УДК 37

Python для обучения основам программирования: количественная оценка

Абдылдаева Жаркынай
Иссык-Кульский государственный университет им. К.Тыныстанова
Преподаватель
Приамурский государственный университет им. Шолом-Алейхема
Студент

Научный руководитель
Баженов Руслан Иванович
к.п.н., доцент, зав.кафедрой информационных систем, математики и правовой информатики

Аннотация
В этой статье сравниваются два различных подхода к вводному обучению программированию путем количественного анализа оценок учащихся в реальном классе. Первый подход заключается в подчеркивании принципов объектно-ориентированного программирования и проектирования с использованием Java с самого начала. Второй подход заключается в том, чтобы сначала обучить базовым понятиям программирования (циклы, ветвление и использование библиотек) с помощью Python, а затем перейти к ориентированному программированию с использованием Java. Каждый подход был принят на один учебный год (2019-20 и 2020-21) со студентами первого курса бакалавриата.
Ключевые слова: Обучение программированию, объектно-ориентированный подход, процедурный подход, java, python.

Python for Learning Fundamentals of Programming: Quantifying

Abdyldaeva Zharkynai
Issyk-Kul State University named after K. Tynystanova
Teacher
Sholom-Aleichem Priamursky State University
Student

Scientific supervisor
Bazhenov Ruslan Ivanovich
Ph.D, Associate Professor, Head of the Department of Information Systems, Mathematics and Legal Informatics

Abstract
This article compares two different approaches to introductory programming instruction by quantifying student grades in a real classroom. The first approach is to emphasize the principles of object-oriented programming and design using Java from the very beginning. The second approach is to first teach the basic concepts of programming (loops, branches, and using libraries) using Python, and then move on to oriented programming using Java. Each approach was taken for one academic year (2019-20 and 2020-21) with first-year undergraduate students.
Keywords: Teaching programming, object oriented approach, procedural approach, java, python

1. Введение

1.1 Актуальность

Обучение программированию — сложная задача. Задача еще более сложная для вводных модулей. Одной из общих проблем, общих для факультетов компьютерных наук, является отсутствие базовых навыков программирования, о которых сообщают руководители модулей курсов после первых курсов программирования, а также то, как вооружить студентов лучшими навыками программирования после вводных курсов. Первые курсы программирования обычно с самого начала делают акцент на принципах объектно-ориентированного программирования и проектирования. Альтернативный подход основан на том, чтобы сначала начать с более традиционного процедурного подхода. Данные прошлых исследований, по-видимому, предполагают, что объектно-ориентированное программирование, как правило, более сложное, чем процедурное (A. Robins [13]). B. McCane (2009) продемонстрировал эффективность Python для обучения основам программирования с использованием качественного анализа.

Поэтому в этой статье предлагается ввести Python (Python Software Foundation) для основных (процедурных) аспектов программирования, а затем ввести Java, чтобы сосредоточиться на объектно-ориентированных аспектах. Ожидается, что использование Python уменьшит накладные расходы, связанные с синтаксисом Java, и обеспечит немедленную обратную связь, пока учащиеся практикуются с базовыми инструкциями (из-за интерпретируемого характера Python). Другими словами, в начале модуля начинающий студент может уделить особое внимание процедурным концепциям, потоку управления, потоку данных и т. д.

1.2 Обзор исследований

Различные подходы к обучению программированию были обобщены в R.S. Lemos [8] и G. Engel [5]. Но в педагогическом сообществе продолжаются дебаты о наилучшем подходе к вводному обучению программированию (R. Lister [9]; A. Pears [11]). Мы нашли четыре экспериментальных исследования, в которых объектно-ориентированный подход сравнивается с традиционным процедурным подходом. Первое

исследование, проведенное A.Decker[4], не выявило различий в успеваемости учащихся между этими двумя подходами. Второе исследование, проведенное S. Reges[14], выявило значительный рост удовлетворенности студентов и числа учащихся после замены первой учебной программы по объектно-ориентированному программированию процедурным подходом. Третье исследование, проведенное T.Vilner [16], не выявило значительного повышения успеваемости учащихся при использовании двух подходов. Параметрами, использованными в этом исследовании для сравнения двух подходов к обучению, являются успешность учащихся и оценки по вопросам, связанным с рекурсией, эффективностью алгоритмов и проектированием классов. Четвертое исследование, проведенное A. Ehlert [6], не выявило значительных преимуществ за счет замены первого подхода объектно-ориентированного программирования на более поздний подход объектно-ориентированного программирования.

1.3 Цель исследования

Цель этой работы — оценить успешность использования Python для введения основных понятий (циклов, ветвления и использования библиотек). Такая оценка основывается на анализе оценок учащихся. Для измерения успеха предложенного метода представлен и обсужден количественный анализ результатов оценок. A.Ehlert [6] подчеркивает важность разработки экспериментальных условий для объективного сравнения и важность тщательного контроля переменных. Экспериментальная установка, используемая в этой статье, тщательно контролирует и определяет переменные и измеряет их с использованием четких индикаторов.

2 Материалы и методы

По словам A. Ehlert [6], разные педагогические аспекты усложняют анализ разных подходов к обучению программированию. Брюс (2005) утверждает, что необходимо больше экспериментальных исследований для изучения различных подходов к обучению программированию.

В рамках этой статьи оценочные материалы Python сравниваются с оценочными материалами Java. Оценки Python и Java очень похожи, поэтому сравнение должно дать надежную оценку. Цель сравнения — количественно измерить способность учащегося освоить основные концепции программирования при использовании Python вместо объектно-ориентированной Java. Обе оценки состояли из студентов, которые должны были реализовать программу Были проведены следующие три сравнения: оценки, программы с ошибками и частота ключевых слов. Оценки представленной программы являются мерой успешности способности учащегося реализовывать программы. Ошибки интерпретируются как мера их общего понимания программирования. Частотные значения ключевых слов интерпретируются как мера знакомства с основными понятиями программирования. В показателе частоты используются четыре ключевых слова: «if», «for», «while» и «import» (в сочетании с использованием класса

«random»). Они являются индикаторами использования учащимися условных выражений, циклов и операторов импорта (случайной библиотеки). Вхождения этих ключевых слов в закомментированные разделы кода были исключены. Не делалось различий между правильным и неправильным использованием формулировок при измерении частоты.

Для сбора данных использовались два разных учебных года. В течение 2019-20 учебного года с самого начала следует подход, в котором основное внимание уделяется принципам объектно-ориентированного программирования и проектирования с использованием Java. В течение 2020-21 учебного года применяется подход, заключающийся в том, чтобы сначала обучать основным понятиям программирования (циклы, ветвление и использование библиотек) с использованием Python, а затем переходить к ориентированному программированию с использованием Java. В обоих случаях студенты оценивались примерно через десять недель обучения. Оценки Java относятся к 2019-20учебному году, тогда как оценки Python относятся к 2020-21 учебному году. Кроме того, все учащиеся в когорте имеют (некоторый) одинаковый опыт, поскольку они должны выполнять аналогичные требования, чтобы быть принятыми. Некоторые могут иметь более высокий уровень владения языком, чем другие, но это в равной мере верно для обеих когорт. Исследование основано на предположении, что когорты студентов 2019-20 и 2020–2021 учебных годов имеют одинаковый академический уровень. Это предположение подтверждается тем фактом, что критерии приема в оба года были одинаковыми. Ни один студент не был в обеих когортах. В таблице 1 представлена студенческая когорта.

Таблица 1

Академический год	Язык	Время оценивания	Количество студентов
2019-20	Java	После 10-недельной учебы	157
2020–2021	Python	После 10-недельной учебы	185

4. Схема модуля

Модули 2019-20 и 2020-21 годов в течение первых десяти недель делают упор на овладение базовыми навыками. В частности, лекции и практические занятия/упражнения были сосредоточены на циклах, условных операторах, массивах/списках и использовании библиотек/пакетов (в частности, тех, которые генерируют случайные числа). BlueJ использовался в модуле 2019-20 в качестве инструмента для практики работы с Java, тогда как в 2009–2010 годах использовались стандартные инструменты Python (т. е. как графический пользовательский интерфейс, так и приложения командной строки). Примерно одинаковое количество времени было посвящено каждой теме в течение обоих лет.

Учащиеся оцениваются по их способности реализовывать программы. Критерии оценки включают: обобщаемость, сложность, покрытие циклов, условные операторы и т. д. Кроме того, требованием для прохождения считалась способность создать программу, которая могла бы компилироваться и работать без ошибок. Однако, как для данных Java, так и для Python небольшие опечатки (отсутствие точки с запятой и т. д.) не считались ошибками (даже если отправленная программа работала некорректно). То есть они не были оценены как неудовлетворительные, хотя они были оценены как чуть выше удовлетворительных. Для оценки стиля представленной программы был проверен исходный код.

У студентов есть час, чтобы выполнить задание, и они могут использовать ту же среду, которую они использовали во время своих лабораторных занятий (BlueJ, Python и т. д.). То есть во время теста они могут реализовывать, запускать и отлаживать свои программы. У них есть доступ как к стандартной документации по Java/Python, так и к ресурсам модуля (конспекты лекций, лабораторные примеры и т. д.). Задания максимально похожи друг на друга в обеих оценках.

5. Результаты и обсуждение

Таблицы 2 и 3 суммируют результаты количественного анализа.

Таблица 2

	Всего студентов	**If, %**	**For, %**	**While %**	**Random, %**	**Bugs, %**
Python	185	67	36	81	44	27
Java	157	36	23	1	1	8

Таблица 3

Всего студентов	**A%**	**B%**	**C%**	**D%**	**E%**	**D(40)%**
Python	185	6	3	15	8	22
Java	157	6	3	10	6	22

Столбцы с 3 по 6 в таблице 2 показывают количество программ, в которых, соответственно, хотя бы один раз имел место условный цикл, цикл for, цикл while и импорт случайного пакета. Во втором столбце (Всего учащихся) указано общее количество оценок, а в последнем столбце указаны случаи, когда программа не могла работать без предварительной отладки.

В таблице 3 показано распределение оценок. Использование циклов, условных выражений и т. д. для выполнения данной задачи входило в критерии оценки. Например, оценка A требует правильного использования

условного выражения и правильного использования циклов, а также правильного использования библиотек в реализованной программе. В последнем столбце (D(40)) показано количество экземпляров программ, которые не запускались без отладки, но в которых ошибки, тем не менее, считались незначительными (и, следовательно, оценивались как пройденные).

Результаты, представленные в таблице 2, показывают, что есть свидетельства значительного увеличения использования всех четырех маркеров, когда Python используется в качестве языка программирования. Более того, количество ошибок в Python меньше, чем в Java. Результаты, представленные в Таблице 3, показывают, что распределение оценок показывает более высокий процент программ, способных удовлетворить минимальные требования для прохода, когда использовался Python.

Таким образом, количество ключевых показателей, используемых студентами, статистически отличается, когда был введен Python, по сравнению с подходом, основанным только на Java. Это означает, что первый подход Python оказал положительное влияние на оценки.

Эти результаты подразумевают, что наш подход к знакомству с Python для основных (процедурных) аспектов программирования с последующим переключением на Java, чтобы сосредоточиться на объектно-ориентированных аспектах, улучшил понимание учащимися вводного программирования.

3 Результаты и обсуждение

В этой статье представлены результаты экспериментов в реальном классе, в которых сравниваются два разных подхода к вводному обучению программированию. Первый подход заключается в подчеркивании принципов объектно-ориентированного программирования и проектирования с использованием Java с самого начала. Второй подход заключается в том, чтобы сначала обучить базовым понятиям программирования (циклы, ветвление и использование библиотек) с помощью Python, а затем перейти к ориентированному программированию с использованием Java.

Первый подход к обучению с использованием Java применялся в 2019-20 учебном году, а второй подход к обучению с использованием Python применялся в 2009–2010 учебном году. Оценки студентов к концу первого семестра каждого учебного года использовались для проведения экспериментов, описанных в этой статье. Для сравнения использовались три показателя, а именно оценки, программы с ошибками и частота ключевых слов. Оценки представленной программы являются мерой успешности способности учащегося реализовывать программы. Ошибки интерпретируются как мера их общего понимания программирования. Частотные значения ключевых слов интерпретируются как мера знакомства с основными понятиями программирования. В показателе частоты используются четыре ключевых слова: «if», «for», «while» и «import» (в сочетании с использованием класса «случайный»).

4 Выводы

Экспериментальные результаты, обсуждаемые в этой статье, показывают измеренное положительное (увеличение) всех трех показателей в пользу второго подхода, который заключается в том, чтобы сначала обучать основным концепциям программирования (циклы, ветвление и использование библиотек) с использованием Python, а затем двигаться дальше. к ориентированному программированию с использованием Java. Использование Python может облегчить освоение базовых понятий, таких как циклы, переходы и использование библиотек для начинающих студентов. Возможным объяснением может быть то, что с помощью Python студенты могут сосредоточиться на важнейших базовых вопросах, не отвлекаясь на накладные расходы. Кроме того, сложность программ, используемых во время их практики, может быть более точно адаптирована к их уровню владения языком.

Одним из ограничений этого исследования является то, что оно рассматривает только один пример. Такие тематические исследования требуют значительного количества времени, поскольку они должны соответствовать академическому календарю, и поэтому мы могли рассмотреть только одно тематическое исследование. Мы надеемся, что подобные тематические исследования будут воспроизведены и другими учреждениями. Результаты нашего тематического исследования являются положительными и статистически значимыми в пользу предложенного нами подхода, который заключается в том, чтобы сначала обучать основным концепциям программирования (циклы, ветвление и использование библиотек) с использованием Python, а затем переходить к ориентированному программированию с использованием Java. Мы надеемся, что наше исследование вызовет споры о различных подходах к обучению вводному программированию и, в свою очередь, приведет к принятию лучших методологий обучения.

Кроме того, это исследование фокусируется только на количественном анализе оценок учащихся и не выявляет мнения учащихся об их учебном опыте. Таким образом, интересным направлением для будущей работы будет сбор и анализ качественных данных от студентов об их точке зрения.

Библиографический список

1. Bennedsen J., Caspersen M. E. Failure rates in introductory programming //AcM SIGcSE Bulletin. 2007. Т. 39. №. 2. С. 32-36.
2. BlueJ - Teaching Java - Learning Java URL: http://www.bluej.org/
3. Bruce K. B. Controversy on how to teach CS 1: a discussion on the SIGCSE-members mailing list //ACM SIGCSE Bulletin. 2005. Т. 37. №. 2. С. 111-117.
4. Decker A. A tale of two paradigms //Journal of Computing Sciences in Colleges. 2003. Т. 19. №. 2. С. 238-246.
5. Engel G., Roberts E. Computing Curricula 2001 Computer Science: final report. IEEE Computer Society and Association for Computing Machinery,

2001, С. 2004
6. Ehlert A., Schulte C. Empirical comparison of objects-first and objects-later //Proceedings of the fifth international workshop on Computing education research workshop. 2009. С. 15-26.
7. Java, Oracle Java Technologies, URL: http://www.oracle.com/us/technologies/java/index.html
8. Lemos R. S. Teaching programming languages: A survey of approaches //Proceedings of the tenth SIGCSE technical symposium on Computer science education. 1979. С. 174-181.
9. Lister R. et al. Research perspectives on the objects-early debate //ACM SIGCSE Bulletin. 2006. Т. 38. №. 4. С. 146-165.
10. McCane B. Introductory programming with python //The Python Papers Monograph. 2009. Т. 1. С. 1-18.
11. Pears A. et al. A survey of literature on the teaching of introductory programming //Working group reports on ITiCSE on Innovation and technology in computer science education. 2007. С. 204-223.
12. Python Software Foundation, Python Programming Language URl: http://www.python.org/
13. Robins A., Rountree J., Rountree N. Learning and teaching programming: A review and discussion //Computer science education. 2003. Т. 13. №. 2. С. 137-172.
14. Reges S. Back to basics in CS1 and CS2 //Proceedings of the 37th SIGCSE technical symposium on Computer science education. 2006. С. 293-297.
15. The R Foundation, (2010), The R Project for Statistical Computing URL: http://www.r-project.org/
16. Vilner T., Zur E., Gal-Ezer J. Fundamental concepts of CS1: procedural vs. object oriented paradigm-a case study //ACM SIGCSE Bulletin. 2007. Т. 39. №. 3. С. 171-175.