УДК 37.04:372.8

ОБУЧЕНИЕ ШКОЛЬНИКОВ ВЫПОЛНЕНИЮ И АНАЛИЗУ АЛГОРИТМОВ ДЛЯ ФОРМАЛЬНОГО ИСПОЛНИТЕЛЯ С ИСПОЛЬЗОВАНИЕМ СИСТЕМ ПРОГРАММИРОВАНИЯ

Козлов С.В.[1], Быков А.А.[2]

[1] ФГБОУ ВО «Смоленский государственный университет», Смоленск, e-mail: svkozlov1981@yandex.ru;
[2]Филиал ФГБОУ ВО «Национальный исследовательский университет «МЭИ», Смоленск, e-mail: alex1by@mail.ru

В статье рассматриваются вопросы обучения школьников анализу алгоритмов для формального исполнителя. Актуальность данной тематики связана с тем, что в современных условиях развития IT-технологий необходима тесная связь теории и практики. Как показывает исследование рассматриваемой проблематики, получение теоретических знаний и решение заданий без использования компьютерных приложений классическими методами стремительно устаревает. IT-индустрия диктует широкое применение систем программирования и реализованных в них методов обработки данных. В связи с этим базовая профессиональная подготовка в области алгоритмизации и программирования требует от школьников приобретения навыков работы в современных инструментальных средах. Это расширяет их предметный кругозор и повышает уровень знаний и умений. Авторами рассматривается возможность применения встроенных методов языка программирования Python для анализа выполнения в нем программ для формального исполнителя. Наличие в его инструментальной среде понятных специальных средств обработки числовых и строковых данных делает ее востребованной для решения такого рода задач. Использование языка программирования Python предлагает школьникам эффективный инструмент анализа и тестирования исследуемых алгоритмов. Это обеспечивает значительный рост их профессиональных навыков в программировании, как основной области приложения теоретической информатики. Результаты педагогического эксперимента, проведенного на различных выборках школьников, подтверждают эффективность применения передовых IT-технологий в обучении анализу и выполнению алгоритмов и программ.

Ключевые слова: информатика, программирование, IT-технологии, алгоритм, формальный исполнитель, язык Python, программное приложение, образовательный процесс.

TEACHING SCHOOLCHILDREN THE IMPLEMENTATION AND ANALYSIS OF ALGORITHMS FOR A FORMAL EXECUTOR USING PROGRAMMING SYSTEMS

Kozlov S.V.[1], BykovA.A.[2]

[1]FGBOU VO "Smolensk State University", Smolensk, e-mail: svkozlov1981@yandex.ru;
[2]Branch of FGBOU VO "National Research University "MPEI", Smolensk, e-mail: alex1by@mail.ru

The article deals with the issues of teaching schoolchildren the analysis of algorithms for a formal executor. The relevance of this topic is due to the fact that in the modern conditions of the development of IT technologies, a close connection between theory and practice is necessary. As the study of the issues under consideration shows, obtaining theoretical knowledge and solving tasks without the use of computer applications by classical methods is rapidly becoming obsolete. The IT industry dictates the widespread use of programming systems and data processing methods implemented in them. In this regard, basic professional training in the field of algorithmizing and programming requires schoolchildren to acquire skills in working in modern tool environments. This expands their subject horizons and increases the level of knowledge and skills. The authors consider the possibility of using the built-in methods of the Python programming language to analyze the execution of programs in it for a formal executor. The presence in its tool environment of understandable special means for processing numerical and string data makes it in demand for solving such problems. The use of the Python programming language offers students an effective tool for analyzing and testing the algorithms under study. This provides a significant increase in their professional skills in programming, as the main area of application of theoretical computer science. The results of a pedagogical experiment conducted on various samples of schoolchildren confirm the effectiveness of using advanced IT technologies in teaching analysis and execution of algorithms and programs.

Keywords: computer science, programming, IT-technologies, algorithm, formal executor, Python language, software application, educational process.

Изучение алгоритмизации и программирования в школьном курсе информатики становится с каждым учебным циклом все более обширным и многогранным [1; 2]. Это

обусловлено тем, что развитие IT-технологий и применение их в обучении сделало в XXI веке стремительный рывок вперед [3; 4]. Информатизация сфер жизнедеятельности человека приобрела всеобщий характер. Во всех областях производства, науки и техники, в системе образования и здравоохранения, экономике и во многих других областях требуются IT-специалисты. В связи с этим необходимо обеспечить на начальных этапах обучения информатике и информационно-коммуникационным технологиям формирование фундаментальной базовой подготовки, особенно в знании алгоритмов и структур данных и умении обоснованно их применять при решении практических задач.

Одной из дидактических линий школьного курса программирования является выполнение и анализ алгоритмов для формального исполнителя [5; 6]. Это направление отвечает за приобретение школьниками навыков алгоритмической мыслительной деятельности в разрешении поставленных перед ними вопросов практики. В то же время изучение данной тематики по старым шаблонам становится малопродуктивным [7]. В отрыве от демонстрации в современных инструментальных средах встроенных методов реализации изученных теоретических алгоритмов школьник не понимает необходимости их применения [8; 9]. Он теряет связь теоретической информатики с практикой ее приложений [10; 11]. Объяснение математических и логических основ алгоритмов требует наглядного их отражения в процедурах и функциях инструментальных систем. Только в этом случае у него формируются необходимые представления о семантике и синтаксисе использования алгоритмических структур в практике программирования.

Цель исследования: проверка эффективности использования методов языка программирования Python при объяснении школьникам методов выполнения и анализа простых алгоритмов для формального исполнителя.

Научная новизна состоит в методологическом использовании средств языка программирования Python при решении задач выполнения и анализа алгоритмов, составленных для формального исполнителя.

Материал и методы исследования. Школьный курс изучения основ алгоритмизации предполагает формирование умений «читать» и выполнять программы для формального исполнителя. Эти знания отражены как в тематических блоках обучения информатике, так и при оценке базовых навыков на государственной итоговой аттестации школьников [12; 13]. При этом, несмотря на вынесение проверки данных умений на базовый уровень сложности, далеко не все учащиеся демонстрируют уверенные навыки применения своих знаний на практике. Это происходит в силу того, что до сих пор во многих учебных заведениях ощущается острая нехватка специалистов в преподавании школьникам основ программирования. В связи с этим упор делается на теоретическое изучение материала, анализ

алгоритмов мало подкрепляется практикой применения. В то же время только параллельное изучение алгоритмизации и программирования обеспечивает получение прочных навыков, необходимых для разработки и создания современных программных приложений.

Так, среди заданий ЕГЭ по информатике в компьютерной форме есть задание № 5, требующее от школьника продемонстрировать свои навыки в выполнении программы для формального исполнителя. Сложность анализа алгоритма обусловлена тем, что от школьника просят вычислить не конечные результаты выполнения программы, а определить исходные параметры задачи, при которых указанные данные были получены. Это существенно затрудняет исследование последовательности алгоритмических действий. От школьника требуется не только понять иногда весьма витиеватый алгоритм, но и учесть все его «подводные камни», сознательно расставленные составителями задания. В решении поставленной задачи может помочь проверка указанных в условии правил и конструкций на языке программирования, например в инструментальной среде Python [14; 15]. Такой подход позволяет как реализовать представленную для формального исполнителя программу, так и автоматизировать его команды и действия с помощью встроенных методов выбранной системы программирования. При этом в случае отсутствия необходимых инструментов ничто не мешает школьнику написать собственные пользовательские методы описанных в условии задачи команд.

Рассмотрим один из примеров, аналогичный тем, которые представлены в формате компьютерного ЕГЭ по информатике, и образующий авторскую систему упражнений для тренировки навыков анализа и выполнения алгоритмов с помощью языка программирования Python. На вход исполнителю алгоритма подается натуральное число N. Исполнитель строит по нему новое число R следующим образом.

1. Строится двоичная запись числа N.

2. К этой записи дописывается справа бит четности: 0, если в двоичном коде числа N было четное число единиц, и 1, если нечетное.

3. Затем полученная запись обрабатывается по следующему правилу:

а) если сумма цифр в полученной двоичной записи числа четная, то к этой записи справа дописывается 0, а затем два левых разряда заменяются на 10;

б) если сумма цифр в двоичной записи числа нечетная, то к этой записи справа дописывается 1, а затем два левых разряда заменяются на 11.

Полученная таким образом запись (в ней на четыре разряда больше, чем в записи исходного числа N) является двоичной записью искомого числа R. Укажите такое наибольшее число N, для которого результат работы данного алгоритма меньше числа 77. В ответе запишите это число в десятичной системе счисления.

Рассмотрим решение поставленной задачи на языке программирования Python.

```
for n in range(1, 101):
    s = bin(n)[2:]
    if s.count('1') % 2 == 0:
        s = s + '0'
    else:
        s = s + '1'
    if s.count('1') % 2 == 0:
        s = '10' + s[2:] + '0'
    else:
        s = '11' + s[2:] + '1'
    r = int(s, 2)
    if r < 77:
        print(n)
```

В представленном решении примера для перебора исходных значений используется цикл с параметром for. Он позволяет для каждого числа в диапазоне от 1 до 100 получить результат исполнения программы и проанализировать его на соответствие заданным в условии задачи критериям. В цикле сначала осуществляется перевод исследуемого натурального числа *n* в двоичную систему счисления. Для этого используется встроенный метод bin(). Кроме того, ввиду получения двоичной записи в текстовом формате появляется возможность работать с числом как со строкой, применяя специально предназначенные для этого методы обработки данных. В частности, для удаления из формата записи числа в двоичной системе счисления префикса '0b' в программе применено понятие «среза», которое позволяет оставить в данном примере в строке *s* все символы, начиная со второго. Здесь следует напомнить, что индексация в языке Python начинается с нуля.

Далее в данном алгоритме используется понятие «бит четности», по правилу, связанному с которым, к полученной записи добавляется справа дополнительный разряд '0' или '1'. Заметим, что правило, сформулированное в пункте 3 алгоритма, также представляет собой понятие «бит четности», представленное в ином виде. Отличие его от пункта 2 составляет только дописывание двух разрядов слева.

Проверить четность числа единиц в полученной на предыдущем этапе алгоритма строке *s* можно, применив команду count. Она возвращает в виде результата целое число, которое соответствует количеству символов '1' в анализируемой строке. После этого остается с помощью оператора % определить остаток от деления количества '1' в строке на 2 и сравнить

его с нулем. В случае равенства нулю, к нему будет необходимо справа к строке справа приписать '0', а в противоположном случае – '1'.

Отметим, что добавление в соответствии с пунктом 3 алгоритма исполнителя двух разрядов слева от формируемого значения значительно усложняет устный анализ задания, на который делается упор при теоретическом подходе. Это обусловлено тем, что в данном случае из меньшего исходного натурального числа *n* получается большее число *r*, как результат работы алгоритма. Это требует от школьника рассмотрения двух ситуаций, соответствующих добавлению слева в записи получаемого числа разрядов '10' и '11', в отличие от более простых случаев, когда в алгоритмах для формального исполнителя предложена последовательность действий, которая реализует возрастающую функцию. При решении задания на языке программирования Python выполнение этой части алгоритма не требует проведения дополнительного анализа возникшей ситуации. Для этого по-прежнему необходимо использовать конкатенацию строк в виде операции сложения строковых данных.

В завершение алгоритма рассматриваемого примера требуется определить наибольшее, а не, как правило, наименьшее, число, которое не превосходит значение 77. При теоретическом анализе и последующем «ручном» исполнении алгоритма это приводит к применению операции «вычитания», а не «сложения» двоичных чисел. А это в свою очередь также усложняет проводимое исследование алгоритма. При использовании возможностей языка программирования Python это практически не изменяет реализуемые в программе действия. Отличие состоит в том, что после перевода строки *s* в десятичную запись и присваивания полученного значения переменной *r* при записи действий в условном операторе if необходимо не указывать команду прерывания цикла break. Это позволит отобразить на экране все найденные значения. Отметим, что для обратного перевода полученной записи в десятичную систему счисления используется встроенный метод языка программирования Python int(s, 2). Он позволяет переводить строковые значения из любого формата в десятичное число, в данном случае значение «2» указывает на преобразование двоичной записи. Также заметим, что в отдельных заданиях такого типа следует расширить диапазон поиска искомых значений до 1000, изменив правый конец полуинтервала на 1001 в цикле с параметром for.

Итак, в данном примере программа, написанная на языке Python, выведет следующий список значений: 14 15 16 17 18 24 25 26 27.

Ответом будет служить последнее число из него – 27, как наибольшее значение *n*, при котором *r* меньше числа 77. Если было бы необходимо найти в данном задании значение переменной *r*, то изменив в команде print(n) вывод переменной *n* на *r*, мы получим следующий ряд значений: 4 10 8 18 20 16 22 34 36 40 46 32 38 42 44 66 68 72 64 70 74 76.

В этом случае искомым значением будет число 76.

Такого типа задания были предложены школьникам при изучении темы «Исполнение алгоритма для формального исполнителя» для тренировки полученных умений.

В качестве методов проводимого исследования на основе обобщения передового педагогического опыта были использованы констатирующий и формирующий педагогический эксперименты, а для обработки результатов исследования – математические методы. Гипотеза исследования состояла в следующем: применение стандартных алгоритмических конструкций и встроенных методов инструментальных систем, в частности языка программирования Python, при решении заданий анализа и выполнения команд для формального исполнителя способствует повышению эффективности обучения школьников.

Результаты исследования и их обсуждение. Педагогический эксперимент по обучению школьников проводился в двух группах. Первую группу составили учащиеся Смоленского физико-математического лицея при МИФИ. Вторую группу – учащиеся IT-класса средней школы № 6 г. Смоленска. Всего в эксперименте приняли участие 33 ученика. Констатирующий этап педагогического эксперимента состоял в обучении школьников анализу и выполнению простых алгоритмов для формального исполнителя решения «на бумаге». Учащиеся изучали алгоритмы перевода чисел из десятичной системы счисления в двоичную систему и обратно. Они также знакомились с понятиями «бит четности», «четность числа», после чего отрабатывали умения дописывать в соответствии с заданными алгоритмами разряды справа и слева к полученной двоичной записи. В завершение данного блока обучения школьникам был дан тест, состоящий из десяти заданий, который они выполняли без использования вспомогательных компьютерных средств. Формирующий этап педагогического эксперимента заключался в обучении решать аналогичные задания с использованием инструментов языка программирования Python, в частности встроенных методов для работы с числовыми и строковыми данными. Отметим, что учащимся на данном этапе предлагались, наряду с простыми, и более сложные алгоритмы для формального исполнителя. Однако это существенным образом не влияло на общие подходы к решению заданий. В завершение данного этапа было также проведено тестирование учащихся. Им был предложен тест, состоящий из десяти заданий, решение которых предстояло выполнить в среде программирования языка Python. Результаты выполнения теста заносились в специально разработанную интеллектуальную оболочку, которая анализировала как ответы, так и способы решения заданий. Данные итоговых диагностик на констатирующем и формирующем этапах педагогического эксперимента представлены в таблицах 1 и 2.

Таблица 1

Результаты констатирующего этапа педагогического эксперимента

Группа	Число школьников, достигших уровня усвоения знаний			Всего
	Высокий	Повышенный	Базовый	
СФМЛ при МИФИ	3	7	8	18
IT-класс школа № 6	2	7	6	15
Всего	5	14	14	33

Таблица 2

Результаты формирующего этапа педагогического эксперимента

Группа	Число школьников, достигших уровня усвоения знаний			Всего
	Высокий	Повышенный	Базовый	
СФМЛ при МИФИ	8	9	1	18
IT-класс школа № 6	7	8	0	15
Всего	15	17	1	33

Качественный анализ условий и результатов эксперимента. На основании результатов эксперимента, представленных в таблицах, можно судить о том, что школьники решают рассматриваемые задания для формального исполнителя более уверенно с использованием языка программирования Python. Количество учащихся, достигших высокого уровня освоения учебного материала, возросло втрое. Произошло существенное перераспределение баллов при тестировании в сторону более высоких групп показателей градации результатов. Учащихся, продемонстрировавших базовый уровень предметной подготовки, практически не осталось. Это позволяет утверждать, что применение инструментов сред программирования значительно упрощает анализ представленных в заданиях алгоритмов. Это происходит в силу того, что тестирование программного приложения происходит в среде приложения. Школьнику нет необходимости рассчитывать результаты работы для различных групп данных «вручную». Он может с помощью написанной программы мгновенно получить результат вычислений. В связи с этим уменьшается как количество вычислительных ошибок, так и время, затрачиваемое учащимися на решение отдельно взятой задачи. Кроме того, унификация действий с помощью одних и тех же командных действий средствами встроенных методов языка программирования Python позволяет уменьшить количество рутинных действий. Это способствует повышению интереса школьников к решению более сложных по классу задач. Таким образом, гипотеза, проверяемая в исследовании, находит свое подтверждение.

Заключение

Итак, результаты педагогического эксперимента свидетельствуют о необходимости применения средств систем программирования для решения заданий анализа и выполнения алгоритмов для формального исполнителя. Во-первых, уровень профессиональных умений и навыков школьников выходит на качественно иной уровень. Учащиеся осознанно решают данный класс задач, что обеспечивает им формирование фундаментальной базы для исследования заданий более высокого уровня сложности. Во-вторых, при таком подходе к организации обучения школьники получают как теоретическую, так и практическую подготовку, навыки использования IT-технологий. Учащиеся не только получают базовые знания, но и находят их применение в области практических приложений современных сред программирования. Это расширяет их возможности при решении заданий практики с учетом цифровых образовательных тенденций. В-третьих, такой подход демонстрирует роль современных инструментальных сред в жизни и деятельности человека. Школьники учатся использовать навыки программирования для решения конкретных задач обработки потоков данных. Увеличивается уровень их владения специальными встроенными методами систем программирования. Таким образом, обеспечивается всесторонняя профессиональная подготовка в области изучения информатики и применения современных IT-технологий.

Список литературы

1. Каган Э.М. Применение визуальных языков программирования для повышения эффективности обучения разделу «Алгоритмизация и программирование» школьного курса информатики // Вестник Московского городского педагогического университета. Серия: Информатика и информатизация образования. 2018. № 1 (43). С. 99-104.
2. Козлов С. В., Быков А.А. О применении методов математического моделирования при обучении алгоритмизации в вузе // Современные проблемы науки и образования. 2021. № 3. URL: https://science-education.ru/ru/article/view?id=30946 (дата обращения: 24.02.2023).
3. Тимофеева Н.М. О цифровизации образовательного процесса в условиях полного его переноса в онлайн // Системы компьютерной математики и их приложения. 2021. № 22. С. 388-394.
4. Киселева О.М. Программные средства поддержки удаленного обучения // Вызовы цифровой экономики: тренды развития в условиях последствий пандемии COVID-19: сборник статей IV Всероссийской научно-практической конференции, приуроченной к Году науки и технологий в России. Брянск, 2021. С. 143-146.

5. Лапшева Е.Е. Профильная информатика в свете введения компьютерного ЕГЭ // Информационные технологии в образовании: материалы XI Всероссийской (с международным участием) научно-практической конференции. 2019. С. 128-130.
6. Никонова П. В. Особенности решения задачи единого государственного экзамена на выполнение алгоритмов для исполнителей // Инновационные процессы в современном образовании: практики, технологии, решения: сборник трудов по материалам международной научно-практической конференции. М., 2021. С. 346-353.
7. Козлов С.В., Быков А.А. Применение методов математического моделирования для диагностики знаний школьников // Современные наукоемкие технологии. 2021. № 4. С. 157-162.
8. Францкевич А.А. О результатах применения методики обучения школьников основам алгоритмизации и программирования с применением визуализированных сред программирования // Физико-математическое образование: цели, достижения и перспективы: материалы Международной научно-практической конференции. Минск: Белорусский государственный педагогический университет имени Максима Танка, 2019. С. 52-53.
9. Козлов С. В. Особенности обучения школьников информатике в профильной школе // Научно-методический электронный журнал «Концепт»». 2014. № 1. С. 31-35. ART 14006. URL: http://e-koncept.ru/2014/14006.htm. (дата обращения: 24.02.2023).
10. Гибадуллин А.А. Программирование компьютерных интеллектуальных игр как метод обучения информатике // Педагогика: материалы 59-й Международной научной студенческой конференции (г. Новосибирск, 12-23 апреля 2021 года). Новосибирск, 2021. С. 15-16.
11. Киселева О.М., Солдатенкова Я.Г. Проектирование образовательных информационных систем // Развитие научно-технического творчества детей и молодежи – НТТДМ 2021: сборник материалов V Всероссийской научно-практической конференции с международным участием. Киров, 2021. С. 93-98.
12. Архангельская Е.В. Организация обучения основам алгоритмизации и программирования с использованием анализа математических задач // Информатизация образования и науки. 2022. № 4 (56). С. 166-175.
13. Козлов С.В., Быков А.А. Особенности изучения междисциплинарных тем школьных курсов математики и информатики с помощью методов математического моделирования // Проблемы современного образования. 2021. № 5. С. 250-261.
14. Рослякова Е.А., Химич А.М. Преимущества использования языка программирования Python при изучении раздела «Алгоритмизация и программирования» в школьном курсе информатики // Современные тенденции развития фундаментальных и прикладных наук:

материалы Всероссийской с международным участием научно-практической конференции / Под ред. С.А. Коньшаковой. 2018. С. 148-152.

15. Ильченко О. Ю., Сырицына В. Н., Кадеева О. Е. Решение задач ЕГЭ по информатике средствами языка Python // Высшее образование сегодня. 2021. № 11-12. С. 42-54.