ДОКЛАДЫ РОССИЙСКОЙ АКАДЕМИИ НАУК. МАТЕМАТИКА, ИНФОРМАТИКА, ПРОЦЕССЫ УПРАВЛЕНИЯ, 2023, том 511, с. 13–20

МАТЕМАТИЧЕСКОЕ ОБРАЗОВАНИЕ ЦИФРОВОГО ВЕКА

УДК 004.434

БЕЗОШИБОЧНЫЙ ДВУМЕРНЫЙ ПИКТОГРАММНЫЙ СИНТАКСИС В УЧЕБНОЙ СРЕДЕ ПРОГРАММИРОВАНИЯ ДЛЯ ДОШКОЛЬНИКОВ

© 2023 г. А. Г. Кушниренко[1,*], А. Г. Леонов[1,], С. А. Поликарпов[2,***]**

Представлено академиком РАН А.Л. Семеновым
Поступило 20.11.2022 г.
После доработки 28.11.2022 г.
Принято к публикации 28.11.2022 г.

При освоении азов программирования дошкольниками серьезные трудности создает необходимость диагностики и исправления синтаксических ошибок. При традиционной методике "экранного" редактирования программы этих трудностей можно избежать, блокируя действия ребенка, приводящие к синтаксическим нарушениям. Сегодня набирает популярность методика составления программ из материальных объектов (tangible objects) с нанесенными на них пиктограммами команд. При использовании такой методики блокировка ошибочных действий пользователя невозможна. В этой ситуации авторы предлагают оградить ребенка от синтаксических ошибок, постулировав двумерность программы и определив с помощью отступов синтаксис и семантику пиктограммного языка программирования для начинающих так, чтобы любое размещение пиктограмм в клетках двумерной таблицы давало синтаксически корректную и выполнимую программу. Этот подход реализован и опробован в отечественной учебной среде "ПиктоМир" пиктограммного программирования для дошкольников.

Ключевые слова: программа, пиктограмма, бестекстовая среда программирования, безэкранное составление программ, ПиктоМир, дошкольник, робот, структурное программирование

DOI: 10.31857/S2686954323700169, **EDN:** IQZKNJ

1. ВВЕДЕНИЕ

В последние годы во всем мире ведутся работы по понижению возраста ознакомления детей с элементами программирования и развитию у детей алгоритмического (вычислительного) мышления [1–10]. Ведутся подобные работы и в России. В статье [11] описана система научных (по Выготскому) понятий программирования, предназначенных для освоения дошкольниками, а также методика ознакомления детей с этими понятиями в годовом систематическом курсе программирования для дошкольников подготовительных групп детских садов [12], который в последние три года успешно прошли тысячи дошкольников города Сургута. В курсе используется среда пиктограммного программирования ПиктоМир [13] разработки ФГУ ФНЦ НИИСИ РАН.

ПиктоМир позволяет детям составлять из пиктограмм программы управления движением реальных роботов-игрушек по полу и виртуальных роботов на экране планшета (рис. 1).

Программы для управления этими роботами могут составляться как из картинок на экране планшета, так и в реальном мире из картинок, нарисованных на материальных объектах.

Например, в репертуаре роботов ПиктоМира есть виртуальный робот *Вертун*, перемещающийся по клетчатой платформе и умеющий выполнять команды-приказы: ПОВЕРНУТЬСЯ НАЛЕВО, ПОВЕРНУТЬСЯ НАПРАВО, ВПЕРЕД, ЗАКРАСИТЬ (см. рис. 2). В тексте статьи мы будем обозначать эти 4 пиктограммы как **(L) (R) (F) (P)**.

2. ВОЗМОЖНОСТИ СРЕДЫ ПИКТОГРАММНОГО ПРОГРАММИРОВАНИЯ ПИКТОМИР

Использование пиктограммного стиля программирования для работы с детьми возраста 5–7 лет и использование материальных объектов для составления программ или практикумов по информатике и математике в начальной школе сегодня

[1] *Федеральное государственное учреждение "Федеральный научный центр Научно-исследовательский институт системных исследований Российской академии наук", Москва, Россия*

[2] *Математический институт им. В.А. Стеклова Российской академии наук, Москва, Россия*

**E-mail: agk_@mail.ru*

***E-mail: dr.l@vip.niisi.ru*

****E-mail: polik@mi-ras.ru*

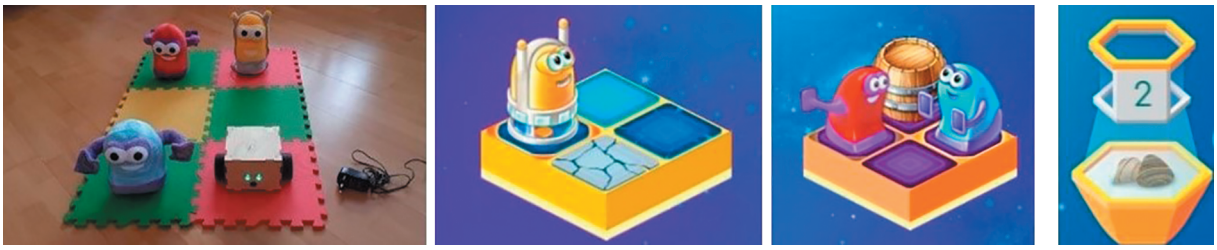

Рис. 1. Реальные и виртуальные роботы среды ПиктоМир.

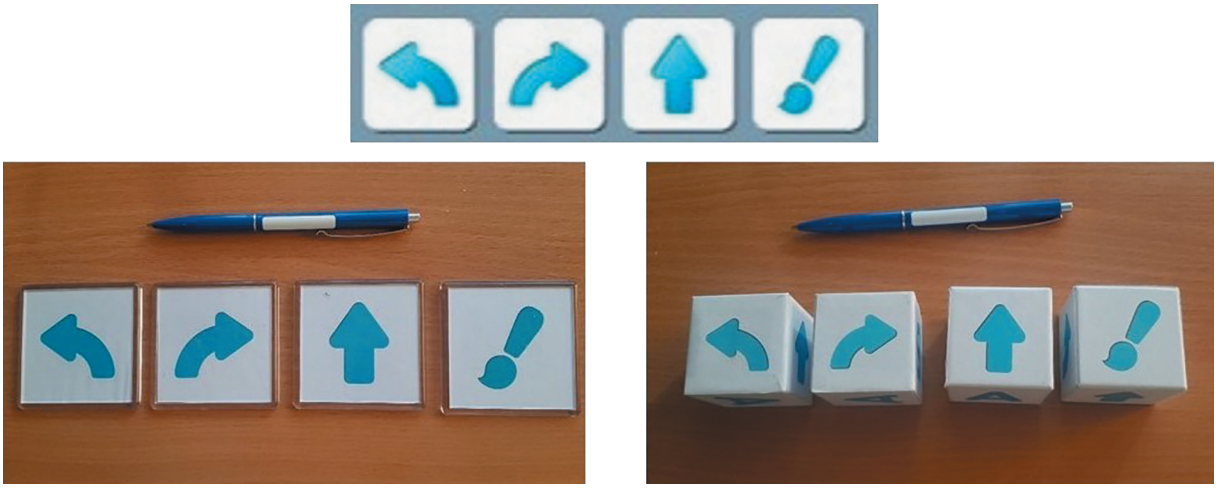

Рис. 2. Пиктограммы команд-приказов Вертуна, размещенные на экране планшета, на магнитных карточках и на деревянных кубиках.

достаточно распространены [14–20]. Наша методика отличается тем, что обеспечивает детям возраста 6–7 лет возможность составления в годовом цикле занятий более сотни программ достаточно сложной структуры с целью освоения всех конструкций структурного программирования, включая конструкцию подпрограммы без параметров. Язык *Пикто* [21], используемый в системе ПиктоМир, позволяет составлять в пиктограммной форме программы управления реальными роботами-игрушками и виртуальными экранными роботами. Последние могут быть подвижными, подобно *Вертуну*, или неподвижными, подобно выполняющему роль счетчика *Волшебному Кувшину* с камнями, изображенному на рис. 1 справа.

Эти роботы способны выполнять императивные приказы (команды-приказы), а также способны отвечать на запросы, возвращая логическое или целочисленное значение (команды-вопросы), что позволяет включить в язык Пикто управляющие конструкции **цикл пока** <условие> **выполнять** и **если** <условие> **выполнять**, где <условие> – это результат ответа некоторого робота на некоторую команду-вопрос, например, на команду-вопрос *Вертуна* МОЖНО СДЕЛАТЬ ШАГ? или команду-вопрос *Волшебного кувшина* КУВШИН ПУСТ? Язык поддерживает также управляющую конструкцию **цикл** <повторитель> **раз**, где <повторитель> может быть задан пиктограммой команды-вопроса некоторого робота, например, командой-вопросом СКОЛЬКО КАМНЕЙ В КУВШИНЕ? робота *Волшебный Кувшин*. Повторитель также может быть задан явно, пиктограммой с изображением грани игрального кубика (рис. 3). В тексте статьи мы будем обозначать эти пиктограммы символами **(1) (2) (3) (4) (5) (6).**

Язык Пикто предусматривает также вложение управляющих конструкций и разбиение программы на секции с предопределенными однобуквенными именами (рис. 4). В тексте статьи мы будем обозначать эти пиктограммы символами **(А) (Б) (В) (Г) (Д).**

3. ПРОБЛЕМА НЕДОПУЩЕНИЯ СИНТАКСИЧЕСКИХ ОШИБОК ПРИ СОСТАВЛЕНИИ ПРОГРАММ ИЗ МАТЕРИАЛЬНЫХ ОБЪЕКТОВ

Среда ПиктоМир, наряду с традиционным подходом, при котором дети составляют программу на экране планшете, поддерживает подход, при котором дети составляют программы из

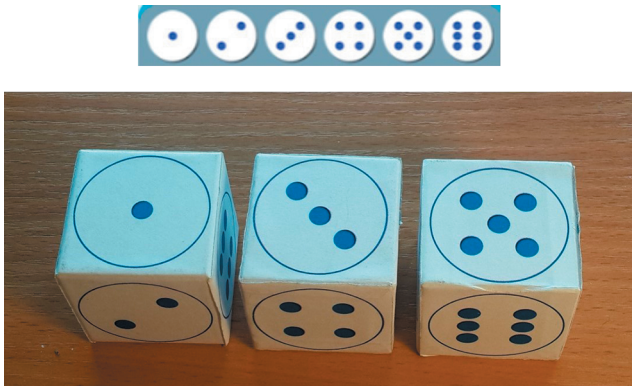

Рис. 3. Пиктограммы повторителей на экране планшета и на гранях кубиков.

Рис. 4. Пиктограммы однобуквенных имен секций на экране планшета и на гранях кубиков.

тех или иных материальных объектов, на грани которых нанесены пиктограммы команд и управляющих конструкций (см. рис. 2–4). Ученик вручную, механически, размещает эти несущие пиктограммы предметы в клетках воображаемой двумерной таблицы. Чтобы помочь ребенку расставлять кубики ровно, таблицу можно нарисовать на картоне или на магнитной доске.

Исследования последних лет показывают [22], что азы программирования успешнее осваиваются в инструментальных средах, ограждающих ребенка от возможности допустить синтаксическую ошибку путем недопущения действий, приводящих к таким ошибкам.

При составлении программ путем манипуляций с материальными предметами, контроль за работой ребенка и недопущение каких-либо действий ребенка практически невозможны. Потому нужно искать другой метод недопущения синтаксических ошибок. Ниже предлагается такой метод, основанный на следующей простой идее:

определим язык пиктограммного программирования так, чтобы любые программы, составленные из заданного набора пиктограмм с командами роботов и заголовками управляющих конструкций, были синтаксически правильны и выполнимы.

Для реализации этой идеи мы, следуя примеру языка Питон, “наделим” программу двумерной структурой, а для задания блочной структуры используем отступы.

Таким образом, мы требуем, чтобы составленные из пиктограмм программы были *a priori* организованы в виде двумерной таблицы. При материальном способе составления программ нарушение этого постулата описывается понятной детям фразой *пиктограммы расставлены неровно*, а при составлении программ на сенсорном экране невозможность подобного нарушения может быть обеспечена автоматически. Таким образом, при составлении программ из материальных объектов единственно возможным синтаксическим (или, скорее, 2D-синтаксическим) нарушением оказывается *неровность*, нарушение двумерной структуры. Эта *неровность* легко осознается и исправляется детьми, начиная с возраста 3–4 года, и потому в дальнейшем мы предполагаем, что *пиктограммы расставлены ровно*, т.е. программа наделена двумерной структурой.

При любом способе составления программы необходимо предварительно объяснить детям, как компьютер будет программу выполнять и какие синтаксические правила составления программы нужно будет соблюдать, чтобы компьютер понял программу. Чем проще будут синтаксические правила, тем легче будет детям их понимать и соблюдать. Идеалом в двумерном пиктограммном программировании является полная отмена каких-либо синтаксических ограничений, кроме нарушения двумерности.

В настоящей статье этот идеал реализован и теоретически, и практически. А именно:

– сформулированы правила интерпретации любого расположения пиктограмм в клетках двумерной таблицы как синтаксически правильной программы, которая может быть выполнена;

– эти правила реализованы в языке Пикто системы ПиктоМир.

Важно, что эти правила методически оказались достаточно естественными и эффективными при освоении дошкольниками и младшеклассниками азов программирования. Хотя использование правил синтаксически неограниченного пиктограммного программирования избавляет от необходимости объяснять детям синтаксические ограничения, оно не отменяет необходимости объяснять детям правила разбиения программы на вложенные конструкции, которые и определяют семантику программы, правила ее выполнения. Выигрыш в смещении акцента с синтаксиса на семантику в том, что в отличие от синтаксиса, семантика наглядна и ее освоение возможно на практике, в эксперименте. Последовательно нажимая кнопку пошагового выполнения программы, ребенок видит наглядно, какую пиктограмму

программы компьютер выполняет и к чему приводит это выполнение. Наглядность особенно важна для ускорения первых шагов в программировании. По нашей методике эти первые шаги организуются следующим образом. Составив программу из материальных объектов, скажем из кубиков на столе, ребенок показывает программу компьютеру (планшету). Компьютер распознает полученное изображение, превращает его в программу в своей памяти и выполняет эту программу. Как бы ребенок не расположил кубики в клетках двумерной таблицы, это расположение распознается компьютером как некоторая правильная программа, и ребенок получает возможность увидеть процесс выполнения этой программы, увидеть, отвечает ли поведение робота при выполнении этой программы замыслам, которые ребенок пытался реализовать при ее составлении. Если поведение не соответствует первоначальным задумкам, ребенок изменяет программу и имеет возможность посмотреть, как выполняется измененная программа. Таким образом, в любой ситуации ребенок может экспериментировать и не попадает в ситуации, в которых он должен вспоминать правила составления программ и пытаться умозрительно понять, какие из этих правил он нарушил.

4. РЕАЛИЗАЦИЯ ИДЕИ ПИКТОГРАММНОГО ПРОГРАММИРОВАНИЯ БЕЗ СИНТАКСИЧЕСКИХ ОГРАНИЧЕНИЙ В СРЕДЕ ПИКТОМИР

Переменных и констант, а стало быть, арифметических или логических выражений в ПиктоМире нет. Значит, об ошибках в выражениях можно не беспокоиться. Синтаксически недопустимые фрагменты программы в ПиктоМире потенциально могли бы возникать в двух ситуациях:

– при разбиении программы на секции-подпрограммы.

– при вложении управляющих конструкций.

Первую ситуацию мы разрешаем следующим образом: игнорируем возможные несогласованности, позволяя им так или иначе проявиться при выполнении программы. А именно,

– вызов подпрограммы с неопределенным именем считаем допустимым и выполняем этот вызов как однотактовую команду *нет операции*;

– код, который можно было бы интерпретировать как повторное определение имени подпрограммы, поскольку выше по программе данное имя уже задано, и весь код, следующий за ним, считаем продолжением кода предыдущей секции.

Правильность же структуры вложенных управляющих конструкций, мы, вдохновленные примером языка Питон, обеспечиваем тем, что задаем блочную структуру неявно, с помощью отступов. В языке Пикто есть управляющие конструкции повторения и ветвления (без ветви **else**) и разрешается их вложение. Значит, внутри одной секции программы должен быть поддержан баланс операторных скобок. Явное нарушение баланса операторных скобок в языке Пикто невозможно, так как в языке Пикто есть пиктограммы, начинающие управляющую конструкцию, но нет пиктограмм, завершающих конструкцию. Информация о том, где завершается управляющая конструкция, задается в языке Пикто неявно, путем размещения начальных отступов в строках программы. И правила эти установлены так, что каковы бы ни были отступы в различных строках программы, можно однозначно определить, где завершается любая управляющая конструкция, не нарушая при этом баланса управляющих скобок, и, значит, можно однозначно определить, как следует выполнять программу. Рассмотрим пример программы, состоящей из одной неименованной секции.

На рис. 5 конструкция “цикл 3 раза” начинается пиктограммой **(3)**, размещенной в первой позиции первой строки с отступом 0. Тело этой конструкции начинается в этой же строке, начиная со второй позиции. **Признаком включения в тело конструкции очередной строки мы всегда считаем отступ, больший отступа начала конструкции** (см. следующий раздел). Вторая строка начинается с такого отступа, поэтому вся вторая строка входит в тело цикла. В третьей строке отступ нулевой, это значит, что тело цикла закончилось во второй строке и команда **(P)**, расположенная в первой позиции третьей строки, в тело цикла не входит и образует новый блок, который будет выполнен после завершения цикла.

Аналогично задаются вложения управляющих конструкций.

5. НАБРОСОК АЛГОРИТМА КОМПИЛЯЦИЯ ОДНОЙ ПИКТО-СЕКЦИИ

Мы рассматриваем задачу компиляции одной секции программы на языке Пикто, представляющей собой прямоугольную таблицу T фиксированного размера, заполненную пиктограммами из алфавита

$$\{(\,)\} \cup \boldsymbol{A} \cup \boldsymbol{B},$$

где **( )** – пиктограмма *фон*, обозначающая незаполненную (пустую) клетку таблицы T,

$\boldsymbol{A} = \{\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{m-1}\}$ – множество пиктограмм действий (**a**ction), т.е. команд-приказов роботов и вызовов секций;

$\boldsymbol{B} = \{\mathbf{b}_0, \mathbf{b}_1, \ldots \mathbf{b}_{n-1}\}$ – множество пиктограммы начал (**b**egin) управляющих конструкций. Клетки, заполненные элементами множеств $\boldsymbol{A}$ и $\boldsymbol{B}$, называем *непустыми*, строку таблицы называем *непустой*, если в ней есть хотя бы одна непустая

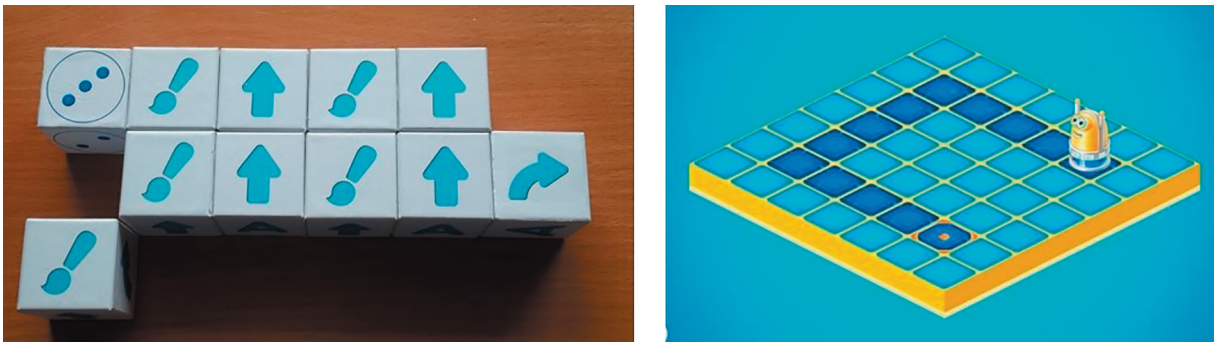

Рис. 5. Программа закрашивания заглавной русской буквы П и результат выполнения этой программы роботом Вертуном. Начальная позиция Вертуна отмечена точкой.

клетка. *Отступом* непустой строки называем количество пустых (фоновых) клеток, предшествующих первой непустой клетке. *Отступ* строки, состоящей только из пустых клеток, считаем равным 0. *Отступом* любой клетки строки называем число клеток строки, предшествующих данной клетке. Множества ***A*** и ***B*** зависят от набора роботов, которыми должна будет управлять программа, и меняются от задачи к задаче. Механизм формирования этих множеств нам не важен, к моменту компиляции секции эти множества предполагаются сформированными.

Например, при компиляции неименованной секции программы на рис. 5 предполагается, что множество ***A*** состоит из 9 пиктограмм

$$(\mathbf{L})(\mathbf{R})(\mathbf{F})(\mathbf{P})(\mathbf{A})(\mathbf{Б})(\mathbf{В})(\mathbf{Г})(\mathbf{Д})(\mathbf{Е}),$$

т.е. в компилируемой секции используются только вызовы команд-действий робота Вертуна и вызовы секций с однобуквенными именами, а множество ***B*** состоит из 6 пиктограмм

$$(\mathbf{1})(\mathbf{2})(\mathbf{3})(\mathbf{4})(\mathbf{5})(\mathbf{6}),$$

т.е. в компилируемой секции в качестве управляющих конструкций используются только циклы с явно заданным числом повторений от 1 до 6.

Наша задача – сгенерировать по таблице T последовательность S элементов в алфавите

$$\{\mathbf{a}_0, \mathbf{a}_1, \ldots, \mathbf{a}_{m-1}\} \cup \{\mathbf{b}_0, \mathbf{b}_1, \ldots \mathbf{b}_{n-1}\} \cup \{(\mathbf{e})\}$$

Здесь **(e)** – отсутствующая в исходной программе явная универсальная закрывающая операторная скобка, нужное количество экземпляров которой должно быть размещено в скомпилированной программе с целью однозначного задания блочной структуры этой программы. После компиляции каждой секции многосекционная программа, включающая вызовы (быть может, рекурсивные) подпрограмм без параметров и вызовы команд-действий и команд-вопросов внешних исполнителей, может быть выполнена по общепринятым правилам.

Последовательность S строится так. Создаем пустую последовательность S и пустой стек неотрицательных целых чисел, в котором будут храниться отступы заголовков незакрытых управляющих конструкций и начинаем построчно сканировать таблицу T. При этом

– каждую встреченную пиктограмму **( )** игнорируем;

– каждую встреченную пиктограмму из ***A*** добавляем в конец S;

– каждую встреченную пиктограмму из ***B*** добавляем в конец S, открывая, тем самым, в программе S новую операторную скобку и, кроме того, записываем в стек отступ этой пиктограммы от начала строки;

– при переходе на новую строку вычисляем ее отступ I (**I**ndent), и, пока на вершине стека обнаруживается отступ, больший или равный I, изымаем этот отступ из стека и записываем в S универсальную закрывающую скобку **(e)**, закрывая, тем самым, одну из ранее открытых операторных скобок;

– окончание секции обрабатываем как строку с отступом 0.

Пример 1. Зададим единственную секцию программы на рис. 5 таблицей 1

Таблица 1

(3)	**(P)**	**(F)**	**(P)**	**(F)**	**( )**
( )	**(P)**	**(F)**	**(P)**	**(F)**	**(R)**
(P)	**( )**	**( )**	**( )**	**( )**	**( )**

Таблица 2

(2)	**( )**	**( )**	**( )**
( )	**(3)**	**( )**	**( )**
( )	**( )**	**(A)**	**(L)**
( )	**(P)**	**(L)**	**( )**

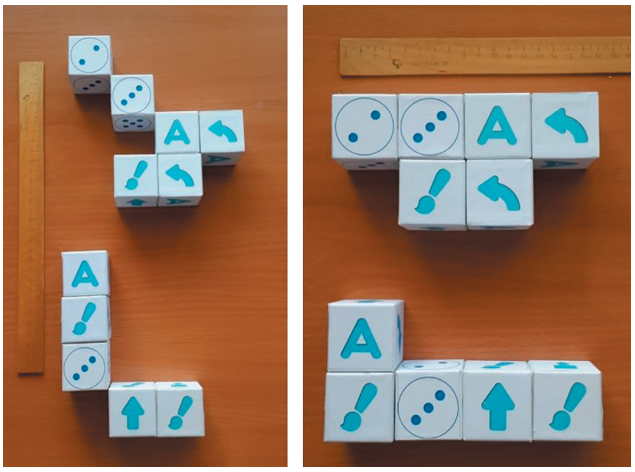

Рис. 6. Задание вложенных циклов отступами. Полная и компактная форма программы закрашивания буквы Е.

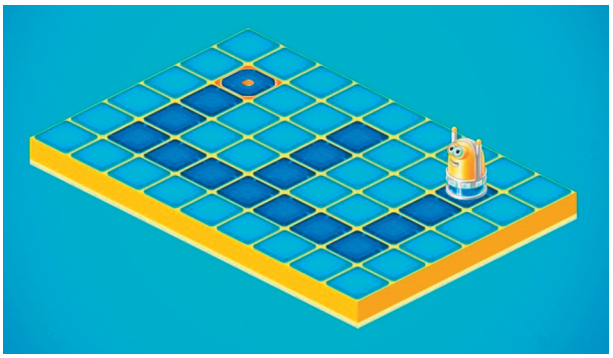

Рис. 7. Результат работы программы закрашивания буквы Е. Начальная позиция робота отмечена точкой.

Результатом компиляции этой секции будет последовательность

$$(3)(P)\,(F)\,(P)\,(F)\,(P)\,(F)\,(P)\,(F)(R)(e)(P)$$

в которой открывающая скобка **(3)** в первой позиции соответствует закрывающей скобке **(e)** в предпоследней позиции.

Пример 2. Зададим неименованную секцию программы на рис. 6 (полная форма) таблицей 2.

Внешний цикл неименованной секции начинается пиктограммой **(2)**, имеющей нулевой отступ. Следующие три строки имеют ненулевой отступ и потому мы считаем их входящими в тело внешнего цикла. Во второй строке этого тела с отступом 1 расположен заголовок **(3)** внутреннего цикла. По нашему алгоритму в тело этого внутреннего цикла должна входить третья строка с отступом 2, но не должна входить четвертая строка, которая находится в теле внешнего цикла и будет выполняться при каждом завершении внутреннего цикла. Результатом компиляции этой секции будет последовательность

$$(2)(3)(A)(L)(e)(P)(L)(e)$$

в которой внутренний цикл **(3) (A) (L) (e)** вложен во внешний цикл **(2) … (e)**.

БЛАГОДАРНОСТИ

Авторы благодарны Н.О. Бесшапошникову, К.А. Мащенко и А.А. Малому за полезные обсуждения.

ИСТОЧНИКИ ФИНАНСИРОВАНИЯ

Работа выполнена совместно сотрудником Математического института им. В.А. Стеклова РАН С.А. Поликарповым (анализ мирового опыта, постановка задачи) в рамках научного проекта № 19-29-14152 мк “Фундаментальные основы формирования математической грамотности для цифрового общества на начальном уровне образования” и сотрудниками

ФГУ ФНЦ НИИСИ РАН (постановка задачи, разработка и реализация алгоритмов) в рамках госзадания по теме FNEF-2022-0010.

СПИСОК ЛИТЕРАТУРЫ

1. *Ershov A.P.* Programming, the second literacy // Computer and Education. Proc. IFIP TC 3 3rd World Conf. on Computer Education. Amsterdam: North-Holland Publishing Company. 1981. Part 1. P. 1–17.
2. *Пейперт С.* Переворот в сознании: Дети, компьютеры и плодотворные идеи. М.: Педагогика, 1989. 224 с. (перевод с англ. Papert S. Mindstorms: children, computers, and powerful ideas. NYC: Basic Books, 1980. 242 p.)
3. *Resnick M., et al.* Scratch: Programming for al. // Commun. ACM 52, 11 (Nov. 2009). P. 60–67. https://doi.org/10.1145/1592761.1592779
4. *Flannery L.P., Kazakoff E.R., Bonta et al.* Designing ScratchJr: Support for early childhood learning through computer programming // In Proceedings of the 12th International Conference on Interaction Design and Children (IDC '13). ACM, New York, NY, USA, 2013. P. 1–10. https://doi.org/10.1145/2485760.2485785
5. *Калаш И.* Возможности информационных и коммуникационных технологий в дошкольном образовании. Аналитический обзор. Институт Юнеско по информационным технологиям в образовании, 2010. 177 с. https://iite.unesco.org/pics/publications/ru/files/3214673.pdf
6. *Richtel M.* Reading, writing, arithmetic, and lately, coding // The New York Times. May 10, 2014. https://www.nytimes.com/2014/05/11/us/reading-writing-arithmeticand-lately-coding.html
7. *Семенов А.Л.* Концептуальные проблемы информатики, алгоритмики и программирования в школе // Вестник кибернетики. 2016. № 2 (22). С. 11–15.
8. *Семенов А.Л.* Цели общего образования в цифровом мире // Информатизация образования и методика электронного обучения: Материалы III Междунар. конф.: в 2 ч. Красноярск: СФУ, 2019. Ч. 2. С. 383– 388.
9. *Бетелин В.Б., Кушниренко А.Г., Семенов А.Д., и др.* О цифровой грамотности и средах ее формирования // Информатика и ее применения. 2020. Т. 14. Вып. 4. С. 100–107.
10. *Agliamutdinova D.B., Besshaposhnikov N.O., Kushnirenko A.G., et al.,* Problems of Early Learning to Program. How to Bridge the Gap Between Pictographic and Textual Programming Styles // International Journal of Education and Information Technologies (NAUN). 2021. V. 15. P. 331–343. https://doi.org/10.46300/9109.2021.15.35
11. *Betelin V.B., Kushnirenko A.G., Leonov A.G., et al.* Basic Programming Concepts as Explained for Preschoolers // International Journal of Education and Information Technologies (NAUN). 2021. V. 15. P. 245–255. https://doi.org/10.46300/9109.2021.15.25
12. *Бесшапошников Н.О., Кушниренко А.Г., Леонов А.Г., и др.* Цифровая образовательная среда "ПиктоМир": опыт разработки и массового внедрения годового курса программирования для дошкольников // Информатика и образование. 2020. № 10. Р. 28–40. https://doi.org/10.32517/0234-0453-2020-35-10-28-40
13. *Rogozhkina I.I., Kushnirenko A.G.* PiktoMir: teaching programming concepts to preschoolers with a new tutorial environment // Procedia—Social and Behavioral Sciences. 2011. V. 28. P. 601–605. https://doi.org/10.1016/j.sbspro.2011.11.114
14. *Bers M.U., Resnick M.* The Official ScratchJr Book: Help Your Kids Learn to Code, No Starch Press, 2015.
15. *Поликарпов С.А., Рудченко Т.А.* Бумажный и цифровой учебники в начальной школе. Преимущества и недостатки подходов // Информатизация образования и методика электронного обучения: Материалы III Междунар. конф.: в 2 ч. Красноярск: СФУ, 2019. Ч. 2. С. 617–621.
16. *Поликарпов С.А.* Математическое образование в России. Новые принципы подготовки учителей математики // Проблемы современного математического образования: Материалы Российско-Американского симпозиума 18–20 ноября 2016 г. / Под ред. А.П. Карпа и С.А. Поликарпова. Москва: МПГУ, 2017. 148 с. С. 74–93. http://mpgu.su/novosti/vyishel-sbornik-statey-aktualnyie-voprosyi-matematicheskogo-obrazovaniya/
17. *Meet Cubetto*. URL: https://www.primotoys.com
18. Matatalab coding set. https://matatalab.com/en/coding-set
19. *Sullivan A., Elkin M., Bers M.U.* KIBO Robot Demo: Engaging young children in programming and engineering. In Proceedings of the 14th International Conference on Interaction Design and Children (IDC'15). ACM, Boston, MA, US, 2015. https://doi.org/10.1145/2771839.2771868
20. Патент США US20140297035A1. https://patents.google.com/patent/US20140297035
21. *Бесшапошников Н.О., Леонов А.Г.* Пиктограммный язык программирования "ПИКТО" // Вестник кибернетики. 2017. Т. 4 (28). С. 173–180.
22. *Monika Mladenović, Saša Mladenović, Žana Žanko,* Impact of used programming language for K-12 students' understanding of the loop concept // International Journal of Technology Enhanced Learning. 2020. V. 12. Issue 1. P. 79–98. https://doi.org/10.1504/ijtel.2020.103817

ERROR-FREE 2D PICTOGRAMMIC SYNTAX IN A PROGRAMMING LEARNING ENVIRONMENT FOR PRESCHOOL CHILDREN

A. G. Kushnirenko[a], A. G. Leonov[a], and S. A. Polikarpov[b]

[a] *Federal State Institution "Scientific Research Institute for System Analysis of Russian Academy of Sciences", Moscow, Russian Federation*

[b] *Steklov Mathematical Institute of Russian Academy of Sciences, Moscow, Russian Federation*

Presented by Academician of the RAS A.L. Semenov

When mastering the basics of programming by preschoolers, serious difficulties are created by the need to diagnose and correct syntactic errors. With the traditional method of "on-screen" program editing, these difficulties can be avoided by blocking the child's actions that lead to syntactic violations. Today, the technique of compiling programs from material objects (tangible objects) with command icons printed on them is gaining popularity. When using this technique, no blocking of user actions is possible. In this situation, the authors propose to protect the child from syntactical errors by postulating the two-dimensionality of the program and defining the syntax and semantics of the pictogram programming language for beginners using indentation so that any placement of pictograms in the cells of a two-dimensional table gives a syntactically correct and executable program. This approach has been implemented and tested in the domestic educational environment "PictoMir" of pictogram programming for preschoolers.

Keywords: program, pictogram, textless programming environment, screenless programming, Piktomir, preschooler, robot, structured programming