13. Третьякова И.А., Похлебаев С.М. Вещество, энергия и информация как факторы сопряжения между организмом и средой его обитания // Адаптация биологических систем к естественным и экстремальным факторам среды: мат-лы VI междунар. науч.-практ. конф. Челябинск, 8–9 ноября 2016 г., г. Челябинск, Российская Федерация. Челябинск: Изд-во Юж.-Урал. гос. гуманит.-педаг. ун-та, 2016. С. 443–448.

14. Третьякова И.А., Похлебаев С.М. Теория и практика формирования и развития сопряженных физиологических понятий «фотосинтез» и «дыхание» в курсе биологии: монография. Челябинск: Изд-во Южно-Урал. гос. гуман.-пед. ун-та, 2018. 245 с.

15. Дубинин Н.П. Общая генетика. М.: Наука, 1976. 572 с.

16. Верзилин Н.М., Корсунская В.М. Общая методика преподавания биологии: учеб. для студ. биол. фак. пед. ин-тов. М.: Просвещение, 1972. 368 с.

17. Похлебаев С.М. Методологические и содержательные основы преемственности физики, химии, биологии при формировании фундаментальных естественнонаучных понятий: дис. … д-ра пед. наук. Челябинск, 2007. 720 с.

18. Реймерс Н.Ф. Краткий словарь биологических терминов: кн. для учителя. М.: Просвещение, 1995. 368 с.

19. Грин Н., Стаут У., Тейлор Д. Биология: в 3 т. Т. 1 / пер. с англ.; под ред. Р. Сопера. М.: Мир, 1990. 368 с.

20. Ленинджер А. Биохимия / пер. с англ.; под ред. А.А. Баева, Я.М. Варшавского. М.: Мир, 1974. 959 с.

METHODOLOGICAL AND METHODICAL ASPECTS OF CARBOHYDRATE METABOLISM STUDY IN PLANT CELLS AS THE ADJOINT SYSTEM

Pohlebaev Sergei Mikhailovich, doctor of pedagogical sciences,
professor of General Biology and Physiology Department
Ageeva Daria Fedorovna, student of Natural Sciences and Technologies Faculty
Tretyakova Irina Anatolyevna, candidate of biological sciences,
associate professor of General Biology and Physiology Department
South Ural State Humanitarian Pedagogical University (Chelyabinsk, Russian Federation)

Abstract. The choice of the research topic is not accidental. Photosynthesis is the basis of anabolism for plants as well as for all living organisms on Earth, while breathing is the basis of their catabolism. In turn, anabolism and catabolism are the links of metabolism, which determines the vital activity of any type of cells and the whole organism. The paper deals with the methodological potential previously justified by the authors of the conjugation category studying such important physiological and biochemical processes as photosynthesis and respiration. The application of this category by students while developing the concepts of photosynthesis and breathing will equip them by an effective methodological means of knowledge. Reflecting the conjugation category extends the ways we understand the principles of the structural organization of matter as a whole, and opens up new prospects, new approaches to solving the most important problems of science and their role in understanding the structure of rational knowledge. The figurative-sign model developed by the authors will allow to develop more effectively the concepts of photosynthesis, breathing and carbohydrate metabolism as a conjugated system at the theoretical level.

Keywords: matter; principles; categories; interaction; coupling; methodology; historical approach; systems approach; modeling; methodology; plant cell; substance; energy; information; evolution; metabolism; photosynthesis; respiration; carbohydrate metabolism; dialectical thinking style.

* * *

УДК 371.38
DOI 10.24411/2309-4370-2019-14311

Статья поступила в редакцию 11.07.2019

ОБУЧЕНИЕ ПРОГРАММИРОВАНИЮ БУДУЩИХ УЧИТЕЛЕЙ ИНФОРМАТИКИ: ЗАДАЧИ СО СПИРАЛЬНО ПОВЫШАЮЩЕЙСЯ СЛОЖНОСТЬЮ

Пугач Валерий Исаакович, доктор педагогических наук,
профессор кафедры информатики, прикладной математики и методики их преподавания
Тюжина Ирина Викторовна, кандидат педагогических наук,
доцент кафедры информатики, прикладной математики и методики их преподавания
Макарова Елена Леонидовна, кандидат педагогических наук,
доцент кафедры информатики, прикладной математики и методики их преподавания
Самарский государственный социально-педагогический университет (г. Самара, Российская Федерация)

Аннотация. В данной статье рассматривается одна из актуальных проблем в подготовке будущих учителей информатики – формирование специальной компетенции: способности использовать методологию программирования для решения задач школьного курса информатики. В противовес классическому подходу в методике обучения, когда каждая тема по разделу «Программирование» закрепляется решением некоторого количества несложных заданий, мы предлагаем решать задачи со спирально повышающейся сложностью (под такими задачами будем понимать крупные проекты, выполнение которых можно совершенствовать на

протяжении изучения всего курса программирования, при этом на каждом этапе результатом работы будет являться полноценная работоспособная программа). Любой «виток» решения, кроме самого первого, может быть опущен без потери функциональности, а каждый блок задания может быть самостоятельно оценен обучающимся. В статье приводится пример подобной задачи, подробно описываются этапы её решения, а также система оценки. Подобная методика была апробирована в Самарском государственном социально-педагогическом университете при подготовке бакалавров направления подготовки 44.03.01 Педагогическое образование (профиль «Информатика»). Предложенная методика показала хорошие результаты как в области мотивации обучающихся, так и с позиции усвоения ими знаний.

Ключевые слова: образование; педагогическое образование; задачи; информатика и информационно-коммуникационные технологии (ИКТ); программирование; высшее образование; метод проектов; практико-ориентированный подход; язык программирования Python; задачи со спирально-повышающейся сложностью; методика обучения программированию; парадигмы программирования.

Постановка проблемы в общем виде и ее связь с важными научными и практическими задачами

На сегодняшний день дисциплина «Программирование» является одной из самых значимых дисциплин при подготовке будущих учителей информатики. Достаточно посмотреть на её объем в учебных планах педагогических вузов (профиль подготовки «Информатика»). Разработана широкая научная база, обеспечивающая подготовку педагогических кадров в области информатики и ИКТ (Т.А. Бороненко [1], Т.В. Добудько [2], А.А. Кузнецов [3], В.В. Лаптев [4], М.П. Лапчик [5], Б.Е. Стариченко [6], М.В. Швецкий [7; 8], Е.К. Хеннер [9] и др.), а также в области алгоритмизации и программирования (А.В. Могилев [10], Д.А. Слинкин [11]). В исследованиях перечисленных авторов достаточно глубоко обсуждаются содержательные и методические аспекты обучения программированию будущих учителей информатики, однако не так много внимания уделено междисциплинарным связям и практико-ориентированным проектам.

В ходе изучения дисциплины будущие учителя знакомятся с несколькими парадигмами и языками программирования. При этом используется различные подходы. Так, М.В. Швецкий [7] предлагает метод демонстрационных примеров, иллюстрирующий определенные аспекты синтаксиса и семантики выбранного языка программирования, реализацию классических алгоритмов (поиск, сортировка и т.д.) и структур данных с помощью средств изучаемого языка. Однако, как показывает практика, получая только готовые примеры, студенты могут переработать программу, но не написать её «с нуля». Как следствие, к сегодняшнему дню зарекомендовал себя с лучшей точки зрения и «де факто» стал классическим подход к обучению программированию, при котором обучаемые сначала знакомятся с теоретическими основами, а затем им предлагается преобразовать или самостоятельно написать ряд несложных программ, используя полученные знания.

Приведем несколько примеров, взятых из задачников по программированию:

Дано целое число N и набор из N положительных вещественных чисел. Вывести в том же порядке дробные части всех чисел из данного набора (как вещественные числа с нулевой целой частью), а также произведение всех дробных частей [12].

В текстовом окне задан многострочный текст. Слова отделяются друг от друга пробелами. Каждое предложение в тексте заканчивается точкой. Последовательно выделите слова, где гласных букв больше, чем согласных [13].

В массиве записана информация о стоимости 20 видов товара. Определить, сколько видов товара имеют стоимость меньшую, чем средняя стоимость всех видов товара [14].

Плюсы таких заданий на начальном этапе изучения программирования очевидны. Они компактны, хорошо проверяют знание конкретного раздела программирования, доступны для понимания большинства студентов.

Однако, если в начальный период обучения классический подход, когда каждая тема закрепляется некоторым количеством несложных заданий, полностью себя оправдывает, то к старшим курсам, когда обучающиеся знакомы с несколькими парадигмами и языками программирования, ценность подхода заметно снижается.

Комплекс разрозненных несложных задач, каждая из которых увязана с изучаемой темой, кажется оторванным от реальности, практическая ценность отдельной задачи неочевидна. Уровень мотивации обучающихся падает.

Анализ последних исследований и публикаций, в которых рассматривались аспекты этой проблемы и на которых обосновывается автор

Отметим, что разрыв между практической ценностью и методическим потенциалом задач часто оказывался в центре внимания педагогов-исследователей. Предлагались различные пути решения этой проблемы. Например, Д.М. Гребнева [15] предлагает изучать программирование при помощи реальных исполнителей – роботов. Использование робототехники в информатике успешно демонстрирует, как на практике работает программный код. Однако такой подход влечет за собой определенные финансовые затраты и не ориентирован на задания, предложенные в государственной итоговой аттестации. Ряд исследователей (П.Ю. Бунаков [16], А.И. Газейкина [17] и др.) предлагают сочетание классического репродуктивного метода и более сложных, но практико-ориентированных курсовых проектов. Также, когда встает вопрос мотивации, многие исследователи [18–21], предлагают использовать метод проектов – метод обучения, основанный на постановке социально-значимой цели и ее практическом достижении. При этом упор делается на предоставлении обучающимся возможности самостоятельно приобретать знания в процессе решения практических задач или проблем, требующих интеграции знаний из различных предметных областей. Пользуется популярностью и кейс-метод – техника обучения, предполагающая анализ

реальной ситуации. Обучающиеся должны проанализировать определенную ситуацию, сделать выводы о причинах проблем, предложить возможные решения и выбрать лучшее из них. Таким образом, академическая теория рассматривается с точки зрения реальных проблемных ситуаций [22; 23].

Нельзя не упомянуть некоторые противоречия. В частности, выполнить крупный проект, стоящий на стыке различных областей знаний, демонстрируя навыки самостоятельной научно-исследовательской деятельности, большинство студентов способно только к моменту выпуска, в рамках выпускной квалификационной работы. Таким образом, мотивация хорошо успевающих студентов повышается, а вот у менее уверенных в своих силах обучающихся она, наоборот, падает.

Второй недостаток – неравномерность распределения усилий во времени. Если речь идет о программировании, то даже спланировать решение сложной задачи, наметить контуры её решения можно, только полностью овладев теоретическим материалом, хорошо представляя себе возможности языка и среды программирования. Как следствие, приступить к выполнению проекта студенты смогут только по завершении всего цикла лекций. При этом, если по ходу лекций знания не закреплялись никакими практическими работами, ожидать хороших результатов не приходится.

Кроме того, получая тему курсовой работы, проекта или кейса в начале семестра, студент зачастую не в состоянии соотнести разделы, которые изучаются в ходе лекционных и практических занятий, с задачами, стоящими перед ним в рамках проекта.

Формирование целей статьи (постановка задания)

Целью предложенного исследования является разработка содержательного и методологического компонентов обучения программированию будущих учителей информатики на финальной стадии изучения языков программирования.

Изложение основного материала исследования с полным обоснованием полученных научных результатов

Рассмотрим содержательную сторону вопроса: примерный набор задач, который предлагается студентам.

Напишите программу «Угадай слово». Программа должна предлагать пользователю слово из категории «Растения», «Животные» или «Еда» (категория выбирается случайным образом). Показывается длина слова и первая буква. За слово, угаданное с первой попытки, пользователь получает 5 баллов, со второй – 4 балла и т.д. После каждых двух попыток компьютер показывает ещё одну букву. Регистр игнорируется (то есть «Кот», «кот» и «КОТ» – это один и тот же ответ).

Компьютер: Это животное, 5 букв, начинается с буквы «К»

Пользователь: Кошка

К.: Неверно!

П.: Крыса.

К.: Неверно! Вторая буква «А».

П: Кабан.

К: Верно, вы заработали 3 балла. Ваша сумма 3 балла. Хотите сыграть ещё?

В зависимости от варианта предлагались задачи «Репетитор по английскому», «Начинаем урок географии», «Анаграммы» и т.д. Общее у этих задач следующее: выбор случайного элемента из базы элементов (в задаче «Угадай слово» – это случайный выбор категории и случайный выбор слова в категории), простейшая обработка текста (длина строки, сравнение строк, обращение к элементу строки), работа с циклами и счетчиками.

В первом приближении такие задачи решаются довольно просто, для их реализации надо знать основы синтаксиса изучаемого языка программирования, понятия списков, кортежей и строк (в случае языка программирования Python нужно уметь работать с подключаемыми модулями, в частности с модулем random). Таким образом, уже после одного-двух лекционных занятий студенты могут построить пусть простую, с текстовым интерфейсом, но работоспособную и интересную программу.

Следующая задача – возможность записи лучших результатов игроков (рейтинг).

Усовершенствуйте задачу 1 следующим образом: если результат игрока входит в топ-5 лучших результатов в игре, ему предлагают ввести свое имя. Имя и результат сохраняются. По завершении игры выводится рейтинг из 5 лучших результатов. Важно! Результаты должны сохраняться даже после закрытия программы.

При каждом запуске программы переменные переопределяются. Таким образом, результаты (имена игроков и набранные ими баллы) должны храниться вне программы. То есть решение этого блока становится невозможным без знания о записи и чтении файлов. При этом мы читаем не абстрактные данные, а совершенствуем ранее созданную программу.

Далее следует изучать процедуры и функции, для того чтобы уметь сокращать код, и изучение библиотек для создания графического интерфейса пользователя. Опять же задача на построение оболочки для своей программы, с одной стороны, более сложная, чем классические задачи, требующие разместить на поле элементы (кнопки, текстовые поля и т.д.), с другой стороны, её практическая ценность гораздо выше. Часто на этом этапе приходится существенно переписать код исходной задачи: переносить часть действий в подпрограммы, изменять логику работы циклов, поскольку это не абстрактное требование, а вопрос целесообразности. Как правило, студенты сами приходят к решению об исправлении кода и начинают лучше понимать алгоритмы, использованные ранее.

На следующем этапе обучаем чтению csv и jason форматов, парсингу сайтов. Очевидно, что даже после всех изменений (улучшения интерфейса, появления возможности записи и хранения результатов) программа остается «игрушечной», так как ещё не содержит достаточно большую базу исходных данных. Угадывать одно из десяти или двадцати слов быстро наскучит, а ручное занесение базы даже из 100 слов – не то, чем должен заниматься программист. Идеальное решение – чтение открытых данных из соответствующих форматов. На этом этапе появляется достаточно широкое расхождение в методах решения в зависимости от варианта задания, однако сложность решения при этом, как правило, варьируется не сильно.

При апробировании этого метода мы использовали как некоторое количество несложных заданий, так и стержневую задачу курса со спирально повышающейся сложностью – программу, которую можно совершенствовать на протяжении всего курса, при этом на каждом этапе результат является полноценной работоспособной программой.

На каждом этапе за решение той или иной проблемы студентам начислялись баллы, причем они были сообщены заранее: так, базовая стоимость задачи – 10 баллов, возможность записи результатов – 4 балла, графический интерфейс – 8 баллов и т.д. Классические задачи оценивались в 1–2 балла. Таким образом, студент, не стремящийся к максимальному баллу, мог пропустить какой-либо виток при решении задачи. Плюсом таких заданий мы считаем то, что ни одна часть задачи, кроме первой – базовой, не является обязательной. Можно приступить к выполнению третьего этапа, минуя второй (программа с графическим интерфейсом, но без возможности записи рейтинга игроков), можно выполнить только первую и четвертую части задачи (программа с текстовым интерфейсом, но большим набором исходных данных) и т.д. Отметим, что на практике студенты редко отказывались от каких-либо этапов. Почти все задачи были выполнены полностью, вызвали интерес студентов к курсу, было отмечено желание заниматься программированием дальше, изучать его самостоятельно. Таким образом, мотивационная составляющая работы оказалась довольно высокой.

С целью проверки и обоснования гипотезы о повышении мотивации студентов при использовании задач со спирально-повышающейся сложностью нами был проведен педагогический эксперимент. На формирующем этапе эксперимента, проводившемся в 2016/2017 и 2017/2018 уч. г. на базе Самарского государственного социально-педагогического университета, были отобраны две группы студентов: контрольная и экспериментальная. Занятия в первой проводились без использования задач со спирально-повышающейся сложностью, вторая занималась по тому курсу, что был разработан нами с учетом обозначенных выше проблем. В контрольную группу вошло 107 студентов (76 в 2016/2017 и 31 в 2017/2018), в экспериментальную – 120 (80 в 2016/2017 и 40 в 2017/2018).

При изучении программирования в Самарском государственном социально-педагогическом университете используется балльно-рейтинговая система (табл. 1).

В качестве аудиторной работы предлагаются лабораторные работы, содержащие задачи по программированию (в контрольной группе только классические, в экспериментальной как классические, так и задачи со спирально-повышающейся сложностью). Самостоятельная работа предполагает самостоятельное решение задач и самостоятельное изучение одного из модулей языка программирования Python; самостоятельная работа на выбор студента – решение задач повышенной сложности в контрольной группе и дополнение к стержневой задаче курса в экспериментальной группе; контрольное мероприятие – тестирование (тест выявляет знание основных понятий программирования и их связи со школьным курсом информатики в соответствии с требованиями образовательных стандартов, терминологии и основных конструкций языка программирования и т.д.).

Интерес представляет распределение баллов по итогам обучения в 2017/18 году в контрольной (рисунок 1) и экспериментальной (рисунок 2) группах.

Несложно заметить, что в контрольной группе максимальные показатели сконцентрированы вокруг пороговых значений – 56, 71 и 86 баллов. Именно столько баллов необходимо набрать, чтобы получить оценки «удовлетворительно», «хорошо» и «отлично» соответственно. Студенты, набрав необходимое для получения положительной оценки число баллов, не стремились продолжить выполнять задания. Разительно отличаются итоги в экспериментальной группе (рисунок 2).

Таблица 1 – Фрагмент балльно-рейтинговой карты дисциплины «Программирование»

Вид контроля		Минимальное количество баллов	Максимальное количество баллов
Модуль 5. Язык программирования Python			
Текущий контроль по модулю:		0	
1	Аудиторная работа	44	56
2	Самостоятельная работа (специальные обязательные формы)	6	12
3	Самостоятельная работа (специальные формы на выбор студента)	0	10
Контрольное мероприятие по модулю		6	12
Промежуточная аттестация		56	100

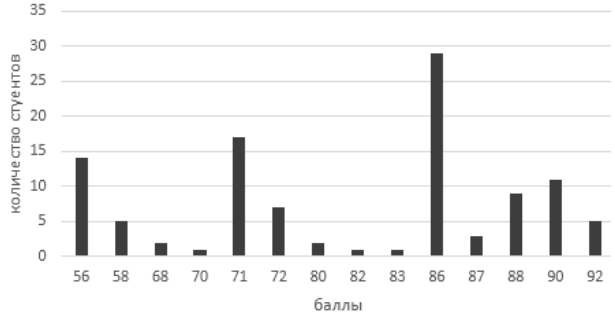

Рисунок 1 – Диаграмма успеваемости в контрольной группе

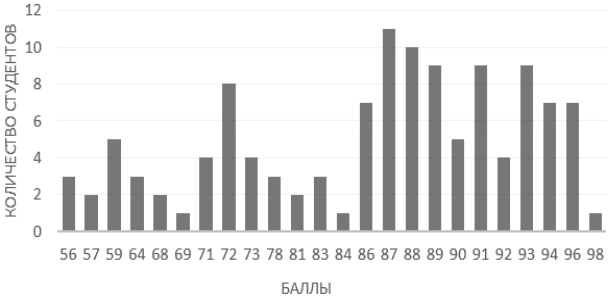

Рисунок 2 – Диаграмма успеваемости в экспериментальной группе

Очевидно, что в экспериментальной группе достижение минимального балла, необходимого для получения желаемой оценки, не становилось сигналом «стоп!». Студенты продолжали работать, проявляя интерес к поставленным задачам и предлагая нетривиальные пути решения. Интересно, что и итоговое тестирование студенты экспериментальной группы в целом прошли лучше, несмотря на то, что вопросы в тесте были едиными для всех студентов и не менялись по сравнению с прошлыми учебными годами.

Для подтверждения эмпирических выводов о различиях в успеваемости групп мы воспользовались статистическими методами. Нами проверялись две гипотезы: 1. В экспериментальной группе больше студентов, набравших более 70 баллов. Различие статистически значимо. 2. В экспериментальной группе больше студентов, набравших более 85 баллов. Различие статистически значимо. В целях достижения максимально точного результата, учитывая объем выборок и шкалу измерения, для проверки гипотезы мы использовали Критерий Хи-квадрат с поправкой Йейтса. Обе гипотезы были подтверждены (таблицы 2 и 3).

Выводы исследования и перспективы дальнейших изысканий данного направления

Таким образом, экспериментально подтверждено, что использование задач со спирально-повышающейся сложностью позволяет усовершенствовать подготовку будущих учителей информатики в области программирования: улучшить мотивацию студентов, равномерно распределить решение сложных задачи по всему курсу, показать практическую ценность программирования. Балльная система оценки различных блоков задач позволит с легкостью включить данные задачи в уровневые фонды оценочных средств. Сформированная специальная компетентность в области программирования позволит стать будущим учителям специалистами высокой квалификации, способными обучать школьников языкам программирования.

Таблица 2 – Распределение баллов в контрольной и экспериментальной группах (относительно оценки «хорошо»)

Распределение баллов в контрольной и экспериментальной группах относительно оценки «хорошо»			
	71 балл и более	менее 71 балла	всего
Экспериментальная группа	104	16	120
Контрольная группа	85	32	107
Всего:	189	48	227
Распределение баллов в контрольной и экспериментальной группах относительно оценки «отлично»			
	86 баллов и более	менее 86 баллов	всего
Экспериментальная группа	79	41	120
Контрольная группа	57	50	107
Всего:	136	91	227

Таблица 3 – Значение критериев

Наименование критерия	Значение критерия	Уровень значимости
Гипотеза 1		
Критерий Хи-квадрат	3,717	0,054
Критерий Хи-квадрат с поправкой Йейтса	3,212	0,074
Гипотеза 2		
Критерий Хи-квадрат	3,717	0,054
Критерий Хи-квадрат с поправкой Йейтса	3,212	0,074

Список литературы:

1. Бороненко Т.А. Методическая система обучения информатике и учебный предмет // Вестник Ленинградского государственного университета им. А.С. Пушкина. 2008. № 2. С. 117–123.

2. Добудько Т.В., Пугач В.И. Методика преподавания информатики: учеб. пособие для студентов. Самара: СамГПИ, 1993. 256 с.

3. Информационно-коммуникационная компетентность современного учителя / А.А. Кузнецов, Е.К. Хеннер, В.Р. Имакаев, О.Н. Новикова и др. // Информатика и образование. 2010. № 4. С. 3–11.

4. Лаптев В.В., Рыжова Н.И. Концепция фундаментализации образования в области информатики и ее реализация в педагогическом вузе // Известия Российского государственного педагогического университета им. А.И. Герцена. 2002. Т. 2, № 3. С. 124–135.

5. Лапчик М.П. О формировании ИКТ-компетентности бакалавров педагогического направления // Современные проблемы науки и образования. 2012. № 1. С. 130.

6. Стариченко Б.Е. Профессиональный стандарт и ИКТ-компетенции педагога // Педагогическое образование в России. 2015. № 7. С. 6–15.

7. Лаптев В.В., Рыжова Н.И., Швецкий М.В. Методическая теория обучения информатике. Аспекты фундаментальной подготовки будущих учителей информатики. СПб.: Издательство Санкт-Петербургского государственного университета, 2003. 350 с.

8. Кудрявцева И.А., Швецкий М.В. Элементы теоретического программирования: комбинаторная логика и теория типов: учеб. пособие к курсу «Программирование». СПб.: Российский государственный педагогический университет им. А.И. Герцена, 2013. 488 с.

9. Проблемы формирования информационно-коммуникационной компетентности учителя российской школы / А.А. Кузнецов, Е.К. Хеннер, В.Р. Имакаев, О.Н. Новикова // Образование и наука. Известия УрО РАО. 2010. № 7 (75). С. 88–96.

10. Могилев А.В. Авторская программа профильного курса по информатике и информационным технологиям // Информатика и образование. 2006. № 8. С. 22–28.

11. Слинкин Д.А. Обучение программированию: выбор системы программирования из нескольких альтернатив // Информационно-коммуникационные технологии в образовании: межвуз. сб. науч. тр.; матлы студ. науч. конф. Екатеринбург: Урал. гос. пед. ун-т, 2013. С. 56–63.

12. Абрамян М.Э. 1000 задач по программированию. Ч. I. Скалярные типы данных, управляющие операторы, процедуры и функции. Ростов-на-Дону: Изд-во РГУ, 2004. 43 с.

13. Мезенцев А.В. Сборник задач по программированию с примерами. Иркутск: Иркутский государственный университет, 2011. 36 с.

14. Златопольский Д.М. Сборник задач по программированию. СПб.: БХВ-Петербург, 2011. 304 с.

15. Гребнева Д.М. Проектирование содержания курса «Основы робототехники» для студентов педагогических вузов // Современные наукоемкие технологии. 2014. № 12–2. С. 313–316.

16. Бунаков П.Ю., Лопатин А.К. Формирование компетентности будущих учителей информатики в области программирования // Современные информационные технологии в образовании: мат-лы XXVIII междунар. конф. Троицк–М.: Московский издательско-полиграфический колледж им. И. Федорова, 2017. С. 321–323.

17. Газейкина А.И. Обучение программированию будущего учителя информатики // Педагогическое образование в России. 2012. № 5. С. 45–48.

18. Слинкин Д.А. Использование метода проектов при обучении программированию в курсе информатики: дис. … канд. пед. наук. Екатеринбург, 2001. 167 с.

19. Беленкова М.А., Рожина И.В. Формирование у будущих учителей информатики компетенции в области проектной деятельности // Актуальные вопросы преподавания математики, информатики и информационных технологий. 2015. № 1. С. 67–71.

20. Дятлова Л.И. Метод проектов в подготовке специалистов в сфере информационных технологий и программирования // Профессиональное образование и рынок труда. 2014. № 5. С. 20–21.

21. Лапчик М.П., Федорова Г.А. Инновационный подход к подготовке педагогических кадров в области информатизации образования // Преподаватель XXI век. 2016. № 4–1. С. 28–41.

22. Конова Е.А., Поллак Г.А. Обучение программированию с использованием метода кейсов // Совет ректоров. 2012. № 2. С. 57–63.

23. Юрьева Т.А., Чалкина Н.А., Лебедь О.А. Применение кейс-метода в обучении бакалавров основам программирования // Педагогические науки. 2016. № 7. 78–82.

TRAINING PROGRAMMING TO PROSPECTIVE IT TEACHERS: TASKS WITH A SPIRAL-INCREASING COMPLEXITY

Pugach Valeriy Isaakovich, doctor of pedagogical sciences,
professor of Computer Science, Applied Mathematics and Teaching Methods Department
Tyuzhina Irina Viktorovna, candidate of pedagogical sciences,
associate professor of Computer Science, Applied Mathematics and Teaching Methods Department
Makarova Elena Leonidovna, candidate of pedagogical sciences,
associate professor of Computer Science, Applied Mathematics and Teaching Methods Department
Samara State University of Social Sciences and Education (Samara, Russian Federation)

Abstract. This paper discusses one of the urgent problems of prospective IT teachers' training – the development of a special competence: the ability to use the programming methodology for solving tasks of the school «Computer Science» course. In contrast to the classical approach in teaching methods, when each topic under the section «Programming» is fixed by the solution of a number of simple tasks, we propose to solve problems with spirally increasing complexity (such tasks will be understood as large projects, the implementation of which can be improved throughout the study of the programming course, while at each stage the result of the work will be a full-fledged workable program). Any «round» of the solution, except for the first one, can be omitted without loss of functionality, and each block of the task can be independently evaluated by students. The paper provides an example of such a problem, describes in detail the stages of its solution, as well as the evaluation system. This technique has been tested at Samara State University of Social Sciences and Education with the students majoring in 44.03.01 Pedagogical Education («Computer Science»). The proposed method has shown good results both in the field of motivation and in terms of learning.

Keywords: education; pedagogical education; tasks; information and communication technologies (ICT); programming; higher education; project method; practice-oriented approach; programming language Python; tasks with spiral-increasing complexity; methods of teaching programming; programming paradigms.

* * *

УДК 371.355.5
DOI 10.24411/2309-4370-2019-14312

Статья поступила в редакцию 05.09.2019

ПРИМЕНЕНИЕ ЛОГИЧЕСКИХ ОПОРНЫХ СХЕМ В КАЧЕСТВЕ СРЕДСТВА ПОВЫШЕНИЯ УРОВНЯ ЭКОЛОГИЧЕСКИХ ЗНАНИЙ СТУДЕНТОВ

Репин Денис Владимирович, кандидат биологических наук, доцент кафедры биоэкологии и химии
Чувашский государственный педагогический университет им. И.Я. Яковлева
(г. Чебоксары, Российская Федерация)
Репина Надежда Васильевна, кандидат биологических наук, преподаватель
Чебоксарский кооперативный техникум Чувашпотребсоюза (г. Чебоксары, Российская Федерация)

Аннотация. В данной статье описываются результаты экспериментальной работы по использованию логических опорных схем в качестве средства повышения экологических знаний студентов технолого-экономи-