УДК 372.862
DOI 10.17513/snt.40846

АНАЛИЗ ИНФОРМАЦИОННЫХ МОДЕЛЕЙ С ИСПОЛЬЗОВАНИЕМ ЯЗЫКА ПРОГРАММИРОВАНИЯ PYTHON

[1]Козлов С. В. ORCID ID 0000-0001-9945-2098, [2]Быков А. А.

[1]Федеральное государственное бюджетное образовательное учреждение высшего образования «Смоленский государственный университет», Смоленск, Российская Федерация, e-mail: svkozlov1981@yandex.ru;
[2]Филиал федерального государственного бюджетного образовательного учреждения высшего образования «Национальный исследовательский университет «МЭИ», Смоленск, Российская Федерация

Цель исследования – рассмотреть возможности языка программирования Python при анализе информационных моделей, представленных в табличном и графическом виде, и сформировать у обучающихся навыки осознанного применения его инструментов. В школьном курсе информатики моделирование применяется для формализации прикладных задач различных областей знаний. Для этого используются табличные модели, схемы, графики. Математические модели служат для анализа свойств изучаемого объекта в разном представлении. Это позволяет выявить значимые параметры, влияющие на поведение объекта, оценить соответствие полученной модели исследуемому явлению. В связи с этим определение критериев моделирования и выбор способа представления информации об объекте играют важную роль. Обучающиеся должны уметь сопоставлять разрозненные факты описания структуры объектов, анализировать связи между ними. Они должны строить логически непротиворечивую систему выводов, которая позволяет получить целостное представление о предмете изучения. В статье описаны принципы анализа взаимосвязей между объектами в ситуации, когда часть информации в условии задачи неизвестна. В качестве примера одной из ключевых задач графического моделирования рассмотрены табличные модели и схемы системы дорог между городами. Авторами описаны принципы исследования данных о населенных пунктах и связывающих их дорогах. Указаны подходы теоретического анализа информации о моделируемой системе городов. Акцент сделан на использовании структур данных языка Python и его функциональных инструментов как эффективного способа моделирования. Авторами охарактеризованы методические особенности построения перестановок для генерации возможных описываемых ситуаций с помощью функций встроенных библиотек языка Python. Рассмотрено применение логических условий для установления соответствия между данными моделей в системе программирования. Представлены результаты экспериментальной деятельности по освоению обучающимися навыков анализа информационных моделей средствами языка Python. Описаны примеры оптимальной записи структуры моделей при формировании у обучающихся навыков осознанного применения инструментов программирования.

Ключевые слова: информатика, обучение, информационная модель, анализ, программирование, таблицы, схемы, язык Python

ANALYZING INFORMATION MODELS USING THE PYTHON PROGRAMMING LANGUAGE

[1]Kozlov S. V. ORCID ID 0000-0001-9945-2098, [2]BykovA. A.

[1]Federal State Budgetary Educational Institution of Higher Education "Smolensk State University", Smolensk, Russian Federation, e-mail: svkozlov1981@yandex.ru;
[2]Branch of the Federal State Budgetary Educational Institution of Higher Education "National Research University "MPEI", Smolensk, Russian Federation

The purpose of the study is to consider the capabilities of the Python programming language when analyzing information models presented in tabular and graphical form, and to form students' skills in the conscious use of its tools. In the school course of computer science, modeling is used to formalize the applied tasks of various fields of knowledge. For this, table models, diagrams, graphs are used. Mathematical models are used to analyze the properties of the object under study in different representations. This makes it possible to identify significant parameters affecting the behavior of the object, to assess the compliance of the obtained model with the studied phenomenon. In this regard, the definition of modeling criteria and the choice of how to present information about an object plays an important role. Students should be able to compare disparate facts of describing the structure of objects, analyze the connections between them. They must build a logically consistent system of conclusions that allows a holistic view of the subject of study. The article describes the principles of analyzing the relationships between objects in a situation where some of the information in the task condition is unknown. As an example of one of the key tasks of graphic modeling, tabular models and diagrams of the system of roads between cities are considered. The authors describe the principles of studying data on settlements and roads connecting them. Approaches of theoretical analysis of information on the modeled system of cities are indicated. The emphasis is on the use of Python data structures and its functional tools as an effective modeling method. The authors described the methodological features of constructing permutations for generating possible described situations using the functions of the built-in libraries of the Python language. The use of logical conditions to establish correspondence between model data in the programming system is considered. The results of experimental activities on mastering the skills of analyzing information models by students using the Python language are presented. Examples of optimal recording of the structure of models in the formation of students' skills of conscious use of programming tools are described.

Keywords: computer science, learning, information model, analysis, programming, tables, diagrams, Python language

Введение

Исследование информационных моделей является основной дидактической линией курса информатики на уровнях основного общего и среднего общего образования. Обучающиеся должны уметь строить разные модели для оценки различных свойств объектов, обоснованно выбирать оптимальные из них [1]. Это является ключевой целью обучения моделированию в информатике. Формирование у обучающихся навыков структурного анализа объектов определяет, насколько цифровая модель будет эффективно описывать ключевые свойства предметов и явлений. Ввиду этого умение обучающихся создавать модели и работать с разными их видами составляет фундаментальную базу подготовки в области информатики [2]. Изучение принципов построения разных моделей с помощью таблиц, диаграмм, графиков воплощает теорию моделирования в соответствии с выбранным подходом в практику при решении задач.

Таким образом табличный способ отражает представление информации в структурированном виде как совокупность строк и столбцов. Он позволяет указать отношения объектов друг с другом или сравнить их свойства. При этом данные могут быть записаны в таблице в числовом формате, в виде текста, как значения логических констант или показателей времени [3, 4]. Диаграммы отражают соотношения между данными, подчеркивают тенденции в сравнении выбранной величины. Столбчатые диаграммы позволяют сравнить количественные характеристики объектов, круговые – доли их частей, линейчатые – изменение процессов во времени. Графики открывают горизонты выявления функциональных зависимостей и изучения корреляционных связей. Они дают возможность оценить динамику и цикличность протекания процессов, выяснить пространственные и временные характеристики.

В то же время в школьном курсе информатики на, казалось бы, очевидный важный междисциплинарный характер не всегда обращают должное внимание [5]. Нередко модели изучают отдельно друг от друга, не демонстрируя их всестороннее применение для целостного представления о системе. Так, во многих случаях табличные модели служат инструментами расчетов. В связи с этим именно математическим операциям и логическим функциям в заготовленных таблицах электронных редакторов уделяют основное время при обучении [6]. При этом вопросу оптимального расположения данных в таблицах, организации эффективного отражения связей между ними времени практически не отводят. Недостаточно уроков остается на визуализацию данных в виде графовых моделей. Основное внимание концентрируется вокруг графиков и диаграмм как средства визуализации числовых зависимостей между двумя величинами [7]. Сетевые модели изучаются только в профильной школе как инструмент распараллеливания информационных потоков. Во многом на практике отсутствует подход комплексного использования разных моделей, особенно в тех случаях, когда они выводят описание объектов и их свойств на новый уровень, отражая их как компонента системы. Кроме того, функциональные возможности систем программирования, несмотря на широкий спектр применения, вообще не рассматриваются в школьном курсе информатики [8]. Даже изучение программирования в школе предполагает акцент на знание базовых алгоритмических конструкций и владение известными алгоритмами обработки данных. Использование сред программирования для моделирования функциональных зависимостей, работы формальных исполнителей, математических расчетов характеристик больших данных только занимает свои позиции в школьном курсе информатики.

Цель исследования – рассмотрение возможностей языка программирования Python при анализе информационных моделей, представленных в табличном и графическом виде, и формирование у обучающихся навыков осознанного применения его инструментов.

Материалы и методы исследования

В педагогическом исследовании для изучения причинно-следственных связей обучения применению инструментов языка программирования Python для анализа информационных моделей использовались наблюдение и тестирование. В экспериментальной деятельности происходило обобщение педагогического опыта оперирования табличными и графовыми структурами. На этапах проведения формирующего и констатирующего эксперимента в инструментальной системе PyCharm изучались правила обработки данных, представленных в табличном и графическом виде.

В педагогическом эксперименте, который длился 4 месяца, принимали участие обучающиеся 11-х классов учебных заведений г. Смоленска. В Смоленском физико-математическом лицее ЯВИР при МИФИ и МЭИ были задействованы 19 чел., в средней школе № 6 г. Смоленска – 14 чел. Программа обучения в них соответствует изучению информатики на профильном уровне.

Результаты исследования и их обсуждение

При изучении методов анализа информационных моделей важно продемонстрировать связь выбранного подхода при исследовании требуемых свойств объекта. Одними из ключевых задач моделирования являются поиск дорог и их длины между пунктами на карте, вычисление кратчайшего пути, определение маршрута, содержащего минимальное или максимальное число промежуточных пунктов. Ответы на эти вопросы можно получить, конструируя и анализируя одну модель, например таблицу, схему или график. При этом выбор вида модели будет находиться в прямой зависимости от постановки вопроса задачи. Компактно записать данные о наличии дорог между пунктами можно табличным способом. Отразить структуру сети и проложить маршрут от одного пункта к другому удобно на дорожной схеме.

В то же время задача построения информационной модели по неполным данным, манипулирования частичной информацией, представленной всевозможными способами, а следовательно, выявления всех характеристических свойств системы дорог между городами является объективно более сложной. При ее решении необходимо исследовать зависимости, отраженные в моделях по-разному.

Так, при определении протяженности дорог между населенными пунктами данные можно отразить и в таблице, и на схеме. Табличный способ позволяет оперировать числовыми данными, записанными на пересечении строки и столбца. Эти числа соответствуют длине пути из одного города в другой. Названия городов будут указаны в первой строке и продублированы в первом столбце таблицы, если они известны. В противном случае города можно обозначить, проиндексировав их и задав общим именем, например «П1». На схеме линии отображают наличие дороги между населенными пунктами, а точки – сами города с подписью названия около них. Длина дороги записывается над линией, если по ней можно двигаться в двух направлениях. В случае одностороннего вектора движения линия изображается в виде дуги, над которой записывается длина в прямом направлении, а под ней – в обратном направлении. Если длина дороги неизвестна, то число не указывается, при этом можно ввести обозначение, используя для этого малые буквы латинского алфавита.

Исследуя зависимости на двух моделях, в табличной форме и на схеме, необходимо сопоставить одни данные с другими. По информации о протяженности пути сначала предположить, а затем подтвердить или опровергнуть, между какими городами на схеме проложена данная дорога. После чего определить все возможные комбинации расположения городов. Можно пойти и от обратного: по наличию дороги между населенными пунктами, а также числу входящих путей в города сделать вывод о том, каким городам в таблице они соответствуют. Проводя системный двусторонний анализ двумя методами, можно определить общую схему дорог между населенными пунктами и их протяженность.

В таких случаях анализ данных, как правило, начинают со схемы. Определяют, сколько дорог ведет в каждый из городов. Затем применяют первый принцип: если есть уникальные города с числом дорог, отличным от всех других городов, то в таблице ищут строку или столбец, так как данные в ней симметричны относительно главной диагонали, соответствующие данному городу на схеме по числу значений в горизонтали или вертикали. Так поступают со всеми такими населенными пунктами, определяя их названия и расположение в таблице. После этого, если в ней есть комбинация «строка – столбец» или «столбец – строка», которая ведет в единственный неизвестный город, то по схеме выясняют его название и заносят сведения о нем в первую строку и первый столбец вместо названия пункта «П». Если же на схеме города неуникальны или если в таблице исчерпан выбор дорог, однозначно ведущих в один из оставшихся для строки или столбца город, то применяют второй принцип. В списке городов с одинаковым числом дорог, определенных на схеме, выясняют значения дополнительного характеристического свойства. Принцип представляет собой формирование кортежа, состоящего из значений числа дорог, ведущих в города, в которые приводят пути из рассматриваемого города. По этим дополнительным показателям определяют остальные города в таблице. Для этого выявляют строку или столбец, в которых числа расположены в столбцах или строках соответственно с указанным набором числа дорог. Если же и после этого остаются города, которые невозможно указать однозначно, то оба принципа применяют последовательно к цепочке городов, увеличивая на каждом шаге их количество в исследуемой группе. Отметим, что второй принцип является более сильным, чем первый. Его можно применять независимо от первого принципа сразу на анализируемой схеме. Тем не менее первый принцип проще в применении и более понятен, так как оперирует только одной характеристикой для города – коли-

чеством дорог, ведущих в него. Для схем с простой структурой или частями, на которые можно разбить основную схему – районами сети, определяющими группу городов с дорогами между ними, следует применять первый принцип. Он дает более быстрый результат в выявлении полного описания информационной модели. Для схем с городами с одинаковой структурой больше подходит второй принцип. Он позволяет выявить сразу совокупность населенных пунктов, как подсеть, и выяснить характер связи городов с одинаковым числом дорог.

В то же время такой подход к анализу информационных моделей применим в случае относительно небольшого числа городов и дорог между ними в исследуемых табличных моделях и схемах. При росте числа городов эффективность его применения снижается. В этом случае можно говорить о необходимости автоматизации рассмотренных принципов. Это позволяет реализовать алгоритмический подход с использованием инструментов систем программирования, например таких, как язык Python функциональной среды PyCharm [9, 10]. Тем самым можно продемонстрировать обучающимся теоретическую значимость изученных алгоритмов и применимость методов программирования на практике.

При реализации данного подхода необходимо задать систему, которая характеризует строки таблицы с помощью кортежей чисел, соответствующих индексам городов «П». А схема определяет пути из одного города в другой по названиям населенных пунктов. Затем остается описать программный код, который обрабатывает всевозможные комбинации городов и дорог между ними, непротиворечащие табличной информации и данным на схеме.

Программный код (составлен авторами) на языке Python для решения таких задач может выглядеть следующим образом. В нем система дорог между городами задана двумя списками table и graph в соответствии с таблицей населенных пунктов и схемой связей между ними. При этом в условии в таблице должны быть определены дороги между городами, как число, звездочка или иное обозначение в ячейках на пересечении строк и столбцов. На схеме указаны дороги между населенными пунктами в виде линий.

```
from itertools import permutations
table = ‘457 346 24 123 167 257 156’.split()
graph = ‘АБ АВ БВ БД ВЕ ДГ ГЕ ДЖ ЖЕ ГЖ’.split()
print(‘1 2 3 4 5 6 7’)
for p in permutations(‘АБВГДЕЖ’):
   if all(str(p.index(c2) + 1) in table[p.index(c1)] for c1, c2 in graph):
      print(*p)
      break
```

Во-первых, необходимо отдельно импортировать функцию *permutations* из библиотеки *itertools* языка Python или все ее инструменты [11, 12]. В этом случае после инструкции *import* следует указать символ «*». Функция permutations позволяет сгенерировать всевозможные перестановки элементов. В рассматриваемом примере это перестановки названий городов ‘АБВГДЕЖ’, описанных как строка.

Во-вторых, необходимо задать два списка, которые соответствуют городам и дорогам. Список *table* представляет собой кортежи строк, где каждая из них указывает доступные связи для вершины с некоторым порядковым номером. Например, вторая запись в списке *table* – «346» – иллюстрирует, что город с номером 2 должен быть связан дорогами с городами 3, 4 и 6. Обычно они перечисляются по порядку следования строк в таблице сверху вниз. При этом порядок можно изменить, он не влияет на построение информационной модели. Но в таком случае необходимо будет поменять и порядок возможных вершин модели. Поэтому делать это в решении нецелесообразно, и последовательность кортежей для строк следует сохранять. Отметим, что особенно важно корректно через пробел указать пути в возможные города, то есть вершины и ребра, которыми они соединены. Список *graph* задает дороги на схеме, они соответствуют ребрам сетевой модели. Так, например, «БВ» означает, что между городами «Б» и «В» на схеме есть путь. Ниже созданных списков выводится строка с названиями городов, в том порядке, в котором они определены в первой строке таблицы или ее первом столбце. Обычно, даже если названия городов указаны в виде «П1», то в команде вывода *print* указывают только их индексы, нумерация которых начинается с «1».

В-третьих, цикл в программе предназначен для генерации перестановок, которые будут сохраняться в переменной *p*. Каждая перестановка – это возможное соответствие букв названий городов номерам в табличной

модели. Например, если на очередном этапе работы программы будет обрабатываться перестановка ‘БАГВЕДЖ’, то она отражает порядок Б – 1, А – 2, Г – 3, В – 4, Е – 5, Д – 6, Ж – 7 для городов на схеме. Для каждой перестановки *p* проверяется условие с помощью встроенной функции системной библиотеки *all*. Она возвращает значение *True*, если проверяемые комбинации для городов и дорог между ними будут истинными. Значение *False* будет сформировано в том случае, если какая-то запись в двух списках *table* и *grap*h противоречит друг другу.

Внутри функции *all* организован цикл *for c1, c2 in graph*, который для каждой дороги из пункта *c1* в пункт *c2* находит в сгенерированной перестановке p позицию первого города и проверяет, что второй город находится в списке доступных населенных пунктов. Функция *str(p.index(c2) + 1)* преобразует номер второго города в строку с учетом нумерации в Python с нуля, а в таблице с единицы. Затем функция *in*, проверяющая вхождения элемента в объект *table[p.index(c1)]*, определяет, есть ли город *c2* в списке доступных пунктов *c1*.

В-четвертых, команда *print(*p)* печатает на экране найденную перестановку, если функция *all* определила значение *True*. Буквы названий городов выводятся в одной строке через пробел. Таких перестановок, удовлетворяющих условию задачи, может быть несколько. В этом случае будет выведена первая подходящая перестановка. Команда *break* прервет выполнение программы при нахождении подходящего ответа. Заметим, что наличие альтернативных перестановок объясняется симметричностью схемы городов в отдельных заданиях. Программа позволяет, исключив команду *break*, вывести на экран все из них. При этом расположение в них городов не будет влиять на дальнейший ответ.

В-пятых, полученные возможные перестановки после запуска программы позволяют дополнить обе информационные модели – таблицу и схему. Для таблицы будет известно соответствие названий пунктам, а для схемы – протяженность дорог между ними. После этого, объединив совокупные данные, можно ответить на разные вопросы заданий. Например, можно указать длину дороги из одного населенного пункта в другой или сумму протяженностей нескольких дорог. Можно вычислить длину кратчайшего пути из одного города в другой. Можно определить соответствие номеров пунктов их названиям.

В ходе экспериментальной работы обучающимся предлагались для выполнения такие задачи. Система содержала задания разного уровня сложности. Обучающиеся в ходе эксперимента сначала учились решать задания «на бумаге», проводя анализ «вручную». Затем они выполняли задания, предъявляемые в разном формате – в виде таблиц и схем, в системе программирования Python с помощью компьютерных средств моделирования данных. Задания были разработаны в программном комплексе «SmartTeacher» с использованием средств генеративного искусственного интеллекта в соответствии с образовательными картами [13, 14]. Это позволяло в учебном процессе предлагать обучающимся наборы индивидуальных заданий, от простых упражнений к сложным, учитывая текущий уровень их учебных достижений.

На начальном этапе обучения практические задания отражали тренировку умений оперировать данными табличных моделей. Затем были изучены особенности моделирования с помощью схем. На втором этапе обучающиеся исследовали алгоритмы анализа на табличных моделях и схемах, изучали приемы сопоставления данных, представленных на них. На третьем этапе задания предлагалось выполнить в среде PyCharm с помощью инструментов программирования. Для этого обучающиеся повторяли правила объявления списков в языке Python, способы их идентификации и приемы обработки данных в них. На четвертом, заключительном этапе обучающимся предлагалось выполнять разнообразные задания с использованием разных подходов, изученных на предыдущих этапах.

В конце обучения была проведена итоговая контрольная работа, которая состояла из восьми заданий. Задания соответствовали разным этапам обучения, по два упражнения на каждый способ решения: табличный, на схему, на агрегацию данных моделей «ручным» и «программным» методами. Результаты проведенной диагностики показали осознанное применение обучающимися изученных подходов к анализу информационных моделей. Так, высокого уровня усвоения учебного материала достигли 14 чел., продвинутого уровня – 15 чел., базового уровня – 4 чел. Это позволяет заявить о высоких показателях обучения, соответствующих нормальному распределению с платформой в области высоких значений. Гипотеза исследования состояла в том, что использование языка программирования Python для анализа табличных и графических информационных моделей способствует формированию у обучающихся навыков осознанного выбора и применения программных инструментов при решении практических задач.

Качественный анализ условий и результатов эксперимента. Результаты экспериментальной работы позволяют утверждать, что обучающиеся усвоили правила представления и анализа табличных и графических информационных моделей. Они научились выделять свойства объекта в зависимости от ситуации в задаче, после чего строить информационную модель, наиболее точно описывающую выбранные для исследования характеристики. У обучающихся были сформированы навыки оперирования числовыми величинами в табличных моделях, обработки связей между городами на схемах дорожной сети. В системе программирования Python ими были получены знания о генерировании перестановок, приобретены умения создавать списки из строковых текстовых данных и обрабатывать информацию в них, соответствующую вершинам и ребрам графа, представленную в табличном виде и на схеме. При этом обучающиеся параллельно закрепили навыки записи составных условий при использовании логических конструкций.

Задания, генерируемые в системе «SmartTeacher» [15], были созданы с учетом ошибок, типичных при выполнении таких заданий. Это позволило детально объяснить логику решения задачи, продемонстрировав особо значимые элементы, приводящие к неправильному ответу. Это позволяет сделать вывод, что гипотеза экспериментального исследования находит подтверждение в проделанной работе.

Заключение

Таким образом, работа, проведенная в ходе эксперимента, демонстрирует важность анализа информационных моделей, представленных в табличном и графическом виде, как с помощью приемов сопоставления и синтеза информации, так и с применением языка программирования Python на основе формирования и обработки списков элементов и проверки взаимосвязей между данными. При этом формирование у обучающихся представлений о структуре программного кода решения позволяет им сократить время на исследование свойств объекта и дать ответ на основе обобщенных характеристик. Это дает возможность заложить у будущих IT-специалистов навыки осознанного выбора инструментов анализа информационных моделей.

Список литературы

1. Бешенков С. А., Шутикова М. И., Лысенкова О. В. Информационное моделирование как инструмент формирования универсальных учебных действий // Преподаватель XXI век. 2015. № 2–1. С. 173–180. EDN: UAXKPL.

2. Гербеков Х. А., Башкаева О. П. Место математического и компьютерного моделирования в системе современного общего образования // Вестник Российского университета дружбы народов. Серия: Информатизация образования. 2017. Т. 14. № 1. С. 17–23. DOI: 10.22363/2312-8631-2017-14-1-17-23.

3. Макаренко М. Д. Образовательный туризм и математическое моделирование // Информатика в школе. 2020. № 5 (158). С. 12–18. DOI: 10.32517/2221-1993-2020-19-5-12-18.

4. Потехина Е. В. Поддержка курса математической логики средствами информационных технологий // Мир науки, культуры, образования. 2018. № 3 (70). С. 329–330. EDN: XUNFCP.

5. Козлов С. В., Быков А. А. Особенности изучения междисциплинарных тем школьных курсов математики и информатики с помощью методов математического моделирования // Проблемы современного образования. 2021. № 5. С. 250–261. DOI: 10.31862/2218-8711-2021-5-250-261.

6. Попов В. С., Алефиренко Е. А., Черницына Л. Ю. Компетенции для вычислений в электронных таблицах: нахождение минимальных и максимальных значений по условию, функции МИНЕСЛИ, МАКСЕСЛИ // Вестник МГПУ. Серия: Информатика и информатизация образования. 2025. № 1 (71). С. 90–100. DOI: 10.24412/2072-9014-2025-171-90-100.

7. Шамсутдинова Т. М. Развитие навыков визуализации и визуальной аналитики в курсе информатики // Информатика и образование. 2023. Т. 38. № 3. С. 16–23. DOI: 10.32517/0234-0453-2023-38-3-16-23.

8. Козлов С. В., Быков А. А. Обучение анализу логических выражений с использованием языка программирования Python // Современные наукоемкие технологии. 2025. № 4. С. 121–126. DOI: 10.17513/snt.40375.

9. Каракозов С. Д., Маняхина В. Г. Python как базовый язык обучения программированию в школе // Информатика в школе. 2020. № 1 (154). С. 26–30. DOI: 10.32517/2221-1993-2020-19-1-26-30.

10. Кондратьева В. А. Обучение основам программирования на языке Python в школьном курсе информатики // Вестник МГПУ. Серия: Информатика и информатизация образования. 2021. № 1 (55). С. 8–16. DOI: 10.25688/2072-9014.2021.55.1.01.

11. Маркелов В. К., Завьялова О. А. Язык программирования Python как альтернативный инструмент для решения заданий ЕГЭ по информатике // Информатика в школе. 2023. № 2 (181). С. 63–72. DOI: 10.32517/2221-1993-2023-22-2-63-72.

12. Ильченко О. Ю., Сырицына В. Н., Кадеева О. Е. Решение задач ЕГЭ по информатике средствами языка Python // Высшее образование сегодня. 2021. № 11–12. С. 42–54. DOI: 10.18137/RNU.HET.21.11-12.P.042.

13. Киселева О. М. Использование математических методов для формализации элементов образовательного процесса // Концепт. 2013. № 2. С. 36–40. EDN: PVXPOV.

14. Быков А. А., Козлов С. В., Исаков М. А. Использование web-платформы “smartteach” как инструмента построения образовательных карт // Естественные и технические науки. 2024. № 10 (197). С. 171–173. EDN: BHYICU.

15. Быков А. А., Козлов С. В. Интеллектуальное формирование наборов тестовых заданий с использованием web-платформы “smartteach” // Естественные и технические науки. 2025. № 5 (204). С. 147–149. EDN: GQUHIM.

Конфликт интересов: Авторы заявляют об отсутствии конфликта интересов.

Conflict of interest: The authors declare that there is no conflict of interest.

Финансирование: Авторы заявляют об отсутствии внешнего финансирования.

Financing: The research was performed without external funding.