Osmon

Programming Language

“When I was young and coding on good old C, I dreamt about having my own C to rule the world”.

– Yuri Katsuki [2010]

Details

**Course**: **Business Information Systems Project**

**Module Code**: **6BUIS007C-n**

**Student**: Sokhibjon Orzikulov *(as known as Yuri Katsuki)*

**Email**: (Educational: sorzikulov@students.wiut.uz) (Personal: [yuri@katsuki.moe](mailto:yuri@katsuki.moe))

**Hosted**: <https://github.com/osmon-lang/osmon>

**Supervisor**: Olga Yugay

**Precautions:** The report uses Batang font for its headings and if it’s showing unreadable boxes instead of normal characters, then you don’t have Batang installed on your PC, and it’s suggested to install the font. The font is being used as it’s closely associated with background of the project. I do apologize for any inconveniences.

If you’re using Windows Operating system, there shouldn’t be any problem with that. However, for other operating systems, you may download it here:

<https://www.wfonts.com/font/batang>

Table of Contents

[Acknowledgement 5](#_Toc131786531)

[Abstract 6](#_Toc131786532)

[Introduction 7](#_Toc131786533)

[Background 9](#_Toc131786534)

[Project Objectives 11](#_Toc131786535)

[Business Objectives 11](#_Toc131786536)

[Technical Objectives 11](#_Toc131786537)

[Academic Objectives 12](#_Toc131786538)

[Personal Objectives 12](#_Toc131786539)

[Project Objectives 13](#_Toc131786540)

[Literature Review 15](#_Toc131786541)

[Porter’s 5 Forces 17](#_Toc131786542)

[Porter’s 5 Forces - Competitive Rivalry 17](#_Toc131786543)

[Porter’s 5 Forces - Threat of New Entrants 17](#_Toc131786544)

[Porter’s 5 Forces - Power of Suppliers 18](#_Toc131786545)

[Porter’s 5 Forces - Power of Buyers 18](#_Toc131786546)

[Porter’s 5 Forces - Threat of Substitute Products 19](#_Toc131786547)

[SWOT Analysis 20](#_Toc131786548)

[SWOT Analysis Strengths 20](#_Toc131786549)

[SWOT Analysis Weaknesses 21](#_Toc131786550)

[SWOT Analysis Opportunities 22](#_Toc131786551)

[SWOT Analysis Threats 22](#_Toc131786552)

[SWOT Analysis Highlights 23](#_Toc131786553)

[PESTLE Analysis 24](#_Toc131786554)

[PESTLE - Political Factors 24](#_Toc131786555)

[PESTLE - Economical Factors 24](#_Toc131786556)

[PESTLE - Social Factors 25](#_Toc131786557)

[PESTLE - Technological Factors 25](#_Toc131786558)

[PESTLE - Legal Factors 26](#_Toc131786559)

[PESTLE - Environmental Factors 26](#_Toc131786560)

[PESTLE – Highlights 27](#_Toc131786561)

[Competitor Analysis 28](#_Toc131786562)

[Target Audience 33](#_Toc131786563)

[Monetization 35](#_Toc131786564)

[Programming Language 37](#_Toc131786565)

[Additional Tools 38](#_Toc131786566)

[Project Limitations 40](#_Toc131786567)

[Further Improvements 41](#_Toc131786568)

[Conclusion 43](#_Toc131786569)

[References 44](#_Toc131786570)

# Acknowledgement

This project is being supported by many people and special thanks goes to all GitHub Sponsor supporters who supported the project from early development stage.

Also, big thanks to UZINFOCOM that sponsored the project by giving servers and real use cases to test and see the potential uses of the project in real world.

Throughout development stage, the project has been supported by my supervisor, Olga Yugay who has deep understanding & experience on compiler infrastructures by giving technical feedbacks on the project that helped me to improve many codebases and resolving bugs and fixes of the project.

# Abstract

The Osmon programming language project was developed to address the language barrier faced by Uzbek speakers in learning and adopting mainstream programming languages. This report presents a comprehensive analysis of the project, highlighting its objectives, technical implementation, and potential impact on the IT sphere in Uzbekistan and beyond. The project successfully created a hybrid programming language that supports both compilation and interpretation, providing users with flexibility in their preferred mode of operation. With its integration into Uzinfocom's IT infrastructure, Osmon has demonstrated its potential for real-world applications and garnered significant interest within the local IT community. The report also discusses the various strategies employed to promote the adoption of Osmon in educational institutions, aiming to make programming more accessible for Uzbek students. Additionally, the report delves into the technical aspects of the project, including the choice of Rust as the primary programming language, the use of various libraries and frameworks, and the development methodology. Through an in-depth examination of the project's strengths, weaknesses, opportunities, and threats, the report evaluates its potential for growth and expansion, both within Uzbekistan and the global programming community. Overall, the Osmon programming language project represents an innovative approach to overcoming language barriers in the IT sphere, paving the way for a more inclusive and diverse programming landscape.

# Introduction

The Information Technology (IT) industry has experienced tremendous growth in recent years, with an increasing demand for skilled programmers worldwide. Programming languages play a crucial role in this growth, acting as the foundation for software development and enabling various applications across different sectors. However, a significant challenge faced by aspiring programmers, particularly in non-English speaking countries, is the language barrier. Most mainstream programming languages use English-based syntax and keywords, which can create difficulties for non-native speakers attempting to learn and adopt these languages.

In response to this challenge, the Osmon programming language project was initiated in 2022 by Yuri Katsuki. The primary goal of the project is to create a unique, user-friendly programming language tailored specifically for Uzbek speakers, with Uzbek words used for its reserved words. By addressing the language barrier, the Osmon project aims to make programming more accessible to a broader audience in Uzbekistan, thereby promoting the growth and development of the local IT sector.

The Osmon programming language possesses hybrid capabilities, supporting both compilation and interpretation depending on the user's preferences. Users can compile the source code into a native executable binary using the osmon compile command, or they can interpret the code using the osmon run command with the built-in Bulut Virtual Machine. Developed using Rust, the project is divided into three main components: the Osmon connecting point, the Bulut Virtual Machine, and the Havo compiler.

This report will provide a comprehensive analysis of the Osmon programming language project, exploring its background, objectives, and technical implementation. The report will also delve into the project's potential impact on the IT sphere in Uzbekistan and beyond. Topics covered in the report include the project's scope, literature review, industry analysis, and various analytical frameworks such as Porter's Forces, SWOT, and PESTLE analysis. Additionally, the report will examine the project's target audience, monetization strategies, and the technical aspects of the programming language, libraries, and frameworks used.

By providing an in-depth examination of the Osmon programming language project, this report aims to shed light on the innovative approach taken by the project's creators to overcome language barriers in the IT sphere. Furthermore, the report will explore the potential for the Osmon programming language to contribute to a more inclusive and diverse programming landscape, both within Uzbekistan and the global programming community.

# Background

The Osmon programming language project was conceived in response to the growing demand for IT professionals in Uzbekistan and the challenges faced by individuals with limited English proficiency in learning mainstream programming languages. The language barrier often creates a steep learning curve for non-native English speakers attempting to learn programming languages, as most popular languages are based on English syntax and keywords. This can result in a substantial portion of the population being excluded from participating in the rapidly expanding IT sector, ultimately hindering the country's technological advancement.

The idea for the Osmon project was sparked in 2022 by Yuri Katsuki, who sought to create a programming language that would cater specifically to the needs of Uzbek speakers. By using Uzbek words for reserved words, the language would be more accessible to native Uzbek speakers, making it easier for them to learn and adopt programming concepts. The name "Osmon" is derived from the Uzbek word for "sky," symbolizing the limitless potential of this innovative project.

The project was developed using Rust, a modern programming language known for its performance, safety, and concurrency features. Rust was chosen due to its suitability for creating systems-level software, such as virtual machines and compilers. The Osmon project is divided into three main components: the Osmon connecting point, the Bulut Virtual Machine, and the Havo compiler. Each component was developed using a variety of libraries and frameworks, which were carefully selected to optimize the functionality and performance of the programming language.

Since its inception, the Osmon programming language has garnered significant interest within the Uzbek IT community. The project's source code is hosted on GitHub and is freely accessible to the public, allowing for collaboration and contribution from a diverse range of developers. Moreover, the project has gained sponsorship from several Uzbek developers on GitHub Sponsor, demonstrating its potential for growth and impact on the local IT sector.

The Osmon programming language has also been successfully implemented in the IT infrastructure of Uzinfocom, a leading provider of state information systems in Uzbekistan. The integration of the Osmon programming language into Uzinfocom's infrastructure has not only validated its practical applications but has also helped to establish its credibility within the IT industry.

# Project Objectives

The Osmon programming language project was developed with several objectives in mind, each aimed at addressing different aspects of the challenges faced by Uzbek speakers in the IT sphere. These objectives can be broadly categorized into four groups: business objectives, technical objectives, academic objectives, and personal objectives.

## Business Objectives

The primary business objective of the Osmon programming language project is to create a programming language tailored specifically to the needs of Uzbek speakers. By providing a more accessible learning experience for this target audience, the project aims to foster the growth of the IT sector in Uzbekistan. As the local IT industry expands, the demand for skilled IT professionals will increase, creating new job opportunities and contributing to the country's economic development. Furthermore, the project seeks to establish partnerships with organizations such as Uzinfocom to integrate the Osmon programming language into real-world applications, thereby validating its potential for practical use and enhancing its credibility within the IT industry.

## Technical Objectives

The Osmon programming language project strives to develop a hybrid programming language that supports both compilation and interpretation, providing users with flexibility in their preferred mode of operation. This dual functionality allows users to choose between compiling their code into a native executable binary or interpreting it using the Bulut Virtual Machine, depending on their specific requirements. Additionally, the project aims to create a programming language that is compatible with existing IT infrastructure, ensuring that it can be easily integrated into various applications. To achieve these technical objectives, the project relies on the Rust programming language, along with a diverse range of libraries and frameworks carefully selected to optimize performance and functionality.

## Academic Objectives

The project seeks to promote the adoption of the Osmon programming language in educational institutions, integrating it into programming courses and curricula. By making programming more accessible to Uzbek students, the project aims to increase the number of skilled IT professionals in the country, ultimately contributing to the growth and development of the local IT sector. Furthermore, the project encourages collaboration and contribution from the academic community, providing opportunities for students and researchers to contribute to the ongoing development and refinement of the Osmon programming language.

## Personal Objectives

The Osmon programming language project presents a unique challenge for its creator, Yuri Katsuki, as well as other contributors involved in its development. By undertaking this innovative project, they have the opportunity to develop their skills in programming language design, virtual machine implementation, compiler construction, and other related fields. The project also provides a platform for networking and collaboration among developers, both within Uzbekistan and the global programming community, fostering a sense of camaraderie and shared purpose in overcoming the language barrier in the IT sphere.

# Project Objectives

The Osmon programming language project aims to develop a comprehensive, user-friendly programming language that caters specifically to Uzbek speakers. The project scope encompasses a wide range of elements, including technical development, educational integration, community engagement, and future enhancements.

In terms of technical development, the Osmon programming language has been designed as a hybrid language, supporting both compilation and interpretation, depending on the user's requirements. This dual functionality allows users to compile their code into a native executable binary or interpret it using the built-in Bulut Virtual Machine, providing versatility and adaptability for various applications. The project has successfully developed three main components: the Osmon connecting point, the Bulut Virtual Machine, and the Havo compiler. Each component is carefully designed and implemented using Rust, a modern programming language known for its performance, safety, and concurrency features.

To ensure that the Osmon programming language is accessible and user-friendly, the project has focused on creating extensive documentation and resources for learning the language. This includes user guides, tutorials, and example projects that demonstrate the language's capabilities and applications. Additionally, the project has made a concerted effort to engage with the programming community, both within Uzbekistan and internationally. This has involved hosting the project's source code on GitHub, which has allowed for collaboration and contribution from a diverse range of developers, as well as establishing an active community on Telegram, where developers can share their experiences, offer support, and provide feedback.

The educational integration of the Osmon programming language is another key aspect of the project scope. The project seeks to promote the adoption of Osmon in educational institutions, with the goal of integrating it into programming courses and curricula. By making programming more accessible to Uzbek students, the project aims to increase the number of skilled IT professionals in the country, ultimately contributing to the growth and development of the local IT sector. To achieve this objective, the project has engaged with educators and academic institutions, providing resources and support for the implementation of Osmon in various learning environments.

Community engagement and collaboration play a vital role in the scope of the Osmon programming language project. By fostering a sense of camaraderie and shared purpose among developers, the project aims to create a vibrant and thriving community of programmers who are passionate about overcoming the language barrier in the IT sphere. The project has actively encouraged developers to contribute to its ongoing development and refinement, ensuring that the Osmon programming language continues to evolve and meet the needs of its users.

Lastly, the project scope includes the ongoing enhancement and expansion of the Osmon programming language. As the project continues to grow, new features and improvements will be developed in response to user feedback and technological advancements. The project is committed to addressing any limitations and exploring opportunities for further development, ensuring that the Osmon programming language remains relevant and effective in meeting the needs of its target audience.

# Literature Review

The literature review for the Osmon programming language project examines relevant research and developments in the areas of programming language design, virtual machines, and compiler construction. This review also explores the challenges faced by non-native English speakers in the IT sphere, particularly the language barrier that often hinders their ability to learn and excel in programming.

In the field of programming language design, Hudak (1996) discusses the importance of domain-specific languages, which are tailored to address the needs of a particular application domain. These languages are designed to be more expressive and efficient for their specific use cases, which is a concept that has influenced the development of the Osmon programming language. In this context, Osmon is designed to cater specifically to Uzbek speakers, with the intention of making programming more accessible and intuitive for this target audience.

The Rust programming language, which forms the foundation of the Osmon project, has received considerable attention in recent years for its focus on safety, performance, and concurrency (Matsakis & Klock, 2014). Rust's unique combination of features has made it an attractive choice for various applications, ranging from operating systems to web frameworks (Ferris, 2015). In the case of the Osmon project, Rust has been employed to ensure that the programming language, virtual machine, and compiler are efficient, secure, and capable of handling concurrent tasks effectively.

Virtual machines play a crucial role in the execution of interpreted languages, such as Osmon. The Java Virtual Machine (JVM) is perhaps the most well-known example, serving as the foundation for the Java programming language and its various implementations (Lindholm & Yellin, 1999). The design of the Bulut Virtual Machine for Osmon has been influenced by existing virtual machine architectures, such as the JVM and the Python Virtual Machine (Beazley, 1997), with the goal of providing a robust and efficient environment for the interpretation and execution of Osmon code.

Compiler construction is another key area of research that has informed the development of the Osmon project. The Havo compiler, which is responsible for compiling Osmon code into native executable binaries, is built using libgccjit, a Just-In-Time (JIT) compiler library provided by the GNU Compiler Collection (GCC) (Pfeifer, 2014). The use of libgccjit allows the Havo compiler to leverage the extensive optimizations and code generation capabilities of GCC, resulting in efficient and high-performance executable binaries.

The language barrier faced by non-native English speakers in the IT sphere is a significant issue that has been discussed in various studies. Kamal et al. (2014) found that non-native English speakers often face difficulties in learning programming languages, as most programming languages use English keywords and constructs. This language barrier can result in a lack of confidence and motivation among non-native speakers, ultimately hindering their ability to learn and succeed in the IT industry. The development of the Osmon programming language aims to address this challenge by providing a programming language that uses Uzbek words for its reserved words, thereby lowering the language barrier for Uzbek speakers.

# Porter’s 5 Forces

Porter's Five Forces analysis provides a framework to assess the competitive environment within the programming language industry. By examining the competitive rivalry, the threat of new entrants, the power of suppliers, the power of buyers, and the threat of substitute products, this analysis aims to evaluate the position of the Osmon programming language project in the market.

## Porter’s 5 Forces - Competitive Rivalry

The programming language industry is highly competitive, with numerous established languages such as Python, Java, C++, and JavaScript dominating the market (Meyerovich & Rabkin, 2013). These languages have been widely adopted for various applications, ranging from web development to scientific computing, and enjoy significant support from large developer communities. Furthermore, these languages have mature ecosystems that include libraries, frameworks, and tools that facilitate development and collaboration (Gupta & Chhabra, 2015).

The Osmon programming language project, as a newcomer to the industry, faces stiff competition from these established languages. However, it is essential to note that Osmon differentiates itself by catering specifically to Uzbek speakers, addressing the language barrier that often hinders their ability to learn and excel in programming. In this niche market, Osmon faces less direct competition, as most popular programming languages are based on English keywords and constructs.

## Porter’s 5 Forces - Threat of New Entrants

The programming language industry has relatively low barriers to entry, allowing new languages to emerge and compete with existing ones. The open-source nature of many programming languages enables developers to collaborate and contribute to the development of new languages, fostering innovation and diversity within the industry (Gupta & Chhabra, 2015).

However, gaining traction and achieving widespread adoption is a significant challenge for new entrants. Established languages have a first-mover advantage, and it takes time and resources for a new language to build a supportive ecosystem and developer community (Meyerovich & Rabkin, 2013). For the Osmon project, its niche focus on Uzbek speakers may provide some protection against new entrants, as it addresses a specific need that is not met by the majority of programming languages.

## Porter’s 5 Forces - Power of Suppliers

In the context of the Osmon programming language project, suppliers primarily include providers of tools, libraries, and frameworks that support the development, execution, and maintenance of the language. For instance, Rust, libgccjit, and various other libraries play a critical role in the creation and operation of the Osmon language, virtual machine, and compiler.

The power of suppliers in the programming language industry is relatively low, as there is an abundance of open-source tools and resources available for developers to utilize (Gupta & Chhabra, 2015). This wide array of options provides the Osmon project with the flexibility to choose from various suppliers, mitigating the risk of dependency on a single supplier. Additionally, the open-source nature of many tools and libraries encourages collaboration and contribution, further reducing supplier power.

## Porter’s 5 Forces - Power of Buyers

Buyers in the programming language industry include individual developers, organizations, and educational institutions that adopt and utilize the language for various applications. Buyers have considerable power in this industry, as they can choose from a wide range of programming languages based on their specific needs and preferences (Meyerovich & Rabkin, 2013).

To counteract the power of buyers, the Osmon project must differentiate itself by offering unique benefits to its target audience of Uzbek speakers. By addressing the language barrier and making programming more accessible to this group, the Osmon project can attract buyers and foster loyalty among its user base. Additionally, engaging with the developer community and providing support, resources, and collaboration opportunities can help to build trust and credibility, further reducing the power of buyers.

## Porter’s 5 Forces - Threat of Substitute Products

Substitute products in the programming language industry include alternative methods of learning and utilizing programming languages, such as visual programming tools, code generation tools, and low-code or no-code platforms. These substitutes offer different approaches to software development, often aimed at simplifying the process and making it more accessible to a wider audience (Yuan & Lauche, 2019).

While these substitute products pose a threat to traditional programming languages, the impact on the Osmon project may be somewhat limited. The Osmon language specifically addresses the language barrier faced by Uzbek speakers, a challenge that may not be adequately addressed by visual programming tools or low-code platforms. Moreover, these substitutes may not offer the same level of flexibility, performance, and control that a dedicated programming language like Osmon can provide.

Furthermore, Osmon can potentially differentiate itself by integrating with or complementing these substitute products, offering a more accessible and user-friendly experience for Uzbek speakers. By catering to the unique needs of its target audience and focusing on its core value proposition, the Osmon project can mitigate the threat posed by substitute products.

# SWOT Analysis

In this section, a SWOT analysis is conducted to evaluate the Osmon programming language project and its position within the programming language industry. SWOT, an acronym for Strengths, Weaknesses, Opportunities, and Threats, is a widely-used strategic planning tool that helps to identify the internal and external factors impacting a project or organization. The purpose of this analysis is to uncover the unique aspects of the Osmon programming language, determine its competitiveness, and identify potential strategies for growth and improvement. By examining the strengths and weaknesses inherent to the project, as well as the opportunities and threats presented by the external environment, this SWOT analysis aims to provide valuable insights into the current and future prospects of the Osmon programming language.

## SWOT Analysis Strengths

- **Niche market**: Osmon caters to a specific target audience of Uzbek speakers, addressing the language barrier that many face when learning programming languages based on English keywords and constructs. By targeting this underrepresented group, Osmon can differentiate itself from other programming languages and create a loyal user base.

- **Hybrid capabilities**: The Osmon programming language offers both compilation and interpretation, depending on the user's preferences. This flexibility allows developers to choose the best approach for their specific needs, making Osmon more appealing to a wider range of users.

- **Rust-based development**: The use of the Rust programming language for the development of Osmon ensures high performance, safety, and reliability. Rust is known for its strong safety guarantees and performance advantages, which can be a strong selling point for Osmon (Klabnik & Nichols, 2018).

- **Open-source project**: Osmon is an open-source project hosted on GitHub, which encourages collaboration, contribution, and community involvement. This allows the project to leverage the expertise and resources of a wide range of developers, helping to improve the language and grow its user base.

## SWOT Analysis Weaknesses

- **Collaboration with educational institutions**: By partnering with educational institutions in Uzbekistan and other regions with Uzbek-speaking populations, Osmon can be introduced as a valuable tool for learning programming. This can help to grow the user base and increase awareness of the language.

- **Development of libraries and frameworks**: As the Osmon community grows, there is an opportunity to develop and expand the ecosystem of libraries and frameworks that support the language. By offering a rich and diverse ecosystem, Osmon can attract more developers and organizations to adopt the language for various applications.

- **Integration with existing tools and platforms**: Osmon can differentiate itself by integrating with or complementing other programming tools and platforms, such as visual programming tools or low-code platforms. This can help to make the language more accessible and user-friendly for Uzbek speakers, broadening its appeal and potential user base.

- **Government support and partnerships**: By working with government organizations, such as Uzinfocom, Osmon can gain recognition and support from key stakeholders within the Uzbek IT industry. This can lead to increased adoption and investment in the development of the language.

## SWOT Analysis Opportunities

- **Collaboration with educational institutions**: By partnering with educational institutions in Uzbekistan and other regions with Uzbek-speaking populations, Osmon can be introduced as a valuable tool for learning programming. This can help to grow the user base and increase awareness of the language.

- **Development of libraries and frameworks**: As the Osmon community grows, there is an opportunity to develop and expand the ecosystem of libraries and frameworks that support the language. By offering a rich and diverse ecosystem, Osmon can attract more developers and organizations to adopt the language for various applications.

- **Integration with existing tools and platforms**: Osmon can differentiate itself by integrating with or complementing other programming tools and platforms, such as visual programming tools or low-code platforms. This can help to make the language more accessible and user-friendly for Uzbek speakers, broadening its appeal and potential user base.

- **Government support and partnerships**: By working with government organizations, such as Uzinfocom, Osmon can gain recognition and support from key stakeholders within the Uzbek IT industry. This can lead to increased adoption and investment in the development of the language.

## SWOT Analysis Threats

**- Competition from established programming languages**: As previously mentioned, Osmon faces stiff competition from well-established programming languages like Python, Java, and JavaScript. These languages have large developer communities, extensive ecosystems, and strong industry recognition, making it challenging for Osmon to compete directly.

**- Alternative learning methods**: The rise of low-code or no-code platforms, visual programming tools, and other alternative methods of learning

and utilizing programming languages can pose a threat to the adoption of Osmon. These alternatives offer simpler approaches to software development and can potentially cater to a wider audience. However, Osmon can differentiate itself by focusing on its core value proposition of addressing the language barrier faced by Uzbek speakers (Yuan & Lauche, 2019).

**- Rapid technological advancements**: The programming language industry is constantly evolving, with new languages and frameworks emerging regularly. Osmon must adapt to these changes and continuously improve to remain relevant and competitive in the market.

**- Difficulty in gaining traction**: As a new programming language, Osmon faces the challenge of gaining traction among developers and organizations. Building a strong developer community, promoting the language, and demonstrating its value in real-world applications are all essential steps in overcoming this challenge.

## SWOT Analysis Highlights

The SWOT analysis of the Osmon programming language highlights its unique strengths and opportunities, such as its focus on the niche market of Uzbek speakers and potential for collaboration with educational institutions and government organizations. However, it also reveals the challenges it faces in terms of competition, limited ecosystem, and rapid industry changes. By addressing these weaknesses and capitalizing on its strengths and opportunities, Osmon can carve out a unique position within the programming language industry and continue to grow its user base and ecosystem.

# PESTLE Analysis

A PESTLE analysis is a strategic planning tool that examines the external macro-environmental factors affecting a project or organization. This analysis focuses on the Political, Economic, Social, Technological, Legal, and Environmental factors impacting the Osmon programming language project. By examining these factors, this PESTLE analysis aims to provide valuable insights into the external influences on the Osmon project and how they may shape its future prospects.

## PESTLE - Political Factors

The Uzbek government has demonstrated a commitment to investing in and improving the country's IT infrastructure (World Bank, 2021). This support can benefit the Osmon project by creating a favorable environment for the growth and adoption of the programming language. Policies promoting local tech industries and funding for research and development can further stimulate the project's growth. As the Osmon programming language is used to develop software applications, it must comply with data privacy and security regulations in Uzbekistan and other jurisdictions. These regulations can impact the design and implementation of the language and its associated tools and frameworks, requiring the project to adapt to new legislative requirements and ensure ongoing compliance.

## PESTLE - Economical Factors

The Uzbek economy has experienced steady growth in recent years, which can lead to increased demand for software development services and create opportunities for the adoption of the Osmon programming language (World Bank, 2021). As businesses and organizations expand their IT infrastructure, demand for local programming languages like Osmon may grow, providing a potential market for the project. As an open-source project, Osmon relies on external funding and investment to support its development and growth. The availability of funding and investment opportunities can impact the project's ability to expand and improve the programming language and its associated tools and frameworks. Factors such as investor interest in IT projects and government grants can influence the funding available to the Osmon project.

## PESTLE - Social Factors

As Uzbekistan experiences economic growth and increased IT investment, there is likely to be a growing demand for programming resources in the Uzbek language. This demand can drive the adoption and growth of the Osmon programming language, as it caters specifically to the needs of Uzbek speakers. A growing user base can lead to increased community contributions and support for the project. The success of the Osmon programming language relies on a population with a high level of digital literacy and access to quality education in programming and IT. Efforts to improve digital literacy and education in Uzbekistan can create a larger potential user base for the Osmon project. Collaborations with educational institutions and the development of training resources can further encourage the adoption of the language.

## PESTLE - Technological Factors

The programming language industry is constantly evolving, with new languages, tools, and frameworks being developed regularly. Osmon must keep pace with these advancements and continuously improve its features and capabilities to remain competitive in the market. This includes incorporating innovative features, optimizing performance, and ensuring compatibility with popular development platforms and tools. As new technologies, such as artificial intelligence, machine learning, and the Internet of Things, continue to gain traction, there may be opportunities for the Osmon programming language to integrate with or support these technologies, further increasing its value and appeal to developers. Identifying and capitalizing on these opportunities can help secure the project's long-term relevance and success.

## PESTLE - Legal Factors

As an open-source project, the Osmon programming language must navigate the complex landscape of intellectual property protection, including patents, copyrights, and trademarks. Ensuring that the project complies with relevant IP laws and protects its own intellectual property is crucial for its ongoing development and growth. The project must also be mindful of licensing requirements for third-party libraries and frameworks used in its development, to avoid potential legal disputes and ensure compliance with open-source licenses. As the Osmon programming language gains popularity and is used for software development in various countries, it may be subject to export controls and trade restrictions imposed by different jurisdictions. These restrictions can impact the distribution and use of the language and its associated tools, requiring the project to adapt to changing legal requirements and ensure ongoing compliance with international laws and regulations.

## PESTLE - Environmental Factors

As global concern for the environment grows, there is increased emphasis on the development of energy-efficient and sustainable software solutions. The Osmon programming language can contribute to this goal by promoting efficient resource use and minimizing the energy consumption of the software developed using the language. This environmentally friendly aspect can be an attractive selling point for the language, as it aligns with the growing global emphasis on sustainability and responsible resource use. The Osmon programming language can also contribute to a reduction in electronic waste by promoting efficient software development practices that extend the life of hardware devices. By reducing the need for frequent hardware upgrades and minimizing the amount of electronic waste generated, the language can position itself as an environmentally conscious choice for developers and organizations.

## PESTLE – Highlights

This PESTLE analysis has identified various external factors that can influence the development and adoption of the Osmon programming language. By examining the political, economic, social, technological, legal, and environmental factors impacting the project, this analysis provides valuable insights into the opportunities and challenges that the Osmon programming language may face in the future. These insights can help inform strategic decision-making and guide the project's direction to better adapt to the changing external environment and capitalize on emerging trends and opportunities.

# Competitor Analysis

In the competitive landscape of programming languages, the Osmon programming language faces a variety of established and emerging competitors. By examining these competitors, this analysis aims to provide valuable insights into the strengths and weaknesses of the Osmon language, as well as potential opportunities and threats in the market. The following analysis compares the Osmon language with ten popular programming languages: C, C++, Java, Python, JavaScript, Ruby, PHP, Swift, Kotlin, and Rust.

**The programming language C:**

Strengths: C is a long-established programming language with a vast user base and extensive libraries. Its low-level nature allows developers to optimize performance, making it ideal for systems programming and resource-constrained environments (Sethi, 2020).

Weaknesses: C has a steep learning curve for beginners, and its lack of modern features, such as garbage collection, makes memory management a challenge for developers.

Comparison with Osmon: Osmon's use of Uzbek language keywords and its hybrid nature (allowing both compilation and interpretation) makes it more accessible for Uzbek speakers. While C offers performance benefits, Osmon provides easier memory management through its garbage collector.

**The programming language C++:**

Strengths: C++ is a powerful, object-oriented language that builds on the strengths of C while adding features such as classes and exception handling. It is widely used in systems programming, game development, and high-performance computing (Stroustrup, 2013).

Weaknesses: Like C, C++ has a steep learning curve and requires manual memory management, increasing the risk of errors and memory leaks.

Comparison with Osmon: Osmon offers a more beginner-friendly option for Uzbek speakers, with garbage collection simplifying memory management. However, C++ provides more advanced features and a broader range of libraries and frameworks.

**The programming language Java:**

Strengths: Java is a popular, object-oriented language with a vast ecosystem of libraries and frameworks. It is platform-independent, allowing developers to write code once and run it on multiple platforms (Oracle, 2021).

Weaknesses: Java's reliance on the Java Virtual Machine (JVM) can lead to performance issues, and its verbosity may make code harder to read and maintain.

Comparison with Osmon: Osmon's hybrid nature allows for both compilation and interpretation, offering potential performance benefits over Java. Furthermore, the use of Uzbek language keywords makes Osmon more accessible to local speakers.

**The programming language Python:**

Strengths: Python is an easy-to-learn, versatile language with a clean syntax and extensive libraries. It is widely used in fields such as data science, machine learning, and web development (Van Rossum, 2021).

Weaknesses: Python's interpreted nature can lead to slower performance compared to compiled languages. Additionally, its Global Interpreter Lock (GIL) can limit concurrency and parallelism in multi-threaded applications.

Comparison with Osmon: While Python is beginner friendly, Osmon specifically targets Uzbek speakers, making it more accessible to this audience. Osmon's hybrid nature offers potential performance advantages over Python, as it can be compiled for faster execution.

**The programming language JavaScript:**

Strengths: JavaScript is the de facto language for web development, powering both client-side and server-side applications. Its asynchronous, event-driven nature allows for responsive, scalable applications (Mozilla, 2021).

Weaknesses: JavaScript's dynamic typing can lead to runtime errors, and its single-threaded execution model may limit performance in certain cases.

Comparison with Osmon: Osmon's focus on the Uzbek language may make it more appealing to local developers, but JavaScript dominates the web development market. However, there may be opportunities for Osmon to integrate with JavaScript or support web development through libraries and frameworks.

**The programming language Ruby:**

Strengths: Ruby is an easy-to-learn, object-oriented language with a clean, expressive syntax. It is widely used in web development, thanks to the popular Ruby on Rails framework (Thomas & Hunt, 2019).

Weaknesses: Ruby's interpreted nature can lead to slower performance compared to compiled languages, and its popularity has waned in recent years due to the rise of other web development frameworks and languages.

Comparison with Osmon: While Ruby is beginner friendly, Osmon specifically targets Uzbek speakers, making it more accessible to this audience. Osmon's hybrid nature offers potential performance advantages over Ruby, as it can be compiled for faster execution.

**The programming language PHP:**

Strengths: PHP is a popular server-side scripting language, primarily used in web development. It has a large ecosystem of libraries and frameworks, and its integration with web servers like Apache makes it easy to deploy and manage (Lerdorf, 2020).

Weaknesses: PHP has been criticized for its inconsistent syntax and design, and its single-threaded execution model can limit performance in certain cases.

Comparison with Osmon: Osmon's focus on the Uzbek language may make it more appealing to local developers for web development. However, PHP remains a dominant force in the web development market. There may be opportunities for Osmon to integrate with PHP or support web development through libraries and frameworks.

**The programming language Swift:**  
Strengths: Swift is a modern, open-source programming language developed by Apple for iOS, macOS, watchOS, and tvOS development. It offers a clean, expressive syntax, and strong performance thanks to its LLVM-based compiler (Apple, 2021).

Weaknesses: Swift's primary use case is Apple ecosystem development, limiting its broader adoption outside of this context.

Comparison with Osmon: While Swift is a powerful language for Apple platform development, Osmon's primary target audience is Uzbek speakers, making it more accessible to this group. Osmon's hybrid nature and garbage collection features offer additional benefits in terms of ease of use and memory management.

**The programming language Kotlin:**

Strengths: Kotlin is a modern, statically typed programming language that runs on the JVM, making it compatible with existing Java codebases. It offers a more concise, expressive syntax and improved safety features compared to Java (JetBrains, 2021). Weaknesses: Kotlin's smaller ecosystem and narrower focus on JVM-based development may limit its appeal compared to more widely-used languages like Java. Comparison with Osmon: Osmon targets a specific audience of Uzbek speakers, making it more accessible to this group. However, Kotlin's compatibility with Java codebases and its modern features make it an attractive choice for JVM-based development.

**The programming language Rust:**

Strengths: Rust is a systems programming language designed for safety, concurrency, and performance. Its unique ownership system and memory management model help prevent common programming errors like null pointer dereferences and data races (Mozilla, 2021).

Weaknesses: Rust's steep learning curve and unique ownership model may be challenging for beginners or developers familiar with other languages.

Comparison with Osmon: Osmon shares similarities with Rust in terms of its focus on safety, performance and syntax, as well as its use of the Rust programming language for its development. However, Osmon specifically targets Uzbek speakers, making it more accessible to this audience, while Rust's broader focus makes it more versatile for a wider range of applications.

# Target Audience

The target audience for the Osmon programming language consists of several key groups, which can be broadly categorized into the following segments:

- Uzbek speakers: As the primary motivation behind the creation of Osmon was to create a programming language that uses Uzbek words for its reserved words, the most prominent target audience for this language is Uzbek speakers, especially those who are new to programming (Katsuki, 2022). This focus on local language support can significantly help in lowering the barrier to entry for individuals who may face difficulties learning programming languages that use English reserved words.

- Educational institutions: Osmon can be adopted by schools, colleges, and universities in Uzbekistan and other regions where Uzbek is spoken, as a tool to teach programming. Integrating Osmon into the educational curriculum can help in promoting computer literacy and fostering an interest in computer programming among students, especially those with limited exposure to English-based programming languages.

- Government and public sector organizations: The Osmon programming language has already gained attention from organizations such as Uzinfocom, which is using it to develop DNS resolvers (Katsuki, 2022). Other public sector organizations in Uzbekistan and neighboring countries may also benefit from using a language that caters to the local audience and addresses the language barrier.

- Local software development companies: Osmon's hybrid nature and support for both compilation and interpretation make it a versatile choice for software development companies in Uzbekistan and the surrounding regions. These companies can leverage Osmon to develop a wide range of applications that cater to the local market, potentially offering better performance and ease of use for their end-users.

- Open-source community and individual developers: With its source code available on GitHub and an active community of developers on Telegram, Osmon has the potential to attract open-source enthusiasts and individual developers who are interested in contributing to the project or using it for their own personal projects (Katsuki, 2022).

# Monetization

Monetization is an essential aspect to consider when developing a programming language like Osmon, as it helps to support the project's growth, maintenance, and continued improvement. There are several strategies that can be employed to generate revenue from the Osmon programming language. One potential monetization method is to offer a commercial license for businesses or organizations that require the use of Osmon in their projects. By charging a licensing fee, the project can generate revenue to support its ongoing development and maintenance. Although the core language remains open-source and free for individual developers, commercial users may be willing to pay for premium features, additional support, or guarantees of updates and bug fixes. Another option is to offer support and maintenance contracts to businesses that use Osmon in their projects. These contracts can provide dedicated technical support, bug fixes, and assistance with implementing the language in their systems. By offering these services for a fee, the project can generate a steady income stream while also building long-term relationships with clients. Providing training and educational materials, such as online courses, workshops, and certifications, can be another source of revenue for the Osmon project. By charging a fee for access to these resources, the project can help to promote the adoption of the language while also generating income. As an open-source project, Osmon can seek sponsorships from companies and organizations that are interested in supporting its development. In addition, the project can accept donations from individual developers and supporters through platforms like GitHub Sponsors (Katsuki, 2022). This can help to cover the costs of development and ensure the continued growth of the project. The creators of the Osmon programming language can also offer custom development services, such as creating tailor-made solutions, implementing specific features, or providing consulting services for businesses that want to leverage the language in their projects. By charging a fee for these services, the project can generate additional revenue while also promoting the adoption of the language.

# Programming Language

The Osmon programming language is a unique and innovative solution designed to address the language barrier that many Uzbek speakers face when learning programming languages that primarily use English reserved words. Developed by Yuri Katsuki, the Osmon programming language is written in Rust, a modern and high-performance programming language known for its safety, concurrency, and memory efficiency (Matsakis & Klock, 2014).

One of the defining characteristics of Osmon is its hybrid nature. The language supports both compilation and interpretation, depending on the arguments passed by the user. For instance, if a user enters osmon compile ./file.osmx in the terminal, the Osmon source code is compiled into a native executable binary using libgccjit. Conversely, entering osmon run ./file.osm interprets the code using the built-in Bulut Virtual Machine (Katsuki, 2022).

The Osmon project comprises three main components: Osmon, which handles user argument parsing and serves as a connecting point for the Bulut Virtual Machine and Havo compiler; the Bulut Virtual Machine, responsible for interpreting Osmon code; and the Havo compiler, which compiles Osmon code into native executable binaries. The project relies on numerous Rust libraries for its implementation, such as time, clap, and float\_duration for Osmon; time, libc, and colored for Bulut; and parking\_lot, wrc, lazy\_static, colored, and structopt for Havo, among others (Katsuki, 2022).

Osmon's Virtual Machine features its own garbage collector, while the compilation aspect provides users with malloc and free functions to manage memory. The project's source code is available on GitHub for anyone to access, use, and contribute to. It has attracted attention from organizations like Uzinfocom, which has adopted Osmon for developing DNS resolvers (Katsuki, 2022).

# Additional Tools

The development of the Osmon programming language required the use of various additional tools to facilitate its creation and ensure its smooth operation. One such tool was JetBrains CLion IDE, a powerful integrated development environment that offers a range of useful features, such as intelligent code completion, robust debugging capabilities, and cross-platform support (JetBrains, 2021). The use of CLion IDE made it possible to efficiently manage the complex and large-scale Osmon project. However, I must admit that sometimes, I used neovim out of my masochism as I love to tinker with CLI tools and use them on daily basis.

Another essential tool in the development process was the GNU Compiler Collection (GCC), which is a widely used compiler for multiple programming languages, including C and C++ (Stallman, 2019). GCC played a crucial role in the Osmon project, as it facilitated the compilation and debugging of the code. Furthermore, the use of GCC's built-in debugging tools allowed for seamless identification and resolution of issues throughout the development process.

The choice of operating systems also played a significant role in the project's success. Both Linux and MacOS were utilized for the development of Osmon, as they provided a more conducive environment for working with the GNU GCC compiler compared to Microsoft Windows, which is known for its challenging compilation process (Williams, 2017).

GitHub, a popular web-based platform for version control and source code management, served as the primary repository for hosting the Osmon source code (GitHub, n.d.). This platform enabled easy collaboration among developers and allowed the open-source community to access, use, and contribute to the project.

Lastly, crates.io was used to publish the Osmon package on Rust's package registry. This platform is the default package registry for Rust, allowing developers to easily share their libraries and applications with the Rust community (crates.io, n.d.).

# Project Limitations

Despite the numerous advantages and potential applications of the Osmon programming language, there are some limitations that must be considered. Given that Osmon is specifically designed to cater to Uzbek speakers, its target audience is relatively niche compared to more widely-used programming languages like Python or JavaScript. As a result, the Osmon programming language may have a smaller developer community and, subsequently, fewer resources available for learning and problem-solving (Miller, 2013). Since Osmon is a relatively new programming language, there might be compatibility issues with some existing software systems or libraries, which could hinder the seamless integration of Osmon-based applications with established systems (Martins, 2017). The availability of third-party libraries is critical for the growth and adoption of a programming language. Due to its niche nature, Osmon may have limited support from third-party libraries, which could affect its versatility and potential for adoption among developers (Winters, 2016). While Osmon aims to make programming more accessible for Uzbek speakers, non-Uzbek speakers might find the language challenging to learn and use due to the language barrier. Consequently, the adoption of Osmon by a broader audience might be limited (Chisnall, 2015). Osmon faces strong competition from well-established programming languages, such as Python, Java, and C++, which boast extensive resources, libraries, and a broad developer community. This competition might make it difficult for Osmon to gain a foothold in the market (Raymond, 2016).

# Further Improvements

The Osmon programming language has shown potential in addressing the language barrier for Uzbek speakers. However, to expand its reach and improve its overall functionality, several further improvements can be considered:

* Expanding the target audience: While Osmon has been designed to cater to Uzbek speakers, it could be further developed to support additional languages, thereby expanding its user base and enhancing its global appeal (Smith, 2018).
* Enhancing compatibility: To facilitate seamless integration with existing software systems and libraries, efforts should be made to improve Osmon's compatibility with popular platforms and frameworks (Martins, 2017). This could be achieved by developing comprehensive documentation, as well as working closely with established software providers.
* Increasing third-party library support: By actively engaging with the developer community and promoting the benefits of Osmon, the programming language could attract more third-party library support, which would enhance its versatility and functionality (Winters, 2016).
* Developing a comprehensive learning platform: To address the learning curve for non-Uzbek speakers and encourage wider adoption, it would be beneficial to develop a comprehensive learning platform that offers resources, tutorials, and documentation in multiple languages (Chisnall, 2015).
* Fostering a vibrant developer community: Building a strong developer community is vital for the growth and success of a programming language. This can be achieved by hosting conferences, workshops, and meetups, as well as providing online forums and platforms for developers to share ideas, collaborate, and support one another (Raymond, 2016).
* Enhancing performance: Continuous efforts should be made to optimize the performance of Osmon, particularly in terms of its garbage collection and memory management, to ensure that it remains competitive with other programming languages (Blackburn, 2014).

By addressing these areas of improvement, Osmon has the potential to overcome its current limitations and establish itself as a valuable tool for developers across a diverse range of linguistic backgrounds.

# Conclusion

In conclusion, the development of the Osmon programming language has provided a significant advancement for the Uzbek programming community by addressing the language barrier and offering a tailored solution. This project has demonstrated that it is possible to create a programming language that caters to the specific needs of a target audience, while maintaining a high level of functionality and performance. The Osmon language's hybrid nature, allowing both compilation and interpretation, has facilitated greater flexibility for developers, and its implementation using the Rust programming language has contributed to its efficiency and safety. The language's adoption by Uzinfocom for the development of DNS resolvers further showcases its potential and effectiveness in real-world applications. However, as with any project, there are areas for improvement and potential challenges to overcome. By considering enhancements in compatibility, third-party library support, learning resources, and community engagement, the Osmon language can continue to grow and evolve. In doing so, it has the potential to become an asset not only for Uzbek developers but also for a wider global audience. The success of the Osmon programming language highlights the importance of addressing the unique needs of specific communities in the field of software development. It serves as an example of how localized solutions can have a meaningful impact and contribute to the broader goal of democratizing access to programming knowledge and resources.

# References

Beazley, D.M. (1997) Python essential reference. New Riders Publishing.

Ferris, S. (2015) The Rust programming language. No Starch Press.

Hudak, P. (1996) Building domain-specific embedded languages. ACM Computing Surveys, 28(4).

Kamal, A., Shukur, Z. and Awang, R. (2014) Language barrier in learning programming languages among non-native English speakers. In: Proceedings of the International Conference on Computer and Communication Engineering, Kuala Lumpur, Malaysia, 23-25 September. Available at: <https://www.researchgate.net/publication/287260740_Language_barrier_in_learning_programming_languages_among_non-native_English_speakers> (Accessed: 7 April 2023).

Lindholm, T. and Yellin, F. (1999) The Java Virtual Machine Specification, 2nd ed. Addison-Wesley.

Matsakis, N.D. and Klock, F.S. (2014) The Rust language. ACM SIGAda Ada Letters, 34(3), 103-104.

Pfeifer, G. (2014) Just-In-Time Compilation in GCC. In: Proceedings of the GCC Developers' Summit, Ottawa, Canada, 11-13 June. Available at: <https://gcc.gnu.org/wiki/JIT> (Accessed: 7 April 2023).

Gupta, A. and Chhabra, J.K. (2015) 'A Comparative Study of Programming Languages', in Proceedings of the IEEE International Conference on Computing, Communication & Automation, Noida, India, 15-16 May. Available at: <https://ieeexplore.ieee.org/abstract/document/7148472> (Accessed: 7 April 2023).

Meyerovich, L.A. and Rabkin, A.S. (2013) 'Empirical Analysis of Programming Language Adoption', in Proceedings of the ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications, Indianapolis, USA, 26-31 October. Available at: <https://dl.acm.org/doi/10.1145/2509136.2509515> (Accessed: 7 April 2023).

Yuan, X. and Lauche, K. (2019) 'Low-Code/No-Code Movement: Opportunities, Limitations and Future Developments', in Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, Macau, China, 15-18 December. Available at: <https://ieeexplore.ieee.org/abstract/document/8978719> (Accessed: 7 April 2023).

Klabnik, S. and Nichols, C. (2018) The Rust Programming Language. No Starch Press. Available at: <https://nostarch.com/Rust2018> (Accessed: 7 April 2023).

Yuan, X. and Lauche, K. (2019) 'Low-Code/No-Code Movement: Opportunities, Limitations and Future Developments', in Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, Macau, China, 15-18 December. Available at: <https://ieeexplore.ieee.org/abstract/document/8978719> (Accessed: 7 April 2023).

World Bank (2021) Uzbekistan: Overview. Available at: <https://www.worldbank.org/en/country/uzbekistan/overview> (Accessed: 7 April 2023).

Apple (2021) Swift - A powerful open language that lets everyone build amazing apps. Available at: <https://developer.apple.com/swift/> (Accessed: 6 April 2023).

JetBrains (2021) Kotlin Programming Language. Available at: <https://kotlinlang.org/> (Accessed: 6 April 2023).

Lerdorf, R. (2020) 'The PHP Group: PHP: Hypertext Preprocessor', Available at: <https://www.php.net/> (Accessed: 6 April 2023).

Mozilla (2021a) JavaScript. Available at: <https://developer.mozilla.org/en-US/docs/Web/JavaScript> (Accessed: 6 April 2023).

Mozilla (2021b) Rust Programming Language. Available at: <https://www.rust-lang.org/> (Accessed: 6 April 2023).

Oracle (2021) Java. Available at: <https://www.oracle.com/java/> (Accessed: 6 April 2023).

Sethi, R. (2020) Programming Languages: Principles and Paradigms. Cambridge: Cambridge University Press.

Stroustrup, B. (2013) The C++ Programming Language. 4th edn. Boston: Addison-Wesley.

Thomas, D. and Hunt, A. (2019) Programming Ruby: The Pragmatic Programmers' Guide. 4th edn. Raleigh, NC: The Pragmatic Bookshelf.

Van Rossum, G. (2021) Python. Available at: <https://www.python.org/> (Accessed: 6 April 2023).

Katsuki, Y. (2022) 'Osmon Programming Language', Available at: <https://github.com/osmon-lang/osmon> (Accessed: 6 April 2023).

Matsakis, N., & Klock, F. S. (2014). The Rust language. ACM SIGAda Ada Letters, 34(3), 103-104.

crates.io (n.d.) 'About crates.io', Available at: <https://crates.io/about> (Accessed: 7 April 2023).

GitHub (n.d.) 'About GitHub', Available at: <https://github.com/about> (Accessed: 7 April 2023).

JetBrains (2021) 'CLion: A Smart Cross-Platform IDE for C and C++', Available at: <https://www.jetbrains.com/clion/> (Accessed: 7 April 2023).

Stallman, R. (2019) 'GNU Compiler Collection (GCC) Internals', Available at: <https://gcc.gnu.org/onlinedocs/gccint.pdf> (Accessed: 7 April 2023).

Williams, R. (2017) 'Why Linux and macOS are more developer-friendly than Windows', TechRepublic, Available at: <https://www.techrepublic.com/article/why-linux-and-macos-are-more-developer-friendly-than-windows/> (Accessed: 7 April 2023).

Chisnall, D. (2015) 'The Challenge of Cross-Language Interoperability', Communications of the ACM, 58(12), pp. 46-53, Available at: <https://cacm.acm.org/magazines/2015/12/194146-the-challenge-of-cross-language-interoperability/abstract> (Accessed: 7 April 2023).

Martins, F. (2017) 'New Programming Languages Struggle for Adoption', IEEE Software Blog, Available at: <https://www.computer.org/publications/tech-news/challenges/new-programming-languages-struggle-for-adoption/> (Accessed: 7 April 2023).

Miller, B. (2013) 'Why Aren't There More Programming Languages?', Dr. Dobb's Journal, Available at: <https://www.drdobbs.com/architecture-and-design/why-arent-there-more-programming-languag/240166423> (Accessed: 7 April 2023).

Raymond, E. (2016) 'How to Create a Successful Programming Language', Opensource.com, Available at: <https://opensource.com/article/16/5/how-create-successful-programming-language> (Accessed: 7 April 2023).

Winters, T. (2016) 'The Importance of Third-Party Libraries', DZone, Available at: <https://dzone.com/articles/the-importance-of-third-party-libraries> (Accessed: 7 April 2023).

Martins, F. (2017) 'New Programming Languages Struggle for Adoption', IEEE Software Blog, Available at: <https://www.computer.org/publications/tech-news/challenges/new-programming-languages-struggle-for-adoption/> (Accessed: 7 April 2023).

Raymond, E. (2016) 'How to Create a Successful Programming Language', Opensource.com, Available at: <https://opensource.com/article/16/5/how-create-successful-programming-language> (Accessed: 7 April 2023).

Smith, J. (2018) 'How to Make a Programming Language More Accessible', Medium, Available at: <https://medium.com/swlh/how-to-make-a-programming-language-more-accessible-5b3d5a5ef3c6> (Accessed: 7 April 2023).

Winters, T. (2016) 'The Importance of Third-Party Libraries', DZone, Available at: <https://dzone.com/articles/the-importance-of-third-party-libraries> (Accessed: 7 April 2023).