RUBY AND C COMPARISON

There are thousands of programming languages used today. The purpose of each programming language and the platforms on which it is used vary. We apply some criteria in order to classify and evaluate these languages. According to these criteria, we will compare the procedural language C with the object-oriented Ruby.

A programming language is an artificial language that can use the features of a computer. It is defined like human languages, using syntactic and semantic evaluations to determine structure and meaning. It is used to organize information, exchange it, express algorithms in full and facilitate relevant communication. Its general purpose is to give a set of instructions to the computer. For this reason, a great degree of precision and integrity is required. With this feature, it is distinguished from human languages. When communicating with people, a natural language is used. However, writings and speeches can be ambiguous and even mistakes can be made. Still, people can expect this to be understood. Computers, on the other hand, do exactly as they are told. The code that the programmer wants to write is unimportant, he literally has to write it. The language definition and the inputs to the program must specify exactly the behavior that will occur when the program is executed. There are 4 types of programming languages: procedural, object-oriented, logical and functional. First of all, we will look at the history of the C and Ruby and then compare the procedural language C with the object-oriented language Ruby with syntax and semantics criteria.

The C language was written at Bell Laboratory in the early 1970s. According to Dennis Ritchie of Bell Laboratory, the C language was developed as a system application language for Unix, the operating system that came out in the early 1970s. At the very beginning, Bell Laboratory employee Ken Thompson began to design a new programming language for the new UNIX platform. Thompson wrote the B language, replacing the BCPL system language. However, utilities could not be written in the B language because it was slow and could not take advantage of the PDP-11 features in the operating system. According to Toptal, UNIX operating systems and most Linux operating systems are also written in C. In addition, databases such as Oracle Database, MySQL, MS SQL Server and PostgreSQL are partially written in C language. It forms the basis of many system kernels.

Other programming languages, such as Python and Perl, use compilers or interpreters written in C. C has changed over the years and is still used in lower-level programs such as kernel. But it is also used for many applications ranging from device drivers to other programming languages' compilers or interpreters. The language also made way for C++, Objective-C, C#, and many more C-based languages that each have their own speciality. Nowadays, it is mostly used in console application development, writing web services and mobile application development, windows application development and embedded systems that can be codable, graphics and games.

Ruby is often called a "language of careful balance.". It was written by Yukihiro Matsumoto in 1995. Matsumoto knew many programming languages such as Eiffel, Ada, Perl, Smalltalk and Lisp. He wanted to create a new language using the best features of these programming languages. This is how the Ruby language appeared. Ruby is a language that the developer has the freedom to change as he wants. Developer can add functions as extras to the basic language features and remove them if necessary. It is also a language that developer can decently switch between operating systems. The code on any operating system will work smoothly on Linux, Mac OSX and Windows. It will even run in UNIX, DOS, BeOS, OS/2, and more. Ruby is dynamically typed, rather than statically typed. The runtime does as much as possible at run-time.

Ruby and C have a solid mutual relationship. Ruby supports so-called "extension modules". These are modules that you can use from your Ruby programs and which, from the outside, will look and act just like any other Ruby module but which are written in C. In this way, you can compartmentalize the performance-critical parts of your Ruby software, and understand those down to pure C. And, of course, Ruby itself is written in C.

There are several similarities between Ruby and C. If you want Ruby language procedural, can be used like C language. However, in the background, it still retains the structure of being object-oriented. Most of the operators are the same (including the compound assignment and also bitwise operators). Although Ruby doesn't have ++ or --. The __FILE__ function, which finds the path to the current file, and the __LINE__ functions, which find the line in the file, are available in both

languages. They also have constants, though there's no special const keyword. Const-ness is enforced by a naming convention instead— names starting with a capital letter are for constants. Strings are shown in double quotes and also strings are mutable in both two languages. Just like manual pages, you can read most docs in your terminal window—though using the "ri" command. You've got the same sort of command-line debugger available.

There are a lot of differences between Ruby and the C language due to their different features. Unlike the C language, there is no need to compile code in the Ruby language. It is a direct executable language. Objects are strongly typed (and variable names themselves have no type at all). There's no macros or preprocessor. No casts. No pointers (nor pointer arithmetic). There are no typedefs, sizeof or enums that are in the C language. There is no header file in Ruby. You just define your functions (usually referred to as "methods") and classes in the main source code files. There's no "#define" in Ruby. Instead, only constants are used. All variables in Ruby live on the heap. Besides, you don't need to free them yourself. The garbage collector takes care of that. In Ruby arguments to methods (i.e., functions) are passed by value, where the values are always object references. It's required 'foo' instead of #include <foo> or #include "foo". You cannot drop down to assembly. In the C language, a semicolon is used at the end of each line. However, this is not necessary in Ruby. Likewise, there is no need to use parentheses for if and while conditional statements in the Ruby language. In contrast to the C language, parentheses for method (i.e., function) calls are usually optional in Ruby. In addition, in Ruby don't usually use braces. Just end multi-line constructs (like while loops) with an end keyword. The do keyword is for generally called for "blocks". There's no "do statement" like in C. The term "block" means something different. It's for a block of code that you associate with a method call so the method body can call out to the block while it executes. You need to declare variables in the C language. There is no such requirement in Ruby. You can assign new names whenever you need. In the C language, 0 and false denote wrong. In Ruby, this is false and nil. 0.0 or even 0 is considered correct in Ruby. There is no char in Ruby. The characters are represented as a 1-letter string. Also, strings don't end with a null byte. Array literals are denoted by braces in the C language, while in Ruby they are denoted by brackets. If you want to expand the size of an array in C, you need to do

dynamic allocation. In Ruby, the number of elements increases automatically. If you add two arrays, you get back a new and bigger array (of course, allocated on the heap) instead of doing pointer arithmetic. This is not possible in the C language. In Ruby, more often than not, everything is an expression (that is, things like while statements actually evaluate to an r-value).

The two languages have advantages and disadvantages due to these features. If we talk about the advantages of the C language; The C language provides a wide platform for the developer to perform all kinds of operations. It also contains many data types and operators for this. A code written in C can run independently on any machine without any changes or minor modifications. This shows that the C language is flexible. C is a language that has an important feature such as the ability to extend itself. The C language has its own function library. Thanks to this, the developer can easily use these functions. If the user wants, they can add their own functions to the C library. As a result, it makes the code easier. C is structure-based. It means that as a result of this feature, problems or complex problems are divided into smaller blocks or functions. This piecework construction helps in an easier testing and maintenance process. C is a middle-level programming language that means it supports high-level programming as well as low-level programming. It supports the use of kernels and drivers in low-level programming and also supports system software applications in the high-level programming language. The presence of algorithms and data structures in the C language has made program calculations easier and faster. The existence of algorithms and data structures makes the C language usable in complex calculations and operations such as MATLAB. C provides dynamic memory allocation that means you are free to allocate memory at run time. For example, if you don't know how much memory is required by objects in your program, you can still run a program in C and assign the memory at the same time. C follows a system-based programming system. It means the programming is done for the hardware devices. These advantages for Ruby can be listed as follows; Ruby has its own web application framework called "Ruby on Rails". It is open-source program. It allows the developer to modify, distribute and use. Because of the syntax design is simple, the code is easy to write, read, and understand. The general goal of the Ruby is made itself simple and fast for web applications. It works on many platforms such as Linux, Windows, Mac and iOS. There are

hundreds of different helpful community-created “gems” and libraries that developer can use as a part of its own software.

If we talk about the disadvantages of the C language; C is a very comprehensive language, but it does not support the concept of OOPs such as inheritance, polymorphism, encapsulation, abstraction and data hiding. The code does not detect problems and problems during writing. This requires a compiler. In this way, problems can be identified. It makes the checking of code very complex in large programs. Exception Handling is one of the most important features of programming languages. While compiling the code, various anomalies and bugs can occur. Exception Handling allows you to catch the error and take appropriate responses. However, C does not exhibit this important feature. C is a small and core machine language that has minimum data hiding and exclusive visibility that affects the security of this language. These disadvantages for Ruby can be listed as follows; Ruby code is difficult to debug and sometimes occurs an error during runtime. It has less information compared to other programs. It has lower flexibility and slow processing.

If we compare the popularity of the two languages over time, C language has long been among the 3 most used languages in the world. Ruby, on the other hand, is not very popular because it is relatively newer than other languages. However, it has a noticeable increase in the usage rate. Due to its similarity to the Python language, Ruby is also a preferred language for beginners in coding. From a reliability point of view, both languages can work exactly same on other platforms besides the written platform.

As a result, although the C and Ruby languages are different languages in terms of classes, they have many similar aspects. Ruby's ability to be a procedural language also contributes to this situation. The C language is used in jobs that require processing speed and maximum efficiency. Ruby, on the other hand, can be used in areas such as web systems and data processing that do not require speed and efficiency.

RESOURCES

- https://data-flair.training
- https://www.section.io/engineering-education/history-of-c-programming-language
- https://www.cs.mcgill.ca/~rwest/wikispeedia/wpcd/wp/p/Programming_language.htm
- https://www.codecademy.com/resources/blog/what-is-ruby-used-for