УДК 004.43+81’42 DOI 10.37972/chgpu.2026.130.1.018

Т. Ю. Мкртчян, Ж. В. Макарова

СОПОСТАВИТЕЛЬНЫЙ АНАЛИЗ СТАТИСТИЧЕСКИХ И ТИПОЛОГИЧЕСКИХ ХАРАКТЕРИСТИК ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

Южный федеральный университет, г. Ростов-на-Дону, Россия

Аннотация. Статья посвящена сопоставительному анализу семи языков программирования – Fortran, ALGOL, C, Assembly, Python, Ruby и Brainfuck – с точки зрения их структурных и статистических характеристик. Исследование ориентировано на лексическое разнообразие, энтропию Шеннона и типологические различия формальных языковых систем. В работе рассматриваются синтаксические особенности программного кода, распределение и вариативность токенов, а также уровень формальной сложности, характерный для каждого из анализируемых языков. Такой подход позволяет проследить, каким образом различия в устройстве языков отражаются на способах построения и восприятия программного текста. Особое внимание уделяется сопоставлению уровня абстракции языка с его статистическими показателями, которые могут быть интерпретированы как косвенный индикатор когнитивной нагрузки при чтении и понимании кода. Проведенный анализ показывает, что высокоуровневые языки, как правило, демонстрируют более широкое лексическое разнообразие и повышенные значения энтропии, тогда как минималистичные языковые системы опираются на ограниченный набор элементов при жестко заданной структуре. В целом результаты подтверждают продуктивность методов цифровой лингвистики при изучении искусственных и формальных языков и указывают на возможность их дальнейшего использования в междисциплинарных исследованиях.

Ключевые слова: *языки программирования, искусственные языки, лексическое разнообразие, энтропия Шеннона, типология языков, цифровая лингвистика, формальные языки*

T. Yu. Mkrtchyan, Zh. V. Makarova

COMPARATIVE ANALYSIS OF STATISTICAL AND TYPOLOGICAL CHARACTERISTICS OF PROGRAMMING LANGUAGES

Southern Federal University, Rostov-on-Don, Russia

Abstract. The article presents a comparative analysis of seven programming languages – *Fortran, ALGOL, C, Assembly, Python, Ruby* and *Brainfuck* – focusing on their structural and statistical characteristics. The study examines lexical diversity, Shannon entropy, and typological differences within formal language systems. Particular attention is paid to the syntactic organisation of program code, the distribution and variability of tokens, and the degree of formal complexity specific to each language under consideration. This approach makes it possible to trace how differences in language design influence the construction and interpretation of program texts. Special emphasis is placed on the relationship between the level of language abstraction and its statistical parameters, which may be interpreted as an indirect indicator of cognitive load during code reading and comprehension. The analysis shows that high-level languages tend to exhibit greater lexical variability and higher entropy values, whereas minimalist languages rely on a limited set of elements combined with a rigid structural framework. Overall, the results confirm the applicability of digital linguistics methods to the analysis of artificial and formal languages and point to prospects for further interdisciplinary research.

Keywords: *programming languages, artificial languages, lexical diversity, Shannon entropy, language typology, digital linguistics, formal languages*

Введение. Современная лингвистика все активнее обращается к формальным языкам, рассматривая их не только как инструмент программирования, но и как полноценный объект филологического анализа. Это вызвано расширением представлений о языке как о системе знаков, которая функционирует в разнообразных условиях и выполняет как когнитивные, так и коммуникативные задачи. Особое внимание уделяется языкам программирования: они строго формализованы, постоянно находятся в развитии, имеют собственные лексические и грамматические правила, а самое главное – имеют широкий спектр решения задач, от лаконичных систем символов до развернутых конструкций, подобных тем, что может создать человек.

Сопоставительные исследования языков программирования позволяют выявлять типологические различия между ними, прослеживать эволюцию языкового формализма и устанавливать взаимосвязи между грамматическими принципами, парадигмами программирования и статистическими характеристиками кода. Применение количественных методов – таких как анализ токенов, оценка лексического разнообразия и вычисление энтропии – обеспечивает объективное сравнение языковых систем, опирающееся на корпусные данные. Подобный подход соответствует общему вектору современной цифровой лингвистики, где традиционные методы исследования дополняются вычислительными и статистическими инструментами.

Цель настоящего исследования – провести сопоставительный анализ семи языков программирования (Fortran, ALGOL, C, Assembly, Python, Ruby и Brainfuck) на основе корпусных данных и набора количественных метрик. Рассматриваемые в работе языки представляют разные исторические периоды, парадигмы и уровни абстракции, что делает их совместное изучение методологически плодотворным. Анализ строится на исследовании трех ключевых параметров: общего количества токенов, числа уникальных токенов и энтропии. Эти метрики отражают степень разнообразия, сложность и структурную плотность текстов программ, что позволяет рассматривать языки программирования в терминах, привычных для исследований естественных языков.

В настоящей работе языки программирования изучаются в хронологической последовательности их появления и охватывают различные этапы развития формальных языковых систем. Ранние языки Fortran (1950-е гг.) и ALGOL (конец 1950-х – 1960-е гг.), сформировавшиеся в период становления вычислительной техники, характеризуются стремлением к формализации и строгой структурированности синтаксиса. Языки C и Assembly, активно используемые с 1970-х гг., отражают этап сближения абстрактных языковых конструкций с архитектурой вычислительных машин и демонстрируют повышенную синтаксическую плотность. Высокоуровневые языки Python и Ruby, появившиеся во второй половине 1990-х гг., ориентированы на читаемость и выразительность, что находит отражение в их лексическом разнообразии. Минималистичный язык Brainfuck, созданный в 1990-е гг. как экспериментальная система, представляет крайний полюс формализации с ограниченным набором лексических элементов. Распределение языков по историческим периодам позволяет соотнести динамику лексического разнообразия и энтропийных характеристик с эволюцией принципов проектирования языков программирования.

Научная новизна работы заключается в интеграции лексико-статистического подхода в сопоставительное исследование языков программирования, причем на материале, включающем как классические, так и современные, а также минималистические и эзотерические языки. Такая комбинация предоставляет возможность выявить общие закономерности и специфические черты их организации, а также продемонстрировать связь между степенью формализации языка и его статистическим профилем.

Полученные результаты позволяют по-новому взглянуть на природу формальных языковых систем и их функциональное разнообразие. Исследование ориентировано

на междисциплинарный диалог между компьютерной лингвистикой, цифровой филологией и теорией языков программирования и может быть использовано как в академической практике, так и в дальнейшем развитии корпусных подходов к формальному языку.

Актуальность исследуемой проблемы. Актуальность настоящего исследования обусловлена устойчивым интересом современной лингвистики к искусственным языкам как особому классу знаковых систем, создаваемых в результате целенаправленного конструирования. В отечественной научной традиции искусственные языки рассматриваются как самостоятельный объект лингвистического анализа, позволяющий моделировать языковые структуры и выявлять фундаментальные свойства языка вне рамок естественной языковой эволюции [2, с. 15–22].

Исследователи подчеркивают, что ключевой характеристикой искусственных языков является их проектируемый характер, предполагающий сознательный отбор и комбинирование языковых элементов. Так, А. С. Зернова и Т. С. Маркова указывают, что использование окказиональных и нестандартных языковых средств выступает важным механизмом формирования искусственно придуманных языков, обеспечивая экспериментирование с формой и функцией языковой системы [1, с. 94–95]. Данный тезис применим и к языкам программирования, в которых лексический состав и синтаксические конструкции формируются искусственно и подчинены заранее заданным принципам.

В ряде отечественных работ искусственные языки анализируются с философских, семиотических и типологических позиций, что позволяет рассматривать их как особый вид формализованных коммуникативных систем ([3, с. 113–118], [4, с. 67–69], [11, с. 456–458]). Вместе с тем языки программирования в подобных исследованиях, как правило, остаются на периферии внимания или же изучаются преимущественно в прикладном / техническом аспекте, без обращения к их лингвистическим характеристикам.

Между тем сопоставительный анализ языков программирования на основе количественных показателей открывает дополнительные возможности для выявления их структурных и типологических различий. Применение статистических методов, заимствованных из корпусной лингвистики и теории информации, позволяет анализировать такие параметры, как лексическое разнообразие и распределение элементов кода, соотнося их с уровнем формализации и абстракции языковой системы. Зарубежные исследования демонстрируют продуктивность подобного подхода при анализе искусственных и формальных языков, в том числе в контексте их эволюции и минимализма ([5, с. 1–12], [6, с. 421–425]).

Таким образом, актуальность настоящего исследования определяется необходимостью комплексного сопоставительного анализа языков программирования как разновидности искусственных языков, основанного на сочетании лингвистической теории и количественных методов. Подобный подход позволяет расширить представления о типологическом разнообразии искусственных языков и интегрировать языки программирования в поле сравнительно-сопоставительной лингвистики.

Материал и методы исследования. Для проведения сопоставительного анализа семи искусственных языков программирования – Fortran, ALGOL, C, Assembly, Python, Ruby и Brainfuck – сформирован корпус из открытых и проверенных примеров кода. Каждый язык представлен пятью типовыми программами, охватывающими базовые конструкции, арифметические функции, циклы и массивы, алгоритмы сортировки и операции с файлами. Такой отбор обеспечивает возможность корректного количественного сравнения языков по уровню синтаксической плотности, лексическому разнообразию и распределению токенов ([10, с. 2–5], [7, с. 47]).

Метаданные о языках и примерах кода собраны из официальной документации, учебных репозиториев, специализированных ресурсов и энциклопедий, включая Rosetta Code, GitHub, официальные сайты языков и страницы Esolang Wiki для Brainfuck. Для каждого

языка фиксировались год создания, автор, парадигма, тип, список ключевых слов, краткое описание и ссылка на источник. Примеры кода также документированы: идентификатор, тип примера, источник и длина текста в символах. Такая организация данных обеспечивает воспроизводимость исследования и прозрачность аналитических процедур [12, с. 12].

Для обработки и анализа корпуса использованы инструменты цифровой лингвистики и вычислительные методы. Основным инструментом был Python 3.13 с библиотеками pandas, Pygments, collections, radon и math, позволяющими токенизировать код, вычислять частоты токенов и оценивать структурное разнообразие через энтропию Шеннона ([6, с. 427–429], [5, с. 5]). Токенизация с помощью Pygments обеспечивала корректное разделение программ на ключевые слова, идентификаторы, литералы, операторы и символы, что позволило сопоставлять различные языки на единой метрической основе. Для языков без готового лексера, например, некоторых вариаций Assembly, применен fallback-режим с использованием базового текстового лексера.

В качестве ключевой метрики при анализе использована энтропия Шеннона, оценивающая структурное и лексическое разнообразие кода. Высокие значения энтропии указывали на богатство синтаксических конструкций, низкие – на минимализм или строгие ограничения языка, что особенно характерно для Brainfuck ([8, с. 135], [13, с. 5]). Результаты анализа сохранялись в формате JSON для последующего экспорта и визуализации, что позволяет строить графовые модели и выполнять сопоставительные исследования языков на уровне токенов и конструкций.

Все данные и скрипты организованы в едином репозитории с каталогами для метаданных, примеров кода и аналитических скриптов. Такая структура позволяет гарантировать воспроизводимость результатов, автоматизировать процедуру анализа и масштабировать исследование на дополнительные языки в будущем [14, с. 73].

Результаты исследования и их обсуждение. Сопоставительный анализ семи языков программирования показал заметные различия в синтаксической плотности и лексическом разнообразии, что отражает их функциональные и когнитивные особенности. Наибольший объем кода и высокую вариативность токенов продемонстрировали языки Fortran, C и Ruby, в то время как минималистичные языки, такие как Brainfuck, характеризуются ограниченным набором токенов и низкой энтропией ([7, с. 47–48], [13, с. 6]).

Наглядное сопоставление исследуемых языков программирования по ключевым количественным параметрам представлено в таблице 1. В ней приведены сводные статистические характеристики корпуса, представлены значения общего числа токенов (N), числа уникальных токенов (V) и энтропии Шеннона (H), рассчитанные для каждого языка на основе единой методики токенизации и анализа.

Таблица 1 – Статистические характеристики корпуса по языкам программирования

	Fortran	**ALGOL**	**C**	**Assembly**	**Python**	**Ruby**	**Brainfuck**
Общее количество токенов (N)	265	263	301	156	138	264	215
Число уникальных токенов (V)	70	56	72	50	53	62	7
Энтропия Шеннона (H)	5,28	5,23	5,46	4,85	5,04	5,18	1,92

Показатели таблицы 1 получены по объединенному корпусу каждого языка, включающему пять типовых программ. Значения N и V отражают объем токенизированного кода и уровень его лексического разнообразия, тогда как энтропия Шеннона (H) характеризует общую структурную вариативность программного текста. Сопоставление данных показывает, что языки с развитым и формально насыщенным синтаксисом (Fortran, ALGOL, C) демонстрируют наиболее высокие значения энтропии. Высокоуровневые языки Python и Ruby достигают сопоставимых показателей вариативности при меньшем

объеме кода. Минималистичные системы, такие как Brainfuck, напротив, характеризуются резким снижением числа уникальных токенов и энтропии, что связано с жесткими ограничениями их формальной структуры.

Детальный анализ различий между языками программирования на уровне элементарных единиц программного текста и десять наиболее частотных токенов каждого языка представлены в таблице 2. Она наглядно демонстрирует, какие синтаксические элементы доминируют в коде и за счет каких компонентов формируются различия, выявленные при анализе агрегированных статистических показателей.

Таблица 2 – Частотный рейтинг токенов (топ-10) в корпусах исследуемых языков программирования

Язык	Топ-10 токенов (токен:частота)
Fortran	(:22,):22, ":20, =:17, end:12, i:12, program:10, n:10, :::9, integer:8
ALGOL	;:20, ::19, =:18, i:17, (:12, 1:12, ":9,):8, n:7, b:6
C	;:24, (:17,):17, ":13, """:12, int:10, =:10, j:10, {:9, }:9
Assembly	":25, mov:18, eax:9, ebx:9, section:7, .text:5, global:5, _start:5, _start::5, 1:5
Python	""":16, (:12,):12, ":9, ::7, =:6, a:5, b:5, x:5, print:4
Ruby	""":8, ":8, x:6, (:6,):6, .:5, puts:4, 1:4, =:4, n:4
Brainfuck	+:126, -:34, >:17, .:17, <:11, [:5,]:5

Частотные списки показывают, что состав наиболее употребительных токенов заметно различается в зависимости от уровня абстракции языка и принципов его проектирования. В языках с более строгой и формализованной структурой (Fortran, ALGOL, C) среди частотных элементов преобладают операторы, служебные слова и символы, связанные с организацией структуры программы. В Python и Ruby, напротив, чаще встречаются токены, отражающие работу с выражениями и вводом-выводом, что соответствует ориентации этих языков на читаемость и компактность кода. Язык Brainfuck резко выделяется на этом фоне: большая доля частот сосредоточена на небольшом наборе символов, что напрямую связано с его минималистичным синтаксисом и экспериментальным характером.

По числу токенов и уникальных токенов можно выделить несколько закономерностей. Например, в Fortran примеры факториала и алгоритма «решето Эратосфена» имеют высокую энтропию (4.42 и 4.49 соответственно), что отражает использование разнообразных синтаксических конструкций и операций. Для языка C картина близка к Fortran: на примере сортировки пузырьком энтропия достигает 4,78. Это ожидаемо для кода, где одновременно работают циклы, условия и операции со структурами данных [10, с. 3]. У Brainfuck результаты иные: энтропия держится в диапазоне 1,96–3,39. Низкие значения здесь связаны не с «бедностью» примеров, а с устройством самого языка – он опирается на крайне ограниченный набор команд и минимальный синтаксис [8, с. 136].

Если перейти к высокоуровневым языкам, Python и Ruby выглядят заметно компактными и менее «жесткими» в синтаксисе. Даже в самом простом случае это видно по объему: «Hello World» укладывается в 9 токенов для Python и в 8 – для Ruby. Такая экономия связана с ориентацией этих языков на читаемость и быстрые наброски решений. При усложнении задач (сортировка, рекурсивные функции) число токенов и значения энтропии растут, однако рост идет без резкого утяжеления конструкции: алгоритм можно записать выразительно, не перегружая текст формальными деталями [6, с. 430–431].

ALGOL и Assembly в этом ряду занимают промежуточную позицию между минималистичными и высокоуровневыми системами. Assembly требует явного управления регистрами и памятью, что приводит к высокой синтаксической нагрузке на примеры среднего объема. ALGOL, в свою очередь, демонстрирует строгую структурированность, но при этом позволяет достаточно лаконично реализовать арифметические и циклические конструкции.

Для визуализации структурных различий между языками по объединенному корпусу каждого исследуемого языка построены графовые модели на уровне токенов (рис. 1 и рис. 2). Узлы графа соответствуют токенам, а ребра отражают переходы между последовательными токенами в коде; вес ребра равен частоте такого перехода в корпусе из пяти типовых программ. В качестве иллюстрации ниже приведены графы для языка C (рис. 1) и для Brainfuck (рис. 2) как двух контрастных систем.

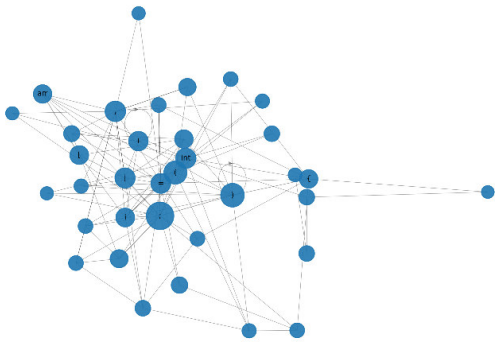

Рисунок 1 – Граф переходов токенов для языка C

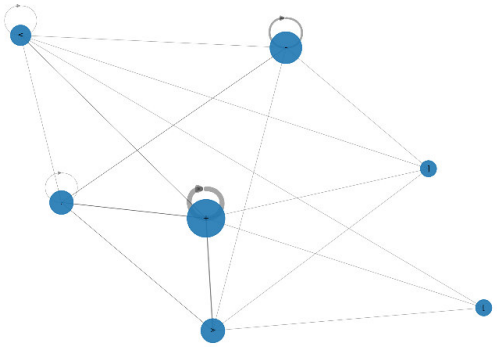

Рисунок 2 – Граф переходов токенов для языка Brainfuck

Сравнение графов наглядно иллюстрирует разницу в организации кода: у языка C сеть переходов более разветвленная, с несколькими «узлами», через которые проходит много связей, что соответствует более разнообразному синтаксису. У языка Brainfuck связи сосредоточены вокруг небольшого набора символов, и сама структура графа получается заметно проще, что напрямую согласуется с низкими значениями энтропии и небольшим числом уникальных токенов.

Энтропийный анализ подтверждает общую зависимость между минимализмом языка и разнообразием синтаксических конструкций. Визуализировать эти результаты можно с помощью диаграмм распределения токенов и графов связей между конструкциями, что позволит выявить скрытые закономерности использования языковых элементов и провести более глубокий сравнительный анализ ([5, с. 6], [12, с. 15]).

Таким образом, сопоставление семи языков программирования на основе корпуса типовых примеров демонстрирует, что различия между языками проявляются не только в уровне абстракции и стиле синтаксиса, но и в структурной вариативности, которую можно количественно оценить через число токенов, уникальные токены и энтропию Шеннона. Эти результаты формируют основу для дальнейшей визуализации и построения интерактивной карты языков, которая станет следующим шагом исследования ([14, с. 73], [9, с. 4–5]).

Полученные результаты показывают, что функциональные и когнитивные особенности языков программирования тесно связаны с их синтаксической структурой и лексической вариативностью. Хотя вариативность таких языков ограничена, именно это делает их удобным материалом для анализа базовых принципов вычислений и восприятия формального синтаксиса. Работа с Assembly и Brainfuck позволяет сосредоточиться на элементарных операциях и проследить, как минимальный набор команд влияет на когнитивную обработку программного текста ([8, с. 138], [13, с. 7]).

Иную картину дают Python и Ruby. Эти языки ориентированы на выразительность и читаемость, при этом не перегружены жесткими синтаксическими ограничениями. По мере усложнения алгоритмов их энтропийные показатели возрастают, что отражает способность языков передавать вычислительные процессы в сжатой, но достаточно гибкой форме [6, с. 431–432]. Именно поэтому Python и Ruby широко используются для быстрого прототипирования, учебных задач и моделирования языковых структур.

Языки Fortran, C и ALGOL занимают промежуточное положение. Они сохраняют формальную строгость, но при этом допускают разнообразие конструкций, особенно при реализации алгоритмов сортировки и рекурсивных вычислений. Высокие значения энтропии в таких примерах указывают на то, что данные языки позволяют эффективно описывать алгоритмы, не теряя предсказуемости и логической прозрачности кода [7, с. 48].

Сопоставление семи языков по числу токенов, уникальным токенам и энтропии выявляет устойчивую закономерность: чем выше уровень абстракции и выразительности языка, тем больше разнообразие токенов при сопоставимом объеме кода. Минималистичные языки демонстрируют узкий синтаксический спектр, что делает их показательными для экспериментов по когнитивному восприятию и формальным ограничениям [8, с. 137].

Интерпретация этих данных имеет практическое значение для цифровой лингвистики и компьютерного образования. Количественные метрики, такие как энтропия Шеннона и частотный анализ токенов, позволяют выявлять структурные закономерности в коде, сравнивать языки по когнитивной нагрузке и предсказуемости, а также оценивать эффективность различных парадигм программирования в обучении ([9, с. 5], [14, с. 73]).

Таким образом, сопоставление языков программирования демонстрирует как различия между минималистичными и высокоуровневыми языками, так и сходства между среднеуровневыми системами, что формирует основу для построения визуальных моделей и интерактивных карт. Предварительные количественные показатели уже позволяют выявить ключевые «узлы» языковых структур, которые станут объектом визуализации на следующем этапе исследования.

Выводы. Проведенное сопоставительное исследование семи языков программирования показало четкую зависимость между уровнем абстракции языка, разнообразием синтаксических конструкций и когнитивной нагрузкой, измеряемой через токенизацию и энтропию Шеннона. Для языков низкого уровня, таких как Assembly и Brainfuck, характерно ограниченное число уникальных токенов и достаточно низкий показатель энтропии. Эти факторы демонстрируют их минималистический характер и строгие формальные ограничения. Для языков высоко уровня (Python и Ruby), наоборот, характерны большая вариативностью токенов и высокая энтропия. Именно это позволяет использовать их для прототипирования, а также образовательных и исследовательских целей. Что касается среднеуровневых языков, включая Fortran, C и ALGOL, они находятся где-то между строгостью синтаксиса и разнообразием конструкций, за счет чего появляются предсказуемость и выразительность кода.

Полученные в результате исследования количественные показатели позволяют сделать вывод, что различия и сходства между языками программирования можно измерять через формализованные метрики. Благодаря этому появляются возможности для системного анализа языковых структур и когнитивных характеристик, которые связаны с их освоением и применением.

Сопоставительный анализ семи языков программирования свидетельствует о том, что искусственные языки как объект исследования позволяют выявлять закономерности формального и когнитивного характера, что имеет прикладное значение в цифровой лингвистике, образовательной практике и изучении взаимодействия человека с языками программирования.

ЛИТЕРАТУРА

1. *Зернова А. С., Маркова Т. С.* Окказионализмы как основа искусственно придуманных языков // Филологический аспект. – 2024. – № 4(108). – С. 93–97.

2. *Мельчук И. А.* Опыт теории лингвистических моделей «смысл – текст». – М. : Языки славянской культуры, 1999. – 371 с.

3. *Морозов Д. О.* Априорный искусственный язык: реальность или фикция // Лингвокультурология. – 2010. – № 4. – С. 113–127.

4. *Романов А. В., Домрачева А. И., Гречишников Н. В., Артемьев И. В.* Искусственные языки для общения между людьми // Филологический аспект. – 2023. – № 2(94). – С. 66–72.

5. *Crafa S.* Modelling the evolution of programming languages // arxiv.org. 2015. – URL : https://arxiv.org/pdf/1510.04440 (дата обращения: 08.09.2025).

6. *Goodall G.* Constructed languages // Annual Review of Linguistics. – 2023. – Vol. 9. – P. 419–437.

7. *He Y.* A comparative study of the significance of different programming languages // Academic Journal of Computing and Information Science. – 2025. – Vol. 8, Iss. 2. – P. 45–50.

8. *Kuhn T.* A survey and classification of controlled natural languages // Computational Linguistics. – 2024. – Vol. 40, № 1. – P. 121–170.

9. *Malik-Moraleda S.* Constructed languages are processed by the same brain mechanisms as natural languages // Proceedings of the National Academy of Sciences of the United States of America. – 2025. – Vol. 122, № 12. – P. 1–11.

10. *Nanz S., Furia C.* Comparative study of programming languages in Rosetta Code // IEEE/ACM 37th International Conference on Software Engineering. – Florence : IEEE, 2015. – P. 1–12.

11. *Novikov P. N.* Constructed languages as semantic and semiotic systems // RUDN Journal of Language Studies, Semiotics and Semantics. – 2022. – Vol. 13, № 2. – P. 455–467.

12. *Sanders N.* A primer on constructed languages // Language Invention in Linguistics Pedagogy / ed. by J. Punske, N. Sanders, A. V. Fountain. – Oxford : Oxford University Press, 2020. – P. 6–26.

13. *Singer J., Draper S.* Let's take esoteric programming languages seriously // Onward! '25. – Singapore : ACM SIGPLAN, 2025. – P. 1–14.

14. *Weiss D. J.* Introduction: The use of artificial languages in bilingualism research // Bilingualism: Language and Cognition. – 2020. – Vol. 23, Iss. 1. – P. 72–73.

Статья поступила в редакцию 18.01.2026

REFERENCES

1. Zernova A. S., Markova T. S. Okkazionalizmy kak osnova iskusstvenno pridumannyh yazykov // Filologicheskij aspekt. – 2024. – № 4(108). – S. 93–97.

2. Mel'chuk I. A. Opyt teorii lingvisticheskih modelej «smysl – tekst». – M. : Yazyki slavyanskoj kul'tury, 1999. – 371 s.

3. Morozov D. O. Apriornyj iskusstvennyj yazyk: real'nost' ili fikciya // Lingvokul'turologiya. – 2010. – № 4. – S. 113–127.

4. Romanov A. V., Domracheva A. I., Grechishnikov N. V., Artem'ev I. V. Iskusstvennye yazyki dlya obshcheniya mezhdu lyud'mi // Filologicheskij aspekt. – 2023. – № 2(94). – S. 66–72.

5. Crafa S. Modelling the evolution of programming languages // arxiv.org. 2015. – URL : https://arxiv.org/pdf/1510.04440 (дата обращения: 08.09.2025).

6. Goodall G. Constructed languages // Annual Review of Linguistics. – 2023. – Vol. 9. – P. 419–437.

7. He Y. A comparative study of the significance of different programming languages // Academic Journal of Computing and Information Science. – 2025. – Vol. 8, Iss. 2. – P. 45–50.

8. Kuhn T. A survey and classification of controlled natural languages // Computational Linguistics. – 2024. – Vol. 40, № 1. – P. 121–170.

9. Malik-Moraleda S. Constructed languages are processed by the same brain mechanisms as natural languages // Proceedings of the National Academy of Sciences of the United States of America. – 2025. – Vol. 122, № 12. – P. 1–11.

10. Nanz S., Furia C. Comparative study of programming languages in Rosetta Code // IEEE/ACM 37th International Conference on Software Engineering. – Florence : IEEE, 2015. – P. 1–12.

11. Novikov P. N. Constructed languages as semantic and semiotic systems // RUDN Journal of Language Studies, Semiotics and Semantics. – 2022. – Vol. 13, № 2. – P. 455–467.

12. Sanders N. A primer on constructed languages // Language Invention in Linguistics Pedagogy / ed. by J. Punske, N. Sanders, A. V. Fountain. – Oxford : Oxford University Press, 2020. – P. 6–26.

13. Singer J., Draper S. Let's take esoteric programming languages seriously // Onward! '25. – Singapore : ACM SIGPLAN, 2025. – P. 1–14.

14. Weiss D. J. Introduction: The use of artificial languages in bilingualism research // Bilingualism: Language and Cognition. – 2020. – Vol. 23, Iss. 1. – P. 72–73.

The article was contributed on January 18, 2026

Сведения об авторах

Мкртчян Тамара Юрьевна – кандидат филологических наук, доцент кафедры лингвистики и профессиональной коммуникации института филологии, журналистики и межкультурной коммуникации Южного федерального университета, г. Ростов-на-Дону, Россия, https://orcid.org/0000-0002-0156-7178, tymkrtchyan@sfedu.ru

Макарова Жанна Валерьевна – бакалавр направления «Интеллектуальные системы в гуманитарной сфере» Южного федерального университета, г. Ростов-на-Дону, Россия, https://orcid.org/0009-0009-3798-254x, zmakarova@sfedu.ru

Author Information

Mkrtchyan, Tamara Yuryevna – Candidate of Philology, Associate Professor of the Department of Linguistics and Professional Communication, Southern Federal University, Rostov-on-Don, Russia, https://orcid.org/0000-0002-0156-7178, tymkrtchyan@sfedu.ru

Makarova, Zhanna Valeryevna – Bachelor's Student in Intellectual Systems in the Humanities, Southern Federal University, Rostov-on-Don, Russia, https://orcid.org/0009-0009-3798-254x, zmakarova@sfedu.ru