, identifiers can include letters, numbers or underscores. In C and Java, the names of identifiers should not conflict with the names of keywords. However, Java enforces more strict rules concerning the use of reserved keywords as identifiers, whereas C provides more flexibility in this regard. Both languages require careful and meaningful selection of identifiers to improve the readability and maintainability of the code, otherwise major problems may arise in the development of the code in the future.

Constants

In programming languages, constants are special variables or values that are considered immutable or fixed. Constants help make code more readable, secure, and easier to maintain. While Java language uses the final keyword for its constant property, C language uses the constant keyword. Java is an object-oriented language. That's why constants often need to be defined in classes. In this case, it is generally preferred to use static final keywords together instead of using only the final keyword.

Special Symbols

In the context of programming and computer science, special symbols refer to characters or symbols used to represent specific operations, concepts, or syntax in a programming language or other computer-related contexts. These symbols have an important place in writing code and defining the structure of programs. The usage purposes of many special symbols in Java and C languages are similar. It can given Parentheses, Square Brackets, Not and Equality, Semicolon as examples. The uses of these special symbols are common with each other for two languages. There are some differences, especially in the context of object-oriented Java and procedural C. For example, In Java, the dot symbol is used to access class or object members. In C, the dot is used to access members of structures. Also, Java provides symbols and language features specific to object-oriented programming. Additionally, C uses pointers and bitwise operations, which are not available in Java.

Operators

"Operators" in programming languages; They are symbols used to perform various operations on variables, constants and other expressions. They serve as building blocks for creating more complex operations and controlling the flow of a program. Operators such as basic arithmetic operators, relational operators, logical operators, assignment operators, increment and decrement operators are used in common areas in both languages. On the other hand, there are some differences in terms of operators between these two languages. For example, C is known for its low-level operations and memory management, while Java focuses on higher-level and object-oriented programming concepts. These differences can lead to differences in operator usage between the two languages.

Comments

In programming languages, a "comment" is a form of explanatory or documentation text that is added to the code to make it more understandable, clarify its purpose, and provide information for other programmers or yourself. Comments are completely ignored by the compiler or interpreter and do not affect the functionality of the program; they exist solely for human readers. Single-line comments and multi-line comments are the same in both C and Java languages. Java has a special type of comments known as Javadoc comments, which are used to create documents. Javadoc makes it easy for Java programmers to document and share their code. It is also a very common tool when creating documentation of Java libraries or APIs. Javadoc documentation makes the Java development process more efficient and collaborative. However, there is no such feature in the C language.

Strings

A datatype string is a datatype modeled on a structured sequence concept. Strings are a data form that is so effective and necessary that they are introduced in almost every computer programming language. They are accessible as primitive data types in a specific languages and synthetic varieties in others. The structure of so many high-level programming languages enables an occurrence of a string datatype to be interpreted by a string, typically referenced in certain manner. C does not have a built-in string data type. Instead, it uses arrays of characters (char) to represent strings. C-style strings are null-terminated character (“\0”) arrays, and string manipulation can be error-prone due to the absence of string-specific functions. C-style strings can be modified directly, leading to potential buffer overflows or memory corruption if not handled carefully. Java offers a dedicated String class to work with strings. Strings in Java are objects and have numerous built-in methods for string manipulation, making it more convenient and safer for handling text. Java strings are immutable, meaning that once created, their contents cannot be changed. This immutability ensures string integrity. Java has a string pool, and all directly assigned values are checked among the strings in this pool. If this string cannot be found in the pool, a new string is created in the pool. However, if we assign a value using the new keyword, the string that needs to be assigned is recorded directly in a memory block outside the pool.

Data Types

Data types are rules and constraints in programming languages that determine what kind of data variables can represent. Programming languages typically offer different data types, and these types define how data is stored, processed, and what operations can be performed on it. C is a statically typed language, meaning variable types must be declared explicitly. Java is also statically typed, requiring explicit type declarations for variables. C provides basic data types like int, float, and char, allowing for low-level memory control. Java offers a rich set of data types, including primitive types (e.g., int, float) and reference types for objects. Custom data structures and user-defined types (structs) are available. C does not have built-in support for object-oriented programming (OOP) features like classes and inheritance. Java is a fully object-oriented language, with classes and support for inheritance, encapsulation, and polymorphism.

Control Structures

Control structures are structures and expressions in programming languages used to organize and direct the flow of a program. Through these structures, programs can make decisions, perform repetitive tasks, and follow different paths based on specific conditions. If else statements, for while loops, switch case structures are very similar to each other in C and Java languages. In the other hand Java offers an enhanced for loop (also known as "for-each") for iterating through collections such as arrays and lists but there is no for-each loop in C language.

Exception Handling

Exception handling is a process where programs identify errors, handle them, and decide how the error should be managed without disrupting the program's normal flow. It helps make software more reliable and error tolerant. If we think about the languages that we compare Java is designed with a specific exception handling system. When exceptional events, called exceptions, occur, Java allows these exceptions to be handled without disrupting the normal flow of the program. In Java, when a method throws an exception, it must be caught at the location where the method is called, or it must be propagated up to higher-level methods. This ensures organized error handling. Java uses try-catch blocks to handle exceptions. You place code that may potentially throw an exception inside a try block and then use one or more catch blocks to handle the exception. This closely associates error handling code with the location where the error occurred and eliminates the need for complex error codes. C does not provide a built-in exception handling system. Instead, errors are often handled using return values or error codes. In C, when a function handles an error condition, it's typically up to the calling code to carefully check and manage errors. There is no standard way to handle errors, which can lead to error handling inconsistencies. Java's exception handling system provides a more powerful and consistent approach to error management. C, on the other hand, relies on lower-level and manual error handling. While Java offers greater safety and code readability, it may come with a steeper learning curve.

Pointers

A pointer is a type that points to the memory address of a piece of data in memory (for example, a variable). In other words, a pointer points to a specific location in the computer's memory. The pointer provides access to data stored in this location. C is known for allowing the use of pointers. C programmers can directly manipulate memory addresses using pointers. Pointers provide direct and efficient access to data in memory. This makes C well-suited for low-level tasks like system programming and hardware control. However, improper use of pointers can lead to issues like memory leaks, buffer overflows, and security vulnerabilities. In C, it's the programmer's responsibility to manage and use pointers correctly. Java does not allow direct use of pointers. Instead, it uses references. References do not point to memory addresses; they reference objects. Java provides automatic memory management. It handles memory allocation and deallocation on behalf of the programmer, reducing the occurrence of memory-related issues like memory leaks. Java restricts the use of pointers to prevent security vulnerabilities and to make code more reliable. It abstracts away low-level memory manipulation. C provides more control and speed but requires more responsibility. Java, on the other hand, provides higher security and easier code development, but offers a more abstract approach.

Memory Management

Memory management refers to the process by which a computer program allocates, uses, and releases memory (RAM - Random Access Memory) during its execution. Memory management is a critical component to ensure the efficient and reliable operation of a program. C language places the responsibility of memory management entirely on the programmer. This means that memory allocation and deallocation processes are manual. In C, memory addresses can be directly accessed using pointers. This allows for direct control over memory management and memory pointers. Memory issues like memory leaks and buffer overflows are common in C due to careless memory management. Programmers must handle memory management with great care. Java provides automatic memory management. Programmers are relieved from the burden of memory allocation and deallocation. The Java Virtual Machine (JVM) automatically handles memory allocation and collects unused objects through a process known as garbage collection. Java does not use pointers. Instead, it uses references that point to objects, abstracting away direct memory access. Java's automatic memory management significantly reduces memory-related issues, such as memory leaks and buffer overflows. It also enhances security by preventing certain types of vulnerabilities.

Compilation

Compilation is the process in programming where written source code is transformed from a human-readable form into a format that a computer can understand and execute. The first stage involves using a special software called a "compiler" to translate the written source code. The second stage is the actual running of the compiled code. The compiled program or application is executed by the operating system or a runtime environment. It is understood and run by the computer's processor.

Java first compiles the source code into an intermediate representation called "bytecode." Bytecode is platform independent. Bytecode is then interpreted by the Java Virtual Machine (JVM) or translated into native machine code through Just-In-Time (JIT) compilation during runtime. This means that the code goes through a kind of two-step compilation process at runtime. Java source code is transformed into bytecode files with a .class extension. Java applications are platform-independent, but the execution speed can sometimes be slower compared to C. C directly compiles source code into native machine code. Source code is transformed into machine code specific to the operating system and hardware using a compiler. C compilation results in an executable file that can be run directly by the operating system. C code is more likely to be platform-dependent, and compiling the same code on different operating systems may require adjustments.

Availability

Java is designed as a platform-independent language, which means that Java applications can be run on different operating systems. This provides a broader range of availability for software developed in Java. Java is an open-source language with a large and active community. This community support is beneficial for Java developers in finding solutions to issues and accessing resources. Java applications are commonly used in a wide range of domains, including web applications, large enterprise systems, mobile applications, and more. This demonstrates the broad availability of Java for various applications.

When codes written in C are compiled and run on another device, they may not work the same way as on the device on which they were written, so it may be necessary to make adjust. C is well-known for its use in developing system-level software, such as operating systems and drivers. This makes C highly available in the field of system software development. C has a rich ecosystem of libraries and resources. These libraries extend the availability of C and accelerate software development processes. C applications can be found in various domains, from embedded systems to operating systems and games, indicating the wide availability of C for different applications.

Learning Curve

The learning curve may vary depending on the complexity of the language, the availability of learning resources and community support, and the experience of the learner. C is a low-level programming language that operates closer to computer hardware. Learning C may require delving deeper into computer science and hardware concepts. The learning curve for C can initially be steep, especially for beginners. C includes complex topics like explicit memory management and pointers.

Java is considered a higher-level language and is platform independent. As a result, Java is often seen as a simpler language to start with. The learning curve for Java is generally lower, making it easier for beginners to get started with coding. Java's automatic memory management reduces the risk of memory leaks and erroneous pointer usage, making the learning process smoother.

In summary, the learning curve for C can be steeper due to its lower-level nature and complex concepts, but it can provide a solid foundation for understanding programming. Java, on the other hand, has a gentler learning curve and is often preferred by beginners, making it an accessible choice.

REFERENCES

Concepts of Programming Languages Eleventh Edition GLOBAL Edition by Robert W. Sebesta

https://www.geeksforgeeks.org/tokens-in-c/

https://www.geeksforgeeks.org/java-tokens/

https://www.youtube.com/watch?v=G1ubVOl9IBw

https://www.geeksforgeeks.org/difference-between-java-and-c-language/

http://cs.williams.edu/~jannen/teaching/f16/cs135/lectures/lecture.31.pdf

https://www.youtube.com/watch?v=Bj9Mx_Lx3q4&t=292s

https://www.javatpoint.com/what-is-string-in-programming