УДК 373

Э.М. Каган

Применение визуальных языков программирования для повышения эффективности обучения разделу «Алгоритмизация и программирование» школьного курса информатики

В статье рассматривается вопрос применимости визуальных языков и сред программирования в качестве средства обучения школьников программированию. Дается описание проблемы формирования навыка программирования на текстовых языках при использовании классических средств обучения. Выделяется возможное место применения визуальных языков и сред программирования в рамках курса, предполагаются достижимые результаты использования новых средств. Делается вывод о возможном пути развития средств обучения программированию.

Ключевые слова: обучение программированию; средства обучения; визуальное программирование; текстовое и блочное описание алгоритма.

Несмотря на значительную переориентацию курса информатики, вызванную повышением доли материала, связанного с информационными технологиями, обучение программированию все еще остается одной из основных линий обучения [3]. Но в связи с этой перестройкой курса раздел «Алгоритмизация и программирование» сокращается и, как следствие, преподаватели вынуждены искать как дополнительные часы обучения, так и более эффективные средства для изучения этого курса.

Качественное преподавание данного раздела необходимо, ведь с точки зрения образовательного стандарта ученики должны сформировать логический, алгоритмический и системный стили мышления, так как без них невозможно достижение личностной ориентации в полном ее смысле [4–6; 8]. Известно, что для создания условий успешного развития личности сам учащийся должен ставить цели, находить пути их достижения и систематически работать над ними.

На данный момент большая часть школьных учителей информатики использует в своей работе классические учебные языки программирования, такие как Basic, Pascal и LOGO [1]. Наличие большого количества учебного материала и хорошо проработанная методическая база для этих языков являются здесь хорошим подспорьем.

Однако современные ученики воспринимают компьютер как обыденную, повседневную вещь и, как следствие, не обладают достаточной мотивацией

для его глубокого изучения. В некоторых случаях данную проблему позволяют решить углубленные курсы за счет увеличения применимости получаемых знаний. Например, курсы по приобретению навыка веб-разработки, программирования микроконтроллеров, создания роботехнических устройств и т. д. Однако в большинстве своем такие курсы основаны на профессиональных языках, таких как C, Python или Java, а их изучение обычно ограничено только старшими классами школы.

В связи с очевидной необходимостью ускорения обучения и повышения эффективности курса информатики иностранные специалисты уже на протяжении нескольких десятилетий занимаются разработкой специализированного программного обеспечения и языков программирования, ориентированных исключительно на обучение школьников [7]. Ярким примером такого решения, которое стало популярным и в России, можно считать среду Scratch, разработанную в медиалаборатории Массачусетского технологического университета. Многие исследователи признают, что применение данного программного обеспечения позволяет повысить качество обучения курсу информатики. Следует заметить, что Scratch, хотя и является в большей степени наследником текстовых языков программирования, тем не менее использует и элементы языков визуальных. Программирование в данной среде ведется при помощи создания блочных диаграмм, в которых связи между блоками отображаются в виде мозаики, при этом для разных элементов программы предусмотрены различные отображения, что позволяет сделать программирование более интуитивным.

Такое совмещение блочного и текстового подходов к описанию алгоритма позволяет улучшить восприятие программы, снижая когнитивную нагрузку, так как отображение использует метафору, сходную по поведению с объектом реального мира (мозаика). Согласно работе Д. Смита для достижения лучшей читаемости, легкости навигации и простоты редактирования алгоритмических структур их нотация должна отображать связность и сами элементы таким образом, чтобы отношения между ними были близки к поведению объектов реального мира [13].

Данное заключение является логическим развитием идеи, заложенной А. Сломаном еще в 1971 году [12]. Он предложил разделить все визуальные отображения на две большие категории: аналогические и предикатные. К аналогическим в данном случае следует отнести все визуальные структуры, сохраняющие атрибуты, необходимые для понимания отображаемой системы. Иллюстративным примером аналогического отображения может служить карта местности: при взгляде на карту пользователь может определить отношения между объектами на ней, при этом отношение между объектами в реальности будет тем же (с точностью до масштаба). Однако следует отметить и тот факт, что карта местности в большинстве случаев сохраняет только внешние атрибуты строений, то есть она не позволяет работать с внутренними помещениями, высотами, температурой, влажностью и т. д. Именно по этой причине

существует большое количество различных видов карт, каждый из которых позволяет аналогически отобразить тот или иной набор атрибутов системы. Наиболее важной для нас в данном случае характеристикой аналогического отображения будет являться отсутствие потребности в предварительных знаниях об отображаемой системе.

Категория предикативных отображений основана, как следует из названия, на логике предикатов, разработанной Г. Фреге (в некоторых источниках предикативные отображения именуются фрежановскими по фамилии автора). Таким образом, в отображении существует всего один вид отношений, что не позволяет воссоздать структуру отображаемой системы. Предикативные отображения воссоздают поведение системы, но не ее структуру. Примером этого может служить программа, написанная на любом императивном языке программирования. После выполнения инструкции происходит изменение состояния некоторой области памяти, зарезервированной за программой, однако в исходном коде мы видим лишь причину этого изменения, заданную в форме некоторой команды. Как следствие, для понимания структуры и процессов, происходящих внутри системы, необходимо пошагово пройти все команды программы, имитируя поведение исполнителя, а также воссоздавая действительную структуру памяти.

Именно по этой причине отладка программ требует зачастую не только знания языка программирования, но и понимания всей структуры программы в целом. На данном примере можно наглядно видеть, что любое предикативное отображение требует дополнительных знаний об исполнителе и структуре окружающего мира, отображаемого системой.

По этой причине для работы с текстовыми языками становится необходимым обладание умением работать с символьными системами, что требует специфических мыслительных навыков, развитого интеллекта и дополнительных знаний. Для овладения подобным представлением знаний, согласно Ж. Брунеру и Ж. Пиаже [9; 10], необходимо пройти эволюционный путь, состоящий из трех представлений: действенное, иконическое и символьное. Действенное представление является наиболее простым и связано с овладением некоторыми навыками действий в реальном мире. Например, навык хождения формируется и развивается у ребенка только как следствие непосредственных попыток ходить. Более того, научиться ходить, изучая книгу или диаграмму алгоритма, невозможно. Иконическое представление знаний можно проиллюстрировать с помощью фотографии. Рассматривая фотографию некоего животного, ребенок может сформировать знание о том, какое это животное. При этом следует отметить, что узнать тип животного можно и при помощи действия: посмотреть на животное в реальном мире. И наконец, символьное представление позволяет нам описывать знания при помощи некоторой формальной нотации. Здесь примером может служить любая статья, журнал или книга. Знание в них передается не с помощью аналогических отображений, а с помощью некоторой символьной метасистемы понятий (схожей

по своей структуре с предикативным отображением), которую принято называть естественным языком (русский, английский и т. д.).

Таким образом, можно прийти к заключению, что даже если предположить, что учащийся уже овладел деятельностным представлением с помощью непосредственного выполнения действия и иконическим отображением, выполняя задачи по оформлению диаграмм алгоритма на уроках информатики, то переход от иконического представления к символьному остается непроработанным в рамках этого курса. Данная проблема может быть частично решена при помощи использования визуальных языков программирования. Упомянутый выше Scratch позволяет оформлять алгоритм программы при помощи блоков. Такой подход к описанию алгоритмов является схожим с классическими диаграммами алгоритмов, навык оформления которых уже есть у учащихся, что в значительной степени снижает когнитивную нагрузку и повышает эффективность обучения [2].

С другой стороны, блоки языка Scratch содержат некоторые элементы текстовой программы, которые по определению являются символьным представлением, что, однако, при поддержке мозаичного механизма и упрощения синтаксиса, позволяет учащемуся практически незаметно для себя начать работу с формальной нотацией языка программирования. Некоторые исследователи также отмечают, что переход к классическим текстовым языкам программирования после практики на одном из визуальных языков происходит значительно легче, а качество полученных знаний и эффективность курса в целом повышаются [11].

Переходя к выводам, следует отметить ряд важных моментов. Развитие современных вычислительных систем постепенно приводит к тому, что все большее количество программ начинают утилизировать возможности параллельного исполнения, а даже современные языки и среды программирования, основанные на редактировании текста, не позволяют в полной мере отобразить поведение такого рода, что в значительной степени усложняет и разработку программ, и их последующую отладку. Графические возможности современных персональных компьютеров уже позволяют взаимодействовать с достаточно большим количеством визуальной информации. По мнению некоторых исследователей, уже сейчас становится возможным вести разработку программ, практически полностью отказавшись от текста. Блочное отображение структуры программы является более наглядным, снижает когнитивную нагрузку и позволяет избежать многих потенциальных ошибок. По этой причине его применение является закономерным в отношении развития образовательных языков программирования. Применение визуальных сред и языков программирования при обучении позволяет учащемуся пройти эволюцию знаний от деятельностного представления к символическому без пропуска одного из этапов, и, как следствие, это должно приводить к повышению качества обучения.

В заключение хотелось бы предположить, что в ближайшие два десятилетия применение визуальных сред и языков программирования станет повсеместным, в том числе и при обучении школьников. Это будет связано не только с развитием самих языков и сред, но и с изменением самого принципа взаимодействия

с компьютером. Таким образом, остается надеяться, что в ближайшее время свет увидят новые среды, а методика преподавания начнет свое систематическое изменение с целью соответствия современным техническим возможностям.

Литература

1. *Безруков Я., Скворцов О.* О выборе языка для обучения программированию в школе // Вестник Новосибирского государственного университета. Серия «Педагогика». 2015. № 1. С. 99–103.

2. *Григорьев С.Г., Гриншкун В.В., Реморенко И.М.* «Умная аудитория»: от интеграции технологий к интеграции принципов // Информатика и образование. 2013. № 10 (249). С. 3–8.

3. *Гриншкун В.В., Левченко И.В.* Особенности фундаментализации образования на современном этапе его развития // Вестник Российского университета дружбы народов. Серия «Информатизация образования». 2011. № 1. С. 5–11.

4. *Кострюкова М., Сафонова Л.* Особенности изучения раздела «Алгоритмизация и программирование» в курсе информатики средней школы // Информационно-телекоммуникационные системы и технологии: материалы Всероссийской научно-практической конференции. Кемерово: КузГТУ им. Т.Ф. Горбачева, 2015. С. 112–115.

5. *Федорова Н.* Структура, содержание и методические подходы к преподаванию языка программирования Python в школе // Современные информационные технологии и ИТ-образование. 2011. № 7. С. 892–897.

6. *Фролова М.* Некоторые аспекты изучения основ объектно-ориентированного программирования в основной школе // Научная дискуссия современной молодежи: актуальные вопросы, достижения и инновации: сб. ст. международной научно-практической конференции. Пенза: Наука и Просвещение (ИП Гуляев Г.Ю.), 2016. С. 99–101.

7. *Чемеков В., Крылов Д.* STEM — новый подход к инженерному образованию // Вестник Марийского государственного университета. 2015. № 20. С. 59–64.

8. *Шалина О.* Обучение основам визуального программирования в школьном курсе информатики // Достижения современной науки: сборник материалов XIII Международной научно-практической конференции. Астрахань: Научный центр «Олимп», 2016. С. 814–818.

9. *Bruner Jerome S.* Toward a theory of instruction. Cambridge MA: Harvard University Press. 1966. Pp. 49–53.

10. *Piaget J.* Intellectual evolution from adolescence to adulthood // Human development. 1972. Vol. 15. № 1. Pp. 1–12.

11. *Sengupta P.* Programming in K-12 science classrooms // Communications of the ACM. 2015. Vol. 58. № 11. Pp. 33–35.

12. *Sloman A.* Interactions between philosophy and artificial intelligence: The role of intuition and non-logical reasoning in intelligence // Artificial Intelligence. 1971. Vol. 2. № 3/4. Pp. 209–225.

13. *Smith D.C., Cypher A., Tesler L.* Novice Programming Comes of Age // Communications of the acm. 2000. Vol. 43. № 3. Pp. 75.

Literatura

1. *Bezrukov Ya., Skvorczov O.* O vy'bore yazy'ka dlya obucheniya programmirovaniyu v shkole // Vestnik Novosibirskogo gosudarstvennogo universiteta. Seriya «Pedagogika». 2015. № 1. S. 99–103.

2. *Grigor'ev S.G., Grinshkun V.V., Remorenko I.M.* «Umnaya auditoriya»: ot integracii texnologij k integracii principov // Informatika i obrazovanie. 2013. № 10 (249). S. 3–8.

3. *Grinshkun V.V., Levchenko I.V.* Osobennosti fundamentalizacii obrazovaniya na sovremennom e'tape ego razvitiya // Vestnik Rossijskogo universiteta druzhby' narodov. Seriya «Informatizaciya obrazovaniya». 2011. № 1. S. 5–11.

4. *Kostryukova M., Safonova L.* Osobennosti izucheniya razdela «Algoritmizaciya i programmirovanie» v kurse informatiki srednej shkoly' // Informacionno-telekommunikacionny'e sistemy' i texnologii: materialy' Vserossijskoj nauchno-prakticheskoj konferencii. Kemerovo: KuzGTU im. T.F. Gorbacheva, 2015. S. 112–115.

5. *Fedorova N.* Struktura, soderzhanie i metodicheskie podxody' k prepodavaniyu yazy'ka programmirovaniya Python v shkole // Sovremenny'e informacionny'e texnologii i IT-obrazovanie. 2011. № 7. S. 892–897.

6. *Frolova M.* Nekotory'e aspekty' izucheniya osnov ob"ektno-orientirovannogo programmirovaniya v osnovnoj shkole // Nauchnaya diskussiya sovremennoj molodezhi: aktual'ny'e voprosy', dostizheniya i innovacii: sb. st. mezhdunarodnoj nauchno-prakticheskoj konferencii. Penza: Nauka i Prosveshhenie (IP Gulyaev G.Yu.), 2016. S. 99–101.

7. *Chemekov V., Kry'lov D.* STEM — novy'j podxod k inzhenernomu obrazovaniyu // Vestnik Marijskogo gosudarstvennogo universiteta. 2015. № 20. S. 59–64.

8. *Shalina O.* Obuchenie osnovam vizual'nogo programmirovaniya v shkol'nom kurse informatiki // Dostizheniya sovremennoj nauki: sbornik materialov XIII Mezhdunarodnoj nauchno-prakticheskoj konferencii. Astraxan': Nauchny'j centr «Olimp», 2016. S. 814–818.

9. *Bruner Jerome S.* Toward a theory of instruction. Cambridge MA: Harvard University Press. 1966. Pp. 49–53.

10. *Piaget J.* Intellectual evolution from adolescence to adulthood // Human development. 1972. Vol. 15. № 1. Pp. 1–12.

11. *Sengupta P.* Programming in K-12 science classrooms // Communications of the ACM. 2015. Vol. 58. № 11. Pp. 33–35.

12. *Sloman A.* Interactions between philosophy and artificial intelligence: The role of intuition and non-logical reasoning in intelligence // Artificial Intelligence. 1971. Vol. 2. № 3/4. Pp. 209–225.

13. *Smith D.C., Cypher A., Tesler L.* Novice Programming Comes of Age // Communications of the acm. 2000. Vol. 43. № 3. Pp. 75.

E.M. Kagan

The Application of Visual Languages of Programming to Improve Efficiency of Teaching the Section "Algorithmization And Programming" of the School Computer Science Course

The article considers the applicability of visual languages and programming environments as a means of teaching students the programming. A description of the problem of formation of the programming skill in text languages using classical learning tools is given. The possible place of application of visual languages and programming environments within the framework of the course is highlighted. The achievable results of using new tools are expected. The conclusion is made about the possible way of development of programming teaching tools.

Keywords: teaching programming; teaching tools; visual programming; text and block description of the algorithm.