ОПЫТ ПРЕПОДАВАНИЯ ПРОГРАММИРОВАНИЯ СТУДЕНТАМ СПЕЦИАЛЬНОСТИ ИФОРМАТИКА

EXPERIENCE FROM TEACHING PROGRAMMING TO IFORMATICS STUDENTS

CZU: 004.43:004.9:378.147 **https://doi.org/10.59295/spd2024n.105**

Георге ЛАТУЛ,
ORCID: 0009-0006-3391-5161
e-mail: gheorghe.latul@usm.md
Государственный Университет Молдовы

Summary. *This article presents ideas for basic education in software development, taking into account the diversity of programming languages used. The aspects that are important for the training of future programmers are discussed, and the experience of conducting classes on programming languages for students of the Faculty of Mathematics and Informatics at the Moldova State University is summarized. It is proposed to shift the focus in programming education from teaching syntax and language constructs to studying general programming concepts. The use of high-performance IDEs and artificial intelligence tools allows for intensification of the coding learning process.*

Keywords: *artificial intelligence, learning to program, no-code platforms, programming languages.*

Ключевые слова: *искусственный интеллект, обучение программированию, платформы no-code, языки программирования.*

ВВЕДЕНИЕ

Обучение программированию в области разработки программного обеспечения преследует несколько ключевых целей, каждая из которых важна для формирования компетентного и успешного разработчика. Во-первых, обучение программированию развивает аналитическое мышление и навыки решения проблем. Программисты сталкиваются с задачами, требующими точного анализа и последовательного подхода к их решению. Этот процесс помогает развить логическое мышление, умение разбивать сложные проблемы на более мелкие части и находить эффективные пути их решения.

Во-вторых, целью обучения программированию является освоение различных языков программирования и технологий, которые используются в индустрии. Разработчики должны быть знакомы с различными инструментами, фреймворками и библиотеками, чтобы создавать качественное и эффективное программное обеспечение. Эти знания позволяют разработчикам адаптироваться к быстро меняющимся требованиям рынка и использовать современные подходы и методологии разработки.

Кроме того, обучение программированию направлено на развитие навыков работы в команде и управления проектами. Современная разработка программного обеспечения зачастую требует совместной работы большого числа специалистов. Обучение программированию включает в себя не только технические аспекты, но и методологии управления проектами, такие как Agile [1,2] и Scrum [3]], а также навыки коммуникации и координации действий внутри команды.

Наконец, обучение программированию способствует развитию творческих способностей и инновационного мышления. Разработка программного обеспечения часто требует креативного подхода и нестандартных решений. Обучение помогает студентам развивать способность генерировать новые идеи, создавать уникальные продукты и находить оригинальные способы их реализации. Эти навыки особенно важны в условиях высокой конкуренции и стремительного развития технологий.

МАТЕРИАЛЫ И МЕТОДЫ

Данная статья написана на основе опыта автора в преподавании таких языков программирования как C, C++, Java, C# на факультете Математики и информатики Молдавского госуниверситета. Исследования проводились в рамках лекционных, семинарских и лабораторных занятий. Самыми важными и результативными для данного исследования были семинарские и лабораторные занятия, так как они предполагают более интенсивное взаимодействие преподавателя со студентами как в группе, так и индивидуально в реальном времени и наблюдения результатов. На семинарских занятиях главный упор был сделан на разработку алгоритмов, обучению навыкам разбиения задачи на подзадачи, что уменьшает сложность разработки алгоритмов решения задач, а в конечном итоге упрощает написание и отладку кода. В качестве средств разработки программ использовались такие инструменты как Microsoft Visual Studio, Intellij IDEA от фирмы JetBrains, а также ChatGPT от OpenAI.

РЕЗУЛЬТАТЫ И ДИСКУССИИ

Учитывая тот факт в разработке программ используются многочисленные языки программирования, в том числе специфичные для конкретной компании либо языки, которые не изучались в университете, следует отметить, что разработка алгоритмов решения задач играет ключевую роль при обучении программированию, поскольку это фундаментальный аспект создания эффективного и функционального программного обеспечения. Алгоритмы представляют собой четко определенные шаги или последовательности инструкций, которые необходимо выполнить для достижения определенного результата. Понимание и умение разрабатывать алгоритмы позволяет программистам создавать программы, способные решать широкий спектр задач, от простых до очень сложных.

Во-первых, разработка алгоритмов способствует развитию логического и структурированного мышления. Процесс создания алгоритма требует от программиста умения анализировать проблему, разбивать её на более мелкие подзадачи и находить наиболее рациональные пути их решения. Это не только помогает в написании кода, но и развивает общие аналитические способности, которые полезны в различных сферах деятельности, не связанных напрямую с программированием.

Во-вторых, алгоритмы являются основой для оптимизации программного обеспечения. Хорошо спроектированные алгоритмы позволяют существенно повысить производительность программ, уменьшить время выполнения задач и снизить потребление ресурсов. В условиях современных требований к быстродействию и эффективности приложений, знание и понимание алгоритмов помогает программистам создавать высококачественные продукты, которые могут работать в реальном времени и обрабатывать большие объемы данных.

Кроме того, изучение алгоритмов помогает освоить и другие важные концепции информатики, такие как структуры данных и сложность вычислений. Эти концепции тесно связаны с алгоритмами и их понимание позволяет более глубоко проникнуть в суть программирования. Например, знание различных структур данных, таких как массивы, списки, деревья и графы, помогает выбирать наиболее подходящие инструменты для решения конкретных задач, что, в свою очередь, делает программы более эффективными и легкими для понимания и поддержки.

Наконец, умение разрабатывать алгоритмы способствует повышению креативности и изобретательности у программистов. Часто встречаются задачи, для которых не существует очевидного решения, и требуется найти новый подход или метод. В таких случаях алгоритмическое мышление помогает разработчикам экспериментировать с различными подходами, изобретать новые методы и оптимизировать существующие решения. Таким образом, обучение алгоритмам и их разработка является неотъемлемой частью образования программистов и необходимым условием для их профессионального роста и успеха.

Использование средств IntelliSense в средах разработки программного обеспечения позволяет сместить акцент при обучении программированию с синтаксиса языка и конкретных конструкций на изучение общих концепций и технологий программирования.

IntelliSense – это средства, встроенные в среды разработки программ, которые предоставляют разработчикам контекстную помощь при написании кода. Основные функции IntelliSense включают автодополнение кода, подсказки синтаксиса, обнаружение ошибок, отображение документации и предложений по коду в реальном времени.

IntelliSense является важным инструментом для повышения производительности и качества кода. Он сокращает количество ошибок, ускоряет процесс написания кода и облегчает понимание существующего кода, особенно в больших проектах с множеством файлов и зависимостей.

Использование средств искусственного интеллекта [4, 5] показывает существенные результаты для самообучения и исправления ошибок в программе, а также в простых или стандартных алгоритмах. Например, при решении комбинаторных задач на лабораторных занятиях. по запросам студентов с предоставлением неправильно работающего кода, ChatGPT обнаруживал лишние циклы, противоречивое использование переменных и т.д.

Кроме того средства искусственного интеллекта могут писать код по сформулированной задаче. По запросу можно получить пояснение к каждой строчке кода, студенту остается только тщательно изучить полученное решение и попытаться его оптимизировать, если это возможно, самостоятельно или при помощи всё тех же средств искусственного интеллекта. Основная проблема здесь наличие базовых знаний и умение писать запросы.

Неплохие результаты показывает также использование платформ или встроенных в системы разработки программ средств без необходимости написания кода. Это так называемые No-code платформы – NCDP (No-code Development Platform), которые позволяют создавать программные приложения с помощью графических пользовательских интерфейсов вместо традиционного программирования. Платформы разработки No-code тесно связаны с платформами разработки low-code, поскольку обе они призваны ускорить процесс разработки приложений. Однако, в отличие от low-code, платформы разработки no-code вообще не требуют написания кода, предоставляя, как правило, готовые шаблоны, с помощью которых пользователи могут создавать приложения. Такие платформы позволяют учиться программированию по готовым решениям. [5, 6]. Однако использование таких средств и понимание предоставляемых решений требуют от студента определенной подготовки и усилий для изучения кода.

ВЫВОДЫ

Наличие множества используемых языков программирования при разработки программного обеспечения, появление новых языков, требует, при обучении программированию, смещения акцентов с изучения синтаксиса и конструкций языка программирования на освоение общих концепций программирования.

Основным навыком для успешной разработки программ является умение составления алгоритмов решения задач, разбиение задач на подзадачи, что позволяет уменьшить сложность задачи и позволяет распределить написание кода между разработчиками программного обеспечения.

При обучении программированию следует широко использовать среды разработки программ с интегрированными средствами InelliSense. Средства искусственного интеллекта предоставляют огромные возможности не только для написания программного кода, но и для обучения программированию.

ССЫЛКИ

1. Agile Modeling. Available: https://www.informicus.ru/items-agile.html [Accessed: 5.06.2024]
2. МАРТИН РОБЕРТ К., НЬЮКИРК ДЖЕЙМС В., КОСС РОБЕРТ С. *Быстрая разработка программ: принципы, примеры, практика.* Вильямс, 2004. 75 с. ISBN 0-13-597444-5.
3. Джефф Сазерленд. SCRUM. *Революционный метод управления проектами*. Манн, Иванов и Фербер, 2016. 288 с. ISBN 978-5-00057-722-6.
4. ABBIE MISHA. *Why Hidden Artificial Intelligence Features Make Such an Impact in Education*, 2023. Available: https://www.edsurge.com/news/2023-03-06-why-hidden-artificial-intelligence-features-make-such-an-impact-in-education [Accessed: 5.06.2024]
5. ChatGPT. Available: https://chatgpt.com/ [Accessed: 5.06.2024]
6. CABALLAR, RINA DIANE. Programming Without Code: The Rise of No-Code Software Development. IEEE Spectrum magazine. 11 March 2020. Available: https://spectrum.ieee.org/programming-without-code-no-code-software-development [Accessed: 5.06.2024]
7. HARRIS, RICHARD . Low code and no code app development benefits. AppDevelopers Magazine. 25 October, 2017. Available: https://appdevelopermagazine.com/Low-code-and-no-code-app-development-benefits/ [Accessed: 5.06.2024]