УДК 378

Стратегии и методы включения больших языковых моделей в преподавание основ программирования

Курбанкулова Махабат Салабатовна
Иссык-Кульский государственный университет им.К. Тыныстанова
Преподаватель

Абдылдаева Жаркынай Кубанычбековна
Иссык-Кульский государственный университет им.К. Тыныстанова
Преподаватель

Салыкова Назгуль Саматовна
Иссык-Кульский государственный университет им.К. Тыныстанова
Преподаватель

Аннотация
В работе исследованы возможности включения инструментов, использующих большие языковые модели(БЯМ) искусственного интеллекта (ИИ), в курсы бакалавриата по информационным технологиям (ИТ), которые преподают основы программного обеспечения. Инструменты БЯМ стали широко доступны и изменили традиционные методы обучения концепциям программного обеспечения. Цели обучения ставятся под угрозу, когда учащиеся отправляют код заданий сгенерированный ИИ, без понимания или проверки кода. Поскольку инструменты БЯМ, включая OpenAI Codex, Copilot от GitHub и ChatGPT используются в промышленности для разработки программного обеспечения, студенты должны быть знакомы с их использованием без ущерба для обучения. Включение инструментов БЯМ в учебную программу готовит студентов к реальной разработке программного обеспечения. Однако студентам по-прежнему необходимо понимать основы программного обеспечения, в том числе то, как писать и отлаживать код. Существует множество проблем, связанных с включением инструментов ИИ в учебную программу по ИТ, которые необходимо решить и смягчить. В этой работе представлены стратегии и методы, позволяющие интегрировать использование студентами инструментов БЯМ, помочь студентам взаимодействовать с этими инструментами и помочь подготовить студентов к карьере, в которой все чаще используются инструменты искусственного интеллекта для проектирования, разработки и обслуживания программного обеспечения.

Ключевые слова: Искусственный интеллект(ИИ), ChatGPT, Генерация кода, Copilot, GitHub, GPT-4, Информационные технологии(ИТ), Большие языковые модели(БЯМ), OpenAI, Программирование, Разработка программного обеспечения, Основы программного обеспечения

Strategies and Techniques for incorporating Large Language Models into programming fundamentals teaching

Salykova Nazgul Samatovna
K.Tynystanov Issyk-Kul State University
Lecturer of Computer Science

Kurbankulova Mahabat Salabatovna
K.Tynystanov Issyk-Kul State University
Lecturer of Computer Science

Abdyldaeva Zharkynay Kubanychbekovna
K.Tynystanov Issyk-Kul State University
Lecturer of Computer Science

Abstract
This paper argues for the inclusion of tools that utilize Artificial Intelligence (AI) Large Language Models (LLM) in information technology (IT) undergraduate courses that teach the fundamentals of software. LLM tools have become widely available and disrupt traditional methods for teaching software concepts. Learning objectives are compromised when students submit AI-generated code for a classroom assignment without comprehending or validating the code. Since БЯМ tools including OpenAI Codex, Copilot by GitHub, and ChatGPT are being used in industry for software development, students need to be familiar with their use without compromising student learning. Incorporating LLM tools into the curriculum prepares students for real-world software development. However, students still need to understand software fundamentals including how to write and debug code. There are many challenges associated with the inclusion of AI tools into the IT curriculum that need to be addressed and mitigated. This paper presents strategies and techniques to integrate student use of LLM tools, assist students' interaction with the tools, and help prepare students for careers that increasingly use AI tools to design, develop, and maintain software.
Keywords AI, Artificial intelligence, ChatGPT, Code generation, Copilot, GitHub, GPT-4, Information technology, IT, Large language model, LLM, OpenAI, Programming, Software development, Software fundamentals

1 Введение

На протяжении многих лет разработчики программного обеспечения использовали различные инструменты для генерации кода, управление версиями, анализ, отладки и тестирования. В последнее время широкое появление и интеграция БЯМ в эти инструменты еще больше повышают автоматизацию этих задач для разработчиков.

БЯМ — это подмножество ИИ, которое обучается без учителя на исключительно больших наборах данных [2]. БЯМ используют это обучение улучшению обработки входных данных на естественном языке включая

подсказки пользователю для получения ответов, включая программное обеспечение, артефакты, такие как тестовые примеры, документация, код и т. д.

Задания такие как генерация кода, отладка, документация и программное обеспечение, тестирование может быть облегчено с помощью инструмента БЯМ [1]. Недавнее исследование обнаружило что код, сгенерированный моделью, сопоставим по сложности и читаемость того, что написано людьми-программистами [3]. В то время как БЯМ инструменты не устраняют необходимость в опытных программистах, они может ускорить разработку и сделать процесс разработки кода проще [4]. Поскольку инструменты БЯМ используются в промышленности, это одновременно уместны и необходимы для освещения их использования в учебной программе по ИТ.

Использование инструментов БЯМ привлекательно для студентов, потому что студенты считают разработку программного обеспечения сложной задачей и часто тратят больше тратят больше времени на отладку своих приложений, чем на их разработку. БЯМ сложны и предоставляют различные способы поддерживать концепции, представленные в курсе по основам программного обеспечения включая объяснения концепций, примеров кода, обработку ошибок и отладки, алгоритмическое мышление, разработку псевдокода и блок-схем, рекомендации по инструментам, лучшие практики, а также ресурсы и ссылки [5].

Современные интегрированные среды разработки (IDE), такие как Visual Studio Code и PyCharm включают расширения БЯМ, которые предоставляют предложения по кодированию с помощью искусственного интеллекта, контекстную документацию, и обратную связь в режиме реального времени об ошибках программирования. Инструмент ChatGPT использует модель Codex, которая позволяет быстро генерировать, отлаживать и описать соответствующий код для традиционного программирования по заданиям на первом курсе. По мере развития технологий БЯМ эти инструменты должны уметь правильно справиться с еще более трудными и сложными проблемами.

Разумно предположить, что учащиеся будут использовать такие инструменты, как ChatGPT, для генерации решений для своих заданий. К сожалению, когда студенты полагаются на инструмент БЯМ и отправляют сгенерированные решения, не выполняя лично никакой проверки и проверки, это нарушает цели обучения, поставленные в задании.

По мере того, как студенты используют инструменты БЯМ, необходимо изменить способы преподавания и оценки основ программного обеспечения для достижения желаемых целей обучения. Реакция на эту революционную технологию требует обсуждения стратегий и методов, которые могут быть использованы для содействия обучению учащихся и помощи в смягчении чрезмерной зависимости от искусственного интеллекта, которая подрывает вовлеченность учащихся и их знания основных принципов программного обеспечения.

2 Предпосылки

Ассоциация вычислительной техники (ACM) определяет руководящие принципы учебных программ для программ бакалавриата в области ИТ. В руководствах определены основные компетенции для программ бакалавриата в области ИТ. В Таблице 1 «Основы программного обеспечения — компетенции для программ бакалавриата в области ИТ» перечислены основные ИТ-компетенции для основ программного обеспечения и потенциальных приложений для помощи с помощью инструментов БЯМ. На эти компетенции влияет появление языковых моделей искусственного интеллекта, которые в реальном времени генерируют документацию, объяснения, сводки и код в ответ на подсказки и запросы пользователей.

Таблица 1. Основы программного обеспечения — компетенции для программ бакалавриата в области ИТ

Основы программного обеспечения — компетенции [12]	**Потенциальное применение инструментов БЯМ [7]**
Использование нескольких уровней абстракции и выбор подходящих структур данных, для создания новой программы, социально значимую и требующую командной работы. (Разработка программы)	Разъяснение требований к данным, операциям и производительности. Объяснение структур данных. Сравнение структур данных. Предложения по объектно-ориентированному проектированию. Ответы, включая примеры и варианты использования.
Оценка с точки зрения стиля программы, предполагаемого поведения при определенных входных данных, правильности компонентов программы и описания функциональных возможностей программы. (практика разработки приложений)	Оценка стиля программы
Разработка алгоритма для решения вычислительной задачи и описание, как программы реализуют алгоритмы с точки зрения обработки инструкций, выполнения программы и запуска процессов. (разработка алгоритма)	Генерация алгоритма. Описание обработки инструкций, выполнения программы и запуска процессов.
Сотрудничество в создании интересного и актуального приложения (мобильного или веб-приложения) на основе дизайна пользовательского опыта, функциональности и анализа безопасности, а также создание приложения, используя стандартные библиотеки, инструменты модульного тестирования и совместный контроль версий. (практика разработки приложений)	Генерация кода для решения конкретной проблемы на поддерживаемых языках. Включает наиболее популярные языки, такие как Python, Java, C++, C#, Swift, R, PHP, HTML/CSS, Perl.

3 Проблемы ИИ в ИТ-образовании

Организация Объединенных Наций по вопросам образования, науки и культуры (ЮНЕСКО) рекомендует использовать ориентированный на человека подход к включению ИИ в образование [8]. Подход, ориентированный на человека, подчеркивает этические и социальные аспекты ИИ и способствует его использованию для определения приоритетности ключевых принципов, включая инклюзивность и равенство, уважение прав и ценностей человека, прозрачность и объяснимость, расширение прав и возможностей учащихся, благополучие, а также сотрудничество и участие многих заинтересованных сторон. В следующих разделах проблемы включения ИИ в учебные курсы по программному обеспечению сопоставляются с ключевыми принципами человекоориентированного подхода ЮНЕСКО.

3.1 Инклюзивность и равенство

За последний год доступ к инструментам БЯМ неуклонно улучшался. Ценообразование влияет на инклюзивность и равенство в образовании. OpenAI, разработчик ChatGPT, предлагает различные модели и варианты ценообразования, включая версию с бесплатным доступом, планы подписки и варианты оплаты по мере использования для своих продуктов, включая ChatGPT. По данным OpenAI, их инструмент GPT-4 превосходит ChatGPT по своим возможностям, но на момент написания этой статьи GPT-4 не является бесплатным для пользователей. GitHub Copilot доступен программистам за ежемесячную плату, а проверенным студентам — бесплатно.

3.2 Уважение прав человека и ценностей

Данные и конфиденциальность учащегося связаны с уважением прав и ценностей человека, особенно в контексте защиты данных и прав на неприкосновенность частной жизни. Из-за быстрого появления языковых моделей искусственного интеллекта они не могут полностью защитить данные и конфиденциальность учащихся при взаимодействии с инструментом БЯМ. Студенты могут забыть защитить свою личную информацию при использовании этих инструментов. Важно, чтобы пользователи систем на базе искусственного интеллекта не разглашали личную информацию о себе или других. Данные, содержащие личную информацию (PII) и личную медицинскую информацию (PHI), не должны передаваться инструментам искусственного интеллекта. Системы искусственного интеллекта могут закреплять предвзятость или дискриминацию в своих ответах и рекомендациях.

3.3 Прозрачность и объяснимость

В ChatGPT отсутствует контекст, в котором учащиеся изучают основы программного обеспечения. Объяснения концепции программного обеспечения, генерируемые ИИ, могут даваться на более продвинутом уровне и мешать обучению студентов. Языковые модели ИИ обучаются в

репозиториях кода, которые могут не подходить для обучения студентов. Например, в ответ на ознакомительную задачу кодирования сгенерированный ИИ код может использовать расширенные концепции, такие как указатели или рекурсия. Кроме того, стили и подходы к кодированию могут не соответствовать стилю и подходу курса. Учащиеся, незнакомые с этим инструментом, могут испытывать трудности с пониманием различных форм предложений кода ИИ, что может помешать учащемуся понять, как работает код и почему он был выбран. Одно исследование показало, что пользователям сложнее отлаживать код, созданный ИИ, чем код, который они написали [9]. Общедоступный код часто требует от любого пользователя ссылки на первоначальный источник, но источники кода непрозрачны в коде, сгенерированном языковой моделью искусственного интеллекта[5]. Хотя модель БЯМ может предоставить документацию кода, объясняющую ее функциональность, в обучающих данных, используемых при ее объяснении, отсутствует прозрачность.

3.4 Подотчетность и ответственность

Студенты могут легко генерировать рабочий код на языке программирования по своему выбору с помощью ChatGPT, но этот код может не соответствовать требованиям задания. Студенты могут не чувствовать ответственности за проверку того, что код работает правильно, логически корректен и соответствует критериям задания. Когда код автоматически генерируется с помощью инструмента БЯМ, используемого студентами, учащиеся могут не чувствовать личной ответственности за качество сгенерированного кода. ChatGPT генерирует ответы, синтезируя данные обучения, и часто выдает ответы, аналогичные существующим источникам, что может привести к плагиату. Подобные ответы, представленные учащимися, когда они используют языковые модели искусственного интеллекта, могут подорвать подотчетность и ответственность учащихся. Существуют опасения по поводу безопасности онлайн-оценки и мошенничества, которое подрывает академическую честность.

3.5 Расширение прав и возможностей учащихся и их благополучие

Целью этого принципа является развитие навыков и знаний учащихся до такой степени, что они смогут генерировать правильный код. Инструмент БЯМ следует использовать в качестве средства программирования, которое помогает студенту разрабатывать программное приложение. Студент должен понимать принципы разработки программного обеспечения, уметь читать и понимать логику кода, сгенерированного инструментом, и иметь навыки внесения необходимых корректировок.

Одной из проблем расширения прав и возможностей учащихся и их благополучия является чрезмерная зависимость от моделей искусственного интеллекта при выполнении лабораторных работ и заданий. Недостаток контекстуального понимания в моделях ИИ может привести к получению ответов, не соответствующих требованиям задания.

Решения, созданные с помощью инструментов БЯМ, могут содержать логические или синтаксические ошибки. Одно исследование показало, что ChatGPT генерирует правильные ответы на 55,6% вопросов и не смог оценить правильность своих ответов [6]. Сгенерированный код может содержать уязвимости и/или предвзятость, вызванные данными обучения модели ИИ [5].

Чрезмерное использование учащимися таких инструментов, как ChatGPT, может привести к снижение творческих способностей учащихся, критического мышления, рассуждения и навыков решения проблем, и у студентов может отсутствовать мотивация к анализу и решению проблемы самостоятельно.

3.6 Сотрудничество и участие заинтересованных сторон

Использование студентами инструментов БЯМ улучшает их понимание взаимодействия человека с машиной, но не заменяет взаимодействие человека с человеком.

Разработка программного обеспечения, особенно крупномасштабного программного обеспечения, требует командной работы – взаимодействия между разными людьми, включая пользователей, менеджеров проектов, системных аналитиков, разработчиков и тестировщиков. Курс по основам программного обеспечения должен включать результаты обучения, связанные с совместным характером разработки программного обеспечения и информировать студентов о том, как участие многих заинтересованных сторон влияет на процесс разработки.

4 Преимущества

Инструменты БЯМ ИИ создают проблемы для существующей учебной программы в области ИТ. Несмотря на эти проблемы, модели широко используются, и запрет на их использование не является практическим подходом, когда курсы в высшем образовании доступны асинхронно и онлайн. Эти инструменты также используются в промышленности, поэтому важно, чтобы студенты были с ними знакомы. Более практичный подход — использовать эти технологии, предоставляя тем самым возможности для преподавания и изучения основ программного обеспечения с помощью ИИ.

4.1 Актуальность в реальной жизни

Языковые модели искусственного интеллекта, включая ChatGPT — это технологии, и реальную разработку программного обеспечения. Включение этих инструментов искусственного интеллекта в курс по программному обеспечению знакомит студентов с новыми программными технологиями и лучше готовит их к динамичной рабочей среде. Платформы No-code/low-code, используемые при разработке программного обеспечения, предлагают среду «что видишь, то и получаешь» (WYSIWYG) с функцией drag-and-drop. Эти платформы все чаще включают модели искусственного интеллекта, такие как AI Builder в Microsoft Power Apps.

4.2 Обращение особого внимания к теории программирования и акцентирование внимания на эффективном поиске решений

Ключевой компонент курсовой работы по программному обеспечению ИТ включает в себя программирование, освоить которое большинству студентов непросто. Модели искусственного интеллекта, такие как Copilot от GitHub, помогают начинающим программистам предлагать код, примеры и объяснения, которые снижают когнитивную нагрузку на студентов, обучающихся программированию. В одном исследовании студенты, использующие модели ИИ для своих заданий по программированию, добились успеха, предоставив точные описания и проверили код перед отправкой [10]. Модели искусственного интеллекта могут сократить время выполнения задач по программированию, уменьшая стресс и разочарование учащихся [10]. Сокращение времени выполнения задач при написании и отладке приложений позволяет преподавателям больше сосредоточиться на теории программного обеспечения и решении проблем, а не на синтаксисе конкретного языка.

4.3 Самостоятельное обучение

Самостоятельное обучение расширяет возможности учащихся и обеспечивает им самостоятельность при обучении и выполнении заданий. Студенты могут подходить к взаимодействию с моделью ИИ множеством самостоятельных подходов. Студент может взаимодействовать с языковыми моделями ИИ, предлагая инструменту ответы на вопросы, генерацию кода, отладку кода и пояснения кода. Учащиеся разрабатывают вопросы для языковой модели. Студенты могут обнаружить, что их подсказки необходимо дополнить или разложить на несколько подзапросов. На ранних этапах процесса студент может использовать такой инструмент, как Copilot или ChatGPT, чтобы изучить варианты разработки кода. Позже в процессе студент может вставить код, содержащий синтаксические ошибки, и предложить модели ИИ выявить ошибки и исправить их. Взаимодействие учащегося с моделью ИИ во время отладки программного обеспечения имеет огромный потенциал для уменьшения разочарования и сокращения времени выполнения задач.

4.4 Знакомство с искусственным интеллектом.

Знакомство с инструментами искусственного интеллекта позволяет учащимся и преподавателям приобретать новые компетенции и способы взаимодействия человека с компьютером. Простой запрос к языковой модели ИИ не гарантирует правильного ответа. Пользователи языковых моделей должны помнить о контекстных ограничениях при использовании этих моделей. Для получения качественных ответов может потребоваться декомпозиция запросов и предоставление информации о контексте в модель ИИ. Любой ответ необходимо проверить и проверить на правильность из-за ограничений данных обучения ИИ [11].

5 Стратегии и методы

Преподаватели могут использовать стратегии и методы, включающие групповые проекты, практические занятия и демонстрации, чтобы повысить эффективность моделей БЯМ и избежать ошибок. Языковые инструменты искусственного интеллекта могут создавать быстрые решения, которые способствуют обучению учащихся, если они используются надлежащим образом. Учащиеся могут достичь лучших результатов, если будут эффективно запрашивать, проверять и проверять результаты, а также размышлять над результатами. Ниже предложены стратегии и методы, которые могут помочь эффективно интегрировать инструменты БЯМ в курс по основам программного обеспечения.

5.1 Разработка политики внедрения ИИ.

Представление политики использования моделей ИИ важно как в программе курса, так и в заданиях. Это устанавливает руководящие принципы использования технологий БЯМ в курсе. В этом документе предлагается добавить такие фразы, как следующие:

• Студент несет ответственность за изучение и оценку информации, генерируемой инструментами ИИ, на предмет актуальности и точности.

• Студент несет ответственность за проверку сгенерированного кода, чтобы убедиться, что он не содержит ошибок, включая синтаксические и семантические ошибки.

• Студент несет ответственность за проверку того, что программное обеспечение соответствует требованиям критериев для таких элементов, как типы файлов, ввод/вывод, типы данных. , стиль программы и документация.

• Если в задании используется инструмент искусственного интеллекта, учащийся должен явно подтвердить использование этого инструмента и описать, как он использовался для выполнения задания.

• Если в задании используется инструмент искусственного интеллекта, контрольный журнал запросов/подсказок и ответов должен быть включен как часть задания.

• Ответственность за защиту своей личной информации и личной информации других лиц при использовании инструмента искусственного интеллекта лежит на учащемся.

• Учащиеся должны иметь возможность понимать и размышлять над любой информацией, генерируемой ИИ, в рамках своего задания.

5.2 Обеспечение равного доступа к инструментам

Обеспечение равного доступа к инструментам БЯМ требует предоставления всем студентам адекватных ресурсов. Например, инструменты разработки программного обеспечения, которые в настоящее время установлены на лабораторных машинах, можно обновить с помощью расширения инструмента искусственного интеллекта, например, добавив программное обеспечение GitHub Copilot в Visual Studio Code. В учебной

программе должны быть описаны технологические требования и ресурсы, доступные для использования этих инструментов модели ИИ.

5.3 Объяснение и демонстрация преимуществ и недостатков моделей ИИ при разработке программного обеспечения

Существуют стратегии, которые можно превратить в курс, чтобы лучше рассказать учащимся о преимуществах и недостатках использования ИИ для выполнения своих заданий. Учащиеся должны понимать, что ответ ИИ может быть неверным из-за ограниченного понимания контекста или неоптимальных подсказок. Одно исследование выявило неправильные решения, сгенерированные Copilot, из-за отсутствия контекстуального понимания инструмента [7]. Это отсутствие контекстуального понимания возникло из-за деталей задания, которые включали анализ текстового файла, адаптированного к заданию. Инструменты искусственного интеллекта, такие как Copilot, могут вызывать разочарование, поскольку они часто не понимают инструкции обучаемого по исправлению или улучшению сгенерированного кода, если только пользовательские подсказки не являются чрезвычайно конкретными. Одной из стратегий может быть включение серии лабораторных занятий, на которых студенты используют инструменты БЯМ для развития идей и изучения конкретной концепции, такой как инкапсуляция. Эти результаты исследования можно обсудить в классе и собрать предложения о том, как лучше взаимодействовать с инструментом БЯМ. Разработка заданий, исследующих ограничения инструментов БЯМ, может рассказать учащимся о подводных камнях использования ИИ. Задания, такие как создание статического веб-сайта, для которого требуются файлы нескольких типов, такие как файлы HTML, файлы CSS и файлы JavaScript, требуют от учащихся успешного связывания файлов. Это задание можно использовать для демонстрации ограничений моделей ИИ, когда задание становится более сложным. Преподаватель может демонстрировать успешные методы взаимодействия с инструментами БЯМ во время занятий и упражнений, чтобы лучше подготовить студентов к их использованию. Демонстрации могут проводиться в классе или прикрепляться в виде демо-видео по мере поступления заданий.

5.4 Требование трехэтапного подхода к заданиям

В одном из цитируемых исследований рекомендуется трехэтапный подход к разработке программных приложений в сочетании с инструментом искусственного интеллекта. Эти шаги включают декомпозицию системы, функциональное программирование и уточнение программы.

• Декомпозиция системы разбивает проблему на последовательный список задач. Эта разбивка выполняется неоднократно, пока алгоритм не будет разложен на подзадачи с использованием структур управления, таких как итерация и принятие решений. Декомпозиция системы более полезна для подсказок ИИ и обеспечивает повышенное качество кода.

• Функциональное программирование определяет атрибуты функции, включая имя, список аргументов и описание. Функциональное программирование - это широко распространенная парадигма программирования, в которой программы можно разбить на функции.

• Уточнение программы включает выполнение и тестирование программного приложения для определения синтаксических и семантических ошибок. Языковые модели ИИ более склонны к семантическим ошибкам, поэтому учащимся следует проверять результаты программы с помощью отладчика.

5.5 Включение проектов с уникальным контекстом и персонализацией

Чрезмерная зависимость учащегося от инструментов может привести к снижению его творческих способностей и навыков решения проблем. Одна из ИТ-компетенций, определенных ACM для основ ИТ-программного обеспечения, включает сотрудничество студентов в создании интересного и актуального мобильного или веб-приложения. Включение проектов с уникальным контекстом и персонализацией учащихся способствует творчеству учащихся и решению проблем. Например, задание, для которого требуется программа Python, рисующая треугольник из трех целочисленных входных данных, легко генерируется с помощью такого инструмента, как ChatGPT. Этот тип заданий не способствует творчеству, поскольку сгенерированный код может быть одинаковым для учащихся. Включение группового проекта, в котором студенты разрабатывают требования к программе и разрабатывают соответствующий алгоритм, структуры данных, модули и пользовательский интерфейс, дает им возможность изучить задачи, необходимые при разработке программного обеспечения, не последней из которых является командная работа. Они принимают решения, уникальные для своего проекта, и способствуют творчеству.

5.6 Требование валидации и верификации кода

В курсе можно использовать несколько методов проверки и проверки кода. Студенты, использующие код, сгенерированный БЯМ, должны поместить сгенерированный код в IDE, например, Visual Studio Code, и использовать встроенный отладчик для переключения точки останова в первой строке основной процедуры и пошагово выполнить код, чтобы убедиться, что весь код используется и является семантически и синтаксически правильным. В курсе веб-систем студенты могут использовать отладчик в VS Code или внешние инструменты, такие как служба проверки разметки W3C, для проверки кода, например HTML и CSS. Поступая таким образом, учащиеся берут на себя ответственность за код и признают, что он может иметь недостатки, такие как логические ошибки, синтаксические ошибки или другие важные проблемы, такие как доступность и другие.

5.7 Требование демонстрации и объяснений учащихся

Размышления учащихся имеют решающее значение для понимания разработанного ими программного обеспечения, включая выбранные структуры данных, стиль программы, структуры управления и то, как программа решает вычислительную задачу. Требование к учащимся продемонстрировать и объяснить разработанное ими программное обеспечение позволяет им поразмышлять над тем, что они узнали. Один из вариантов — предложить учащимся записать видео, показывающее их работающий код. Эти демонстрации и объяснения можно использовать при оценивании, чтобы определить, полностью ли учащийся понимает созданное им программное обеспечение. При демонстрации и объяснении полученного программного обеспечения учащимся требуется полное понимание. Это объяснение может включать предполагаемое поведение, конкретные входные данные, компоненты программы и функциональность.

6 Заключение

В этом документе предлагается интеграция инструментов БЯМ ИИ в фундаментальные курсы по программному обеспечению по той простой причине, что они используются в промышленности и со временем станут только более эффективными. Работодатели будут ожидать, что студенты не только умеют программировать, но и знают, как эффективно использовать различные инструменты, в том числе инструменты БЯМ ИИ, поэтому важно включить эти инструменты в учебную программу по ИТ.

Хотя инструменты искусственного интеллекта могут генерировать код, они пока не могут заменить опытного разработчика программного обеспечения. Они действительно представляют собой полезный инструмент, который может облегчить разработку кода, но пользователь любого инструмента должен знать, как его использовать и его ограничения. Поскольку сгенерированный код может не соответствовать требованиям, пользователь должен иметь возможность использовать фундаментальные концепции и практические знания для проверки и проверки сгенерированного кода. Использование ИИ меняет методы преподавания фундаментальных курсов по ИТ-программному обеспечению. В данной работе предлагается активно рассматривать и использовать стратегии и методы, способствующие обучению студентов в развивающейся среде разработки программного обеспечения, где инструменты БЯМ легко доступны как студентам, так и преподавателям.

Библиографический список

1. Tian H. et al. Is ChatGPT the Ultimate Programming Assistant--How far is it? //arXiv preprint arXiv:2304.11938. 2023. URL: https://arxiv.org/pdf/2304.11938.pdf (дата обращения: 2.12.2023).

2. Birhane A. et al. Science in the age of large language models //Nature Reviews Physics. 2023. Т.5. С. 1-4. URL: https://www.nature.com/articles/s42254-023-00581-4 (дата обращения: 4.12.2023).
3. Al Madi N. How readable is model-generated code? examining readability and visual inspection of github copilot //Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering. 2022. С. 1-5.
4. White J. et al. Chatgpt prompt patterns for improving code quality, refactoring, requirements elicitation, and software design //arXiv preprint arXiv:2303.07839. 2023. URL: https://arxiv.org/pdf/2303.07839.pdf (дата обращения: 6.12.2023).
5. Becker B. A. et al. Programming is hard-or at least it used to be: Educational opportunities and challenges of ai code generation //Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1. 2023. С. 500-506.
6. Jalil S. et al. Chatgpt and software testing education: Promises & perils //2023 IEEE International Conference on Software Testing, Verification and V
a
7. Puryear B., Sprint G. Github copilot in the classroom: learning to code with AI assistance //Journal of Computing Sciences in Colleges. 2022. Т. 38. №. 1. С. 37-47. URL: https://dl.acm.org/doi/abs/10.5555/3575618.3575622 (дата обращения: 11.12.2023).
8. Holmes W. et al. Технологии искусственного интеллекта в образовании: Руководство для лиц, ответственных за формирование политики. UNESCO Publishing, 2022. URL: https://unesdoc.unesco.org/ark:/48223/pf0000382446 (дата обращения: 12.12.2023).
9. Barke S., James M. B., Polikarpova N. Grounded copilot: How programmers interact with code-generating models //Proceedings of the ACM on
P
k
10. Kazemitabaar M. et al. Studying the effect of AI Code Generators on Supporting Novice Learners in Introductory Programming //Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. 2023. С. 1-23.
11. Tlili A. et al. What if the devil is my guardian angel: ChatGPT as a case study of using chatbots in education //Smart Learning Environments. 2023. Т. 10. №. 1. С. 15.
12. Computing Curricula 2020 CC2020 Paradigms for Global Computing Education URL: https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf (дата обращения: 26.12.2023).
Languages. 2023. Т. 7. №. OOPSLA1. С. 85-111. URL:
W
)
.
I
E
E