УДК 378.016
DOI: http://dx.doi.org/10.21686/1818-4243-2019-2-31-41

Д.В. Моглан
Бельцкий государственный университет им. А.Руссо,
Бельцы, Республика Молдова

Дидактический потенциал использования систем визуализации алгоритмов в процессе обучения программированию

В статье рассматривается дидактический потенциал по использованию визуализатора алгоритмов как программного обеспечения, наглядно демонстрирующего работу алгоритмов по обработке набора входных данных. Описаны трудности, возникающие в ходе изучения основ алгоритмизации и программирования. В качестве вспомогательного наглядного средства обучения алгоритмизации в статье предложено использовать визуализатор алгоритмов для более успешного усвоения механизма работы алгоритмов на лекционных занятиях. В статье приведены функциональные требования к визуализатору алгоритмов, описаны основные его элементы и методические рекомендации по использованию визуализаторов алгоритмов в учебном процессе. В целях повышения эффективности учебного процесса предлагается использование визуализатора алгоритмов при обучении алгоритмизации и программированию студентов направления «Информатика» как будущих специалистов в области информационных технологий.

***Целью выполненного исследования** является повышение эффективности процесса развития алгоритмического мышления у студентов, обучающихся по направлению «Информатика», за счёт поддержки и активного применения визуализатора алгоритмов в качестве дополнительного наглядного средства обучения при изучении курса «Основы программирования».*

***Материалы и методы** исследования включают анализ научно-методической литературы по исследуемой тематике, анализ проблемы развития алгоритмического мышления у студентов, практическая реализация процесса обучения основам алгоритмизации и программирования с применением разработанного визуализатора алгоритмов, анализ результатов опроса студентов.*

***Результаты.** Проведённые автором исследования позволили создать инструментальную среду – визуализатор алгоритмов, обеспечивающую поддержку формирования навыков решения прикладных задач, способствующую повышению эффективности процесса обучения основам алгоритмизации и программирования и развитию алгоритмического мышления. Реализованы визуализаторы, позволяющие в интерактивном режиме продемонстрировать работу алгоритмов по обработке статических структур данных (массивы), что способствует осмыслению и закреплению теоретического материала. Визуализаторы внедрены в учебный процесс в Бельцком государственном университете им. А.Руссо и используются при изучении курса «Основы программирования».*

***Заключение.** В заключении сформулированы выводы о целесообразности визуального описания алгоритма, предложены способы применения визуализатора алгоритмов в рамках учебного процесса, выделены положительные факторы, влияющие на осмысление механизма выполнения алгоритма при организации обучения с использованием визуализаторов.*

***Ключевые слова:** информатика, программирование, алгоритм, визуализация алгоритмов, визуализатор, учебный процесс*

Diana V. Moglan
Alecu Russo Balti State University, Bălți, Republic of Moldova

Didactic potential of using systems for visualization of algorithms in the process of teaching programming

The article is devoted to the didactic potential on the use of the visualizer of algorithms as software, which graphically demonstrates the work of algorithms for processing input data. There are described the difficulties arising during the study of the fundamentals of algorithmization and programming. The author proposed to use the algorithm visualizer, as an auxiliary visual means of teaching algorithmization, for more successful mastering the mechanism of the algorithms at lectures. The article discusses the functional requirements for the visualizer of algorithms, describes its main elements and guidelines for the use of visualizers of algorithms in the educational process. In order to increase the effectiveness of the educational process, it is proposed to use the visualizer of algorithms for teaching algorithms and programming for students of the "Applied Informatics" direction as future specialists in the information technologies.

***The purpose of the study** is to increase the efficiency of the development of algorithmic thinking among students of the specialty "Computer Science", due to the support and active use of the algorithm visualizer as an additional visual tool for learning when studying the course "Fundamentals of Programming".*

***Materials and research methods** include the analysis of scientific and methodological literature on the subject under study, the analysis of the problem of developing algorithmic thinking by students, the practical implementation of the learning process of the basics of algorithmization and programming using the developed algorithms visualizer, the analysis of students' survey results.*

***Results.** The research conducted by the author allowed the creation of an instrumental environment - an algorithm visualizer that supports the formation of problem solving skills, enhancing the effectiveness of the learning process in the fundamentals of algorithmization and programming and the development of algorithmic thinking. Have been implemented visualizers, that allowing online demonstration of the work of algorithms for processing static data structures (arrays), which contributes to the comprehension and consolidation of theoretical material. Visualizers are introduced into the educational process at Balti State University A. Russo and are used in studying the course "Fundamentals of Programming".*

***Conclusion.** The author comes to the conclusions about the expediency of the visual description of the algorithm, proposes methods for using the algorithms visualizer in the educational process, and highlights positive factors that affect the understanding the mechanism of algorithm execution when organizing training using visualizers.*

***Keywords:** computer science, programming, algorithm, visualization of algorithms, visualizer, educational process*

Введение

В настоящее время спрос на квалифицированных ИТ-специалистов на рынке труда остаётся высоким за счёт интеграции информационных технологий в различные области народного хозяйства и науки, информационного обслуживания, государственного управления и образования, поэтому качественная подготовка будущих выпускников вузов в области информатики и информационных технологий (ИТ), способных применять современные информационные технологии для решения прикладных задач в профессиональной деятельности, является актуальной проблемой.

В Бельцком государственном университете (БГУ) им. А. Руссо Республики Молдова осуществляется подготовка студентов по направлению «Информатика». Согласно учебному плану специальности 444.1 «Информатика» и Национальной рамке профессиональных квалификаций Республики Молдова [1] будущий специалист в области информатики и информационных технологий по окончанию обучения должен обладать следующими профессиональными компетенциями:

– способностью оперировать основами речевой профессиональной культуры с использованием понятий из области информатики и математики;

– способностью разрабатывать модели для описания реальных ситуаций и процессов;

– способностью проектировать, разрабатывать и анализировать алгоритмы для решения задач из профессиональной деятельности;

– способностью программировать, разрабатывать и сопровождать ИТ-приложения на языках высокого уровня;

– способностью внедрять информационные технологии в различные отрасли профессиональной деятельности;

– способностью обрабатывать, анализировать и интерпретировать данные.

Для успешной профессиональной деятельности будущих выпускников по специальности «Информатика» определяющее значение имеет изучение основ алгоритмизации и методов обработки структур данных, освоение которых позволит создать фундамент (базовые знания, умения и навыки) для формирования профессиональных компетенций выпускников. Обучение программированию может сыграть решающую роль в улучшении когнитивных навыков студентов путём преподавания базовых алгоритмов и методов программирования, так как способствует развитию умений студентов составлять план для решения конкретной задачи, описывать в формализованном виде последовательность действий для достижения желаемого результата, прогнозировать результат выполнения спроектированных действий и вносить исправления в зависимости от полученных результирующих данных. При составлении алгоритмов у студентов вырабатываются навыки анализировать, сравнивать, обобщать, делать логические выводы, описывать план деятельности и излагать собственные мысли в строгой логической последовательности [2].

В рамках учебного плана по специальности 444.1 «Информатика» одной из базовых дисциплин является «Основы программирования», изучаемая студентами первого курса. Данная дисциплина посвящена изучению основных форм организации и хранения данных в компьютере, управляющих структур, используемых в структурном программировании (линейная структура, разветвляющаяся структура, циклическая структура), абстракции в программировании и стандартным методам обработки структур данных [3].

Роль изучения основ алгоритмизации и программирования в профессиональном становлении будущих ИТ-специалистов

Алгоритмизация является важным этапом в решении любой задачи с помощью компьютера. Развитие алгоритмического мышления является одной из основных задач при изучении программирования, так как даже простые алгоритмы трудны для понимания первокурсникам, и поэтому, сложны для изучения [4, с. 166]. Практика обучения студентов показала, что изучение языка программирования является «менее сложным» процессом, чем проектирование алгоритмов.

Успешная деятельность специалиста в области информационных технологий зависит не только от знания используемого им языка программирования при разработке приложений для решения прикладных задач, но и, в большей степени, зависит от «богатства» знаний в области построения хорошо структурированных алгоритмов обработки информации, представляемой различными структурами данных [5, 6], и накопленного опыта при решении прикладных задач.

При обучении программированию студенты довольно часто испытывают значительные трудности при поиске ответа на вопрос, почему данный алгоритм подходит для решения задачи и позволяет получить необходимые результирующие данные. Для обоснования корректности алгоритма без осознания студентами свойств объекта, выделения логических связей между отдельными объектами и отвлечения от других несущественных свойств и связей довольно сложно. Например, обладание знаниями о структуре цикла со счётчиком, не означает наличие умения на

практике построить алгоритм для вычисления среднемесячной температуры.

Согласно А.И. Газейкине, умение создать алгоритм предполагает наличие мыслительных схем, которые способствуют видению проблемы в целом, её решению крупными блоками с последующей детализацией и осознанным закреплением процесса получения конечного результата в языковых формах [7]. Эти мыслительные операции характеризуют уровень развития алгоритмического мышления студентов. Под способностью алгоритмически мыслить понимается умение решать задачи различного происхождения, требующие составления плана действий для достижения желаемого результата [8].

Основная трудность программирования состоит в том, что написание алгоритма для программы – это творческий процесс, который просто невозможно автоматизировать. В лучшем случае можно лишь создать условия для студентов, в которых этот процесс будет стимулироваться и реализовываться наиболее эффективно [9].

При подготовке к занятиям преподаватели довольно часто испытывают затруднения в отборе и систематизации теоретического материала для лекционных занятий и составлении практических заданий для самостоятельной работы с учётом количества часов по дисциплине, наличия предварительной базовой подготовки студентов и их заинтересованности в освоении нового материала по дисциплине.

При построении обучения основам алгоритмизации и программирования процесс изучения механизма работы алгоритмов с использованием дидактических материалов (книг, учебников, презентаций) реализуется обычно одним из следующих способов:

1. Статическое восприятие исходного кода алгоритма с динамической прокруткой его работы в уме.

2. Написание алгоритма на одном из языков программирования или копирование исходного кода алгоритма из учебника с целью дальнейшего его пошагового выполнения для отслеживания операций алгоритма на тестовых наборах входных данных.

Изучение работы алгоритма с помощью первого способа является достаточно трудным для большинства студентов, так как требует алгоритмического мышления и программистского воображения. Данный способ приемлем лишь для опытных программистов, а не для студентов младших курсов университета, которые только приступают к изучению основ алгоритмизации. Второй способ является более доступным и прямолинейным для первокурсников. Однако в данном случае студенты акцентируют своё внимание на программной реализации алгоритма и синтаксисе используемого языка программирования, а не на проектировании алгоритма.

Статическое изложение материала, на наш взгляд, является неэффективным с точки зрения изучения механизма работы алгоритмов. Помимо того, описанные способы изучения алгоритмов с точки зрения их использования на лекциях, также мало эффективны. Первый способ требует определённых усилий со стороны студентов в усвоении лекционного материала по реализации алгоритмов обработки информации, а второй способ и вовсе невозможно реализовать на лекционных занятиях.

На наш взгляд, для успешного усвоения механизма работы алгоритмов необходимым условием является визуальное описание алгоритма для каждого этапа его выполнения. Визуализация – это процесс представления информации, данных, знания в виде изображения, имеющий целью максимальное удобство их восприятия, понимания и анализа [10].

Визуальное представление алгоритма позволит студенту лучше понять и увидеть какие структурные элементы использованы при проектировании алгоритма, каким образом описана логика взаимодействия структурных элементов алгоритма, каким реальным объектам соответствуют определённые элементы в решаемой задачи. Следовательно, обучение алгоритмизации и программированию необходимо осуществлять с опорой на проектирование алгоритмов и дальнейшее написание программ на основе разработанных алгоритмов с помощью конструкций языка программирования. Кроме того, при изучении механизма работы алгоритмов основу должны составлять наглядные средства обучения. Для понимания структуры алгоритма и действий, которые происходят внутри алгоритма, необходимо наглядно увидеть каждый шаг выполнения алгоритма, что позволит студентам построить эффективную ментальную модель [11], имитируя поведение исполняемого алгоритма.

Использование визуализатора алгоритмов в учебном процессе

В качестве наглядного средства обучения, позволяющее реализовать такой подход, может быть использован *визуализатор алгоритмов*. Визуализаторы моделируют процессы работы алгоритмов, давая возможность учащимся при помощи интуитивно понятного пользовательского интерфейса проходить алгоритм шаг за шагом на заданном наборе данных [12].

Визуализатор алгоритмов поддерживает взаимосвязанную работу двух категории пользователей: преподаватель и студент.

Преподаватель определяет условия по выполнению алгоритма в зависимости от целей обучения (подробное или обобщённое изучение алгоритма). Например, определить набор входных данных, которые будут использованы для решения некоторой задачи; определить шаг выполнения алгоритма и др. От студентов не требуется владеть определёнными знаниями и навыками для работы с визуализатором, они осмысливают ход выполнения алгоритма и пытаются предсказать следующий шаг работы алгоритма. Кроме того, визуализаторы алгоритмов могут быть использованы для развития алгоритмического мышления не только у студентов, изучающих информатику.

Одной из сильных сторон визуализатора алгоритмов является возможность задействования слухового и зрительного канала для улучшения восприятия учебной информации в процессе обучения. Исследования Z. Katai [13] подтвердили, что, если методы и средства обучения воздействует на различные органы чувств учащихся, то они способны усвоить больше информации и эффективно поддерживать обучение при изучении алгоритмов.

Однако, логику алгоритма нельзя полностью раскрыть лишь показывая, как меняются значения переменных, участвующих в алгоритме. Студентам необходимо предоставить правильные графические представления, которые лучше соответствуют их ментальным моделям по выполнению конкретного алгоритма. Визуализатор алгоритмов позволяет помочь студентам построить правильные ментальные модели, связать структурные компоненты в иерархии построения алгоритма и обобщить модель алгоритма при решении подобных задач, визуализируя абстрактные концепции и разворачивая основную логику исследуемого алгоритма.

Отметим отличительные характеристики визуализаторов [14]:

1. Интерактивность и наглядность при управлении процессом визуализации со стороны пользователя.

2. Простота использования интерфейса визуализатора.

3. Отображение хода выполнения алгоритма. Визуализатор отображает как изменения значений переменных алгоритма, так и все действия над ними.

4. Наличие пояснений. По ходу работы визуализатора отображаются текстовые комментарии, поясняющие каждый шаг алгоритма.

5. Поддержка двух режимов визуализации: пошагового и автоматического.

В учебном процессе визуализаторы алгоритмов могут быть использованы при соблюдении следующих требований:

– Отображение входных и выходных данных в наглядной форме, на которых демонстрируется выполнения алгоритма – учащимся должна быть предоставлена возможность определять произвольные наборы входных данных и просматривать для них исполнение алгоритма.

– Просмотр действий алгоритма для всех визуализируемых элементов – учащимся должна быть предоставлена возможность просматривать не только изменения, произведённые визуализатором в наборе данных, но и другие действия, например, сравнения при поиске минимального элемента в массиве.

– Вывод комментариев к выполнению алгоритма – в процессе работы алгоритма на каждом его шаге должны выводиться комментарии, которые бы поясняли все производимые действия.

– Пошаговое отображение работы алгоритма – учащиеся, исходя из собственных способностей в усвоении материала, должны иметь возможность проследить за действиями выполнения алгоритма от начала и до конца, шаг за шагом.

Согласно исследованиям C. Hundhausen [15] студенты должны быть настолько активными в процессе обучения алгоритмизации и программированию, насколько это возможно во время визуализации. Кроме того, студенты более вовлечены в процесс обучения, когда они могут самостоятельно использовать средства по визуализации механизма выполнения алгоритма. В работе T. Naps [16] описана таксономия, которая определяет уровень и вид деятельности студента в процессе работы с визуализатором, включающая шесть уровней:

1. Без просмотра: в этом случае визуализатор не используется совсем.

2. Просмотр: студенты рассматривают шаги выполнения алгоритма в визуализаторе.

3. Ответ: студенты отвечают на вопросы преподавателя во время визуализации.

4. Изменение: студенты могут изменять входные данные во время визуализации для исследования поведения алгоритма в различных ситуациях.

5. Построение: студенты самостоятельно строят алгоритм после визуализации без его кодирования на языке программирования.

6. Представление: студенты объясняют шаги алгоритма на примере визуализируемого алгоритма и участвуют в обсуждении с коллегами группы.

Данная таксономия соотносится с таксономией учебных целей по Блуму [17]. Рассмотрим соотношение познавательных способностей с конкретными действиями студентов и примерами заданий для темы «Одномерные массивы».

Знание.

Описание действия: называть и неформально определять конкретные понятия, связанные с типом данных «одномерный массив».

Примеры заданий:

– определить следующие понятия: массив данных, одномерный массив, индекс элемента, значение элемента;

– назвать базовые алгоритмы для обработки элементов массива;

– перечислить алгоритмы сортировки и поиска.

Понимание.

Описание действия: понимать общий принцип алгоритма и объяснять словесно, как он работает; определять понятия формально, то есть распознать их существенные характеристики; понимать ключевые понятия, связанные с алгоритмом и их роль в алгоритме; понимать поведение алгоритма в наихудшем и в наилучшем случаях.

Примеры заданий:

– каким образом структура данных (одномерный массив) описывает исследуемый объект;

– представить данные в форме одномерного массива;

– привести примеры для одномерных массивов;

– объяснить механизм прямого доступа к элементу одномерного массива;

– объяснить алгоритм для последовательной обработки элементов одномерного массива;

– объяснить алгоритм для заполнения (вывода) одномерного массива;

– проиллюстрировать поиск минимального (максимального) элемента в массиве;

– привести примеры для алгоритма поиска в массиве элемента с заданным свойством;

– выделить закономерности в работе алгоритмов сортировки;

– записать базовые алгоритмы для обработки одномерного массива (ввод и вывод элементов массива, суммирование элементов массива, подсчёт количества элементов массива, обладающих заданным свойством и др.) с помощью некоторого языка программирования.

Применение.

Описание действия: интерпретировать алгоритмические конструкции в конструкции языка программирования; проверять корректность выполнения алгоритма; адаптировать ранее изученный алгоритм для определённой задачи и конкретного представления данных; определять наилучший и наихудший случай при анализе базовых алгоритмов.

Примеры заданий:

– применить алгоритм для сортировки элементов массива, используя сортировку вставкой, и продемонстрировать корректность его работы;

– записать алгоритм, используя конструкции языка программирования, для поиска в массиве заданного элемента с помощью алгоритма двоичного поиска и определить промежуточные результаты в работе программы;

– решить подобные задачи, используя аналогичный алгоритм (например, вычисление среднемесячной температуры и вычисление среднего дохода предприятия за месяц).

Анализ.

Описание действия: понимать связь алгоритма с другими алгоритмами, применяемые для решения подобных задач; рассуждать, противоречить и/или доказывать правильность алгоритма; анализировать сложную задачу, выявлять в ней значимые объекты и разбивать её на более мелкие подзадачи.

Примеры заданий:

– определить необходимые входные и промежуточные параметры для корректной работы алгоритма (поиска, сортировки);

– определить алгоритмы, необходимые для решения конкретной задачи (например, определить, если была зарегистрирована температура выше 20 градусов);

– реализовать трассировку выполнения алгоритма сортировки массива из 10 целых чисел;

– проверить влияние различных входных параметров на работу алгоритма, в том числе и особые случаи;

– сравнить производительность работы алгоритмов сортировки;

– выявить сходства и отличия в работе алгоритмов поиска;

– провести анализ текста программы для выявления синтаксических и логических ошибок.

Синтез.

Описание действия: разрабатывать алгоритмы для сложных задач, в которых требуются различные структуры данных, алгоритмы и методы; анализировать сложность и эффективность алгоритма; определять критерии для сравнения различных решений задачи.

Примеры заданий:

– разработать программу с использованием массивов и конструкций языка программирования, адекватных алгоритму, для решения сложных задач практической направленности (например, описать необходимые параметры для хранения и обработки двух многочленов степени n, заданных своими коэффициентами, и реализовать меню для выполнения следующих команд: вычислить значение многочлена для заданного значения переменной x, выполнить сложение двух многочленов, выполнить вычитание двух многочленов);

– определить набор тестовых значений для проверки правильности выполнения программы.

Оценка.

Описание действия: аргументировать необходимость модифицировать используемый алгоритм или объединить его с другими алгоритмами для более эффективного решения новой более сложной задачи; обсуждать плюсы и минусы различных алгоритмов, которые используются для реше-

ния одинаковых или подобных задач; проводить оценку разработанных алгоритмов.

Примеры заданий:

– оценить разработанный алгоритм (универсальность, сложность и др.) и аргументировать решение о его соответствии поставленной задаче;

– обсудить дизайн решения задачи и его преимущества перед другими решениями;

– определить возможные пути усовершенствования алгоритма.

Предложенные формулировки действий и заданий позволяют определить взаимосвязь между формой взаимодействия учащегося с визуализатором и уровнями результатов обучения (согласно учебным целям по Блуму), на которые влияет это взаимодействие.

Существует множество примеров систем для визуализации алгоритмов, которые могут быть использованы на уроках информатики. В отношении уже разработанных визуализаторов алгоритмов следует отметить следующие: AlgorihtmVisualizer (http://algo-visualizer.jasonpark.me/), анимированные визуализации структур данных VisuAlgo (https://visualgo.net/ru), курс «Дискретная математика: Алгоритмы» (http://rain.ifmo.ru/cat/view.php/vis). Данные визуализаторы стали опорными трудами для разработки собственных программных продуктов для поддержки лекционного курса по изучению алгоритмов. К разработке программ-визуализаторов были также привлечены студенты в рамках выполнения дипломной работы.

В качестве среды разработки программ-визуализаторов была выбрана среда программирования Delphi, включающая достаточно многообразные инструменты для создания оболочек визуализаторов алгоритмов. Помимо того, её использование, когда в создании программ-визуализаторов занята группа разработчиков (студенты и преподаватели), обеспечивает единый подход в реализации дизайна готовых продуктов.

Рис. 1. Визуализация алгоритма суммирования элементов массива

Пример разработанного программного продукта для визуализации работы алгоритмов по учебному разделу «Статические структуры данных. Одномерные массивы» представлен на рисунке 1.

Графический пользовательский интерфейс программы-визуализатора выполнен в форме оконного приложения, которое содержит три функциональные области (визуализация, код алгоритма, комментарии) и панель управления.

Окно визуализации используется для размещения элементов на экране и для просмотра работы алгоритма. *Окно кода* используется для просмотра пользователем кода алгоритма. *Окно комментариев* используется для вывода пояснений к текущему действию алгоритма.

Управление процессом визуализации алгоритмов осуществляется с помощью панели управления, на которой расположены следующие элементы интерфейса:

– *Инициализировать* – кнопка для инициализации работы визуализатора, при запуске визуализатора входные наборы данных генерируются случайным образом;

– *Далее* – кнопка для пошагового выполнения алгоритма, то есть при каждой активации кнопки выполнится один шаг алгоритма;

– *Все сразу* – кнопка для автоматического выполнения алгоритма без вмешательства в процесс работы алгоритма;

– *Пауза* – кнопка для временного приостановления работы визуализатора;

– *Остановить* – кнопка для полного приостановления работы визуализатора алгоритма;

– *Время задержки* – кнопка для управления скоростью визуализации процесса выполнения алгоритма (указывается в миллисекундах).

Рассмотрим применение разработанного визуализатора на примере реализации алгоритма поиска минимального элемента в одномерном массиве из *n* целых чисел. Задача поиска минимального элемента в массиве решается с помощью следующего алгоритма:

Min: = A[1];
For i: =2 to n do
If A[i] < Min then
Min: = A[i];

В данном алгоритме *A* представляет собой одномерный массив, в котором производится поиск минимального элемента, *Min* – значение текущего минимума (среди первых *i* элементов массива). Изначально значение минимума

Рис. 2. Визуализация алгоритма поиска минимального элемента в массиве

инициализируется значением первого элемента массива.

В визуализаторе основными данными являются элементы массива и значение текущего минимального элемента (рис. 2). Для наглядного представления работы алгоритма значения элементов массива и минимального элемента должны быть постоянно показаны в окне визуализации. При поиске минимального элемента массива наиболее «интересными» событиями в алгоритме являются операции сравнения значения текущего элемента массива с минимальным значением в условном операторе и связанная с ним операция обновления текущего минимального значения.

На каждом шаге алгоритма текущий элемент массива выделяется синим цветом, а в случае нахождения нового минимального значения соответствующий элемент выделяется красным цветом. Для пояснения работы алгоритма в окне комментариев выводится сообщение о выполненной операции на текущем шаге алгоритма. Отметим, что на шаге проверки на обновление минимального значения, который соответствует условному оператору, возможны два случая: для ложного и истинного условий. На каждом шаге алгоритма значения параметров вычисляются и подставляются в ходе выполнения алгоритма.

Рассмотрим работу визуализатора на примере реализации алгоритма сортировки методом простого выбора в одномерном массиве из *n* целых чисел. Задача сортировки элементов массива решается с помощью следующего алгоритма:

```
For i:= 1 to n-1 do
begin
   Min:= A[i];
   i_min:= i;
   For j:= i+1 to n do
      If A[j] < Min then
      begin
         Min:= A[j];
         i_min:= i;
      end;
   A[i_min]:= A[i];
   A[i]:= Min;
end;
```

В данном алгоритме *A* представляет собой одномерный массив, в котором выполняется сортировка методом простого выбора, *i* – позиция текущего элемента, *Min* – значение текущего минимума и *i_min* – позиция минимального элемента среди элементов от *i+1* до *n*. Изначально значение минимума инициализируется значением текущего элемента массива. После нахождения минимального элемента, он меняется местами с текущим элементом массива.

Визуализатор отображает массив в виде смежных ячеек (рис. 3), которые могут перемещаться во время выполнения алгоритма, чтобы таким образом помочь студентам преодолеть трудности в осво-

Рис. 3. Визуализация алгоритма сортировки методом простого выбора

ении механизма работы алгоритма и построить эффективные ментальные модели [10], основываясь на предыдущем полученном опыте в процессе усвоения материала курса. Действия, которые являются критическими для внутренней логики алгоритма, описываются как «интересные» события (например, замена местами двух элементов массива при сортировке). Во время выполнения алгоритма в визуализаторе каждый шаг, соответствующий текущему событию, детализируется в окне комментариев, что помогает студентам связать действие алгоритма с соответствующим графическим представлением. Другой важной особенностью визуализатора является анимация управляющих переменных (индексы *j, i_min*), которые выделяются соответственно синим и красным цветом, что способствует пониманию студентами механизма выполнения алгоритма при автоматическом изменении значений индексов массива.

По сравнению с другими способами ведения лекционного занятия по объяснению алгоритмов можно выделить наиболее значимые преимущества в применении визуализаторов:

– визуализация передаёт динамическое понятие алгоритма;

– визуальное представление работы алгоритма может помочь преподавателям на лекционных занятиях затратить меньше времени на объяснение теоретического материала и продемонстрировать студентам на практике работу алгоритмов. Преподаватели могут продемонстрировать поведение алгоритма на различных наборах входных данных, используя мультимедиа-проектор в аудитории, что позволит студентам получить более обширное представление о работе обсуждаемого алгоритма, сложности алгоритма и различиях в его поведении в ответ на определённые наборы данных.

– динамическая визуализация выполнения алгоритма позволяет наглядно показать такую значимую характеристику алгоритма, как его трудоёмкость, особенно в ходе пошаговой демонстрации, аналогично режиму трассировки программ.

В заключение систематизируем результаты, полученные в процессе усвоения студентами содержание раздела «Статические структуры данных» в рамках курса «Основы программирования» с использованием визуализатора алгоритмов. Опыт показал, что при изучении теоретического материала с использованием визуализатора алгоритмов в ходе лекционных занятий целесообразно использовать следующие формы и приёмы обучения:

– предварительное самостоятельное знакомство студентов с изучаемым алгоритмом с помощью визуализатора;

– коллективный разбор различных алгоритмов для заданного набора входных данных с использованием компьютера и мультимедиа-проектора, поясняющих в наглядной форме основные понятия изучаемого алгоритма;

– систематический опрос в ходе лекционного занятия;

– предоставление студентам после лекции всех визуализаторов в электронном формате;

– определение результатов работы визуализатора для определённого алгоритма (пошаговая трассировка).

По окончании изучения раздела «Статические структуры данных» в рамках курса «Основы программирования» было проведено анкетирование студентов с целью выявления мнения студентов относительно влияния визуализатора алгоритмов на их учебную деятельность, практичности и полезности применения визуализатора алгоритмов в ходе усвоения механизма работы алгоритма. В опросе приняли участие студенты 1 курса (26 участников), обучающихся по специальности «Информатика». Результаты опроса приведены в таблице.

Результаты проведения анкетирования студентов также показали следующее:

1. 78% студентов отметили, что визуализация работы алгоритма помогла им определить свои логические ошибки в алгоритме и изменить код программы для решения поставленных задач, 90% – осмыслить процесс выполнения алгоритма, так как динамическая визуализация позволяет быстрее и легче понять работу алгоритма, чем словесное объяснение.

Таблица

Результаты опроса студентов

№	Вопросы	Ответы студентов	
		Положит. (%)	Отрицат. (%)
1.	Мне было легко понять, как работает визуализатор алгоритмов	87	13
2.	Мне было легко использовать визуализатор алгоритмов	89	11
3.	Вывод комментариев при пошаговой демонстрации помогли мне понять механизм работы алгоритма	78	22
4.	Динамическая визуализация работы алгоритма помогла мне связать команды алгоритма с их назначением	92	8
5.	Работа с визуализатором помогла мне решить задачи, в которых применяются алгоритмы по обработке статических структур данных (массивы).	82	18

2. 84% опрошенных среди плюсов применения визуализаторов алгоритмов в учебном процессе указали возможность экспериментировать с визуализацией работы алгоритма для различных ситуаций, 88% – возможность улучшить понимание как теоретических, так и практических вопросов, поскольку визуализатор отображает ход выполнения алгоритма и изменение состояний выходных данных, 54% – возможность активного вовлечения в учебный процесс, 72% – возможность получить немедленную обратную связь на выполненное действие, а для 56% опрошенных важной является возможность обучения в своём собственном режиме и ни один студент не отметил, что применение визуализатора алгоритмов не является эффективным средством при изучение алгоритмов.

3. 86% студентов отметили, что применение визуализаторов алгоритмов повышает увлекательность занятий, 61% – увеличивает активность познавательного процесса и интенсивность учёбы, 75% – повышает активность студентов на занятиях и продуктивность взаимодействия с преподавателем.

В целом, студенты положительно оценили визуализатор и вклад визуализации в развитие алгоритмического мышления, преодоление трудностей в изучении алгоритмов по обработке статических структур данных (массивы) и решение задач на основе рассмотренных алгоритмов.

Заключение

Визуализация алгоритма предоставляет альтернативный способ в объяснении тем по основам алгоритмизации и программирования студентам, которые легче усваивают материал при его визуальном способе представлении по сравнению со словесным способом. На лекционном занятии студенты в результате извлекут больше полезной информации, если визуализаторы будут использованы для объяснения хода работы алгоритмов. Конечно, визуальное представление материала не ново, так как большинство преподавателей используют дополнительные наглядные материалы (слайды, плакаты) в ходе объяснения лекционного курса. Визуализаторы алгоритмов используются нами как вспомогательное средство обучения для наглядного представления работы алгоритма.

Визуализаторы могут быть использованы для демонстрации работы алгоритмов как в старших классах на уроках информатики, так и в университете при изучении основ алгоритмизации и программирования, так как позволяет в наглядной и простой форме изучить простейшие алгоритмы и усвоить механизм их работы.

Литература

1. Cadrul național al calificărilor: Învățământul Superior. Chișinău: Bons Ofices, 2015. 493 с. [Электрон. ресурс] Режим доступа: https://mecc.gov.md/sites/default/files/cnc_22_31_32_33_34_38_42_44_55_85.pdf

2. Нефедова В.Ю., Прилепина А.В. О дидактических средствах изучения содержательной линии информатики «Алгоритмизация и программирование» [Электрон. ресурс] // Письма в Эмиссия. Оффлайн. 2018. №4 (апрель). Режим доступа: http://www.emissia.org/offline/2018/2607.htm

3. Дейнего Н.В., Кабак В.И., Моглан Д.В. Основы программирования: Базовый курс. Бэлць: Бельцкая университетская пресса, 2016. 234 с.

4. Дробушевич Л.Ф., Конах В.В. Использование визуальных технологий в процессе обучения программированию // Информатизация образования – 2010: педагогические аспекты создания информационно-образовательной среды: материалы межд. науч. конф. Минск, 2010. С. 166-170.

5. Кнут Д. Искусство программирования. Том 1. Основные алгоритмы. М.: Вильямс, 2002. 720 с.

6. Кормен Т., Лейзерсон Ч., Ривес Р. Алгоритмы. Построение и анализ. М.: МЦНМО, 2001. 960 с.

7. Газейкина А.И. Стили мышления и обучение программированию студентов педагогического вуза [Электрон. ресурс] // Информационные технологии в образовании. 2006. Режим доступа: http://ito.edu.ru/2006/Moscow/I/1/I-1-6371.html

8. Еремин О.Ф. Методическое пособие по программированию на языке Паскаль. М.: Моздок, 2009. 49 с.

9. Смольянинов А.В. Возможности среды визуального программирования BLS для использования в учебном процессе // Компьютерные инструменты в образовании. 1999. № 1. С. 15–21.

10. Баженова И.В. Визуальное моделирование в обучении программированию // Вестник Челябинского государственного педагогического университета. Педагогические науки. 2017. № 6. С. 15–20.

11. Калитина В.В. Методика ментального обучения программированию студентов информационных направлений подготовки // Вестник Красноярского государственного педагогического университета им. В.П. Астафьева. 2015. №1 (31). С. 45–48.

12. Казаков М.А., Шалыто А.А. Использование автоматного программирования для реализации автоматов // Компьютерные инструменты в образовании. 2004. №2. С. 19–33.

13. Katai Z., Juhasz K., Adorjani A.K. On the role of senses in education // Computers & Education. 2008. Vol. 51 (4). C. 1707-1717. DOI: 10.1016/j.compedu.2008.05.002.

14. Avancena A.T., Nishihara A. Usability and pedagogical assessment of an algorithm learning tool: A case study for an introductory programming course for high school [Электрон. ресурс] // Issues in Informing Science and Information Technology. 2015. Vol. 12. P. 21-43. DOI: 10.28945/2184. Режим доступа: http://iisit.org/Vol12/IISITv12p021-043Avancena1817.pdf

15. Hundhausen C., Douglas S.A., Stasko J.T. A meta-study of algorithm visualization effectiveness [Электрон. ресурс] // Journal of Visual Languages and Computing. 2002. Vol. 13. C. 259-290. DOI: 10.1006/S1045-926X(02)00028-9. Режим доступа: https://www.cc.gatech.edu/~john.stasko/papers/jvlc02.pdf

16. Naps T. L., Roessling G., Almstrum V., Dann W., Fleischer R., Hundhausen C.D. Exploring the role of visualization and engagement in computer science education // SIGCSE Bulletin. 2003. Vol. 35(2). P. 131-152. DOI: 10.1145/960568.782998.

17. Bloom B.S. Taxonomy of educational objectives: The classification of educational goals: Handbook I: Cognitive domain. New York: Longman, 1994. 207 p.

References

1. Cadrul național al calificărilor: Învățământul Superior. Chișinău: Bons Ofices; 2015. 493 p. [Internet] URL: https://mecc.gov.md/sites/default/files/cnc_22_31_32_33_34_38_42_44_55_85.pdf

2. Nefedova V.Yu., Prilepina A.V. On the didactic means of studying the informatics content line “Algorithmization and Programming” [Internet]. Pis’ma v Emissiya. Offlayn = Letters to the issue. Offline. 2018; 4 (April). URL: http://www.emissia.org/offline/2018/2607.htm (In Russ.)

3. Deynego N.V., Kabak V.I., Moglan D.V. Osnovy programmirovaniya: Bazovyy kurs. = Basics of programming: Basic course. Balti: Balti University Press; 2016. 234 p. (In Russ.)

4. Drobushevich L.F., Konakh V.V. The use of visual technology in the process of learning programming.Informatizatsiya obrazovaniya – 2010: pedagogicheskiye aspekty sozdaniya informatsionno-obrazovatel’noy sredy: materialy mezhd. nauch. konf. = Informatization of education – 2010: pedagogical aspects of creating an informational and educational environment: materials of scientific conference. Minsk; 2010: 166-170. (In Russ.)

5. Knut D. Iskusstvo programmirovaniya. Tom 1. Osnovnyye algoritmy. = The art of programming. Volume 1. The main algorithms. Moscow: Williams; 2002. 720 p. (In Russ.)

6. Kormen T., Leyzerson Ch., Rives R. Algoritmy. Postroyeniye i analiz = Algorithms. Construction and analysis. Moscow: MTSNMO; 2001. 960 p. (In Russ.)

7. Gazeykina A.I. Thinking styles and programming training for students of a pedagogical university [Internet].Informatsionnyye tekhnologii v obrazovanii = Information technologies in education. 2006. URL: http://ito.edu.ru/2006/Moscow/I/1/I-1-6371.html (In Russ.)

8. Eremin O.F. Metodicheskoye posobiye po programmirovaniyu na yazyke Paskal’ = Methodical manual on programming in Pascal. Moscow: Mozdok; 2009. 49 p. (In Russ.)

9. Smol’yaninov A.V. Features of the visual programming environment BLS for use in the educational process.Komp’yuternyye instrumenty v obrazovanii = Computer tools in education. 1999; 1: 15-21. (In Russ.)

10. Bazhenova I.V. Visual modeling in teaching programming.Vestnik Chelyabinskogo gosudarstvennogo pedagogicheskogo universiteta. Pedagogicheskiye nauki = Bulletin of the Chelyabinsk State Pedagogical University. Pedagogical sciences. 2017; 6: 15-20. (In Russ.)

11. Kalitina V.V. Methodology of mental training in programming students of information areas of training.Vestnik Krasnoyarskogo gosudarstvennogo pedagogicheskogo universiteta im. V.P. Astaf’yeva = V.P. Astafieva Bulletin of the Krasnoyarsk State Pedagogical University. 2015; 1 (31): 45–48. (In Russ.)

12. Kazakov M.A., Shalyto A.A. The use of automaton programming for the implementation of automata. Komp’yuternyye instrumenty v obrazovanii = Computer tools in education. 2004; 2: 19–33. (In Russ.)

13. Katai Z., Juhasz K., Adorjani A.K. On the role of senses in education.Computers & Education. 2008. Vol. 51 (4): 1707–1717. DOI: 10.1016/j.compedu.2008.05.002.

14. Avancena A.T., Nishihara A. Usability and pedagogical assessment of an algorithm learning tool: A case study for an introductory programming course for high school [Internet]. Issues in Informing Science and Information Technology. 2015. Vol. 12: 21-43. DOI: 10.28945/2184. URL: http://iisit.org/Vol12/IISITv12p021-043Avancena1817.pdf

15. Hundhausen C., Douglas S.A., Stasko J.T. A meta-study of algorithm visualization effectiveness [Internet].Journal of Visual Languages and Computing. 2002. Vol. 13: 259-290. DOI: 10.1006/S1045-926X(02)00028-9. URL: https://www.cc.gatech.edu/~john.stasko/papers/jvlc02.pdf

16. Naps T. L., Roessling G., Almstrum V., Dann W., Fleischer R., Hundhausen C.D. Exploring the role of visualization and engagement in computer science education. SIGCSE Bulletin. 2003. Vol. 35(2): 131-152. DOI: 10.1145/960568.782998.

17. Bloom B.S. Taxonomy of educational objectives: The classification of educational goals: Handbook I: Cognitive domain. New York: Longman; 1994. 207 p.

Сведения об авторе

Диана Васильевна Моглан
К. пед. н., доцент кафедры математики и информатики факультета точных наук, экономики и окружающей среды
Бельцкий государственный университет им. А.Руссо,
Бельцы, Республика Молдова
Эл. почта: di_2008@mail.ru

Information about the author

Diana V. Moglan
Cand. Sci. (Pedagogy), Associate Professor of the Department of Mathematics and Computer Science, Faculty of Exact Sciences, Economics and the Environment
Alecu Russo Balti State University, Bălți, Republic of Moldova
E-mail: di_2008@mail.ru