**THEORY OF PROGRAMMING LANGUAGES**

**A picture containing text, gear, metalware

Description automatically generated**

**Name: M. Hassan Sattar**

**Roll No: 22F-3773**

**Submitted to: Dr. Sajid Anwar**

**National University of Computer and Emerging Sciences**

**A survey of programming languages**

**Abstract**

For a professional programmer, learning multiple languages might be a turning point in their career. The study of new programming languages is important now a days. Computer researchers often develop new programming languages. In recent years, many new languages have been developed. Each language has its own benefits and drawbacks. **C++, PHP, C#, Java, Python, VB** are popular programming languages that many programmers are using in different set of domains. These languages are often used in **web2.0**. These languages are used from the criteria of Reusability, dependability, portability, availability of compilers and tools, readability, efficiency, familiarity, and expressiveness. And also, in **web3.0** domain blockchain technology also has play significant impact on a few areas, including finance, cloud computing, privacy, security, etc. **Blockchain** programmers have been using many programming languages like **Solidity, Pact, and Liquidity**. But the most widely used programming languages are solidity. Using these languages developers write their own smart contracts and agreements. that are used in **web3.0** domain.

**1. Introduction**

Learning multiple languages might be a turning point in their career. When a developer understands that programming principles of any language, the doors open to knowledge that truly makes a difference in quality and productivity. Each language has its own pros and cons. The couples of popular programming languages are **C++, PHP, C#, Java, Python, and VB**. Many developers use these languages comparing the criteria of Reusability, dependability, portability, availability of compilers and tools, readability, efficiency, familiarity, and expressiveness. Software developers and designers should be careful when making logical decisions and be aware of the advantages and disadvantages each language may bring to their software solution. **Blockchain technology** has played significant impact on a few areas, including finance, cloud computing, privacy, security, etc. **Smart contracts** are the meaning of the agreement between multiple parties that are directly written into lines of code. Smart contracts give trusted transactions and agreements between unidentified groups. Smart contracts are also facing dares. because it’s not fully mature.Blockchain programmers have been using many programming languages like **Solidity, Pact, and Liquidity**. But the most popular and widely used programming languages is **solidity.**

# 2. Compare of Programming Languages

# The couples of popular programming languages are C++, PHP, C#, Java, Python, VB. Many developers use these languages. Software developers and designers should be careful when making logical decisions and be aware of the advantages and disadvantages each language may bring to their software solution

**2.1 PHP**

PHP is a server-side programming language created for web development. The main aim of creating these scripts was to improve personal websites and boost its performance. Most of the PHP's language is derived from C, Java, and Perl. Native Boolean types are available in PHP, same as in Java and C++. Additionally, PHP shares keywords and syntax with mostly of high-level languages that use the C style. PHP is a flexible language. When compared to other scripting languages, it is viewed as the quickest. PHP is a weak language with dynamic typing. variables in PHP must always be specified before usage. For PHP applications, memory management is essential. PHP should use a garbage collection technique to release memory and remove circularly referenced variables to reduce memory use. PHP also support to eclipse IDE and combine HTML tags and PHP code in one file

**2.2 C++**

C++ is a high-level general-purpose programming language. C++ is a multi-paradigm language: procedural, object oriented, generic, and functional. C++ requires much more space than low-level languages, it is easier to both learn and use, and it is compatible with C code. C++ added several features likes: Virtual functions, Operator overloading, Function name, References, Constants, Improved type checking, One line comments support

C++ has good support for generic programming and also supports classes, structs, and unions and allocates them into the heap. C++ programs generally have great performance compared to other high-level languages like Java and C#.

There are different methods to improve memory management and their program performance in C++. There are four methods supported in C++. Static memory allocation is used to allocate a static variable a value at compile time. Another method used by C++ is automatic memory allocation. In this method, an automatic variable is declared within its class name. When this variable is assigned a value, it is stored in the allocated stack. The dynamic memory allocation is done through the keywords new and delete, that are quite like to C’s malloc and free keywords. One last technique used by C++ to manage its memory is garbage collection. Programmer could find memory leaks and double allocations

**2.3 VB**

Visual Basic was first invented by two mathematicians. The main goal in creating this new language was to make it simple to learn and utilize for college students. Visual Basic was originally an object-based programming language. Thus, it has the concepts of objects, classes, data abstraction, and strong type checking. Now a part of the.NET framework is Visual Basic. As a result, the developer can use VB.net to construct apps for Windows, the Web, and console. And with VB.NET, programmers can produce robust, high-quality apps.

Like JAVA and C#, Visual Basic contains a garbage collector that automatically manages the memory of the programs. Garbage collector runs automatically, searches for unwanted items, and frees up memory. Static type checking is used in Visual Basic. Visual Basic is a tightly typed language. During compilation, the type of variable is given more attention. For example, the object type of the calling function must match that of the called function.

**2.4 Java**

Java is a programming language with a strong emphasis on static typing, however it also allows dynamic typing for some OOPs principles like polymorphism. A large number of the constructs were impacted by their equivalents in the C and C++ computer languages. A great OOPs language is Java. It provides encapsulation with access modifiers like private, public, protected, etc., greatly enhancing the code's dependability and security. Additionally, Java offers procedural programming. Java, however, restricts the use of global variables and outside class methods. JVM executes Java. Because the JVM manages all the resources required by the programmed, it is also known as a managed language. Java is a good OOPs programming language since it includes most OOPs ideas like inheritance, polymorphism, and encapsulation and removes some complex ideas like multiple inheritance.

**2.5 Python**

Python is a dynamically typed programming language. The main benefit of dynamic typed language is the readability of the code and also a drawback of python as a dynamic programming language is that there would be no way to guarantee that a particular piece of code would run successfully for all the different data-types scenarios. Multiple paradigms are supported by Python, including Object-Oriented, Imperative, Functional, Procedural, and Reflective. Because of the flexibility it offers due to its dynamically typed nature, Python is frequently used as a scripting language in the development of small applications that are being developed with static-typed languages. Python's preference for rapid application development makes it suitable for prototyping. Python is used in the development of websites to a certain level

**2.6 C#**

Microsoft released the C# programming language and the.Net platform simultaneously, and the various C# versions were released concurrently with Microsoft. In addition to being an object-oriented programming language and drawing inspiration from the C-family of computer languages, C# also borrowed several aspects of functional programming languages, such as anonymous types and lambda expressions. C# was created with the intention of utilizing the best aspects of other programming languages like Java and C++ while avoiding their flaws. C# is a powerful programming language that increases programmers' productivity by streamlining the development process and lowering the probability that they will make the most common programming faults.

**3. Smart Contract Programming Languages on Blockchains**

A blockchain is basically a chain made up of sequential lists of data called blocks that are connected and safeguarded via encryption. Many peers on a network keep copies of this chain.Blockchain technology has played significant impact on a few areas, including finance, cloud computing, privacy, security, etc. Smart contracts is the meaning of the agreement between multiple parties that are directly written into lines of code. Smart contract programming languages talk about the popular smart contract languages like Solidity, Pact, and Liquidity.

**3.1 Solidity**

Solidity is a programming language designed for creating smart contracts which has syntax close to Ecma (JavaScript). Ethereum is a decentralized, open-source platform for creating smart contracts. Ethereum has its own IDE that is called Ethereum Virtual Machine (EVM). Solidity contracts run on EVM. A peer-to-peer network protocol is part of Ethereum. Several networked nodes are responsible for maintaining and updating the Ethereum blockchain database. while the EVM is running then the same set of instructions are carried out by every node on the network

**3.2 Pact**

Pact is a programming language that is used to create smart contracts on Kadena blockchain. Pact gives developers the ability to design transactional logic that is reliable, fast, and secure for mission-critical business operations. Pact is an immutable language and favors a declarative approach over complex control-flow. Pact smart contracts are designed to enforce business rules guarding the update of a system-of-record: complex, speculative application logic simply does not belong in this critical layer

**3.3 Liquidity**

Liquidity is a high-level typed smart-contract language on Tezos blockchain. It is a fully typed functional language that closely meets with the security requirements set forward by Michelson. It follows closely to Michelson security requirements and use the OCaml syntax.It is being developed with a formal verification mechanism to demonstrate the accuracy of smart contracts written in Liquidity. The Liquidity language has three essential characteristics: (1) Complete coverage of the Michelson language: Liquidity allows for the writing of anything that can be expressed in Michelson. (2) Local variables rather than stack manipulations: Local variables can hold values. high-level types: Liquidity programs can define and use kinds like record-types and sum-types.

**4. Related Work**

The first goal of experimental repetition is to repeat the analyses from the (FSE article) and to obtain the same results. A repeat should ideally follow a straightforward process in which a draught produces effects that correspond to the final result of the published piece. The only necessary section of code needed to generate the expected number of graphs and tables is that one. The data's results are contained in two CSV files, the first of which is larger and contains one row and one commit per file, as well as the marks for bug fixes. The second file contains less merged rows and uses a similar commit and wording.

Comparative analysis is done between the Ethereum, IBM Open Blockchain (Hyperledger project), Intel Sawtooth Lake, Block Stream Sidechain Elements, and Eris platforms. According to this analysis, in terms of scalability, development, documentation, and support, Ethereum should be the top platform of choice. The authors come to the conclusion that Ethereum is a superior choice for developers because it had no security flaws at the time. Although Solidity is the most approachable language for a new programmer to create smart contracts, it is also the language most vulnerable to flaws. However, Liquidity and Pact have lower usability but appear secure at the moment.

As compared to Liquidity and Pact, the RESULT revealed that Solidity had a significantly shorter average execution time for each contract than the other two. The data extracted from the investigation was carried out to be coherent about all three languages of all test subjects and test contracts, such as the first test subject are implemented, and the result of Solidity is faster than implemented the contracts of Pact or Liquidity. The subject who implements test contracts using liquidity does so more quickly than the subject who implements test contracts using the pact, according to a likely observation in the case of liquidity and pact.

The second goal of the examination of the FSE article is ANALYSIS. The analysis differs from repetition in that it suggests various statistical analyses and data processing to report the methodological errors of the original study. The first way of analysis is data processing, in which the activity of data gaining in the original task was more closely watched. Future was used in this step as quality control, not data change. Create the software that will be downloaded automatically, and GitHub history will be validated against commits from projects.

There are 618 downloads of the 729 projects that were utilized in the FSE study. The opposite was either deleted or made secret. By name, the projects available for download matched. Next, employ three steps of data cleaning. Deduplication, TypeScript removal, and C/C++ accounting come first, second, and third, respectively. (b) The second method, statistical modelling, allows for the methodological flaws in the original script's statistical analyses to be exposed. 1. Zero-sum Contrasts include the original document's selection of a set of programming languages with arbitrary disparities. Such comparisons allow for the understanding of the negative binomial coefficients. When compared to the normal log-expected number of databases commits that correct bug, regression shows variations in the number of languages commits that do so. (2) Multiplicity of Hypothesis Testing is a common error in data- obsessed software engineering is failed to justification for several hypothesis testing. When at once testing various hypotheses, certain p-values can decrease in the meaning range by random chance. (3) Statistical Significance versus Practical Significance is the FSE article that are focused on statistical significance of coefficients regression. This is quite limited, that the p-values mostly focused by the number of explanations in the dataset. When p-values are small not certainly suggest practically important relations. (4) Accounting for Uncertainty is the FSE analyses expected that the sums of bug-fixing commits have no error. Labeling of commits is focused to doubt the experimental used to able that commits have several false positives, which have been factored into the outcomes. A quite simple method to reach this relies on factor estimation by statistical procedure is called the bootstrap. OUTCOME Of the reanalysis unsuccessful to confirm most of the privileges of reference. The results of 11 of 7 languages were cancelled by the numerous steps of enhanced statistical modelling and data cleansing. Even if the recommendations are statistically significant, their practical value is minimal.

**Conclusion**

The FSE study's primary result was cancelled due to many and major methodological mistakes that were found. It might be difficult to create statistical analysis software using large-scale code sources. There are many opportunities for errors to surface. It is not true that eleven programming languages have statistically significant bug relationships. The general public may only come to believe in these findings and have a better understanding of the challenges and solutions associated with such investigations by concluding comprehensive re-confirmation of such studies.

The study that was done to connect the security and usability flaws of the particular languages was part of the future work of the supplied valuation. According to the findings of the trials, Solidity is the least protected language against invincibility even if it is a highly useful language for new creators to construct smart contracts. Alternately, Liquidity and Pact, which have poorer usefulness but appear secure right now. There are various data that can be covered to draw more reliable conclusions about the variations in smart-contract programming languages. These abilities include the ability to perform studies with (i) additional object courses that will include larger systems with wide-ranging context boundaries such as application size or domain; (ii) more spread-background of human subjects; (iii) new coming smart-contract programming language is being created.