5. Шабанова, И. А. Основные подходы обучения диалогической речи на иностранном языке / И. А. Шабанова // Вестник научных конференций. – 2019. – № 9-1(49). – С. 133–134.

6.Олимова, Д. Ш. Психология усвоения иностранного языка на среднем этапе обучения / Д. Ш. Олимова, Н. К. Адамбаева // Научный альманах. – 2016. – № 2(16). – С. 191–195.

7.Пассов, Е.И. Урок английского языка в средней школе / Е.И. Пассов. – М. : Просвещение, 1988. – 324 с.

8.Мильруд, Р.П. Современный методический стандарт обучения иностранным языкам в школе / Р.П. Мильруд // Иностранные языки в школе. – 1996. – № 1. – С. 5–12.

9.Гальскова, Н. Д. Современная методика обучения иностранным языкам : пособие для учителя / Н. Д. Гальскова. – Москва : Аркти-Глосса, 2000. – 165 с.

10.Ваулина, Ю. Е. Английский язык. 5 класс : учебник для общеобразовательных организаций / Ю. Е. Ваулина, Д. Дули, О. Е. Подоляко, В. Эванс. – Москва : Express Publishing : Просвещение, 2026 – 168 с.: ил. – (Английский в фокусе).

11.Ваулина, Ю. Е. Английский язык. 6 класс : учебник для общеобразовательных организаций / Ю. Е. Ваулина, Д. Дули, О. Е. Подоляко, В. Эванс. – Москва : Express Publishing : Просвещение, 2026 – 144 с.: ил. – (Английский в фокусе).

METHODOLOGICAL PRINCIPLES OF TEACHING DIALOGICAL SPEECH DURING FOREIGN LANGUAGE LESSONS AT SECONDARY SCHOOLS

Vorobeva I. A., Mikhailova M. S.
Saint Petersburg, Russia
RSHU

Summary. *The article explores methods for teaching dialogical speech at secondary schools, examining theoretical bases, current approaches and practical exercises to foster students' communicative skills during foreign language classes.*

Keywords: *dialogical speech; foreign language teaching; secondary school; communicative competence; system of exercises.*

УДК 811.161.1'27'28:821.161.1-31

ВЛИЯНИЕ ЕСТЕСТВЕННЫХ ЯЗЫКОВ НА ХАРАКТЕР СЕМИОЗИСА В ИСКУССТВЕННЫХ ЯЗЫКАХ ПРОГРАММИРОВАНИЯ

Галимова Ю. О., Иркабаева З. Р.
Уфа, Российская Федерация
БАГСУ при главе РБ

Аннотация. *Авторы анализируют искусственный язык программирования Python как знаковую систему, интегрирующую влияние естественных языков в синтаксис, семантику и прагматику. Проанализированы знаковые средства, роль интерпретаторов и норм сообщества. Продемонстрировано, как механизмы, наблюдаемые в естественных языках, проявляются в развитии языков программирования.*

Ключевые слова: *семиозис; искусственные языки; Python; лингвистика программирования.*

В условиях цифровой трансформации общества программирование является не просто техническим инструментом, но приобретает статус ключевой когнитивной и коммуникативной практики. Количество практикующих программистов активных пользователей языка Python исчисляется миллионами и ежегодно растет. При этом, языки программирования, будучи изначально формальными системами, в реальности функционируют как знаковые структуры, подверженные интерпретации, переформулировке и переосмыслению в профессиональных сообществах – они активно развиваются, трансформируются и приобретают новые роли, подобно естественным языкам [2]. Уникальность искусственных языков программирования состоит в том, что, хотя они и создаются как технический инструмент, обеспечивающий работу современных информационных технологий, их носителем и пользователем является человек, программист. Так, языки программирования обслуживают не столько машину, но в первую очередь человека, вступая в сложные отношения с естественным языком, культурным контекстом и когнитивными стратегиями программиста. Это превращает искусственные языки в значимые объекты семиотического анализа [3].

При этом семиозис кода представляет собой распределённый процесс, в котором участвуют не только автор и машина, но и коллективное сообщество как культурный носитель норм интерпретации. На примере языка программирования Python мы отчетливо видим как интерпретация кода зависит от социально закреплённых норм, формируемых профессиональным сообществом разработчиков. Эти нормы могут быть формально закреплены в нормативной документации языка, например рекомендации PEP, «Zen of Python», а так же существуют и корректируются сообществом пользователей посредством негласных правил «питоничности».

Кроме того для Python характерна жанровая организация кода (библиотека, тест, скрипт, документация), которая обнаруживает структурные и институциональные признаки характерные и для естественных языков, в том числе стабильность формы, функциональную направленность, нормативную закреплённость, специфические лексико-синтаксические маркеры, социальную воспроизводимость, предсказуемость интерпретации и рефлексивное описание в метатекстах. Это сближает жанры программного кода с жанрами естественных языков и подтверждает их семиотическую сопоставимость

Язык программирования Python, как и естественные языки, оперирует категориями, которые можно соотнести с понятием частей речи [1]. При этом в Python, благодаря философии «читаемости» и близости к естественным языковым структурам, это сопоставление становится особенно наглядным. Python намеренно стремится к тому, чтобы синтаксис был ближе к естественному языковому мышлению. Например, конструкция цикла, которую на языке Python можно записать следующим образом:

for item in items:

print (item)

почти дословно соответствует естественному языковому выражению: "для каждого элемента в коллекции — напечатать элемент".

В ходе исследования было продемонстрировано, что искусственные языки программирования, включая Python, обладают полноценной семиотической структурой и подлежат анализу на синтаксическом, семантическом и прагматическом уровнях [2]. Это позволяет рассматривать программирование не только как техническую, но и как когнитивную и коммуникативную деятельность. Анализ языка Python также показал, что его синтаксис и структурные маркеры, такие как отступы, соглашения об именах и встроенная документация, функционируют как знаковые средства, чьё значение формируется в конкретных практиках программирования и зависит от культурного контекста сообщества разработчиков. Проведенный анализ позволяет конкретизировать способы семантического образования терминологии в языках программирования и специфику с учётом расширения её роли в современных языках программирования. В результате исследования были выделены следующие способы:

1. Логическая связь с предметом реального мира: a. Логическая связь с действием, которое можно осуществить с предметом реального мира. b. Логическая связь с признаками, которые присущи предмету реального мира.

2. Визуальная связь с предметом реального мира (образование визуальных метафор).

Понимание принципов и способов семантического формирования терминов позволяет разработчикам понимать использование определённых терминов в разных контекстах. Это помогает им лучше понимать сложные системы и проекты, а также принимать взвешенные решения о том, как называть те или иные компоненты кода. Понимание роли в передаче лингвистических значений естественного языка в искусственных языках в сфере информационных технологий в динамическом процессе обновления IT – терминологии в современном мире может быть полезна в практическая плане для специалистов IT-сферы.

Естественный язык системно влияет на формирование и функционирование Python: английская лексика служит основным источником номинации ключевых слов, встроенных функций и идентификаторов, обеспечивая когнитивную прозрачность и снижая порог вхождения. Номинативные практики при этом воспроизводят модели, присущие естественным языкам:

1) Именование по наблюдаемым свойствам.

Имена фиксируют броские, «диагностические» признаки объекта и часто несут оценку. В инженерной практике это проявляется в именах-метафорах и «именах-диагнозах», которые не только идентифицируют сущность, но и маркируют её качество/дефект; тем самым имя выступает знаком с денотативной и аксиологической составляющими.

2) Именование «от вещи» (метафоризация).

Широко используется перенос материальных образов на абстрактные цифровые объекты (конвейер, стена, шина и т. п.), что соответствует модели «от вещи». Такая метафора обеспечивает семантическую компрессию и интуитивную доступность термина, выступая маркером профессиональной компетенции.

3) Именование «от прецедента» (культурный код).

Аллюзии на научные, мифологические и массово-культурные тексты создают компактные, высокоузнаваемые термины; понимание аллюзии выполняет функцию групповой идентификации сообщества разработчиков. Эти имена конденсируют сложные технические характеристики в кратких метках и одновременно «социализируют» термин.

4) Именование «от притчи» (нарративная маркировка).

Имена фиксируют уникальные события разработки (горячие исправления, временные обходы), превращаясь в «память проекта». Они задают будущую интерпретацию фрагмента и служат внутренним навигационным механизмом команды (прагматический слой номинации).

Семиотически важны и «ловушки переноса»: совпадающие формы естественного и программного языка могут иметь различную семантику (связка в английском vs. оператор идентичности в Python), что требует институциональных норм и контекста для стабилизации интерпретации. В целом номинативные практики Python воспроизводят универсальные когнитивные механизмы естественного языка—метафоризацию, культурную кодированность, нарративизацию, показывая, что даже в формализованной знаковой системе смысл конструируется по правилам социальной и культурной коммуникации

Процесс семиозиса в языке Python представляет собой многоуровневое взаимодействие между знаками кода, интерпретируемыми объектами и субъектом-интерпретатором. При этом интерпретант возникает как результат когнитивной обработки кода в зависимости от целей, задач и опыта программиста.

Исследование подтвердило важную роль профессиональных сообществ в формировании правил интерпретации кода, в частности через норму «питоничности», стандарты документации (PEP) и коллективные практики рецензирования. Эти нормы формируют прагматическую рамку, в которой осуществляется интерпретация программного текста. Наши результаты подтверждают тезисы ранних исследований [4; 5: 6] о том, что интерпретация кода программистами сильно зависит от их опыта, участия в профессиональных сообществах и степени знакомства с негласными "культурными кодами" языка. Например, новичок может синтаксически верно написать рабочий код, который, тем не менее, будет отвергнут на code review за нарушение норм "питоничного" стиля — даже если интерпретатор Python его успешно исполняет.

Формируются своеобразные метаязыковые практики, в которых программисты обсуждают не только "работает ли код", но и "насколько он соответствует принятым культурным стандартам". Появляются такие категории, как "чистый код", "антипаттерны", "питоничность", которые невозможно полностью вывести из грамматики языка — они формируются и поддерживаются именно сообществом. Кроме того, сами сообщества выступают как механизм эволюции языка. Роль вторичных знаковых систем — таких как стиль оформления, именование и встроенная документация — в создании прагматической читаемости и понятности кода так же очень высока. Эти элементы рассматриваются как визуальные и символические маркеры, регулирующие интерпретацию текста программы.

Исследования ментальных процессов, сопровождающих понимание программных конструкций [4] с опорой на данные когнитивной психологии и eye-tracking исследований, демонстрируют, как структура и стиль кода влияют на скорость и точность его восприятия. Показано, что код воспринимается не линейно, а блоками, а «питоничность» кода снижает когнитивную нагрузку.

Вопрос "является ли код питоничным?" регулярно обсуждается в сообществе и на форумах (Stack Overflow, Reddit r/Python).

Например, конструкция, представленная ниже формально корректна:

```
if len(my_list) = = 0:
      print( “Empty list”)
```

Но опытный программист пользователь искусственного языка оставляет в коде корректирующий комментарий: «Лучше использовать питоничную идиому:

```
if not my_list:
       print (“Empty list”)»
```

Хотя поведение кода идентично в обоих случаях, из комментария мы можем заключить, что профессиональное сообщество воспитывает у программиста ожидание "читаемого", "питоничного" стиля. Таким образом, норма задаётся не языком, а сообществом. Этот пример наглядно показывает, что в Python допустимо множество способов выразить одно и то же действие, и выбор способа часто зависит от прагматических установок — читаемости, краткости, эстетики, принадлежности к определённой профессиональной культуре. Внешний вид текста становится носителем метасмысла, сигнализируя другим программистам о квалификации и опыте автора, согласованности коллектива, качестве менеджмента и зрелости управленческих практик в команде, задействованной в разработке кода. На этом основании код предстает не только как формальная инструкция для машины, но и как знак, включённый в коммуникативное пространство профессионального сообщества.

Эмпирический анализ отрывков кода в репозиториях и комментариев пользователей к ним показывает, что интерпретатором кода выступает не абстрактная машина, а человек, обладающий предзаданной системой категорий, ожиданий и прагматических целей. Программист не только читает код, но и придаёт ему значение в зависимости от задач, среды и личного опыта. Язык Python следует рассматривать через призму «сообществ практики», где нормы и жанры закрепляются в ходе коллективной языкотворческой деятельности. В этой рамке «Дзен Python» (PEP 20) выступает метанормативным текстом сообщества: его афоризмы и лексемы — *import this*, *Zen of Python*, *pythonic*, *PEP 8*, *The Python Way*, «Readability counts.» — систематически используются в дискуссиях как аргументы оценки и руководства к действию. Эти единицы выполняют три функции: (1) деонтическую (предписание: *“import this and meditate on it”*), (2) оценочную (*beautiful, readable, intelligible*), (3) идентификационную (маркирование принадлежности к норме *pythonic*). Сообщество одновременно фиксирует «дух» и «букву» нормы: *“Think of it as PEP 8 is the letter; the Zen is its spirit”*; при этом подчёркивается, что «*it should never be taken as rules*», а *“each entry … gets you to THINK”* задаёт рефлексивный режим интерпретации. Институционально идиомы из «Дзена» цитируются в заметках к коду как обоснование для внесения правок в код программного обеспечения, даже при условии отсутствия проблем с функциональными характеристиками кода, что подтверждает его нормативный статус.

Анализ таких идиом выявил три ценностно-идеологических блока: (1) эстетико-семиотический (*Beautiful is better than ugly; Simple…; Readability counts*), нормирующий визуально-композиционные свойства кода; (2) когнитивно-герменевтический (*Explicit is better than implicit; one obvious way; refuse the temptation to guess*), снижающий интерпретативную неопределённость; (3) практико-идеологический (*Practicality beats purity; Errors should never pass silently; If the implementation is easy to explain…; Namespaces…*), институирующий прагматизм, дисциплину ошибок и требование объяснимости.

Частотный срез по корпусу (n = 200) подтверждает трёхконтурную организацию «социальной семиотики» Python-сообщества: доминирует эстетико-семиотический блок (48 %), далее когнитивно-герменевтический (31 %) и практико-идеологический (21 %). Тем самым PEP 20 функционирует не как «кодекс правил», а как набор интерпретативных опор, через которые сообщество согласует значения «питоничности» и нормативные ожидания читаемости, явности и прагматической достаточности.

Языки программирования демонстрируют структурное сходство с естественными языками, поскольку обладают лексикой, синтаксисом, семантикой и прагматикой употребления. Их знаки организованы в системы правил, допускают порождение потенциально неограниченного числа высказываний-программ и функционируют в сообществе носителей — разработчиков, которые формируют нормы стиля, жанры, идиомы и критерии «правильного» кода. Это делает искусственные языки программирования перспективным объектом сопоставительного анализа и позволяет рассматривать их через призму общей теории языка.

Особая значимость таких языков состоит в том, что они дают возможность наблюдать процессы смыслообразования и языковой эволюции в более формализованной и эмпирически фиксируемой среде. Изменение синтаксиса, появление новых идиом, закрепление стилевых норм и институционализация стандартов в сообществах разработчиков позволяют проследить, как языковая система развивается под влиянием когнитивных, коммуникативных и социальных факторов. В этом смысле анализ языков программирования не только расширяет объектную область сравнительного языкознания, но и углубляет теоретическое понимание механизмов развития языков как знаковых систем.

Полученные результаты имеют важное прикладное значение на стыке лингвистики, теории языка и программирования, делая шаг на встречу развитию языковых компетенций в преподавании программирования, проектировании языков и интерфейсов, а также в исследовании цифровой лингвистики и когнитивных систем. Исследование продемонстрировало перспективность применения семиотического анализа к цифровым текстам и позволяет переосмыслить теоретические границы понимания языка не только как коммуникативного средства между людьми, но и в качестве медиатора между человеком и техникой.

Список литературы

1. Бурмашева, Н. В. Лингвистические основы языка программирования С : учеб. пособие / Н. В. Бурмашева. – Улан-Удэ : Изд-во ВСГУТУ, 2010. – 120 с. – ISBN 978-5-89610-201-3.

2. Камалова, Н. И. Роль семиотического подхода в обучении языкам программирования / Н. И. Камалова, А. Ш. Самадов // International Conference on Educational Discoveries and Humanities : материалы Междунар. науч. конф., Plano, Texas, USA, 1 Dec. 2023. – Plano : Econference Series, 2023. – С. 82–87.

3. Мартынец, М. С. Естественные и искусственные языки: общее, особенное и единичное / М. С. Мартынец // Профессиональное образование в современном мире. – 2011. – № 2. – С. 69–73.

4. Eye Tracking in Computing Education / T. Busjahn, R. Bednarik, A. Begel [et al.] // Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education. – 2015. – P. 3–10.

5. Scalabrino, S. A Comprehensive Model for Code Readability / S. Scalabrino, M. Linares-Vásquez, R. Oliveto, D. Poshyvanyk // Journal of Software: Evolution and Process. – 2018. – Vol. 30, № 6. – Art. e1958. – DOI: 10.1002/smr.1958.

6. Zlatev, J. Cognitive Semiotics: An Emerging Field for the Transdisciplinary Study of Meaning / J. Zlatev // Public Journal of Semiotics. – 2012. – Vol. 4, № 1. – P. 2–24. – DOI: 10.37693/pjos.2012.4.8837.

INFLUENCE OF NATURAL LANGUAGES ON THE CHARACTER OF SEMIOSIS IN ARTIFICIAL PROGRAMMING LANGUAGES

Galimova Yu. O., Irkabaeva Z. R.
Ufa, Russian Federation
BAGSU under the Head of RB

Abstract. *The authors analyze the artificial programming language Python as a sign system integrating the influence of natural languages into its syntax, semantics, and pragmatics. The study examines sign structures, the role of interpreters, and the influence of community norms. It demonstrates how mechanisms characteristic of natural languages are reflected in the development and functioning of programming languages.*

Keywords: *semiosis; artificial languages; Python; programming linguistics.*

УДК: 81'27 + 004.056

СЕМАНТИЧЕСКИЕ ФАНТОМЫ В ПРОФЕССИОНАЛЬНОЙ РЕЧИ СТУДЕНТОВ, ОБУЧАЮЩИХСЯ ПО СПЕЦИАЛЬНОСТИ «КИБЕРБЕЗОПАСНОСТЬ»

Гончарова С. А.
Витебск, Республика Беларусь
ВГУ имени П. М. Машерова

Аннотация*. В статье рассматривается проблема семантических фантомов — англоязычных заимствований, которые присутствуют в лексиконе студентов, но не вызывают внутреннего зрительного образа. На материале эксперимента с участием студентов факультета математики и информационных технологий (специальность «Компьютерная безопасность») выявляется разница в образном наполнении англицизмов и их русских аналогов.*

Ключевые слова*: семантический фантом, экология языка, англицизм, опора на родной язык, психолингвистика.*

Введение. Современное языковое образование сталкивается с вызовом массового проникновения англицизмов в профессиональную речь студентов. Студенты активно используют слова, заимствованные из английского языка, определяя значение, ориентируясь на контекст, часто не всегда верно. Семантические фантомы представляют собой заимствованные слова из английского языка, которые функционируют наряду с русскими, но не имеют образа либо ассоциаций в сознании носителя. Кроме этого, многие заимствованные слова включаются в синонимические ряды с уже употребляющимися в современном литературном языке словами [1]. При этом студенты отда-