УДК 37.016:004
DOI 10.25688/2072-9014.2021.55.1.01

В. А. Кондратьева

Обучение основам программирования на языке Python в школьном курсе информатики

В статье рассматриваются методические аспекты обучения программированию на языке Python в школьном курсе информатики.

Ключевые слова: школьный курс информатики; методика обучения; основы программирования; язык программирования Python.

Сегодня язык программирования Python является одним из наиболее популярных языков программирования, поскольку он предназначен для разработки различных приложений, позволяет работать с xml/html-файлами, http-запросами, изображениями, аудио- и видеофайлами, используется в робототехнике, программировании математических и научных вычислений, разработках искусственного интеллекта [5]. Этим объясняется высокая потребность IT-компаний в программистах, владеющих именно этим языком программирования.

Вместе с тем Python — достаточно простой для освоения язык программирования, имеет лаконичный и понятный синтаксис, поэтому он стал одним из языков программирования, изучаемым в школьном курсе информатики. Язык Python, наряду с алгоритмическим языком, языками Basic, Pascal и C++, используется в заданиях Государственной итоговой аттестации по информатике.

В данной статье описаны некоторые методические особенности обучения языку программирования Python, что не только может послужить поддержкой при изучении раздела алгоритмизации, но и позволит более детально рассматривать вопросы искусственного интеллекта в рамках школьного курса информатики [3; 4].

Обучение основам программирования на языке Python целесообразно начинать после приобретения учащимися алгоритмических умений в курсе

информатики основной школы [1; 2]. Рекомендуется проводить занятия, организуя смену видов деятельности учащихся за счет чередования объяснения материала и выполнения учащимися практических заданий. Такое построение занятий наиболее эффективно и отвечает санитарно-гигиеническим требованиям по работе учащихся с электронными средствами.

Сформулируем планируемые результаты обучения.

Предметные результаты обучения:

- развитие умений и навыков использования среды IDLE для разработки Python-программ;
- овладение синтаксисом основных конструкций языка программирования Python;
- овладение умениями и навыками работы с различными типами данных языка программирования Python;
- развитие навыков процедурного, а также объектно-ориентированного программирования.

Метапредметные результаты обучения:

- развитие умений организации собственной учебной деятельности, включающих целеполагание, планирование, контроль;
- развитие умений постановки и формулирования проблемы, структурирования и визуализации информации, выбора наиболее эффективных способов решения.

Личностные результаты обучения:

- формирование целостного мировоззрения, соответствующего современному уровню развития науки и общественной практики;
- развитие познавательной активности, интеллектуальных и творческих способностей;
- развитие алгоритмического мышления;
- личностное и предпрофессиональное самоопределение.

Рассмотрим особенности языка программирования Python, на которые следует обратить внимание при обучении школьников.

Выбор среды разработки

Для написания программ на языке Python можно использовать разные интегрированные среды разработки и специализированные редакторы кода, предоставляющие программисту полезные функции, такие как: подсказки кода, подсветка и проверка синтаксиса, средства просмотра файлов и баз данных и т. п. Существуют среды разработки, созданные специально для языка программирования Python. Это, например, PyCharm, Spyder, Thonny. Есть универсальные среды, такие как Visual Studio, которые позволяют разрабатывать программы на различных языках программирования, в частности и на языке Python. Есть и другие среды и редакторы кода, поддерживающие Python.

В учебных целях, на наш взгляд, наиболее рационально и эффективно использование интегрированной среды разработки и обучения IDLE (Integrated Development and Learning Environment), которая содержит необходимый набор инструментальных средств и встроенный текстовый редактор. Среда IDLE позволяет осуществлять ввод, редактирование и сохранение программы, ее тестирование и отладку, трансляцию и исполнение.

Аргументируем выбор именно этой среды разработки для использования в школьном курсе. Прежде всего заметим, что дистрибутивы среды разработки IDLE поставляются вместе с дистрибутивами языка программирования Python и доступны для бесплатного скачивания на официальном сайте Python. Это облегчает для школьников задачу установки Python на их домашние компьютеры, что немаловажно для обеспечения комфортной и своевременной работы. Но это техническая сторона вопроса.

Гораздо более важным здесь является методический момент. Среда разработки, используемая школьниками, должна быть достаточно простой, позволяющей сразу писать программы без дополнительного предварительного овладения технологией работы в этой среде. Чем проще среда разработки, тем меньше она отвлекает обучаемых от собственно процесса изучения программирования. Именно этой цели отвечает среда разработки IDLE. С другой стороны, весь необходимый набор инструментов среда IDLE содержит, что говорит в ее пользу при сравнении с редакторами кода, которые могут обладать меньшей функциональностью.

Есть и другие среды разработки, обладающие подходящими характеристиками и отвечающие целям обучения программированию. Причем если дома обучаемый использует одну среду разработки, а в школе установлена другая, то обычно проблем при переходе не возникает: Python-программа может быть разработана в одной среде, а открыта и запущена на исполнение — в другой.

Отдельно хотелось бы остановиться на рассмотрении популярного веб-приложения Jupyter Notebook, часто используемого для Python-разработок. Jupyter Notebook представляет собой мощный инструмент, предназначенный для работы с данными, но использовать это приложение при обучении программированию не стоит сразу по нескольким причинам.

Во-первых, это многофункциональное приложение, и обучаемым необходимо будет овладеть навыками работы в нем. Во-вторых, оформление Python-программ в блокноте Jupyter несколько отличается от традиционного способа. В-третьих, при разработке более сложных программ фрагменты кода записывают в отдельных ячейках блокнота Jupyter и эти фрагменты уже не являются в полном смысле единой программой и не сохраняются единым файлом, т. е. у обучаемых нарушается представление о программе как о целостной структуре, реализующей некоторый алгоритм. В-четвертых, программы, написанные в блокноте Jupyter, плохо переносимы в иные среды и редакторы.

И, наконец, еще один важный момент: в связи с указанными особенностями этого приложения на олимпиадах различного уровня по информатике Python-программы, разработанные в блокноте Jupyter, к проверке не принимаются[1].

Таким образом, выбирая среду разработки для обучения программированию, следует отдавать предпочтение программным продуктам, отвечающим поставленным учебным целям. Среда разработки IDLE, на наш взгляд, представляет собой оптимальный вариант.

Структурный стиль программирования на языке Python

При обучении программированию на языке Python прежде всего необходимо обратить внимание на то, что язык Python изначально является объектно-ориентированным языком программирования, но при этом он поддерживает структурный (процедурный) стиль программирования. Именно структурному стилю программирования следует обучать школьников в рамках базового курса информатики. Это позволяет продолжать линию развития алгоритмических умений школьников путем обучения записи алгоритмов на языке программирования. Особенности синтаксиса и типизации языка Python таковы, что можно научиться структурному программированию на этом языке, не имея каких-либо знаний в области объектно-ориентированного программирования и не используя этих возможностей языка в своих программах [5].

Таким образом, обучение школьников программированию на языке Python следует вести придерживаясь традиционной методики, опираясь только на структурную парадигму языка. Возможности объектно-ориентированного программирования языка Python можно использовать по мере необходимости, ориентируясь на уровень готовности учащихся к восприятию дополнительного материала. При этом необходимо отметить, что для решения соответствующих заданий Государственной итоговой аттестации по информатике как в 9-м, так и в 11-м классах достаточно владения навыками структурного программирования.

Особенности типизации переменных в языке Python

Отметим особенности языка Python, связанные с типизацией переменных: Python относится к языкам с ***неявной, динамической*** типизацией. Программистам это существенно упрощает работу, однако на начальном этапе изучения основ программирования может стать причиной возникновения ошибок в программах. Рассмотрим, почему так происходит.

---

[1] Калинин П. Почему я не советую учиться программированию в Jupyter notebook [Электронный ресурс] // Блог Петра Калинина. URL: https://blog.algoprog.ru/jupiter/ (дата обращения: 01.10.2020).

Итак, во-первых, Python является языком с неявной типизацией, поэтому объявлять имена и указывать типы используемых в Python-программе переменных, как это делается в других языках программирования, не нужно. Казалось бы, такое удобство не может нанести вреда процессу обучения программированию. Однако языки с явной типизацией накладывают жесткие требования на использование переменных в программе, поэтому объявление переменных и указание их типов заставляет разработчика программы хорошо структурировать свой алгоритм, продумывать необходимость введения каждой переменной и затем использовать ее в соответствии с указанным типом. Все эти действия ведут к развитию алгоритмических умений, которые являются базовыми и, бесспорно, полезными для учащихся с методической точки зрения.

При разработке Python-программы эти действия остаются без должного внимания. Возможность в любой момент написания программы инициализировать переменную позволяет обучаемым не задумываться о структуре программы, создавать «лишние» переменные, оперировать переменными без учета типов данных, с которыми эти переменные связаны.

Во-вторых, Python является языком с ***динамической*** типизацией, поэтому в процессе работы программы можно изменить тип переменной, присвоив ей значение другого типа (в языках с динамической типизацией тип переменной определяется непосредственно при выполнении программы в отличие от языков программирования со статической типизацией, в которых тип переменной определяется на этапе компиляции).

Эта особенность языка Python также требует внимания при обучении программированию, поскольку возможность использования одного имени переменной в разных частях программы может спровоцировать и нарушение логики программы, и возникновение так называемых ошибок типов.

Таким образом, плюсы, за которые ценят язык Python программисты и IT-специалисты, могут стать минусами при обучении программированию на этом языке.

Чтобы минимизировать количество ошибок, связанных с особенностями типизации в языке Python, необходимо формировать и развивать у обучаемых навык грамотной работы с переменными. Добиться этого можно, постоянно акцентируя внимание учащихся на типах используемых в программах переменных. Кроме того, полезно выполнение специальных заданий на определение типов данных, а также на поиск ошибок типов.

Особенности реализации основных алгоритмических конструкций на языке программирования Python

В соответствии с парадигмой структурного программирования при разработке алгоритмов используются три базовые управляющие структуры:

следование, ветвление и повторение. Программа на языке Python, как и на других языках структурного программирования, представляет собой последовательность инструкций или, другими словами, команд. Отметим отличия инструкций ветвления и повторения в Python-программах от аналогичных конструкций в других языках. Синтаксические особенности инструкций Python-программ рассматривать не будем, поскольку они в данном случае несущественны.

При изучении инструкции ветвления в языке Python следует обратить внимание на то, что и команда ветвления (полная и неполная), и команда выбора реализуются с помощью одной синтаксической конструкции `if-elif-else`, поскольку части `elif` и `else` являются необязательными и могут быть опущены. Это, с одной стороны, упрощает процесс запоминания синтаксиса, а с другой — позволяет гибко и вариативно использовать одну конструкцию для решения различных задач.

Структура повторения в языке Python реализуется с помощью инструкции цикла `while` или инструкции цикла `for`. Отметим два момента, связанных с инструкциями циклов. Во-первых, конструкций для реализации цикла с постусловием в языке Python нет. Во-вторых, если цикл `while` традиционен и по оформлению, и по использованию (аналогичные конструкции присутствуют и в учебном алгоритмическом языке, и в других языках программирования), то цикл `for` имеет определенные особенности.

Характерной чертой инструкции `for` в языке Python является то, что переменная цикла может пробегать по любому итерируемому объекту, т. е. объекту, предоставляющему возможность поочередного доступа ко всем своим элементам. Итерируемыми объектами являются строки, списки, кортежи, множества, словари, сгенерированные последовательности чисел. В связи с этим цикл `for` имеет гораздо более широкий спектр возможностей использования.

Специфической в языке Python является организация итерации по последовательности чисел, поскольку эту последовательность необходимо сгенерировать с помощью встроенной функции `range()`, но это достаточно просто реализуется и обычно не вызывает затруднений у обучаемых.

Использование инструкций `continue`, `break` и `else` для циклов. Для оптимизации работы циклов в языке Python используются инструкции `continue`, `break` и `else`. Инструкция `continue` используется, когда необходимо перейти к следующей итерации, минуя оставшиеся инструкции тела цикла `for` или `while`. Инструкция `break` необходима для досрочного выхода из цикла (другими словами, для его прерывания). Инструкция `else` для циклов может работать только в паре с инструкцией `break`. Инструкция `else` для циклов осуществляет проверку, не произошло ли прерывание цикла; блок инструкций, помещенный в инструкцию `else`, выполняется только в том случае, если выход из цикла произошел без помощи `break`.

Подчеркнем очень важный момент. Большинство задач по программированию в школьном курсе информатики может быть решено без использования инструкций `continue`, `break` и `else`. Более того, составление программ без применения указанных инструкций дает возможность школьникам лучше овладеть умениями работы с основными управляющими структурами. Однако при разработке некоторых программ сложно или даже невозможно обойтись без них. Поэтому их изучение школьниками должно быть своевременным и с методической, и с практической точки зрения.

Кроме того, инструкция `else` для циклов по своей логике использования сильно отличается от инструкции `else` для ветвления, что, естественно, может вызвать сложности при изучении. Поэтому переходить к изучению этой инструкции следует только после приобретения учащимися устойчивых умений программирования с помощью основных способов.

В связи с этим изучение оптимизирующих работу циклов инструкций при обучении программированию на языке Python возможно в том или ином объеме в зависимости от уровня подготовки учащихся.

При изучении инструкций `continue`, `break` и `else` в школьном курсе информатики необходимо сформировать у учащихся представление о том, что эти инструкции предназначены для повышения эффективности работы сложных программ.

Использование встроенных функций и методов

Таким же своевременным должно быть изучение встроенных функций и методов языка программирования Python. Наличие большой библиотеки встроенных функций и методов является одной из привлекательных сторон программирования на языке Python. Большинство алгоритмов, которые традиционно используют учащиеся при изучении основ программирования: нахождение длины строки, максимального и минимального элемента в списке, сортировка списка, добавление элемента в список и т. д., — реализованы в языке Python в виде встроенных функций и методов.

Несмотря на это, для развития у обучаемых навыков алгоритмизации и программирования необходимо ставить перед ними задачи разработки этих алгоритмов вручную, с использованием основных приемов программирования. В дальнейшем, когда определенный уровень знаний и умений учащимися будет достигнут, встроенные функции и методы можно будет использовать для решения более сложных задач.

Здесь следует подчеркнуть, что реализация алгоритмов с ограничением на использование каких-то определенных конструкций или методов, является хорошим упражнением для развития алгоритмических навыков.

Изменяемые и неизменяемые типы данных

При изучении способов работы с различными типами данных языка Python необходимо обратить внимание на то, что некоторые объекты данных являются изменяемыми объектами, а некоторые — неизменяемыми. Эта необходимость продиктована тем, что разные способы записи данных в памяти компьютера требуют различных подходов при программировании.

Неизменяемость объекта данных означает, что в созданном объекте в процессе работы программы его значение измениться не может. К неизменяемым типам данных в языке Python относятся, например, числа, логические значения, строки. Любое изменение значения переменной неизменяемого типа приведет к тому, что будет создан новый объект, связанный с этой переменной, которому будет отведена новая ячейка памяти компьютера.

Если объект данных изменяемый, то в процессе работы программы его значение может изменяться, адресация этого объекта в памяти компьютера при этом остается прежней, созданной при его инициализации. К изменяемым объектам в языке Python относятся, например, списки.

При разработке программ обычно не задумываются о том, сохранилась или нет адресация объекта после некоторых действий с ним, но помнить о различиях между изменяемыми и неизменяемыми объектами необходимо, чтобы использовать в программах допустимые для каждого типа операции.

Понятия изменяемых и неизменяемых типов данных являются достаточно сложными для школьников. Если времени на занятиях не хватает для рассмотрения дополнительного материала, достаточно ввести эти понятия, указать на отличия изменяемых и неизменяемых объектов, привести примеры, демонстрирующие допустимые и недопустимые (т. е. приводящие к возникновению ошибок в программе) операции с объектами, и в дальнейшем при работе с определенным типом данных обращать внимание учащихся на способы работы с этим типом, продиктованные особенностями его реализации.

Если учащиеся хорошо подготовлены и способны к восприятию дополнительного материала, то можно для демонстрации различий между изменяемыми и неизменяемыми типами данных рассмотреть работу встроенной функции `id()`, возвращающей идентификатор объекта, и отследить, как изменяется адресация изменяемых и неизменяемых объектов при попытке перезаписи их значений.

В заключение заметим, что остальные особенности языка программирования Python, на наш взгляд, являются менее существенными и не требуют каких-либо специфических подходов при обучении программированию в рамках школьного курса информатики.

Литература

1. Григорьев С. Г., Гриншкун В. В., Левченко И. В., Заславская О. Ю. Проект примерной программы по информатике для основной школы // Информатика и образование. 2011. № 9. С. 2–11.
2. Левченко И. В. Применение методических средств организации алгоритмической деятельности на уроках информатики основной школы // Информатика и образование. 2006. № 2. С. 107–112.
3. Левченко И. В., Левченко И. В., Садыкова А. Р., Абушкин Д. Б., Михайлюк А. А., Павлова А. Е., Тамошина Н. Д. Эллективный курс «Основы искусственного интеллекта»: учеб. пособие М.: Образование и Информатика, 2019. 96 с.
4. Левченко И. В., Левченко Е. С., Михайлюк А. А. Практические работы элективного курса «Основы искусственного интеллекта». М.: Образование и Информатика, 2019. 64 с.
5. Лутц М. Изучаем Python. СПб.: Символ-Плюс, 2011. 1280 с.

Literatura

1. Grigor`ev S. G., Grinshkun V. V., Levchenko I. V., Zaslavskaya O. Yu. Proekt primernoj programmy` po informatike dlya osnovnoj shkoly` // Informatika i obrazovanie. 2011. № 9. S. 2–11.
2. Levchenko I. V. Primenenie metodicheskix sredstv organizacii algoritmicheskoj deyatel`nosti na urokax informatiki osnovnoj shkoly` // Informatika i obrazovanie. 2006. № 2. S. 107–112.
3. Levchenko I. V., Levchenko I. V., Sady`kova A. R., Abushkin D. B., Mixajlyuk A. A., Pavlova A. E., Tamoshina N. D. E`llektivny`j kurs «Osnovy` iskusstvennogo intellekta»: ucheb. posobie M.: Obrazovanie i Informatika, 2019. 96 s.
4. Levchenko I. V., Levchenko E. S., Mixajlyuk A. A. Prakticheskie raboty` e`lektivnogo kursa «Osnovy` iskusstvennogo intellekta». M.: Obrazovanie i Informatika, 2019. 64 s.
5. Lutcz M. Izuchaem Python. SPb.: Simvol-Plyus, 2011. 1280 s.

V. A. Kondratyeva

Teaching the Basics of Programming in Python in the School Computer Science Course

The article discusses the methodological aspects of teaching Python programming in a school computer science course.

Keywords: school computer science course; teaching method; basics of programming; Python programming language.