УДК 37.04:372.8

ОБУЧЕНИЕ ШКОЛЬНИКОВ ВЫПОЛНЕНИЮ ЦИКЛИЧЕСКИХ АЛГОРИТМОВ ДЛЯ ИСПОЛНИТЕЛЯ С ПОМОЩЬЮ СИСТЕМ ПРОГРАММИРОВАНИЯ

[1]Козлов С.В., [2]Быков А.А.

[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 обсуждаются вопросы обучения школьников выполнению циклических алгоритмов для формального исполнителя. Авторами объясняется необходимость всесторонней практической подготовки учащихся применению современных систем программирования для выявления связей между исходными и выходными данными. Подчеркивается необходимость поиска баланса между изучением теоретических методов анализа программ и использованием инструментов компьютерных сред для проверки особенностей их выполнения. На примере циклических алгоритмов обработки строковых данных с использованием оператора ветвления демонстрируются простота и понятность методов языка программирования Python для получения результата преобразования исходной строки. Раскрываются особенности записи условий принадлежности множеству значений в алгоритмических структурах цикла и ветвления. Разъясняется назначение параметров встроенной функции замены строк и специфика их использования в алгоритмах поиска данных. Авторами приводятся модификации рассматриваемого задания, состоящие в иной формулировке вопроса при неизменности представленного алгоритма. Указывается, что такие незначительные изменения приводят при теоретическом решении задачи к существенному увеличению ее сложности, а при использовании средств системы программирования Python практически не влияют на код программы. Это расширяет как набор инструментов анализа массивов данных, которым владеют школьники, так и круг решаемых ими задач. Анализ результатов констатирующего и формирующего этапов педагогического эксперимента, результаты которого представлены в статье, доказывают эффективность выбранного авторами подхода применения систем программирования для составления и анализа программ для формального исполнителя.

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

TEACHING SCHOOLCHILDREN TO EXECUTE CYCLIC ALGORITHMS FOR THE CONTRACTOR WITH THE HELP OF PROGRAMMING SYSTEMS

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

[1]Smolensk State University, Smolensk, e-mail: svkozlov1981@yandex.ru;

[2]Branch of National Research University "MPEI", Smolensk, e-mail: alex1by@mail.ru

The article discusses the issues of teaching schoolchildren to perform cyclic algorithms for a formal performer. The authors explain the need for comprehensive practical training of students in the use of modern programming systems to identify the links between source and output data. The need to find a balance between studying theoretical methods of analyzing programs and using tools from computer environments to check the specifics of their execution is emphasized. The example of cyclic algorithms for processing string data using the branch operator demonstrates the simplicity and clarity of the Python programming language methods to obtain the result of the transformation of the original string. Features of recording conditions of belonging to multiple values in algorithmic structures of cycle and branching are disclosed. The purpose of the parameters of the built-in string replacement function and the specifics of their use in data search algorithms are explained. The authors present modifications of the task under consideration, consisting in a different formulation of the question with the invariability of the presented algorithm. It is indicated that such minor changes lead to a significant increase in its complexity when theoretically solving the problem, and when using Python programming system tools, they practically do not affect the program code. This expands both the set of tools for analyzing data arrays owned by schoolchildren and the range of tasks they solve. Analysis of the results of the stating and forming stages of the pedagogical experiment, the results of which are presented in the article, prove the effectiveness of the approach chosen by the authors to use programming systems to compile and analyze programs for a formal performer.

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

Дидактическая линия алгоритмизации и программирования является одним из основных направлений базовой подготовки школьника как будущего специалиста в сфере IT-индустрии [1, 2]. В то же время в XXI в. компьютерные технологии настолько далеко шагнули вперед, что обучения только азам теоретической мысли в области информатики и отдельных примеров ее приложений становится недостаточно. Практика применения инструментальных средств компьютерных приложений должна не только постоянно расширяться, но и становиться доминантой при изучении

информатики и информационных и коммуникационных технологий на всех этапах современной системы образования [3, 4].

Изучение базовых алгоритмических структур, таких как следование, ветвление и цикл, и основных алгоритмов обработки разных типов данных в школе в условиях цифровой трансформации системы образования [5, 6] требует не только всестороннего анализа их применения, но и проверки выдвинутых гипотез о результатах выполнения программ при различных входных параметрах на практике. При этом школьник должен применять инструменты востребованных в сфере современных разработок языков программирования [7, 8]. Он должен знать специфику работы встроенных в инструментальную среду методов обработки данных и использовать их при реализации алгоритмов, написанных для формального исполнителя программ. Ввиду этого учителю необходимо находить требуемый баланс между теорией и практикой, делать упор на демонстрацию изученных научных положений через средства компьютерных приложений, которые служат для написания новых программ.

Цель исследования – описание и анализ использования методов языка программирования Python при объяснении школьникам методов выполнения и анализа циклических алгоритмов, содержащих ветвления.

Научная новизна состоит в выявлении особенностей применения школьниками инструментальных средств языка программирования Python для решения задач выполнения циклических алгоритмов, составленных для формального исполнителя.

Материалы и методы исследования

При изучении линии алгоритмизации и программирования в школьном курсе информатики учащиеся последовательно изучают линейные алгоритмы, ветвления и циклы. При этом в заданиях повышенной сложности циклы естественным образом включают разветвляющиеся структуры как квинтэссенцию базовых алгоритмических конструкций. Это требует от школьника значительно более высокого уровня понимания происходящих в задаче событий, особенно при обучении информатике на профильном уровне [9, 10]. В то же время только теоретический анализ условий, определенных в исследуемом задании, нередко останавливает большинство учащихся в поиске правильного подхода алгоритма решения. В этом случае использование на практике встроенных методов систем программирования позволяет грамотно организовать тестирование входных и выходных данных, всесторонне проанализировать их.

Так в заданиях компьютерного ЕГЭ по информатике, как отражении знаний и умений школьного курса, содержится № 12, в котором от учащихся требуется продемонстрировать умения выполнить циклический алгоритм для исполнителя. Школьники должны знать методы обработки строковых данных, такие как определение длины строки, поиск и выделение из нее подстроки, удаление и вставка символов в строку, а также объединение строк. При этом предполагается, что эти инструменты анализа текстовых данных учащиеся будут применять при «ручном» исполнении предлагаемых в условиях заданий алгоритмов. Это безусловно позволяет развивать у них поисковый метод решения задачи, выявление частных случаев и общих закономерностей при выполнении циклического алгоритма. В то же время параллельное с решением задач в тетради использование современных систем программирования, например Python [11, 12], позволяет существенно расширить горизонты анализа и выполнения циклических алгоритмов для формального исполнителя.

Рассмотрим один из примеров подобных задач, предлагаемых на компьютерном ЕГЭ по информатике. Они образуют авторскую систему заданий, используемую при изучении темы «Циклические алгоритмы для формального исполнителя» школьного курса. Исполнитель получает на входе в программу строку цифр и обрабатывает ее. Он может выполнять две команды, в которых *x* и *y* обозначают цепочки цифр, заданные в обрабатываемой строке. Команда «нашлось *x*» проверяет, встречается ли указанная цепочка в обрабатываемой исполнителем строке. Команда «заменить *x*, *y*» определяет первое вхождение цепочки *x* в заданную строку, если таковое есть, то есть заменяет ее на цепочку *y*. Если цепочка *x* не найдена, то исследуемая строка не изменяется.

Какая строка получится в результате применения нижеследующего алгоритма к строке, состоящей из 71 идущих подряд цифр 4?

```
ПОКА нашлось (444) или нашлось (888)
    ЕСЛИ нашлось (444)
    ТО заменить (444, 8)
    ИНАЧЕ заменить (888, 4)
    КОНЕЦ ЕСЛИ
КОНЕЦ ПОКА
```

Рассмотрим решение данной задачи на языке программирования Python.

```
s = '4' * 71
while '444' in s or '888' in s:
    if '444' in s:
        s = s.replace('444', '8', 1)
    else:
        s = s.replace('888', '4', 1)
print(s)
```

Представленное решение примера начинается с задания строки *s*. Для этого используется операция умножения, которая позволяет получить заданное количество символов '4' с помощью их дублирования и склейки. Также в отдельных заданиях для получения строки, состоящей из различных символов, требуется применять операцию сложения, которая позволяет получить объединение совокупности заданных символьных значений. Остальной алгоритм подразумевает знание конструкций цикла и ветвления и владение встроенной функцией *replace*. В то же время, если посмотреть на запись решения на языке Python, то программа практически в точности повторяет структуру алгоритма, представленную в условии задания. От учащихся требуется только помнить, что функция *replace* содержит три параметра. Первый из них указывает, какую символьную последовательность ищем в строке, второй – на какую совокупность символов происходит их замена, а третий – сколько раз повторяется команда при наличии искомых для замены символов в заданной строке. Отметим, что если не указать значение третьего параметра, он является необязательным, то произойдет замена всех найденных цепочек символьных значений. Если же задать значение этого параметра равным 1, то замена будет осуществлена однократно при поиске в строке *s* заданных символов. Цикл завершается в программе в том случае, если в строке не осталось на очередном шаге трех подряд идущих цифр '4' или '8'. Таким образом, остается вывести значение полученной после преобразования строки *s*. В результате получится строка 44844, которая будет ответом на вопрос данной задачи.

Заметим, что условие данного задания можно усложнить. Для этого достаточно поставить вопрос следующим образом: «Сколько четверок было удалено в ходе работы данной программы?» При решении «вручную» школьнику будет необходимо после анализа представленного алгоритма разделить 71 на 3 с остатком по числу удаляемых четверок, так как именно они указаны в условии оператора ветвления *if*. На данном этапе будет удалено 69 цифр '4'. Затем преобразуемая строка будет состоять из 23 восьмерок и идущих за ними 2 четверок. Так как в ней при проверке не будет записанных подряд трех цифр '4', то на следующих трех витках цикла будут последовательно заменены три цифры '8' на три цифры '4'. После этого три цифры '4' опять будут преобразованы в одну цифру '8'. Такая процедура повторится два раза. Следовательно, будет еще дважды удалено по три цифры '4', то есть шесть таких символов. Таким образом, всего с начала выполнения алгоритма будет удалено из заданной строки 75 цифр '4'. При решении задания в среде программирования Python для ответа на тот же вопрос достаточно до начала цикла *while* объявить переменную k4, служащую для подсчета количества удаленных четверок при работе программы, обнулить ее значение, а в ветке алгоритма при выполнении условия в операторе ветвления *if* увеличивать ее значение на три. Для этого можно использовать команду k4 += 3. После завершения цикла с помощью команды print(k4) останется вывести найденное значение. В данной задаче оно будет равно 75.

Если поставить еще один вопрос: «Сколько восьмерок было удалено в ходе работы данной программы?», то, рассуждая аналогичным образом, после удаления 69 четверок из заданной строки будет получено 23 цифры '8'. После этого на каждом из двух шагов, состоящих в последовательной замене три раза трех цифр '8' на одну цифру '4', будет удалено 18 восьмерок. При этом две восьмерки будут возвращены назад, следовательно, таким образом, в преобразуемой строке *s* останется 7 цифр '8'. Первые две тройки из них на завершающих витках цикла будут преобразованы в две четверки, а одна цифра '8' останется без изменений. В результате данных преобразований будет суммарно удалено 18 + 6 = 24 цифры '8'. В то же время ответ на этот вопрос также требует аналогичных действий по объявлению переменной k8. Только подсчет искомого числа восьмерок, присваивание ей исходного нулевого значения и увеличения ее значения на 3 с помощью команды k8 += 3, осуществляется в ветке *else* условного оператора *if* при ложности проверяемого в нем значения вхождения трех цифр '4' в преобразуемую строку *s*. Таким образом, будет получено искомое значение 24 на поставленный вопрос задачи.

Как видим, дополнения в программе на языке программирования Python состоят из классических действий по объявлению переменной-счетчика, присваивания ей начального значения и ее изменения в теле цикла. Они не требуют дополнительного анализа проводимых в задаче действий и позволяют значительно быстрее получить искомый результат. Также возможность мо-

дификации исходной строки *s* открывает более широкие перспективы для проведения всестороннего анализа входных данных и выявления закономерностей в получении совокупности результатов обработки значений строковых переменных. Это в значительной степени расширяет спектр обучения школьников программированию наряду с теоретической подготовкой анализу алгоритмов для формального исполнителя.

Аналогичные задания разного уровня сложности были предложены школьникам при изучении темы «Циклические алгоритмы для формального исполнителя» для отработки в практике программирования полученных умений.

Ход проводимого педагогического исследования предполагал обобщение передового педагогического опыта и обработку полученных результатов эксперимента. На практике были проведены констатирующий и формирующий этапы педагогического эксперимента, для анализа данных которых были использованы математические методы. Гипотеза исследования заключалась в том, что применение базовых алгоритмических конструкций и встроенных функций языка программирования Python для анализа и выполнения циклических алгоритмов с условием для формального исполнителя будет способствовать повышению эффективности обучения школьников.

Результаты исследования и их обсуждение

Констатирующий и формирующий этапы педагогического эксперимента параллельно проводились в двух группах школьников. Одну из этих групп составили 18 учеников 10 класса физико-математического лицея при МИФИ в г. Смоленске. Другую группу образовали 15 учеников 10 IT-класса средней школы № 6 г. Смоленска. Таким образом, в совокупности в педагогическом эксперименте приняли участие 33 десятиклассника. На констатирующем этапе педагогического эксперимента учащиеся обеих групп учились решать задачи, содержащие циклические алгоритмы с ветвлением, для формального исполнителя «на бумаге». Они знакомились с обработкой цифр, записанных как элементы строки, в цикле. При этом цикл содержал как простое ветвление с альтернативным выбором, так и ветвления с множественным выбором, но не более трех альтернатив. В завершение данного этапа обучения с учащимися было проведено промежуточное тестирование, включающее в себя 10 заданий. Эти задания школьники решали без использования инструментов языков программирования. На этапе формирующего педагогического эксперимента школьники тренировались решать как те же задачи, так и новые с переформулированным вопросом к алгоритму для исполнителя в среде языка программирования Python. Обучение состояло в записи команд для исполнителя в программах с помощью встроенных функций языка Python. Они изучали возможности применения этих команд и специфику использования в различных алгоритмических ситуациях. В завершение данного этапа педагогического эксперимента школьникам был предложен аналогичный тест, который также состоял из десяти тестовых заданий. При этом одни пять заданий были такого же уровня сложности, а другие – повышенного уровня. Тестирование проводилось в специально разработанном программном приложении «Advanced Tester», предназначенном для диагностики знаний школьников с применением методов математического моделирования [13]. В данном приложении в автоматизированном виде подбирались системы индивидуальных и групповых заданий для подготовки учащихся на протяжении всей экспериментальной деятельности. Заметим, что возможности данной интеллектуальной информационной среды также использовались для организации системы дополнительных занятий с помощью удаленного обучения школьников [14, 15]. Результаты диагностических работ, выполненных школьниками, приведены в табл. 1 и 2.

Таблица 1

Результаты констатирующего этапа педагогического эксперимента

Группа	Число школьников, достигших уровня усвоения знаний			Всего
	Высокий	Повышенный	Базовый	
СФМЛ при МИФИ	1	7	10	18
IT-класс школа № 6	2	6	7	15
Всего	3	13	17	33

Таблица 2

Результаты формирующего этапа педагогического эксперимента

Группа	Число школьников, достигших уровня усвоения знаний			Всего
	Высокий	Повышенный	Базовый	
СФМЛ при МИФИ	7	9	1	18
IT-класс школа № 6	7	7	1	15
Всего	14	17	2	33

Качественный анализ условий и результатов эксперимента

Результаты проведенного эксперимента, представленные в таблицах, свидетельствуют, что школьники намного более успешно выполняют программы для формального исполнителя на компьютере. Использование встроенных команд среды программирования Python существенным образом изменяет картину успеваемости в сторону увеличения показателей высоких баллов. Как видно из таблиц, произошло перераспределение баллов между всеми тремя уровнями усвоения знаний. Большинство школьников, продемонстрировавших повышенный уровень знаний и умений на контролирующем этапе эксперимента, достигли высокого уровня на формирующем этапе. Из 17 учащихся стабильно показывавших в обучении теоретическому анализу циклических алгоритмов базовый уровень знаний, только 2 школьника не улучшили показатели успеваемости при использовании для решения инструментальных компьютерных средств. Такие данные об обучении решению заданий с циклами для формального исполнителя можно объяснить особенностями применения мыслительных операций для получения логических выводов о выполнении программ при разных входных данных. При «ручном» решении задачи школьник играет роль компьютера, ему на бумаге необходимо анализировать массивы исходных данных. При этом нередко он может не учесть те или иные нюансы алгоритма, представленного в задаче. При компьютерном решении важно правильно реализовать программу с помощью имеющихся функций и алгоритмических структур. А затем, изменяя диапазон входных параметров, вычислять искомые значения и проверять их на соответствие сформулированным условиям алгоритма. Иными словами перенесение решений в компьютерную плоскость позволяет как обработать большие объемы информации, так и ускорить процессы их аналитического анализа. Это выступает совместно с привлекательностью применения современных IT-технологий триггером повышения результатов обучения решению такого рода заданий.

Заключение

Ход педагогического эксперимента и полученные результаты демонстрируют эффективность и обуславливают необходимость более широкого внедрения средств IT-технологий в учебный процесс. Использование систем программирования, таких как язык Python, смещает акценты обучения в практику применения прикладных программных сред. Это соответствует общим тенденциям развития IT-технологий на современном этапе развития общества. Учащиеся наряду с изучением теоретических основ получают опыт их практического приложения. Они становятся мотивированными в изучении предмета информатики, у них активно формируется фундаментальная база необходимых профессиональных знаний и навыков работы в специализированных программах на компьютере. Таким образом, школьники видят итоги своих достижений в реальном использовании инструментов решения прикладных задач, что находит отражение и в результатах практического познания приложений информатики.

Список литературы

1. 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. Белорусский государственный педагогический университет имени Максима Танка. 2019. С. 52–53.

2. 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.

3. Каган Э.М. Применение визуальных языков программирования для повышения эффективности обучения разделу «Алгоритмизация и программирование» школьного курса информатики // Вестник Московского городского пе-

дагогического университета. Серия: Информатика и информатизация образования. 2018. № 1 (43). С. 99–104.

4. Козлов С.В. Особенности обучения школьников информатике в профильной школе // Концепт. 2014. № 1. С. 31–35. URL: http://e-koncept.ru/2014/14006.htm (дата обращения: 18.04.2023).

5. 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.

6. Пушкарева Т.П., Калитина В.В., Брит А.А. Особенности обучения информатике в условиях цифровизации экономики и образования // Бизнес. Образование. Право. 2021. № 1 (54). С. 320–325.

7. Архангельская Е.В. Организация обучения основам алгоритмизации и программирования с использованием анализа математических задач // Информатизация образования и науки. 2022. № 4 (56). С. 166–175.

8. Гибадуллин А. А. Программирование компьютерных интеллектуальных игр как метод обучения информатике // Педагогика: материалы 59-й Международной научной студенческой конференции (Новосибирск, 12–23 апреля 2021 г.). Новосибирск, 2021. С. 15–16.

9. 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.

10. Козлов С.В., Быков А.А. Особенности обучения школьников олимпиадному программированию с использованием методов математического моделирования // Современные наукоемкие технологии. 2022. № 6. С. 141–146.

11. 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.

12. 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.

13. Козлов С.В., Быков А.А. Применение методов математического моделирования для диагностики знаний школьников // Современные наукоемкие технологии. 2021. № 4. С. 157–162.

14. 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.

15. Senkina G.E., Timofeeva N.M., Kiseleva O.M. Modernization of traditional educational forms in the context of distance learning // Journal of Higher Education Theory and Practice. 2022. Т. 22, № 3. P. 160–165.