УДК 001

Python: роль языка программирования в разработке философских концепций на примере философии Р. Карнапа

Агошкова Татьяна Валерьевна

Преподаватель,
Московский авиационный институт,
125993, Российская Федерация, Москва, Волоколамское ш., 4;
e-mail: agoshkovatv@mai.ru

Аннотация

В данной статье исследуется взаимосвязь между языком программирования Python и философскими концепциями, в контексте логического позитивизма, разработанными Р. Карнапом. Автор рассматривает аспект применения программирования для формализации и анализа философских логических понятий. В статье подчеркивается, что проблемы логики в философии возникают в связи с неясностью языка, а также что логическая формальная интерпретация могут помочь устранить эту неопределенность, либо сгладить её. В статье затрагивается вопрос практического применения теории экстенсионала и интенсионала, подробно рассматриваемой Р. Карнапом в работе «Значение и необходимость». Данное исследование может быть расширено в дальнейшем с целью сравнительно-сопоставительного анализа натуральных языков при помощи философских теорий, а также моделирования когнитивных процессов, отраженных в философских трудах.

Для цитирования в научных исследованиях

Агошкова Т.В. Python: роль языка программирования в разработке философских концепций на примере философии Р. Карнапа // Контекст и рефлексия: философия о мире и человеке. 2024. Том 13. № 7А. С. 22-28.

Ключевые слова

Программирование, Python, интерпретация, экстенсионал, интенсионал, Р. Карнап.

Введение

Программа для создания кода Python, разработанная Гвидо ван Россумом, стала одним из самых популярных языков программирования благодаря своей простоте. Однако, успех Python связан не только с техническими характеристиками, но и с философией, которая лежит в основе языка. Дзен Python, созданный Тимом Петерсом, содержит 19 принципов, служащих основой для написания кода. Эти принципы подчеркивают важность простоты и ясности системы, что делает Python идеальным инструментом для реализации философских идей в программировании [Lindstrom, 2005]. Он может быть применен в различных сферах IT, а также там, где необходимо написать читаемый код. Принципы, изложенные в Дзене Python, помогают разработчикам принимать решения при проектировании программного обеспечения, избегая излишних сложностей и следуя идеям, предложенных Т. Петерсом. Например, такие принципы, как «Простое лучше, чем сложное», находят свое применение в создании интуитивно понятного интерфейса. Однако, эти идеи полезны не только в программировании, но и в смежных областях, как управление данными, автоматизация процессов и при исследовании натуральных языков, в том числе и логики языка.

Основное содержание

Рудольф Карнап, один из ведущих представителей логического позитивизма, считал, что проблемы логики в философии возникают зачастую из-за неясности и неоднозначности языка. Ученый подчеркивал важность формализации и логического анализа, позволяющего устранить языковые неопределенности [Карнап, 1959, с. 338-339]. Эти идеи Р. Карнапа имеют некоторые параллели с современными языками программирования, которые можно рассматривать также как одну из форм логической формализации. В данном контексте программирование может рассматриваться как способ формализации абстрактных понятий, что позволяет их определить и проанализировать на качественно новом уровне. К примеру, при разработке программного обеспечения программист создает модель, переводя абстрактные концепции такие как данные или процессы, в формальные конструкции языка программирования. Эти конструкции строго определены и поскольку их поведение внутри конкретной системы предсказуемо, то, соответственно, они поддаются анализу. Таким образом, программирование становится способом устранения неоднозначности и позволяет формализовать и исследовать абстрактные идеи, в том числе и выраженные при помощи логики философии.

Программирование на Python предоставляет уникальную возможность для реализации на практике философских концепций. Мы можем использовать код для создания моделей, которые иллюстрируют философские идеи и концепции, наделяя их признаками математической логики и затем представить их в виде набора показателей или коэффициента. Такой подход представляет философию не только как форму общественного сознания, но и как объект экспериментальных исследований, позволяющих верифицировать и визуализировать информацию посредством кода.

Для Р. Карнапа язык – это не столько средство общения, сколько инструмент для анализа и построения научных теорий. Ученый выделял язык науки, который приводит к мысли о единстве науки [Суровягин, 2012]. Чтобы избежать двусмысленностей и достигнуть единой картины мира, языку науки необходимо быть точным и формализованным, а также прибегнуть к единой методологии. То есть язык науки по Карнапу – идеализированный язык, построенный

на строгих логических принципах, обладающий определенными характеристиками, к которым мы можем отнести точность формулировок, формализованность и методологическая универсальность. Первый аспект связан с тем, что каждая концепция или высказывание в языке должны быть строго определены в целях исключения многозначности или субъективных интерпретаций. Формализованность означает способность быть основанным на логических и математических правилах в целях избегания неясностей и противоречий. Такой язык позволяет строить теории как логические системы, где каждое утверждение вытекает из правил вывода. Универсальность же соотносится с единством методов исследования абсолютно для всех областей знания, способствующего созданию единой картины мира.

Программирование на Python может создать формализованные структуры, которые могут служить моделями научных теорий [Черкашин, Орлова, 2017]. Например, мы можем создать определенные классы и функции, которые будут моделировать определенные философские концепции. В частности, Р. Карнап в своих работах вводит понятия экстенсионала и интенсионала, которые имеют параллели с практикой программирования на Python.

Экстенсионал соотносится с объектами или классами объектов, определяемых через набор элементов. В программировании на Python это может быть проиллюстрировано на примере списков или множеств, определяющих структуру данных. Они позволят хранить и обрабатывать множества объектов для создания более сложного кода [Карнап, 1959]. Экстенсионалы в программировании могут быть использованы для представления философских концепций, связанных с множествами, классификациями или принадлежностью, например, представляя категорию через конечный набор элементов.

Интенсиональность же касается понятий, определяемых через их содержание – выражаемое свойство. В программировании данный параметр будет соответствовать функциям и модулям, в которых акцент делается на логике и поведении, а не на конкретном наборе элементов [Там же]. Интенсионал в программировании проявляется, например, посредством объектно-ориентированного программирования, где взаимодействие моделей предметной области образуют иерархию исследования.

Так интенсионал и экстенсионал могут быть взаимодополняющими аспектами одного и того же исследования. Если в данной интерпретации экстенсионал определяет конкретный набор данных, с которыми работает программа, то интенсионал будет задавать логику обработки данных и определять, как данные будут интерпретироваться.

По мнению Р. Карнапа, многие философские проблемы возникают из-за неопределенности и возможного смешения интенсиональных и экстенсиональных понятий: «Любые подлинные философские проблемы, по мнению Рудольфа Карнапа, должны быть сведены к проблемам логики, которые, в свою очередь, сводимы исключительно к одним только проблемам синтаксиса» [Нехаев, 2018, с. 108]. При таком подходе метафизические вопросы будут бессмысленны, так как они не будут поддаваться эмпирической проверке, ни тем более логическому анализу. Например, вопрос, касающийся сущности понятия термина бытие, не имеет значения, если его невозможно выразить в терминах формализованного языка. В таком случае, философские проблемы можно свести к логическим проблемам, связанным с анализом процесса того, как формализуется, используется, а также интерпретируется язык. Например, вопрос о том, что является бытие, а что им не является, превращается в анализ синтаксических и семантических правил, которые будут использоваться для утверждения истинности высказываний.

Аналогично в программировании важно различать эти два аспекта, чтобы избежать ошибок

и создать более гибкие модульные и масштабируемые системы. Например, при проектировании классов в Python важно определить, какие атрибуты и методы должны быть определены как экстенсиональные - через определенные значения, а какие – интенсионально, то есть через абстрактные правила и логику. Целью Р. Карнапа была разработка металогики языка или «логического синтаксиса языка», позволяющего отделить теоретический язык от всего субъективного [Griftsova, Kozlova, 2024, с. 129]. Под синтаксисом Р. Карнап понимал правила, определяющие как строятся логические высказывания в языке и как из одних высказываний логически выводятся другие высказывания. В связи с этим Карнап считал, что философия должна отказаться от вопросов, которые не могут быть выражены в терминах логического синтаксиса и вместо изучения термина философия должна анализировать, как это понятие используется в языке и как его можно формально определить.

По мнению финского философа и математика Я. Хинтикка, главная задача, к которой стремится Р. Карнап, является безнадежной, поскольку невозможно систематически изучит отношение языка к миру или выражать подобные исследования в языке. Например, невозможно комплексно изучить то, как язык соотносится с реальностью, так как сам язык не способен адекватно выразить все аспекты этого отношения. Язык, как знаковая система, не может адекватно описать все аспекты реальности, поскольку натуральный язык ограничен в своих возможностях. Следовательно, всякая крупномасштабная теория моделей не может быть выражена в языке, в том числе и та, которую предлагал Р. Карнап [Hintikka, 1991]. Я. Хинтикка подчеркивает, что философские и логические исследования, направленные на исследование отношений языка и реальности, сталкиваются с фундаментальными ограничениями, которые усложняют исследования, в том числе и делает их невозможными.

Логично предположить, что концепция логики языка Р. Карнапа не может быть целиком и полностью реализуема в рамках положения о всеобщности языка – языка как универсального средства. Однако, когда мы переходим к концепции языка как инструмента систематического анализа и построения теорий, как это делал Карнап в своих работах, становится возможным варьировать интерпретацию языка и разрабатывать жизнеспособную модель языка. В таком случае, мы можем говорить как о попытках компьютерной интерпретации философских теорий, так и философии программирования в целом [Хорольский, Фесикова, 2018].

Таким образом, опыт программирования на языке Python может служить наглядной иллюстрацией философских идей, в том числе и логики Р. Карнапа об экстенсионале и интенсионале. Понимание этого различия на модели программирования способствует более глубокому пониманию роли языка и логики в решении как практических, так и теоретических философских проблем.

Создание кода в Python, основанного на философских идеях, может иметь значительное влияние на философию, сравнительно-сопоставительное языкознание и когнитивистику по многим причинам [Макулин, 2016; Копытова, 2016]. Во-первых, принципы программирования могут помочь в анализе и формулировании философских идей. Во-вторых, программирования может использоваться для анализа и сравнения натуральных языков в целях более глубокого изучения языковых структур и особенностей. В-третьих, использование программирования для моделирования когнитивных процессов, отраженных в философских трудах, может способствовать развитию когнитивных теорий, так как Python удобен для создания алгоритмов и моделей. На наш взгляд, язык программирования способен передать те логико-философские концепции, которые могут быть отражены в виде четкой и систематичной структуры. Python способен решить ряд актуальных прикладных задач обработки и анализа информации на

естественных языках, которые невозможно осуществить прибегая исключительно к эмпирическому анализу. Среди прочего, важными компонентами таких исследований являются библиотеки данных, которые представляют необходимые инструменты для построения более сложных моделей исследования, позволяющих максимально приблизить экспериментальный результат к логическим исследованиям, проведенным в рамках теоретизированных логических структур.

При этом исследования данного типа могут столкнуться с некоторыми трудностями, в том числе указанными нами выше. Речь идет в первую очередь о семантических трудностях, которые влияют на интерпретируемость философских идей на практике. В том числе и сам Карнап пишет в труде «Значение и необходимость», что приводит в своих исследованиях только тот набор правил и закономерностей, который с его точки зрения будет существенным для данной работы [Карнап, 1959, с. 30]. Также исследования логической концепции могут иметь ограничения для применения в конкретных областях знания. То есть в данном случае при понимании языка как строгой системы мы отходим от понятия произвольности знака, которое установил де Соссюр. Знак представляется как знаковое выражение, что фактически сводит его к односторонней сущности - форме знака. Кроме этого, Р. Карнап исключает значение из научного исследования, сосредоточившись лишь на синтаксисе и формальных свойствах языка, что в целом также усложняет всесторонний и всеохватывающий характер исследований общей системы языка Р. Карнапа [Тренихина, 2018]. Такое сравнение языкового знака отражает переход от гуманитарного подхода к исследованию его природы к математическому.

Заключение

На наш взгляд, при всех ограничениях, связанных с реализацией на практике данной концепции построения логики языка в Python, создаются новые перспективы для междисциплинарных исследований в области философии языка, языкознания и программирования на Python. Создание новых методов позволит стимулировать развитие новых теорий и подходов при исследовании уже существующих, но недостаточно изученных философских теорий. Основой методологии такого логико-философского исследования будет являться сравнение натуральных языков по синтаксическим характеристикам, что позволит выявить закономерности и различия существования натуральных языков и создание новой классификации, в соответствии с исследованиями, Р. Карнапа.

Библиография

1. Lindstrom G. Programming with python //IT professional. – 2005. – Т. 7. – №. 5. – С. 10-16.
2. Карнап Р. Значение и необходимость. Исследование по семантике и модальной логике/ Перевод Н.В. Воробьева; Общая ред. Д.А. Бочвара; предисл. проф. С.А. Яновской. – М.: Изд-во иностр. лит. 1959. – 382 с.
3. Суровягин Д. П. Физикалистский редукционизм Рудольфа Карнапа // Вестник Пермского университета. Философия. Психология. Социология. – 2012. – №2. – С.87-97.
4. Черкашин Е. А., Орлова И. В. Инструментарий создания цифровых архивов документов на основе связанных данных // Современные технологии. Системный анализ. Моделирование. – 2017. – №4 (56). – С. 100-107.
5. Нехаев А. В. Рудольф Карнап и «Вавилонские башни» мира логики // ОНВ. ОИС. – 2018. – №2. – С. 106-110.
6. Griftsova I., Kozlova N.Y. Rudolf Carnap's Ideas in Philosophy of Language in the Context of Conceptual Engineering // Epistemology & Philosophy of Science. – 2024. – С. 122-133.
7. Hintikka J. Carnap, the Universality of Language and Extremality Axioms // Erkenntnis Orientated: A Centennial Volume for Rudolf Carnap and Hans Reichenbach/ ed. W. Spohn. — Dordrecht: Springer Netherlands. – 1991. – P. 325-336.
8. Хорольский О.С., Фесикова О.В. Философские аспекты программирования / О. С. Хорольский, О. В. Фесикова

// Научный альманах. – 2018. – № 4-2(42). – С. 210-213.
9. Макулин А. В. Интеллектуальные системы в гуманитарной сфере и цифровая философия // Вестник Северного (Арктического) федерального университета. Серия: Гуманитарные и социальные науки. – №2. – 2016. – С. 76-86.
10. Копытова М.А. Актуальность языка программирования Python// Экономика и социум. – №10(29). – 2016. – С. 943-945.
11. Тренихина М. Физикалистская модель сознания: искусственный интеллект в рамках классической теории знака // Парадигма: философско-культурологический альманах. – 2018. – № 28. – С. 21-26.

Python: the role of a programming language in the development of philosophical concepts using the example of R. Carnap's philosophy

Tat'yana V. Agoshkova

Lecturer,
Moscow Aviation Institute,
125993, 4, Volokolamskoe ave., Moscow, Russian Federation;
e-mail: agoshkovatv@mai.ru

Abstract

This article explores the interdependence between the Python programming language and philosophical concepts, particularly in the context of logical positivism developed by R. Carnap. The author examines the aspect of applying programming to formalize and analyze philosophical logical concepts. The article highlights that logical problems in philosophy arise due to the ambiguity of language, and that logical formal interpretation can help eliminate or reduce this uncertainty. The article addresses the practical application of the theory of extensionality and intensionality, thoroughly discussed by R. Carnap in his work *Meaning and Necessity*. This research can be extended further to facilitate a comparative analysis of natural languages using philosophical theories, as well as modeling cognitive processes reflected in philosophical works.

For citation

Agoshkova T.V. (2024) Python: rol' yazyka programmirovaniya v razrabotke filosofskikh kontseptsii na primere filosofii R. Karnapa [Python: the role of a programming language in the development of philosophical concepts using the example of R. Carnap's philosophy]. *Kontekst i refleksiya: filosofiya o mire i cheloveke* [Context and Reflection: Philosophy of the World and Human Being], 13 (7A), pp. 22-28.

Keywords

Programming, Python, interpretation, extension, intension, R. Carnap.

References

1. Lindstrom, G. (2005). Programming with Python. IT Professional, 7(5), 10-16.
2. Carnap, R. (1959). Meaning and necessity: A study in semantics and modal logic (N.V. Vorobyov, Trans.; D.A. Bochvar, Ed.; S.A. Yanovskaya, Preface). Moscow: Foreign Languages Publishing House.
3. Suravyagin, D. P. (2012). Rudolf Carnap's physicalist reductionism. Vestnik Permskogo Universiteta: Filosofiya, Psikhologiya, Sotsiologiya, 2, 87-97.
4. Cherkashin, E. A., Orlova, I. V. (2017). Tools for creating digital document archives based on linked data. Sovremennye

Tekhnologii: Sistemyi Analiza i Modelirovaniya, 4(56), 100-107.
5. Nekhaev, A. V. (2018). Rudolf Carnap and the "Babylonian towers" of the world of logic. ONV: OIS, 2, 106-110.
6. Griftsova, I., Kozlova, N.Y. (2024). Rudolf Carnap's ideas in the philosophy of language in the context of conceptual engineering. Epistemology Philosophy of Science, 122-133.
7. Hintikka, J. (1991). Carnap, the universality of language and extremality axioms. In W. Spohn (Ed.), Erkenntnis Orientated: A Centennial Volume for Rudolf Carnap and Hans Reichenbach (pp. 325-336). Dordrecht: Springer Netherlands.
8. Khorolsky, O.S., Fesikova, O.V. (2018). Philosophical aspects of programming. Nauchny Almanakh, 4-2(42), 210-213.
9. Makulin, A. V. (2016). Intelligent systems in the humanitarian sphere and digital philosophy. Vestnik Severnogo (Arkticheskogo) Federal'nogo Universiteta: Seriya: Gumanitarnye i Sotsial'nye Nauki, 2, 76-86.
10. Kopytova, M.A. (2016). The relevance of the Python programming language. Ekonomika i Sotsium, 10(29), 943-945.
11. Trenikhina, M. (2018). The physicalist model of consciousness: Artificial intelligence within the framework of classical sign theory. Paradigma: Philosophical and Cultural Almanac, 28, 21-26.