

DOI: 10.32517/0234-0453-2023-38-2-66-74

# СИСТЕМА КОНТРОЛЯ ЗНАНИЙ И ПРАКТИЧЕСКИХ НАВЫКОВ ПО ПРОГРАММИРОВАНИЮ

И. А. Жуков<sup>1</sup> ✉, Ю. Л. Костюк<sup>1</sup><sup>1</sup> *Национальный исследовательский Томский государственный университет, г. Томск, Россия*

✉ lgZhukov963@yandex.ru

## Аннотация

В статье описан программный комплекс (система контроля знаний), позволяющий на практике применять новый тип заданий для автоматизированного контроля знаний и навыков по программированию и основанный на конструктивно-выборочном методе. В таких заданиях обучающийся из предложенных компонентов (строк программного кода) составляет ответ (программу), причем задание предполагает множество правильных ответов. Новый тип заданий позволяет контролировать знание ранее изученных алгоритмов и навыки составления программ, дополняя тестирование методом черного ящика. Разработанная система автоматизированного контроля состоит из двух подсистем: «Обработка» и «Демонстрация». Подсистема «Демонстрация», которая является веб-приложением, дает минимально необходимый набор возможностей для применения такого типа заданий в учебном процессе. Также она служит образцом, который может быть использован при разработке расширений для других систем. Подсистема «Обработка» позволяет интегрировать новый тип заданий по программированию в различные системы автоматизированного контроля и LMS, если те предоставляют возможность для расширения. Перед применением нового типа заданий преподаватель вручную или при помощи разработанных авторами инструментов создает шаблон и список компонентов. Алгоритм контроля оценивает соответствие ответа обучающегося шаблону и дает оценку соответствия в интервале [0; 1].

**Ключевые слова:** конструктивно-выборочный метод, автоматизированный контроль знаний, система контроля знаний, обучение программированию.

## Для цитирования:

Жуков И. А., Костюк Ю. Л. Система контроля знаний и практических навыков по программированию. *Информатика и образование*. 2023;38(2):66–74. DOI: 10.32517/0234-0453-2023-38-2-66-74

# THE SYSTEM FOR AUTOMATED ASSESSMENT OF KNOWLEDGE AND PRACTICAL SKILLS IN PROGRAMMING

I. A. Zhukov<sup>1</sup> ✉, Yu. L. Kostyuk<sup>1</sup><sup>1</sup> *Tomsk State University, Tomsk, Russia*

✉ lgZhukov963@yandex.ru

## Abstract

The article describes a software package (automated knowledge assessment system) that allows in practice to apply a new type of tasks for automated control of knowledge and skills in programming, based on a constructive-selective method. In such tasks, the student constructs an answer (program) from the proposed components (lines of program code), and the task involves many correct answers. A new type of task allows assessing the knowledge of previously studied algorithms and skills of writing programs, that complement black-box testing. The developed automated control system consists of two subsystems: “Processing” and “Demonstration”. The “Demonstration” subsystem, which is a web application, provides the minimum necessary set of features for using this type of tasks in the educational process, and it also serves as an example that can be used when developing extensions for other systems. The “Processing” subsystem allows integration of a new type of programming tasks into various automated knowledge assessment systems and LMS, if they provide an opportunity for expansion. A teacher creates manually or using the tools developed by the authors of this article a pattern and a list of components before applying a new type of task. A student makes up the answer (program) from the proposed components (lines of program code). The assessment algorithm evaluates the compliance of the student’s answer with the pattern and assesses compliance in the interval [0; 1].

**Keywords:** constructive-example method, automated knowledge control, knowledge control system, programming training.

## For citation:

Zhukov I. A., Kostyuk Yu. L. The system for automated assessment of knowledge and practical skills in programming. *Informatics and Education*. 2023;38(2):66–74. (In Russian.) DOI: 10.32517/0234-0453-2023-38-2-66-74

## 1. Введение

В настоящее время информационные технологии активно интегрируются в образовательный процесс. Можно выделить три основных направления применения информационных технологий:

- организация взаимодействия преподавателя с обучающимися;
- автоматизация контроля знаний и практических навыков;
- разработка автоматизированных «ассистентов», выполняющих часть функций преподавателя.

Для организации взаимодействия преподавателя с обучающимися при изучении программирования обсуждается применение LMS (Learning Management System) [1, 2] и систем контроля версий [3].

Самым распространенным способом контроля теоретических знаний является применение заданий, основанных на альтернативно-выборочном методе. Такие задания используют, например, при проверке знаний конструкций изучаемого языка программирования [4–7]. В работе Д. В. Шагбазяна и А. А. Штанюка [5] применяются тестовые задания с выбором одного правильного ответа. В [6, 7] приведены задания, в которых обучающемуся дан фрагмент программного кода и вопрос «каким будет результат выполнения приведенного фрагмента?». Обучающийся выбирает свой ответ из предложенного списка. Кроме того, применяются задания с открытым ответом [2, 7] и задания на соответствие [7].

При проверке практических навыков популярно автоматизированное тестирование программ [8–10]. В системе проверки преподаватель задает наборы входных и выходных данных для каждого задания. Обучающийся загружает программу, затем система проводит тестирование методом «черного ящика» — на вход программы подаются наборы данных, и результат выполнения сравнивается с соответствующим набором выходных данных.

Автоматизированные «ассистенты» имеют две области применения: визуализаторы отладки [11] и генераторы обратной связи [12, 13]. В генераторах обратной связи для каждого задания имеется «эталонная» программа преподавателя, которая сопоставляется с программой обучающегося. Разница между ними может быть предъявлена в качестве подсказки. Авторы [14] провели эксперимент, в котором участвовало 42 обучающихся. Каждому были даны три задачи и возможность при работе использовать либо Python Tutor [11], либо CLARA [12], либо тестирование методом «черного ящика». Обработка результатов эксперимента не дала четких представлений об эффективности, однако отмечено, что применение инструментов уменьшило количество попыток обучающегося для получения правильной программы.

Из изложенного выше можно сделать вывод, что для обучения программированию не представляет сложности организовать взаимодействие преподавателя с обучающимися. Разнообразные типы тестовых

заданий позволяют контролировать теоретические знания. При автоматизированной проверке практических навыков существует проблема: тестирование методом «черного ящика» дает представление лишь о функционировании программы, но не о ходе решения и структуре программы. Для качественного обучения программированию важно развить навык выбора подходящего алгоритма и средств языка.

Указанная проблема легла в основу исследования авторов данной статьи, которые в течение четырех лет работают над реализацией автоматизированного контроля знаний и практических навыков при обучении программированию с применением конструктивно-выборочного метода\*. Вместе с заданием выдается список допустимых смысловых единиц (компонентов), из которых обучающемуся предлагается составить ответ. По сравнению с альтернативно-выборочным методом конструктивно-выборочный требует творческого подхода от обучающегося [15]. Применение конструктивно-выборочного метода, с одной стороны, позволит контролировать знание ранее изученных алгоритмов и навыки составления программ, дополняя тестирование методом «черного ящика». С другой стороны, оно позволяет создать «ассистента», который может выявлять ошибки обучающихся.

Программная реализация конструктивно-выборочного метода не встречается в автоматизированных системах контроля знаний, и в литературе его применение практически не обсуждается. В связи с этим представляется актуальной разработка комплекса программ для контроля знаний (система контроля знаний) с использованием конструктивно-выборочного метода. Такой комплекс включает в себя программный интерфейс приложения, позволяющий применять этот метод в существующих системах контроля знаний и системах управления обучением путем разработки соответствующих расширений.

## 2. Исследование и формализация предметной области

На первом этапе авторами данной статьи был проведен анализ задач по основам программирования, их решений (текстов программ) на Python, C++ и Pascal. Был сделан вывод, что для автоматизированной проверки и оценки решений таких задач может быть применен конструктивно-выборочный метод. Применение этого метода позволяет создать новый тип заданий, в которых обучающийся из предложенных компонентов (строки программного кода) составляет ответ (программу). Как правило, существуют различные варианты программ для решения одной конкретной задачи, и обучающийся может сконструировать любой из них. Поэтому был разработан способ

\* Довгялло А. М., Ющенко Е. Л. Программированный учебник. Глушков В. М., Амосов Н. М., Артеменко И. А. Энциклопедия кибернетики. В 2 т. Киев: Главная редакция Украинской советской энциклопедии; 1975;2:243–244. Режим доступа: [https://scask.ru/f\\_book\\_kiber2.php?id=358](https://scask.ru/f_book_kiber2.php?id=358)

анализа программ, который позволяет для данной задачи оценить соответствие ответа обучающегося правильным программам.

Затем была проведена формализация и создана модель представления задания [16]. Введены базовые понятия модели: компонент, элемент и шаблон.

**Компонент** — строка программного кода. Ответ обучающегося состоит из компонентов. Для представления множества правильных ответов используются элементы и шаблоны.

**Элементом** названо логическое объединение компонентов. В модели выделено три типа элементов:

- «Один компонент»;
- «Один из компонентов»;
- «Перестановка компонентов».

Элементы типа «Один из компонентов» объединяют одинаковые по смыслу, но различные по форме компоненты, которые являются взаимозаменяемыми. Элемент типа «Перестановка компонентов» объединяет компоненты, последовательность которых не важна. При автоматизированном контроле отсутствие компонентов элемента считается ошибкой. Для указания степени серьезности ошибки преподаватель может пометить элемент как «Рубежный» или «Допускается отсутствие». Отсутствие компонентов рубежного элемента считается критической ошибкой, т. е. сильнее снижает общую оценку ответа обучающегося. Отсутствие компонентов элемента с меткой «Допускается отсутствие» считается некритической ошибкой, т. е. незначительно снижает общую оценку ответа обучающегося. Элементы являются основой шаблонов.

**Шаблон** описывает семейство схожих программ, которые незначительно отличаются синонимами. Пример — изменение последовательности компонентов, которое не влияет на результат выполнения программы. Множество шаблонов, описывающих различные семейства программ, решающих конкретную задачу, названо U-шаблон.

U-шаблон имеет два способа записи: запись на языке шаблонов и промежуточное представление в виде массива словарей. Промежуточное представление необходимо для автоматической обработки и использует формат JSON\*. Несмотря на то, что формат позиционируется как удобный для чтения и написания человеком, промежуточное представление шаблона получается сложным для использования преподавателями. С целью упрощения чтения и создания шаблонов для преподавателя разработан язык шаблонов.

Для проверки правильности ответа обучающегося (программы) авторами данной статьи разработан специальный контролирующий алгоритм [17]. Алгоритм контроля использует понятие эксперта. Эксперт — это автомат, который оценивает соответствие ответа обучающегося одному шаблону. Ответом является программа, которую обучающийся составил из предложенных компонентов, т. е. ответ

представлен как последовательность компонентов. Если в U-шаблоне для задания имеется более одного шаблона, то ответ будут оценивать несколько экспертов. Оценкой эксперта является обыкновенная дробь в интервале  $[0; 1]$ , которая характеризует степень соответствия ответа шаблону. Помимо оценки, формируется структура данных «мнение эксперта», которая состоит из причины завершения анализа, списка компонентов, которые остались в ответе после завершения анализа, и таблицы ошибок. Причины завершения анализа могут быть трех типов:

1. «Закончился шаблон». Это означает, что были проанализированы все элементы шаблона, но остались не просмотренные компоненты в ответе обучающегося. В данном случае наиболее вероятно, что программа содержит лишние компоненты.
2. «Закончился ответ». Это означает, что были проанализированы не все элементы шаблона, но были просмотрены все компоненты в ответе обучающегося. В данном случае наиболее вероятно, что в программе упущена часть компонентов.
3. «Закончился и ответ, и шаблон». Это означает, что были проанализированы все элементы шаблона и просмотрены все компоненты ответа. При этом возможны две ситуации: получен полностью правильный ответ или частично правильный ответ. Например, при анализе перед элементом были найдены лишние компоненты.

Таблица ошибок содержит элементы, при анализе которых было выявлено несоответствие, тип ошибки, лишние компоненты и отсутствующие компоненты. Существует три типа ошибок:

- «элемент частично присутствует»;
- «элемент присутствует с лишними компонентами»;
- «элемент присутствует, но часть компонентов отсутствует, и имеются лишние компоненты».

Если по завершении анализа указана причина «Закончился шаблон», то все оставшиеся в ответе компоненты будут добавлены в таблицу ошибок с пояснительным текстом «Лишний компонент после завершения анализа».

Таким образом, проведена формализация части предметной области, связанной непосредственно с заданием: представление задания, ответа на задание, контроль правильности выполнения. Кроме того, для разработки системы контроля знаний выделены роли пользователей и варианты использования для каждой роли: преподаватель составляет задания и просматривает результаты контроля, обучающийся выполняет задания и получает оценки.

### 3. Описание системы

На основе результатов исследования предметной области авторами данной статьи разработана система автоматизированного контроля. Она состоит из двух основных подсистем, названных «Обработка

\* Официальная страница формата на русском языке. <https://www.json.org/json-ru.html>

и «Демонстрация». Пользователи (преподаватели и обучающиеся) взаимодействуют с подсистемой «Демонстрация». Подсистема «Обработка» содержит реализацию модели задания и предназначена для машинного взаимодействия, т. е. взаимодействует с подсистемой «Демонстрация» или любой другой системой для автоматизации процесса обучения, если та предоставляет возможность для расширения.

Подсистема «Обработка» является сервером, написанным на языке Python, с использованием фреймворка FastAPI\*. Для взаимодействия сервер использует REST API. Подсистема реализует четыре функции:

- трансляция размеченного файла исходного кода;
- компиляция U-шаблона;
- декомпиляция U-шаблона;
- проверка ответа.

*Трансляция размеченного файла исходного кода* принимает на вход размеченный файл исходного кода и возвращает список компонентов для представления задания обучающемуся и U-шаблон в промежуточном представлении.

*Компиляция* предназначена для перевода U-шаблона с языка записи шаблонов в промежуточное представление.

*Декомпиляция*, наоборот, переводит промежуточное представление в запись на языке шаблонов.

*Проверка ответа* принимает U-шаблон и ответ обучающегося в виде строки, содержащей последовательность номеров компонентов, разделенных точкой с запятой. U-шаблон может быть передан как в промежуточном представлении, так и в виде записи на языке шаблонов. Во втором случае также будет использована функция компиляции. Проверка ответа возвращает мнение эксперта с наивысшим баллом, информацию обо всех экспертах и список мнений всех экспертов. За счет описанных выше функций и способа взаимодействия с сервером возможна интеграция с другими системами автоматизированного контроля знаний или LMS.

Подсистема «Демонстрация» является веб-приложением, которое включает в себя все необходимое для создания и редактирования заданий, представления их обучающимся и контроля знаний. Это приложение дает минимально необходимый набор возможностей для применения в учебном процессе, также оно служит образцом, который может быть использован при разработке расширений для других систем. Серверная часть приложения (Backend) написана на языке Python с использованием фреймворка Django\*\*, клиентская часть (Front-end) — на языке TypeScript с использованием библиотек React\*\*\* и Redux\*\*\*\*.

В ходе описания работы пользователей с подсистемой «Демонстрация» в качестве примера будем использовать задачу «Сложение всех элементов массива» со следующей формулировкой: «Из заданных ниже компонентов составьте программу, результатом работы которой будет вычисление суммы всех элементов массива  $m$ . Длина массива  $n$ . Результат суммирования должен быть записан в переменную  $S$ ».

Рассмотрим процесс подготовки задания. Преподавателю необходимо указать название задания, внештатный текст, список компонентов и U-шаблон. Список компонентов и U-шаблон могут быть созданы при помощи конструктора. Дополнительно предусмотрена возможность их создания из файлов исходного кода, которые нужно разметить при помощи специального языка. После трансляции полученное задание будет автоматически открыто в конструкторе. Страница конструктора приведена на рисунке 1. В столбце «Компоненты» преподаватель может добавлять и редактировать содержимое компонентов. Одна строка соответствует одному компоненту. Нумерация компонентов происходит автоматически. В столбце «U-шаблон» преподаватель описывает шаблоны для задания. Шаблоны нумеруются с нуля. Добавление шаблона осуществляется при помощи кнопки «+» рядом с текстом «U-шаблон». После этого в списке появляется следующий шаблон. Пустые шаблоны во время сохранения задания будут удалены. Для добавления элементов в шаблон используется кнопка «+», расположенная рядом с номером шаблона. После нажатия создается элемент с типом «Один компонент» и меткой «Без метки». Тип и метка могут быть изменены преподавателем при помощи соответствующих выпадающих списков. Для добавления компонентов в элемент преподаватель перетаскивает компонент из списка в специальное поле элемента. Если выбран тип «Один компонент», то добавление более чем одного компонента не допускается. Когда процесс редактирования закончен, преподаватель должен нажать на кнопку «Сохранить».

Созданное задание получает в системе идентификатор, который позволяет организовать выдачу задания обучающемуся с возможностью последующего автоматизированного контроля. Страница выдачи задания приведена на рисунке 2. Обучающемуся предоставляется название задания, текст, список компонентов и поле для ввода ответа. Обучающийся перетаскивает компоненты в поле для ответа, формируя программу. Можно удалить компоненты из поля ответа или при помощи перетаскивания изменить их порядок. Составив свой ответ, обучающийся нажимает на кнопку «Проверить». В системе сохраняется информация о попытке: ответ, оценка и сведения об анализе ответа. Обучающемуся доступна только оценка в виде вещественного числа в интервале  $[0; 1]$ . Пример результата контроля для обучающегося приведен на рисунке 3.

Преподаватель может посмотреть результаты работы контролирующего алгоритма. На рисунке 4 приведен пример списка результатов контроля, в ко-

\* Официальный сайт проекта. <https://fastapi.tiangolo.com>

\*\* Официальный сайт фреймворка. <https://www.djangoproject.com>

\*\*\* React. A JavaScript library for building user interfaces. <https://reactjs.org>

\*\*\*\* Redux. A predictable state container for JavaScript apps. <https://redux.js.org>

## Редактировать задание

Название \* Сумма всех элементов массива

---

Текст \* Из заданных ниже компонентов составьте программу, результатом работы которой будет вычисление суммы всех элементов массива m. Длина массива n. Результат суммирования должен быть записан в переменную S.

Компоненты + ≡

|                  |  |   |
|------------------|--|---|
| Текст            |  |   |
| S:=0;            |  | × |
| Текст            |  |   |
| for i:=1 to n do |  | × |
| Текст            |  |   |
| S:=S + m[i];     |  | × |
| Текст            |  |   |
| i:=1;            |  | × |
| Текст            |  |   |
| while i <= n do  |  | × |
| Текст            |  |   |
| begin            |  | × |
| Текст            |  |   |
| i:=i + 1;        |  | × |
| Текст            |  |   |
| inc(i);          |  | × |
| Текст            |  |   |
| end;             |  | × |
| Текст            |  |   |
| repeat           |  | × |
| Текст            |  |   |
| until i > n      |  | × |
| Текст            |  |   |
| i += 1;          |  | × |
| Текст            |  |   |
| i++;             |  | × |
| Текст            |  |   |
| repeat do        |  | × |
| Текст            |  |   |
| until i < n;     |  | × |

U-шаблон + ≡

**Шаблон 0** + ≡

|                  |                        |   |
|------------------|------------------------|---|
| Тип              | Метка                  |   |
| Один компонент   | Допускается отсутствие | × |
| S:=0;            |                        |   |
| Тип              | Метка                  |   |
| Один компонент   | Без метки              | × |
| for i:=1 to n do |                        |   |
| Тип              | Метка                  |   |
| Один компонент   | Рубежный               | × |
| S:=S + m[i];     |                        |   |

**Шаблон 1** + ≡

|                     |                        |   |
|---------------------|------------------------|---|
| Тип                 | Метка                  |   |
| Перестановка        | Без метки              |   |
| S:=0;               |                        |   |
| i:=1;               |                        |   |
| Тип                 | Метка                  |   |
| Один компонент      | Рубежный               | × |
| while i <= n do     |                        |   |
| Тип                 | Метка                  |   |
| Один компонент      | Без метки              | × |
| begin               |                        |   |
| Тип                 | Метка                  |   |
| Один компонент      | Допускается отсутствие | × |
| S:=S + m[i];        |                        |   |
| Тип                 | Метка                  |   |
| Один из компонентов | Без метки              | × |
| i:=i + 1;           |                        |   |
| inc(i);             |                        |   |
| Тип                 | Метка                  |   |
| Один компонент      | Без метки              | × |
| end;                |                        |   |

**Шаблон 2** + ≡

|                     |                        |   |
|---------------------|------------------------|---|
| Тип                 | Метка                  |   |
| Перестановка        | Без метки              |   |
| S:=0;               |                        |   |
| i:=1;               |                        |   |
| Тип                 | Метка                  |   |
| Один компонент      | Рубежный               | × |
| repeat              |                        |   |
| Тип                 | Метка                  |   |
| Один компонент      | Допускается отсутствие | × |
| S:=S + m[i];        |                        |   |
| Тип                 | Метка                  |   |
| Один из компонентов | Без метки              | × |
| i:=i + 1;           |                        |   |
| inc(i);             |                        |   |
| Тип                 | Метка                  |   |
| Один компонент      | Без метки              | × |
| until i > n         |                        |   |

СОХРАНИТЬ

ОЧИСТИТЬ

Рис. 1. Визуальный редактор задания

Fig. 1. Visual task editor

| Сумма всех элементов массива                                                                                                                                                                                                                                                                                                |                                 |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------|
| Из заданных ниже компонентов составьте программу, результатом работы которой будет вычисление суммы всех элементов массива m. Длина массива n. Результат суммирования должен быть записан в переменную S.                                                                                                                   |                                 |
| Компоненты                                                                                                                                                                                                                                                                                                                  | Ответ                           |
| <div>S:=0;</div> <div>for i:=1 to n do</div> <div>S:=S + m[i];</div> <div>i:=1;</div> <div>while i &lt;= n do</div> <div>begin</div> <div>i:=i + 1;</div> <div>inc(i);</div> <div>end;</div> <div>repeat</div> <div>until i &gt; n</div> <div>i += 1;</div> <div>i++;</div> <div>repeat do</div> <div>until i &lt; n;</div> | <div></div> <div>ОЧИСТИТЬ</div> |
| <div>ПРОВЕРИТЬ</div>                                                                                                                                                                                                                                                                                                        |                                 |

Рис. 2. Выдача задания

Fig. 2. Issuing of a task

| Сумма всех элементов массива                                                                                                                                                                                                                                                                                                |                                                                                                                                     |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| Из заданных ниже компонентов составьте программу, результатом работы которой будет вычисление суммы всех элементов массива m. Длина массива n. Результат суммирования должен быть записан в переменную S.                                                                                                                   |                                                                                                                                     |
| Компоненты                                                                                                                                                                                                                                                                                                                  | Ответ                                                                                                                               |
| <div>S:=0;</div> <div>for i:=1 to n do</div> <div>S:=S + m[i];</div> <div>i:=1;</div> <div>while i &lt;= n do</div> <div>begin</div> <div>i:=i + 1;</div> <div>inc(i);</div> <div>end;</div> <div>repeat</div> <div>until i &gt; n</div> <div>i += 1;</div> <div>i++;</div> <div>repeat do</div> <div>until i &lt; n;</div> | <div>S:=0; X</div> <div>begin X</div> <div>for i:=1 to n do X</div> <div>S:=S + m[i]; X</div> <div>end; X</div> <div>ОЧИСТИТЬ</div> |
| <div>ПРОВЕРИТЬ</div>                                                                                                                                                                                                                                                                                                        |                                                                                                                                     |
| Результат: 0.758                                                                                                                                                                                                                                                                                                            |                                                                                                                                     |

Рис. 3. Результат контроля для обучающегося

Fig. 3. An assessment results for a student

| Попытки                      | Search X         |                      |        |
|------------------------------|------------------|----------------------|--------|
| Задание                      | Обучающийся      | Дата                 | Оценка |
| Сумма всех элементов массива | Третьякова К. А. | 17.02.2022, 14:46:25 | 1      |
| Сумма всех элементов массива | Захарова В. С.   | 17.02.2022, 14:46:42 | 0.758  |
| Сумма всех элементов массива | Измайлова М. Б.  | 17.02.2022, 14:47:15 | 0.5    |
| Сумма всех элементов массива | Соколова О. С.   | 17.02.2022, 14:47:23 | 0.65   |
| Сумма всех элементов массива | Сергеев А. К.    | 17.02.2022, 14:48:10 | 1      |
| Сумма всех элементов массива | Пугачев М. Б.    | 17.02.2022, 14:49:01 | 0      |
| Сумма всех элементов массива | Кузнецов М. А.   | 17.02.2022, 14:49:13 | 1      |
| Сумма всех элементов массива | Степанова А. И.  | 17.02.2022, 14:49:33 | 0.7    |
| Сумма всех элементов массива | Белов А. Е.      | 17.02.2022, 14:50:03 | 0.125  |
| Сумма всех элементов массива | Ушакова Н. Д.    | 17.02.2022, 14:51:27 | 0      |

Рис. 4. Список результатов контроля

Fig. 4. List of assessments results

Задание: Сумма всех элементов массива

Обучающийся: Захарова В. С.

Время попытки: 17.02.2022, 14:46:42

Оценка: 0.758

**Ответ**

```

S:=0
begin
for i:=1 to n do
S:=S + m[i]
end;
```

**Анализ ответа**

Причина завершения анализа: Закончился шаблон

**Таблица ошибок**

| Элемент                                                                 | Ошибка                                    | Лишние компоненты | Отсутствующие компоненты |
|-------------------------------------------------------------------------|-------------------------------------------|-------------------|--------------------------|
| Тип: Один компонент<br>Метка: Без метки<br>Компоненты: for i:=1 to n do | Присутствует с лишними компонентами       | begin             |                          |
|                                                                         | Лишний компонент после завершения анализа | end               |                          |

Рис. 5. Детальные сведения об ответе студента

Fig. 5. Detailed information about the student's response

тором указывается задание, ФИО обучающегося, дата ответа и оценка.

Нажав на какую-либо строчку ответа, преподаватель перейдет к детальным сведениям. Детальные сведения включают в себя данные попытки, ответ обучающегося и мнение эксперта. Пример страницы с детальными сведениями приведен на рисунке 5.

Рассмотрим, какую информацию преподаватель может получить из детальных сведений (рис. 5). Поскольку в анализе ответа указаны лишние компоненты и в таблице ошибок указан лишний компонент «begin», можно сделать вывод, что обучающийся использовал операторные скобки «begin» и «end» там, где они не требуются для корректной программы. Эксперт оценил программу обучающегося как частично правильную и выставил оценку 0,758 (приблизительно 76 %).

**4. Заключение**

Авторами предложен новый тип заданий для автоматизированного контроля знаний и навыков по программированию, в которых обучающийся из предложенных компонентов (строк программного кода) составляет ответ (программу). Для практического применения таких заданий авторами раз-

работан комплекс программ для контроля знаний (система контроля знаний), состоящий из двух подсистем: «Обработка» и «Демонстрация». Подсистема «Демонстрация» дает минимально необходимый набор возможностей для применения нового типа заданий в учебном процессе. Также она служит образцом, который может быть использован при разработке расширений для других систем. Подсистема «Обработка» позволяет интегрировать новый тип заданий по программированию в различные системы автоматизированного контроля и LMS, если те предоставляют возможность для расширения.

Проведены тестовые испытания системы, в ходе которых успешно созданы задания нового типа для задач дисциплины «Основы программирования». Применение системы автоматизированного контроля позволяет обучающимся осуществлять самоконтроль. Сведения об автоматизированной оценке ответа обучающегося, предоставляемые контролирующим алгоритмом, помогают преподавателю анализировать ошибки и формировать рекомендации для обучающихся.

**Список источников / References**

1. Шумихина М. С., Яшина И. А. СДО Moodle в обучении программированию. Актуальные проблемы авиации

и космонавтики. Сборник материалов XIV Международной научно-практической конференции, посвященной Дню космонавтики. 2018;3(4(14)):803–804. EDN: YWPLUL

[Shumikhina M. S., Yashina I. A. LMS Moodle in teaching programming. *Actual Problems of Aviation and Cosmonautics. Proc. XIV Int. Scientific and Practical Conf. Dedicated to the Day of Cosmonautics*. 2018;3(4(14)):803–804. (In Russian.) EDN: YWPLUL]

2. Барабанова П. С., Лямкин Р. А., Макарова И. С. Разработка электронного курса по программированию для самоподготовки студентов технического вуза. *Бюллетень лаборатории математического, естественнонаучного образования и информатизации: материалы Международной научно-практической конференции «Математическое, естественнонаучное образование и информатизация»*. М.: ГАОУ ВО МГПУ; 2015;6:62–66. EDN: VHJSIB

[Barabanova P. S., Lyamkin R. A., Makarova I. S. The development of the electronic course on programming for independent learning of technical university students. *Bulletin of the Laboratory of Mathematical, Natural Science Education and Informatization: Proc. Int. Scientific and Practical Conf. "Mathematical, Natural Science Education and Informatization"*. Moscow, MCU; 2015;6:62–66. (In Russian.) EDN: VHJSIB

3. Bihás M. Version Control System in CS1 course: Practical experience. *2013 IEEE 11th Int. Conf. on Emerging eLearning Technologies and Applications (ICETA)*. 2013;23–28. DOI: 10.1109/ICETA.2013.6674398

4. Баженова И. Ю. Особенности обучения языкам программирования при дистанционной форме образования. *Современные информационные технологии и ИТ-образование*. 2014;(10):190–196. EDN: TJTXXF

[Bazhenova I. Yu. Peculiarities of teaching a programming language in the remote form of education. *Modern Information Technologies and IT-education*. 2014;(10):190–196 (In Russian.) EDN: TJTXXF]

5. Шагбазян Д. В., Штанюк А. А. Методы контроля знаний студентов вузов по программированию. *Поколение будущего: взгляд молодых ученых. Сборник научных статей 4-й международной молодежной научной конференции*. Курск: ЗАО «Университетская книга»; 2016;2:119–121. EDN: XDXIKH

[Shagbazyan D. V., Shtanyuk A. A. Methods for knowledge assessment of university students in programming. *Generation of the Future: the View of Young Scientists. Collection of Scientific Articles of the 4th Int. Youth Sci. Conf. Kursk, CJCS "University book"*; 2016;2:119–121. (In Russian.) EDN: XDXIKH]

6. Магомедов А. М., Лавренченко С. А., Ибрагимова З. И. Алгоритм автоматизации создания тестов. *Вестник Дагестанского государственного университета. Серия 1: Естественные науки*. 2019;34(2):63–71. EDN: CHLRYE. DOI: 10.21779/2542-0321-2019-34-2-63-71

[Magomedov A. M., Lawrenchenko S. A., Ibragimova Z. I. Algorithm for automating test creation. *Herald of Dagestan State University. Series 1. Natural Sciences*. 2019;34(2):63–71. (In Russian.) EDN: CHLRYE. DOI: 10.21779/2542-0321-2019-34-2-63-71]

7. Голанова А. В., Голикова Е. И. Организация итогового контроля знаний по дисциплине «Программирование» будущих бакалавров педагогического образования. *XXII Царскосельские чтения. Материалы международной научной конференции*. Санкт-Петербург: Ленинградский государственный университет имени А. С. Пушкина; 2018;3:75–79. EDN: YXOZUS

[Golanova A. V., Golikova E. I. Organization of the final control of knowledge on the discipline "Programming" of future bachelors of teacher education. *XXII Tsarskoye Selo Readings. Proc. Int. Sci. Conf.* Saint Petersburg, Pushkin Leningrad State University; 2018;3:75–79. (In Russian.) EDN: YXOZUS]

8. Перязева Ю. В. Возможности автоматической проверки заданий в LMS Moodle. *Современные информационные технологии и ИТ-образование*. 2019;15(4):876–885. EDN: BECCUB. DOI: 10.25559/SITITO.15.201904.876-885

[Peryazeva J. V. Extending moodle functionality with computer-assisted assessment. *Modern Information Technologies and IT-education*. 2019;15(4):876–885. (In Russian.) EDN: BECCUB. DOI: 10.25559/SITITO.15.201904.876-885]

9. Ramos J., Trenas M. A., Gutiérrez E., Romero S. E-assessment of Matlab assignments in Moodle: Application to an introductory programming course for engineers. *Computer Applications in Engineering Education*. 2013;21(4):728–736. DOI: 10.1002/cae.20520

10. Карманова Е. В. Автоматизированный контроль при обучении программированию на Python с использованием плагина CodeRunner LMS Moodle. *Наука. Информатизация. Технологии. Образование. Материалы XIV международной научно-практической конференции*. Екатеринбург: Российский государственный профессионально-педагогический университет; 2021:102–108. EDN: TEDZUF

[Karmanova E. V. Automated control for teaching programming on the Python using the plugin CodeRunner in LMS Moodle. *Science. Informatization. Technologies. Education. Proc. XIV Int. Scientific and Practical Conf.* Yekaterinburg, Russian State Vocational Pedagogical University; 2021:102–108. (In Russian.) EDN: TEDZUF]

11. Guo P. J. Online python tutor: Embeddable web-based program visualization for cs education. *Proc. 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13)*. New York (United States), Association for Computing Machinery; 2013:579–584. DOI: 10.1145/2445196.2445368

12. Gulwani S., Radiček I., Zuleger F. Automated clustering and program repair for introductory programming assignments. *Proc. 39th ACM SIGPLAN Conf. on Programming Language Design and Implementation (PLDI 2018)*. New York (United States), Association for Computing Machinery; 2018:465–480. DOI: 10.1145/3192366.3192387

13. Suzuki R., Soares G., Head A., Glassman E., Reis R., Mongiovi M., D'Antoni L., Hartmann B. TraceDiff: Debugging unexpected code behavior using trace divergences. *Proc. 2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. Raleigh, NC, IEEE; 2017:107–115. DOI: 10.1109/VLHCC.2017.8103457

14. Reis R., Soares G., Mongiovi M., Andrade W. L. Evaluating feedback tools in introductory programming classes. *Proc. 2019 IEEE Frontiers in Education Conf. (FIE)*. Covington, KY, USA, IEEE; 2019:1–7. DOI: 10.1109/FIE43999.2019.9028418

15. Шевелев М. Ю., Вишнякова Л. А., Шевелев Ю. П. Контроль знаний в компьютерном обучении. Неантропоморфный подход. Саарбрюккен: LAP LAMBERT Academic Publishing; 2014. 404 с.

[Shevelev M. Yu., Vishnyakova L. A., Shevelev Yu. P. Knowledge assessment in e-learning. Non-anthropomorphic approach. Saarbrücken, LAP LAMBERT Academic Publishing; 2014. 404 p. (In Russian.)]

16. Жуков И. А., Костюк Ю. Л. Модель представления многовариантных заданий для автоматизированного контроля знаний по программированию. *Вестник Томского государственного университета. Управление, вычислительная техника и информатика*. 2020;(53):110–117. EDN: TERGOW. DOI: 10.17223/19988605/53/11

[Zhukov I. A., Kostyuk Yu. L. Model of representation of multivariate tasks for automated control of programming knowledge. *Tomsk State University Journal of Control and Computer Science*. 2020;(53):110–117. (In Russian.) EDN: TERGOW. DOI: 10.17223/19988605/53/11]

17. Жуков И. А., Костюк Ю. Л. Расширенный алгоритм контроля знаний для модели представления многовариантных заданий. *Вестник Томского государственного*

университета. *Управление, вычислительная техника и информатика*. 2021;(57):101–109. EDN: AOWESF. DOI: 10.17223/19988605/57/11

[Zhukov I. A., Kostyuk Yu. L. An extended assessment algorithm for multivariate task presentation model. *Tomsk State University Journal of Control and Computer Science*. 2021;(57):101–109. (In Russian.) EDN: AOWESF. DOI: 10.17223/19988605/57/11]

#### **Информация об авторах**

**Жуков Игорь Андреевич**, аспирант кафедры теоретических основ информатики, Институт прикладной математики и компьютерных наук, Национальный исследовательский Томский государственный университет, г. Томск, Россия; *ORCID*: <https://orcid.org/0000-0001-7995-3688>; *e-mail*: IgZhukov963@yandex.ru

**Костюк Юрий Леонидович**, доктор тех. наук, профессор, профессор кафедры теоретических основ информатики, Институт прикладной математики и компьютерных наук,

Национальный исследовательский Томский государственный университет, г. Томск, Россия; *ORCID*: <https://orcid.org/0009-0001-3730-8071>; *e-mail*: kostyuk.yu48@mail.ru

#### **Information about the authors**

**Igor A. Zhukov**, a postgraduate student at the Computer Science Theory Department, Institute of Applied Mathematics and Computer Science, Tomsk State University, Tomsk, Russia; *ORCID*: <https://orcid.org/0000-0001-7995-3688>; *e-mail*: IgZhukov963@yandex.ru

**Yuriy L. Kostyuk**, Doctor of Sciences (Engineering), Professor, Professor at the Computer Science Theory Department, Institute of Applied Mathematics and Computer Science, Tomsk State University, Tomsk, Russia; *ORCID*: <https://orcid.org/0009-0001-3730-8071>; *e-mail*: kostyuk.yu48@mail.ru

**Поступила в редакцию / Received:** 20.09.2022.

**Поступила после рецензирования / Revised:** 11.11.2022.

**Принята к печати / Accepted:** 15.11.2022.