МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

УТВЕРЖДАЮ

Зав.кафедрой,

к. ф.-м. н.

_____ М. В. Огнева

ОТЧЕТ О ПРАКТИКЕ

студента 1 курса 173 группы КНиИТ

Пронина Антона Алексеевича

вид практики: производственная распределенная (научно-исследовательская работа)

кафедра: Кафедра информатики и программирования

курс: 1

семестр: 1

продолжительность: 18 нед., с 01.09.2022 г. по 15.01.2023 г.

Руководитель практики от университета,

старший преподаватель _____ Е. Е. Лапшева

Тема практики:«Разработка комплекса аппаратно-программных и методических средств изучения робототехники в школе с использованием языка программирования КуМир»

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Стремительное вхождение в повседневную жизнь информационных и коммуникационных технологий стало возможным благодаря широкому распространению персональных компьютеров и созданию глобальной сети Интернет. В связи с этим вопрос совершенствования и модернизации сложившейся образовательной системы остается актуальным [1] . Одним из наиболее интересных вопросов, требующих особого внимания в обучении информатике, является вопрос системы обучения программированию. Это связано с тем, что профессия специалистов в области информатики и информационных технологий в какой-то мере начинается со школы. Одним из прямых приложений программирования является робототехника. Многие исследователи отмечают актуальность развития данного направления в рамках обучения школьников [2] , [3] , [4] . Образовательная робототехника - уникальный инструмент обучения, который помогает сформировать привлекательную для детей учебную среду с практически значимыми и занимательными мероприятиями, подкрепляющими интерес учащихся к изучаемым предметам [5] . В то же время, изучение языка программирования КуМир является неотъемлемой частью обучения в школе в рамках дисциплины “Информатика”. Ввиду растущего интереса в сфере образования к обучению робототехнике в школе и широкой распространенности и глубокой степени интеграции современной образовательной системы с этим языком, появилась идея о разработке транслятора с языка КуМир в язык C++. Актуальность идеи заключается в снижении входного порога в область робототехники как со стороны ученика, предоставляя возможность, используя полученные навыки программирования в системе КуМир, заниматься разработкой роботов, так и со стороны преподавателя, уменьшая затраты на приобретение программных и аппаратных средств разработки.

В качестве аппаратной платформы данной задачи был выбран электронный конструктор и удобная платформа быстрой разработки электронных устройств для новичков и профессионалов - платы Arduino, ввиду низкой стоимости устройств, периферийных модулей, простоты разработки аппаратных устройств на базе этих плат, высокой модульности систем и их высокой распространенности среди робототехников. Цель работы - разработка комплекса аппаратно-программных и методических средств изучения робототехники в школе с использованием языка программирования КуМир.

Для выполнения поставленной цели, требуется выполнить следующие задачи:

- анализ методики использования системы программирования Кумир в учреждениях

основного, среднего и дополнительного образования;

- анализ существующих систем программирования роботов в школьном обучении информатики: функционал, достоинства, недостатки;
- поиск и анализ существующих разработок взаимосвязи системы программирования КуМир и робототехнических систем;
- разработка требований к программному, методическому и аппаратному комплексу взаимосвязи языка программирования КуМир с платформой Arduino;
- создание программного обеспечения взаимосвязи языка программирования КуМир и Arduino;
- разработка технической и методической документации для комплекса;
- апробация разработанной системы в одном из образовательных учреждений города.

1 КуМир

1.1 Среда исполнения и язык программирования КуМир

КуМир (Комплект Учебных МИРов) - система программирования, предназначенная для поддержки начальных курсов информатики и программирования в средней и высшей школе с открытым исходным кодом [6] . Спектр возможностей применения данной системы программирования ограничен, ряд стандартных команд позволяет разрабатывать небольшие приложения с целью изучения основ программирования на процедурных языках. На данный момент язык КуМир - это язык, с которого хорошо начать, чтобы освоить основы алгоритмического подхода и процедурный стиль программирования (https://foxford.ru/wiki/informatika/sredaprogrammirovaniya-kumir). Система КуМир [7] в современном ее состоянии (с подсистемой ПиктоМир) состоит из расширяемого набора исполнителей (или роботов), набора систем программирования и вспомогательных утилит и программ. Расширяемый набор исполнителей (роботов) представляет собой отдельные самостоятельные программы и (или) электронно-механические устройства, имеющие собственное пультовое управление (интерфейс), а также возможность локального или сетевого управления из выполняющей системы.

Набор систем программирования:

- система программирования КуМир на школьном алгоритмическом языке, состоящая из редактора-компилятора алгоритмического языка с многооконным интерфейсом, интегрированная с выполняющей системой. Система не является полноценным интерпретатором или компилятором школьного алгоритмического языка. Зачастую, программа на школьном алгоритмическом языке компилируется в некий промежуточный код (подобный подход был использован в языке Java), который затем интерпретируется и исполняется с автоматической генерацией точек останова по событиям и шагам (при пошаговом выполнении). В системе программирования КуМир также отсутствует отладчик в его классическом понимании. Это имеет свои причины, так как сильно упрощает процесс освоения и написания системы программирования КуМир, а также отладку программ. Все изменения используемых в школьной программе величин автоматически визуализируются на полях программы. Визуализируются и результаты логических операций. Кроме того, к системе можно подключать любого исполнителя (робота) из набора и программно управлять им;
- система бестекстового, пиктографического программирования ПиктоМир. В ней учащийся собирает программу из команд исполнителя (робота). Управляющие конструкции

языка представлены определенной параметрической организацией алгоритмов. Созданная учащимися программа при выполнении передает команды робота и получает его состояние;

- система программирования на производственном языке программирования C++, Python и т.п; Вспомогательные утилиты и программы – это всевозможные редакторы миров исполнителей (роботов), конвертеры и препроцессоры из пиктографического языка в школьный алгоритмический язык, из школьного алгоритмического языка в C++ и Python [8] .

1.2 Исполнители в системе КуМир

Среда программирования КуМир обладает рядом исполнителей с разными уровнями сложности управления - “Робот”, “Чертежник”, “Вертун”, примеры программ исполнителей приведены на рисунках 1, 2.

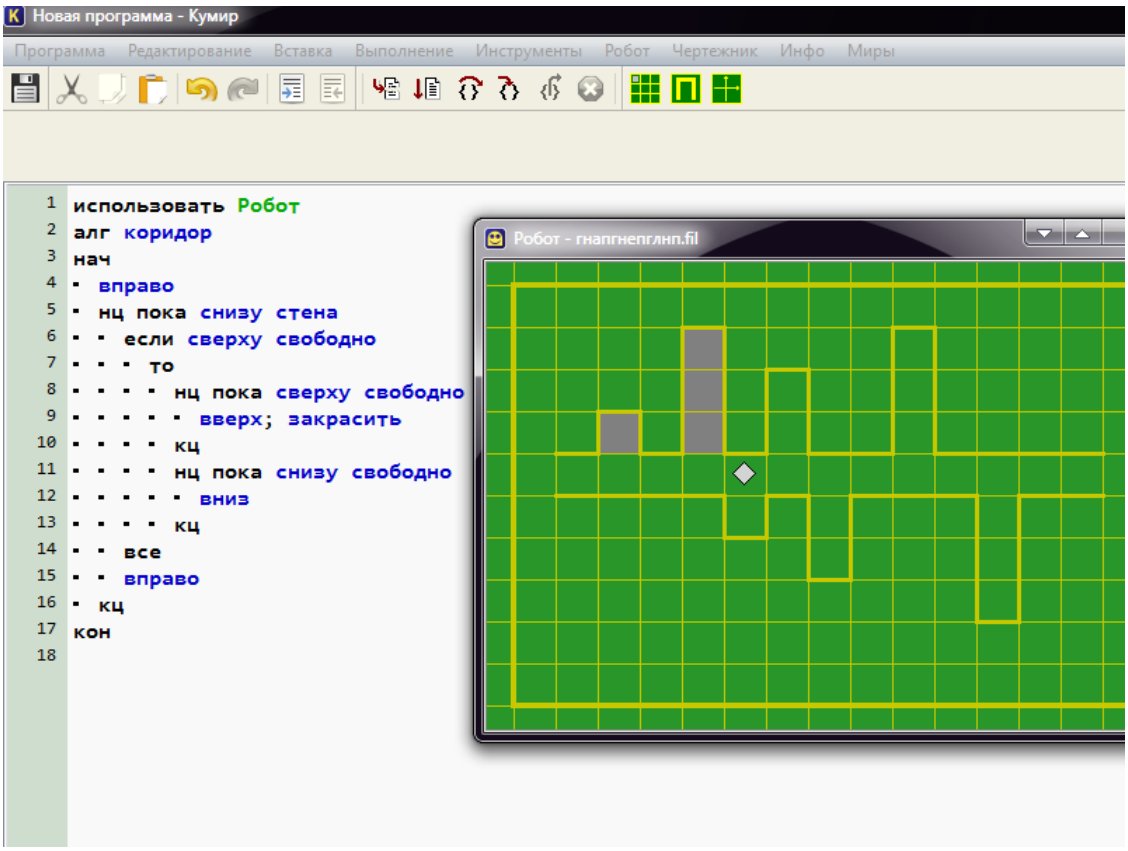

Рисунок 1 — Пример программы для исполнителя “Робот”

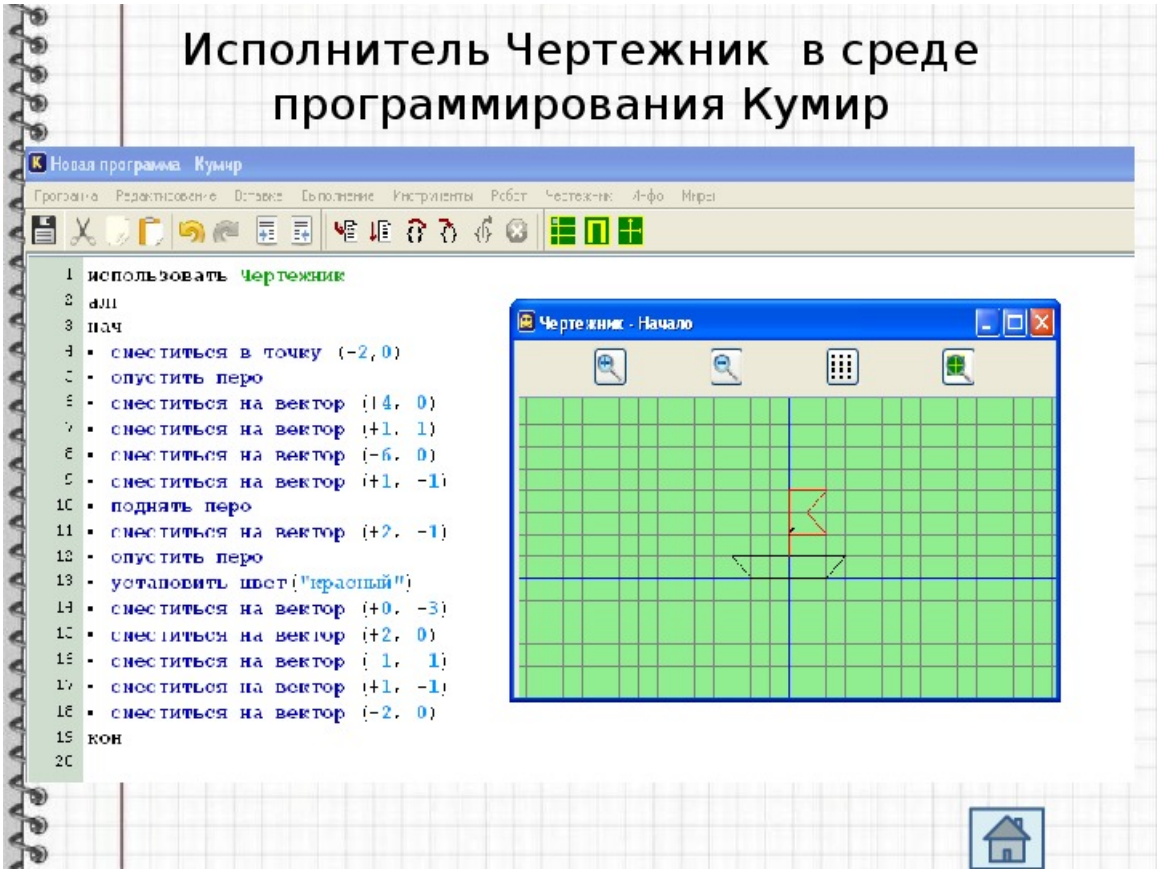

Рисунок 2 — Пример программы для исполнителя “Чертежник”

Обучающиеся могут плавно переходить между изучением исполнителей, постепенно повышая сложность разрабатываемых программ. КуМир-исполнители могут быть расположены либо в файле вместе с основной программой, либо в отдельном файле. Они доступны редактированию и выполнению в пошаговом режиме, как и обычная КуМир-программа. Таким образом, КуМир предоставляет базу для перехода от изучения основ объектно-ориентированного программирования в учебной среде к изучению производственных языков ООП [9] .

Исходный код системы программирования и языка программирования КуМир расположен в github-репозитории [10] . Данный факт предоставляет широкие возможности для развития языка программирования КуМир - любой желающий может разработать свою библиотеку, найти и исправить ошибку в работе некоторой части системы программирования. Такой доработкой может стать транслятор из языка КуМир в другой высокоуровневый язык программирования, например - C++.

1.3 Актуальный статус развития языка программирования КуМир

Последняя версия языка программирования КуМир (2.x) была выпущена в 2014 году. Разработчиками был решен ряд следующих задач:

1. возможность простого расширения функциональности системы путем реализации самодостаточных модулей;
2. возможность создания на базе системы Кумир различных конфигураций, адаптированных под специфичные применения, например, для проведения олимпиад;
3. радикальное увеличение скорости выполнения программ;
4. поддержка системы практикумов с возможностью самопроверки (для учеников) и отслеживания хода работы учеников (для учителей) [11] .

На момент 2022 года, развитие языка программирования КуМир приостановлено. Имеется ряд нерешенных проблем, вызывающих сложности при разработке. Например, поскольку в языке отсутствуют пространства имен, разделение функционала программы на библиотеки невозможно без знаний внутреннего устройства языка программирования КуМир и механизмов разработки исполнителей. Последние наработки создателей языка КуМир связаны с решением проблемы вынужденной смены формата обучения школьников с очного на дистанционный во время пандемии COVID-19. Разработчики предлагают решение описанной проблемы путем интеграции платформы КуМир с облачным сервисом Мирера. Данная доработка не позволяет полностью перейти от использования клиента системы программирования КуМир к работе при помощи облачного веб-сервиса, поскольку в рамках работы лишь был перенесен транслятор языка программирования с поддержкой стандартных исполнителей, визуальные среды для них не были проинтегрированы. Как отмечают авторы, даже при создании большого числа практикумов(500 задач), проделанной работы недостаточно. Без визуализации выполнения и отладки разработанной учащимся программы, значительно снижается скорость и качество восприятия изучаемого материала [12] .

1.4 Статистика использования языка программирования КуМир

КуМир активно используется при решении задач высокой сложности ГИА. В задании 15 учащимся предлагается выбор: задание 15.1 - написать программу для исполнителя “Робот” или задание 15.2 - разработка программы на универсальном языке программирования. Для выполнения задания 15.1 рекомендуют использовать среду программирования КуМир [13] .

Анализ результатов ГИА последних лет, позволяет понять, что школьники чаще и успешнее решают задачу 15.1. Для анализа были рассмотрены результаты проведения ГИА в нескольких регионах - Калининградская область, Чукотский автономный округ, Ленинградская область и Саратовская область.

Данные статистики результатов ГИА по выбранным регионам позволяют выделить следующие тезисы, касательно выбора и успешности решения заданий 15.1 и 15.2:

- данные по Чукотскому автономному округу за 2022 год, демонстрируют ухудшение результатов при решении задания 15.2 среди учащихся [14] ;
- результаты проведения ГИА по Калининградской области за 2022 год отмечают низкий процент успешного решения задания 15.2 среди школьников - из 274 учащихся, выбравших данное задание, всего 3 смогли получить за него максимальный балл. В основном, школьники выбирают задание 15.1 и справляются с ним [15] ;
- по данным результатов проведения ГИА в Ленинградской области за 2019 год 70% учащихся, приступивших к заданию 15.1 не получили за него баллов, 2% получили 1 балл и 27% учащихся получили максимальный балл за данное задание. В то же время, результаты школьников по решению задания 15.2 следующие: 90% не справились с заданием, получив 0 баллов, 2% учащихся получили 1 балл и лишь 8% школьников смогли получить максимальный балл за решение задания 15.2 [16] ;
- в Саратовской области в 2022 ГИА по информатике сдавали 6501 школьников. Среди них только 2780 учащихся приступали к решению заданий 15.1 и 15.2. Среди учащихся, выбравших задание 15.1, 42% не справились с заданием, набрав 0 баллов, 3% не полностью выполнили задание, получив 1 балл и 55% успешно справились с заданием, набрав максимальный балл. Статистика по результатам решения задания 15.2 следующая: 16% не справились с заданием, получив 0 баллов, 5% учащихся не полностью выполнили задание, получив 1 балл и 78% успешно справились с заданием, набрав 2 балла. Стоит отметить количественное соотношение учащихся - из 2780 учащихся, приступивших к решению задания, 2397 выбрали номер 15.1. Данные по результатам ГИА по саратовской области получены из данных, предоставленных региональным центром оценки качества (http://sarrcoko.ru/).

Проанализировав имеющиеся данные, можно сделать вывод о большей популярности задания 15.1 над заданием 15.2 - в Саратовской области школьников успешно справившихся с заданием 15.1 почти в 5 раз больше, чем справившихся с заданием 15.2! Данные свидетельствуют о высоком проценте использования полученных при изучении курса информатики навыков в среде программирования КуМир, подтверждающих актуальность данной работы.

На момент 2022 года, в составе ЕГЭ по информатике была изменена задача 6 - теперь это задание на анализ циклического алгоритма, выполняемого исполнителем. Авторы ЕГЭ

(https://vk.com/video-36510627_456239849) прямо говорят о том, что это можно и нужно решать с помощью исполнителей среды программирования [17] .

1.5 Анализ учебно-методических комплексов по информатике

В качестве материалов исследования рассматривались учебники из федерального перечня на 2022-2023 годы. Рассматриваемые учебники можно разбить на три группы по авторству:

- учебники за авторством Гейна А.Г. и Сенокосова А.И.;
- учебники за авторством Босовой Л.Л. и Босовой А.Ю.;
- учебники за авторством Полякова К.Ю. и Еремина Е.А. [18]

В учебниках написанных А.Г. Гейном и А.И. Сенокосовым изучению языка Кумир и его исполнителям отводится очень мало времени. На протяжении всего курса с 6 по 11 класс используется алгоритмический язык, подобный языку программирования КуМир, а для изучения понятия исполнителей приводятся задачи, требующие использования исполнителя “Робот”, но по факту, знакомство с языком КуМир происходит лишь в одной главе, сравнивающей программный код, написанный на алгоритмическом языке программирования, код на КуМире и Паскале.

В учебниках за авторством Босовых изучение языка программирования КуМир начинается в 6 классе со знакомства с исполнителем “Чертежник”. Помимо разбора команд исполнителя, материалы затрагивают понятие цикла n раз. Основы алгоритмизации и начала программирования раскрываются в учебнике для учащихся 8 классов. Для изучения понятия алгоритма и работы с исполнителем “Робот” используются примеры программ, реализованных при помощи языка КуМир. На этом обучение языку программирования КуМир заканчивается. Для более глубокого изучения программирования выбран язык Паскаль. Материал учебников не позволяет обучающимся познакомиться со всеми операторами и возможностями языка программирования и среды КуМир.

Больше всего изучению языка программирования КуМир посвящают время Поляков и Еремин. В 7 классе обучающиеся знакомятся с исполнителями “Робот”, “Чертежник”, “Черепаха” и “Вычислитель”, а также базовыми понятиями, такими как условная конструкция, цикл, вспомогательная программа и т.д. Для закрепления полученных знаний на практике приводятся примеры программ, написанных на языке программирования КуМир. В учебнике 8 класса на протяжении всего курса приводятся примеры программного кода на языке программирования КуМир. В третьей главе к изучению предлагаются два языка программирования на выбор

- КуМир или Паскаль. В учебнике подробно разбирают составляющие программного кода: комментарии, операторы, идентификаторы; количественно и качественно расширяется набор возможностей языка программирования КуМир - ранее пройденные элементы разбираются более детально, а также к рассмотрению приводятся новые конструкции языка, например сложные операторы ветвления и различные типы циклов. В учебнике 9 класса авторы так же предлагают к изучению языки программирования КуМир и Паскаль. Глава “Программирование” посвящена расширению списка конструкций языка программирования КуМир и решению задач с их использованием - символьные строки, массивы. матрицы, функции и процедуры. В учебнике 10 класса повышается сложность рассматриваемых задач - к изучению предлагаются задачи бинарного поиска и сортировки массива, реализуемые на языке программирования КуМир, а также приводятся функции работы с файловой системой, подкрепленные примерами. В учебнике для 11 класса язык программирования КуМир используется для разбора сложных задач, например, “Решето Эратосфена”. На данном этапе изучения языка программирования КуМир заканчивается ввиду недостаточности множества операторов. Часть программы обучения, посвященная изучению языка программирования КуМир рассматривает весь стандартный набор операторов и функции работы с файловой системой. Изучаемая теория подкрепляется заданиями, позволяющими закрепить полученные знания на практике. В качестве задач выступают широко распространенные задачи, изучаемые как в курсах среднего, средне-специального, так и высшего образования. Подход авторов позволяет изучить и использовать язык программирования КуМир в качестве скриптового языка программирования и решать с его помощью небольшие задачи, поскольку в серии учебников язык программирования КуМир рассматривается не только в контексте решения задач для исполнителей. Последовательное изложение материала с постепенным увеличением сложности предоставляет возможность плавно погрузиться в язык программирования.

Можно сделать вывод, что на изучение языка программирования КуМир отводится небольшое количество времени. Авторы учебников, составляющих федеральный перечень предлагают обучающимся изучать более популярные языки программирования с большим функционалом, такие как Паскаль и Python. Во многом причиной данного факта является ограниченность конструкций языка КуМир, малое количество библиотек, отсутствие шаблона объектно-ориентированного программирования, невозможность дальнейшего применения полученных знаний, а также сложности с разработкой программ на языке программирования КуМир на разных платформах.

2 Робототехника в школе

2.1 Задачи по внедрению робототехники

В педагогике отмечается важность творческого мышления для получения новых знаний и применения их на практике. Особенно интенсивно творческое мышление, в том числе инженерное творческое мышление, развивается в школьном возрасте. В условиях современной школы мощным средством развития творческого инженерного мышления являются специализированные робототехнические комплексы. Изучение робототехники обеспечивает в дальнейшем освоение большого числа прикладных технических наук. В свою очередь, быстрое развитие робототехнических средств обучения диктует необходимость проектирования соответствующих дидактических систем. Образовательная робототехника занимает достаточно большое место в школьной программе. Она преподается как в рамках специально отводимого учебного времени, так и в процессе внеурочной деятельности [19] .

Робототехника - это проектирование и конструирование всевозможных интеллектуальных механизмов-роботов, имеющих модульную структуру и обладающих мощными микропроцессорами. Робототехника - прикладная наука, занимающаяся разработкой автоматизированных технических систем. Занятия робототехникой способствуют развитию творческой активности учащихся, включению учащихся в различные виды творческой деятельности, внесению проблемности в дидактический процесс, развитию следующих качеств творческого мышления: самостоятельности, критичности и т. д. [20]

В ходе обучения робототехнике школьники активно участвуют в проведении теоретических и экспериментальных исследований робототехнических систем различного назначения, изучают методы и средства их проектирования и моделирования. Разрабатывают новые методы управления и обработки информации, осуществляют поиск новых конструктивных решений и робототехнических систем широкого назначения, их подсистем и отдельных модулей. Анализируя возможности робототехники как учебной дисциплины, следует выделить компетенции, которые развивать в рамках других дисциплин сложно:

1. способность адаптироваться к решению различных научных, производственно-технологических задач;
2. возможность эффективно применять современные информационные технологии в проектировании программно-аппаратных комплексов и систем;
3. способность мыслить на стыке наук математики, информатики и физики.

Существенный объем содержания обучения робототехнике тесно связан с алгоритмизацией и программированием. Без освоения понятий алгоритма, основных управляющих алгоритмических структур (следование, разветвление, цикл, вспомогательный алгоритм) и использования их при программировании невозможно освоение управления робототехнической конструкцией, а также решение таких задач, как движение робота в пространстве и объезд препятствий с использованием датчиков, движение по заданной траектории и др.

При обучении алгоритмизации традиционно используются исполнители, например, в учебнике Л. Л. Босовой – целый ряд исполнителей: “Робот”, “Чертежник”, “Черепаха” в виртуальной среде КуМир.

Робототехника в настоящее время рассматривается в образовательной практике и как новое направление в содержании обучении, как новая технология обучения. Стоит определить следующие педагогические задачи использования робототехники в обучении школьников:

- демонстрация возможностей робототехники как одного из основных направлений научно-технического прогресса и новой технологии;
- усиление профильной подготовки учащихся, их ориентация на профессии инженерно-технического профиля;
- развитие у школьников познавательного интереса и мотивации к изучению ряда школьных предметов [21] .

Повышение качества образования за счет следующих факторов:

1. углубление и расширение предметного знания;
2. совершенствование знаний в области прикладных наук;
3. интеграция материальных и информационных технологий, интеграция конструирования и программирования;
4. интеграция знаний в области математических, физических, инженерно-технических наук и информатики, углубление межпредметных связей;
5. формирование умений и навыков проектирования, моделирования и конструирования;
6. развитие учебно-исследовательских умений и навыков, овладение методами проведения экспериментов.

2.2 Анализ подходов к внедрению изучения робототехники в школе

На данный момент существует большое количество подходов к внедрению робототехники в школе. Проанализировав актуальные исследования и разработки, можно группировать

самые популярные подходы:

- использование комплектов Arduino и языка C++;
- использование комплектов Lego MIndstorm и среды Trik Stduio;
- использование собственных разработок, например, комплект “УМКА” [22] .

Самым популярным комплектом для занятий робототехникой по уровню доступности является Arduino. Низкая стоимость компонентов, большое сообщество разработчиков и среда для программирования с открытым исходным кодом предоставляют массу возможностей по созданию собственных роботов и модификации существующих. Имеется масса вариантов для старта изучения робототехники на основе данной платы - можно самостоятельно приобрести контроллер Arduino и набор периферийных модулей или рассмотреть готовые комплекты, например серию “Матрешка” от компании “Амперка” (https://amperka.ru/page/matryoshka-comparison).

Другим популярным вариантом является программирование роботов LEGO Mindstorm в специализированной среде программирования. Среда TRIK Studio – это дальнейшее развитие QReal:Robots, поддерживавшей конструкторы LEGO Mindstorms NXT. Данная среда программирования использовалась при проведении олимпиады «Робофест» в 2020 году. Также оборудование ТРИК используется при реализации проектов в рамках Олимпиады Кружкового движения Национальной технологической инициативы — всероссийских инженерные соревнования для школьников 5?11 классов по 30 профилям. TRIK Studio поддерживает все основные возможности конструктора ТРИК: работа с видеокамерой на роботе, синтез речи, обмен сообщениями между роботами в группе, работа с Android-пультом, работа с гироскопом и акселерометром. Разумеется, поддержаны и обычные для робототехнических конструкторов возможности, такие как работа с датчиками, моторами, дисплеем робота, кнопками. В TRIK Studio также была реализована поддержка LEGO NXT и LEGO EV3 (в ограниченном объёме). Переключение между режимами осуществляется в настройках программы и на панели инструментов [23] .

Имеется ряд работ, сравнивающих дидактический потенциал и возможности платформ Arduino и микроконтроллеров семейства LEGO. Исследователи отмечают более высокую сложность платформы Arduino, предоставляющую более гибкую и изменяемую на разных уровнях разработку.

Сравнивая платформы Lego и Arduino можно выделить следующие их общие отличительные особенности:

1. стоимость комплекта Arduino составляет в среднем 6000 рублей. Детали можно докупать по отдельности. В то время как стоимость комплекта Lego mindstorms ev3 начинается от

40000 рублей по данным на 2022 год. Детали нельзя докупать по отдельности;

2. комплектующие Arduino представляют собой стандартные радиодетали. Детали Lego являются уникальными;
3. габаритные размеры компонентов конструктора Arduino значительно меньше чем у lego;
4. к Arduino можно подключить различные модули и компоненты (в том числе и спроектированные самостоятельно), в то время как у Lego используются только определенные компоненты и специализированные разъемы;
5. микроконтроллер Arduino программируется на C-подобном языке, а для Lego разработан свой собственный язык и графическая среда.

Из данного анализа можно сделать вывод, что платформа Arduino является более универсальным и гибким средством, позволяя конструировать и программировать произвольные программно-аппаратные комплексы, тогда как платформа Lego ограничена набором типовых деталей конструктора, включая датчики и исполнительные механизмы [24] , конструктора на базе данного микроконтроллера в формировании технической культуры школьников [25] , образовательной робототехники в обучении дошкольников и младших школьников (на основе опыта разработки и реализации образовательной программы «Хорошо играть») [26] .

2.3 Аппаратно-программный комплекс Arduino

Использование платформы Arduino в педагогической деятельности открывает новые возможности для студента и школьника. Проекты, реализуемые в учебном процессе в образовательных учреждениях, могут выполняться и развиваться дома. Помимо раскрытия творческих способностей создается благодатная основа для реализации индивидуализации учебного процесса. В последние годы во многих школах активно организуется основная и дополнительная образовательная деятельность учащихся, связанная с освоением элементов робототехники (https://www.arduino.cc/en/Guide/Introduction).

Развитие современных информационных технологий будет связано не только с появлением новых технических совершенных устройств, но и с развитием робототехники. Основы робототехники еще не является обязательной составляющей ФГОС ООО, поэтому обучение этому курсу возможно в рамках внеурочной деятельности или предпрофильной подготовки (элективные курсы), а также в профильном обучении в 10–11 классах.Исследователи вопроса актуальности изучения робототехники на базе платформы Arduino в образовательном процессе отмечают повышение креативности учащихся [27] , активное формирование и оттачива-

ние профессиональных навыков и умений, а также развиваются и осваиваются компетенции естественно-научной направленности, связанные с физико-техническими дисциплинами [28] .

Arduino - комплекс аппаратно-программных средств с открытым исходным кодом, обладающий низким порогом вхождения как со стороны программирования, так и со стороны электроники. Электронные платы способны считывать и генерировать входные и выходные сигналы. Для передачи последовательности команд управления плате используется язык программирования Arduino и соответствующая среда исполнения. Многолетняя история развития платформы Arduino включает в себя тысячи проектов - от небольших устройств до крупных научных инструментов. Сообщество разработчиков данной платформы состоит из программистов различного уровня - от любителей до профессионалов, собравших и систематизировавших свой опыт взаимодействия с Arduino для помощи в вопросах разработки проектов, полезный как новичкам, так и профессионалам. Arduino стала самой популярной платформой для разработки встроенных систем благодаря простоте взаимодействия и дизайну. Программное обеспечение может быть легко использовано на любой операционной системе. Студенты и преподаватели используют комплекс Arduino для разработки недорогих научных инструментов, для исследования физических и химических законов, а также для изучения основ программирования и робототехники. Дизайнеры и архитекторы создают интерактивные прототипы, музыканты и художники используют его для создания инсталляций и экспериментирования с новыми инструментами. Большим плюсом является огромное число инструкций по созданию устройств шаг-за-шагом, позволяющим любому разработчику повторить и попробовать существующее решение.

Основные плюсы:

- низкая стоимость плат и компонентов платы Arduino сравнительно недороги в сравнении с прочими программно7 аппаратными платформами. Минимальная конфигурация системы может быть собрана вручную, используя готовые модули, стоимостью не более 50$;
- кросс-платформенность среда исполнения Arduino исполняется на Windows, Macintosh OSX и Linux, хотя прочие микроконтроллерные системы доступны лишь для разработки, используя ОС Windows;
- простая, чистая среда для программирования редактор кода Arduino прост и гибок как для новичков, так и для профессионалов в данной сфере. Для преподавателей его удобность кроется в среде программной обработки, позволяя студентам создавать программы

на языке программирования высокого уровня без глубоких знаний в низкоуровневом программировании;

- открытое расширяемое ПО среда программирования Arduino - свободно распространяемая среда исполнения программного кода, написанного на языке программирования, схожем с AVR C, на котором основана платформа;
- открытая и расширяемая аппаратная часть схемы плат Arduino публикуются в сети с использованием открытой лицензии, позволяющей любому разработчику плат воспроизвести плат в соответствии с чертежом и модернизировать её. Даже неопытные пользователи могут создать кальку модуля, используя макетную плату для понимания механизмов её работы и экономии денег [29] .

2.4 Соревнования по робототехнике в школе

Соревновательная робототехника — это прикладное направление, в основе которого практическая работа конкурсантов. С целью тестирования возможностей роботов и их разработчиков придумывают множество разных заданий. Например, на протяжении многих лет не угасает интерес к боям роботов. И на то есть ряд причин. В первую очередь, это открытость мероприятия — принять участие в нем может любой, кто способен создать своего бойца.

Как правило, принять участие в соревнованиях может любой желающий, проходящий по возрасту. Соревнования по робототехнике проводятся для школьников и студентов, есть ряд мероприятий для дошкольников. В конкурсах в зависимости от условий могут принимать участие как команды от 2-х человек, так и одиночки. Каждому соревнованию предшествует этап подготовки и, как правило, участники готовятся к заранее определенному турниру: некоторые подразумевают, что конкурсанты принесут готовые модели на конкурс, другие — сборку модели на месте.

Начиная с 2015 года, в Российской Федерации ежегодно проводится Национальная технологическая олимпиада - командные инженерные соревнования для школьников и студентов, объединяющие самых разных людей, которые хотят и могут решать приоритетные технологические задачи, стоящие перед Россией (https://ntcontest.ru/).

Участвуя в НТО, школьники и студенты со всей России обучаются у лучших и решают задачи, поставленные государственными компаниями, лидерами технологических отраслей, прорывными технологическими компаниями. Участники знакомятся с самыми разными областями: от искусственного интеллекта и «умной» энергетики до нейротехнологий и геномного

редактирования.

Участие в подобных олимпиадах позволяет школьникам получить уникальный опыт решения практических задач за рамками программы обучения. Победа дает абитуриентам льготы на 100 баллов ЕГЭ при поступлении в ведущие инженерные вузы; победителей приглашают на стажировки; студенты выпускных курсов могут получить бонусы при поступлении в магистратуру.

3 Разработка транслятора с языка КуМир в язык C++

3.1 Модуль для взаимодействия с аппаратно-программным комплексом Arduino

Программный код для языка программирования КуМир группируется в библиотеки для возможности переиспользования, поддержки и распространения. Одной из таких библиотек является разработанный модуль, предоставляющий набор основных команд, используемых при программировании роботов на базе аппаратной платформы Arduino.

Разработанный модуль определяется в виде использования исполнителя в среде программирования КуМир.

Исполнитель системы «Кумир 2.0» представляет собой подключаемый программный модуль, который может быть использован при написании программ как на языке Кумир, так и на других языках программирования, поддерживаемых системой.

Существует следующие типы реализации исполнителей:

1. реализация исполнителя на целевом языке программирования (Кумир, Python и т. д.);
2. реализация исполнителя с использованием технологий QML/JavaScript;
3. реализация исполнителя на базе WebKit с использованием технологий HTML 5;
4. реализация исполнителя на языке C++ с использованием средств Qt.

Для реализации исполнителей на языке C++ в системе Кумир 2.0 был создан инструмент, позволяющий создать единый шаблон разработки исполнителей. Преимущества, предоставляемые данной разработкой:

1. обеспечение единого программного кода исполнителя, назависящего от изменений в программном интерфейсе системы Кумир 2x;
2. реализация только функциональной части исполнителя, абстрагированной от деталей взаимодействия с системой Кумир;
3. возможность создания программных интерфейсов исполнителя для различных языков программирования на основе декларативного описания команд исполнителя.

Создание нового исполнителя на языке C++ системы Кумир 2.x состоит из следующих этапов:

1. декларативное описание исполнителя: его системы команд и, при необходимости, пользовательского интерфейса;
2. создание на основе декларативного описания проекта на языке C++, содержащего заготовки файлов, необходимых для реализации;

3. реализация функциональности исполнителя.

Под декларативным описанием подразумевается файл в формате JSON, который является частью проекта. Каждый исполнитель представляется отдельным проектом при компиляции исходных кодов.

Для создания проекта на основе декларативного описания, необходимо использовать скрипт генерации кода исполнителя при помощи команды — gen_actor_source.py, в качестве аргументов указываются флаг –project и имя файла с декларативным описанием исполнителя в формате JSON.

После этого будут созданы файлы: CMakeLists.txt — файл проекта CMake, имя_исполнителямodu и имя_исполнителямodule.h – файлы заготовок для реализации функциональности исполнителя. Данные файлы создаются только один раз при создании проекта и впоследствии не перезаписываются автоматически.

Замечание: на этапе генерации исходных текстов выполняется только синтаксический, но не семантический анализ на возможные ошибки. Таким образом, если результатом работы скрипта gen_actor_source.py является заведомо некомпилируемый текст, это означает, что исходный файл описания содержит ошибку.

Разработанный модуль приведен в приложении А.

3.2 Реализаця транслятора с языка КуМир в язык C++

Сложность разработки Приложение разрабатывается на языке C++ с использованием фреймморвка QT 4 или 5 версии, а также системы автоматизации сборки программного кода CMake. Для сборки проекта на операционной системе Windows, требуется программа-сборщик MS Build 2012, устаревший на данный момент и труднодоступный для скачивания, поэтому было предпринято решение разрабатывать приложение на операционной системе Linux Mint. Для сборки проекта требуется установить ряд дополнительных библиотек, среди которых: ZLib, Boost и интерпретатор языка программирования Python.

Проект состоит из ряда библиотек, используемых внутри проекта, в том числе - AST-дерево, библиотека для работы с низкоуровневой виртуальной машиной, библиотеки для рендеринга pdf-документов, . . . ; плагинов, а также исполнителей и сред разработки, состоящих из файлов с описанием использующихся методов, процедур и сущностей и файлов с описанием слоя представления данных частей. Ввиду большого объема приложения, высокого уровня связности элементов друг с другом и устаревших библиотек, используемых для разработки, сборка

проекта осуществляется при помощи командной строки.

Изначально, для сборки исходных кодов представления проекта использовалась 4 версия фреймворка QT. Разработчики добавили возможность сборки проекта с использованием 5 версии фреймворка для повышения поддерживаемости и актуальности библиотек. Как оказалось, добавление возможности сборки проекта с использованием более современной версии фреймворка QT повлекло за собой негативные последствия, затрудняющие разработку — изучив ряд современных сред разработки для работы с используемыми в проекте библиотеками, оказалось, что ни одна из них не способна скомпилировать проект. Из-за объема различий фреймворка QT, отладка проекта в момент исполнения в привычном понимании недоступна — возможность отслеживания изменений при помощи точек останова отсутствует. Единственный способ отладки приложения в момент исполнения — вывод данных в формате сообщений об ошибках в момент исполнения программы.

Ещё одной сложностью, повышающей трудоемкость задачи является отсутствие документации к исходным кодам системы программирования КуМир. Названия методов и сущностей не всегда позволяют понять предназначение программного кода. Зачастую в программном коде встречаются некорректно именованные блоки кода, не позволяющие понять их роль в программе. Иногда встречаются комментарии, поясняющие процесс выполнения кода программы, но их объем не покрывает и доли процента программного кода проекта, что снижает доступность кода.

Разработка плагина для трансляции В плагине kumirCodeGenerator осуществляется трансляция и компиляция программного кода с языка КуМир в язык низкоуровневой виртуальной машины. По имеющемуся файлу с исходным кодом происходят лексический и синтаксический анализы, в случае, если они завершились успешно и не было выявлено ошибок, трансляция продолжается. В результате лексического анализа имеется набор лексем кода исходного языка программирования, использующийся для синтаксического анализа. В результате синтаксического анализа программа создает набор токенов для генерации по ним дерева разбора. Вычисленное дерево разбора усекается до AST-дерева, для трансляции кода. На этом этапе происходит наращивание кода на выходном языке по имеющемуся дереву разбора. Описанные выше этапы осуществляются при помощи вызова функций сторонних модулей.

Для разработки транслятора на основе измененного и переработанного программного кода плагина kumirCodeGenerator был создан модуль

arduinoCodeGenerator. Основные изменения затрагивают сущность Generator, содержащую основной объем операций по обработке данных AST-дерева и наращиванию программного кода по нему на выходном языке.

Поскольку в изначальном варианте программный код транслировался в язык виртуальной низкоуровневой машины, был кардинально переработан список команд. Список команд для трансляции с языка программирования КуМир в язык программирования C++ приведен в таблице 1.

Таблица 1 – Список команд, используемый для трансляции с языка программирования КуМир в язык программирования C++

Название операции	Код операции
ForLoop	0
WhileLoop	1
VAR	2
ARR	3
FUNC	4
CONST	5
IF	6
ELSE	7
SWITCH	8
CASE	9
INPUT	10
OUTPUT	11
BREAK	12
END_ARG_DELIM	13
END_FUNC_DELIM	14
END_ARR_DELIM	15
STR_DELIM	16
END_ST_DELIM	17
END_VAR_DELIM	18
END_ST_HEAD_DELIM	19
RET	20
SUM	21
SUB	22
MUL	23
DIV	24
POW	25
NEG	26
AND	27
OR	28
EQ	29
NEQ	30
LS	31
GT	32
LEQ	33
GEQ	34
ASG	35
DCR	36
INC	37

Были изменены инструкции для трансляции циклов, условий, вызова и объявления функций. Были добавлены инструкции для объявления переменных и констант. Блок операций подвергся минимальным изменениям — были добавлены инструкции инкремента и декремента, а также оператор присваивания.

Также изменилась основная сущность, используемая при трансляции — сущность инструкции выходного языка. Были удалены поля для хранения данных о регистре, спецификации строки, спецификации модуля и были добавлены поля для хранения имени операнда и типы операнда, если присутствует.

Трансляция циклов Существует 4 типа циклов в языке программирования КуМир: стандартный для языков программирования высокого уровня цикл, выполняющийся некоторое количество раз, определяемое набором элементов в указанном диапазоне “нц для“, цикл с предусловием “нц пока“, цикл, выполняющийся n раз “нц для n раз“, а также цикл, выполняющийся постоянно до остановки исполнения “нц всегда“. При трансляции в низкоуровневый язык, этап трансляции циклов требовал одной инструкции — “LOOP“, а также тела операций, выполняющихся до данной метки. В начале транслировалось тело цикла, в случае цикла с предусловием до тела транслировалось условие исполнения, на последнем этапе трансляции добавлялась инструкция “LOOP“, объявляющую метку завершения определения цикла.

Для трансляции циклов в язык C++ были добавлены инструкции для соответствующих типов циклов — ForLoop, WhileLoop. Циклы “ нц пока“и “ нц всегда“ определяются при помощи инструкции WhileLoop, “нц для“ и “нц для n раз“ - при помощи ForLoop.

В начале трансляции цикла любого типа указывается заголовок, определяющий тип цикла и возможные предусловия. В случае цикла “нц пока“, заголовок содержит условие окончания работы, состоящее из ряда выражений.

После трансляции заголовка инструкции цикла транслируется тело цикла. Трансляция завершается инструкцией END_ST_DELIM.

Трансляция подвыражений Трансляция подвыражений осуществляется при помощи метода calculate. Любое подвыражение представляется бинарным деревом, в вершине которого находится операнд, а на листьях - константы или переменные. Для корректной трансляции подвыражений требуется разобрать дерево подвыражения снизу вверх — найти узел, листья которого не содержат дальнейшей вложенности, затем добавить в стек инструкций левый лист

узла, затем сам узел, в конце — правый лист. Метод calculate используется во множестве мест и содержит логику для трансляции переменных, констант, вызовов функций и подвыражений. Поскольку метод является рекурсивным, невозможно добавить дополнительный блок операций трансляций, описанный выше, в тело метода calculate, поэтому было принято решение вынести логику данного метода в новый - innerCalculation, превратив метод исходный в обертку, куда и была добавлена вспомогательная логика.

Трансляция условных выражений Изначально, трансляция условных конструкций состояла из определения количества подвыражений транслируемого выражения и поиска ошибок. На каждую конструкцию “если то“ и “иначе“ добавлялась инструкция безусловного перехода, изменяя ход исполнения программы. Для установки места программы низкоуровнего языка, куда совершался переход используется регистр IP.

В разработанной реализации трансляция условных выражений повторяет трансляцию цикла с предусловием за исключением первой инструкции — вначале, в стек инструкций добавляется инструкция “IF“ или “ELSE“, указывающая трансляторy на объявление соответствующего блока. Процесс повторяется пока сущность, содержащая данные об условном выражении не опустеет.

Трансляция блоков ‘выбор‘ происходит следующим образом: вначале в стек заносится инструкция с кодом “SWITCH“, далее добавляется переменная, значения которой перебираются и на каждый блок ‘при условие‘ добавляется инструкция “CASE“ и константа, хранящая значение переменной. В случае наличия метки “иначе“ в блоке “выбор“, добавляется инструкция “CASE“ без константы, транслирующаяся в инструкцию “default“ в коде выходного языка.

Трансляция переменных и констант В исходной версии при трансляции констант в выражениях или при инициализации, в стек виртуальной машины заносился индекс переменной или константы среди всех встреченных при трансляции, извлекаемый по ссылке на этапе исполнения. В разработанной реализации в случае использования переменной в стек добавляется инструкция “VAR“, хранящая ссылку на название переменной и на тип, в случае объявления. Трансляция констант начинается с занесения в стек инструкции “CONST“, хранящей индекс значения и тип константы в случае инициализации.

3.3 Разработка системы автоматизированного тестирования транслятора

Процесс тестирования Под тестированием компилятора понимается сборка кросс-компилятора из исходных кодов, трансляция тестовой программы и сравнение результатов работы кросс-компилятора с ожидаемым результатом. Было решено автоматизировать процесс тестирования, для повышения качества и снижения сложности процесса. Т.к. язык программирования и среда КуМир - кроссплатформенные проекты, в начале процесса автоматизации стоило решить вопрос с внешними условиями - операционной системой, набором используемых пакетов и библиотек, . . . , т.о. первым шагом к автоматизации тестирования явилось создание централизованной среды для сборки компилятора, а именно разработка dockerfile файла, содержащий набор инструкций для сборки кросс-компилятора из исходных кодов. В качестве операционной системы была выбрана Ubuntu, ввиду широкой распространенности, большого сообществу разработчиков и активной поддержке пакетов с данной системой в репозитории docker. Далее встал вопрос автоматизации запуска кросс-компилятора. Для решения этой проблемы был разработан скрипт на языке Python 3. Для автоматизации запуска тестовой системы в репозитории разрабатываемого кросс-компилятора были настроены службы автоматического запуска(Github Actions) при отправке изменений и создании запроса на внесение изменений в целевую ветвь.

Тесты Для тестирования были разработаны набор тестов, покрывающие основные инструкции языка программирования КуМир, а именно: работа с переменными(инициализация, присваивание), арифметические и логические выражения, ветвления, циклы, а также функции и процедуры. Тестирование построено на принципе сравнения ожидаемого результат с результатом работы кросс-компилятора. Тесты сгруппированы по подмножествам инструкций исходного языка. Группы тестов приведены на рисунке 3.

Рисунок 3 — Группы тестов

Группа состоит из секций, представляющих отдельные инструкции. Для создания тестовой секции, например, для тестирования трансляции функций, необходимо 2 файла - файл с расши-

рением .kum на исходном языке программирования, а также файл с расширением .exp - файл с предполагаемым результатом работы кросс-компилятора, содержащий программу на языке C++. Пример секции тестов приведен на рисунке 4.

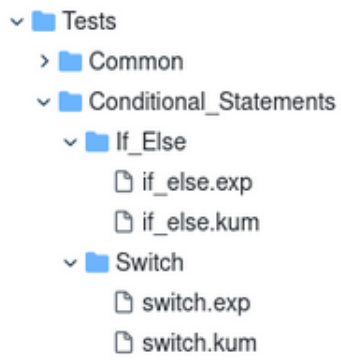

Рисунок 4 — Пример секции тестов

Под тестом кросс-компилятора понимается 2 файла, содержащие 1 или несколько инструкций на исходном и выходном языках программирования. На данный момент разработано чуть больше 200 тестов.

Программа-тестер Для автоматизации тестирования было разработано консольное приложение-тестер кросс-компилятора. В качестве языка программирования был выбран Python 3. Для работы с программой был разработан ряд аргументов командной строки, а также справка. Список аргументов командной строки приведен ниже:

```
[-h] [--help] - show help.
[-tr] [--translator] ["the path to pre-builded kumir2 to
 arduino translator instance"] - show app what translator instance to use.
[-t] [--path-to-tests] ["the path to tests folder"] -
show app  the folder with test files.
[-o] [--output] ["the path to log file"] - show app where to store test logs.
[-d] [--duplicate] - duplicate output to console.
[-ss] [--skip-successfull] - skip open log info
about succesfully completed tests.
[-sf] [--skip-failed] - skip open log info about failed tests.
[-swe] [--skip-without-expectation] - skip open log info
 about tests for which the file with expectations was not found.
[-sce] [--skip-with-compiler-error] - skip open log info
```

```
14  about tests ended with compiler error.
15 [-b] [--brief] - skip open log info about all tests.
```

Было решено разработать программу гибкой и информативной. Для работы необходимы 2 аргумента –translator(путь к кросс-компилятору), а также –path-to-tests(путь к папке с тестами). По умолчанию результат тестирования записывается в файл, также есть возможность продублировать вывод в консоль. Существует 4 состояния, отражающие результат работы компилятора:

- успех;
- провал;
- ошибка компиляции;
- не хватает файла с ожиданиями.

Состояния 3 и 4 могут быть получены по причине некорректного содержания тестовых файлов - либо написанная на языке программирования КуМир программа имеет ошибки и не может быть скомпилирована, либо не был найден файл с расширением .exp, содержащий ожидания работы компилятора. В других случаях, текст сообщения с результатами тестирования содержит предложение вызвать команду vimdiff с путями до файлов и прореферировать результат. Можно посмотреть как три файла сразу (исходный код, ожидаемый результат, результат работы компилятора), так и только ожидания и результат работы. Пример сообщения о результатах работы приведен на рисунке 5.

```
Test #0. Test /home/anton/Sources/kumir2/kumir_tests/Sources/2.
Test is not completed(
Sources for the test: /home/anton/Sources/kumir2/kumir_tests/Sources/2.kum
Test expectations: /home/anton/Sources/kumir2/kumir_tests/Expectations/2.c
Test results: /home/anton/Sources/kumir2/kumir_tests/Results/2.kumir.c.
To get more detail info about comparison results, print:
vimdiff /home/anton/Sources/kumir2/kumir_tests/Sources/2.kum /home/anton/Sources/kumir2/
kumir_tests/Expectations/2.c /home/anton/Sources/kumir2/kumir_tests/Results/2.kumir.c
or vimdiff /home/anton/Sources/kumir2/kumir_tests/Expectations/2.c /home/anton/Sources/kumir2/
kumir_tests/Results/2.kumir.c[0m
```

Рисунок 5 — Пример сообщения с возможностью подробно посмотреть результаты тестирования

В начале вывода информации приведена краткая статистика - сколько тестов завершилось и с каким состоянием. Объем вывода можно фильтровать и убирать подробную информацию о сообщениях в состояниях 1-4, либо полностью отключить подробный вывод информации, добавив флаг -b. При дублировании вывода в консоль, в зависимости от состояния, сообщение будет выделено цветом. Пример краткого вывода по результатам тестирования приведен на рисунке 6.

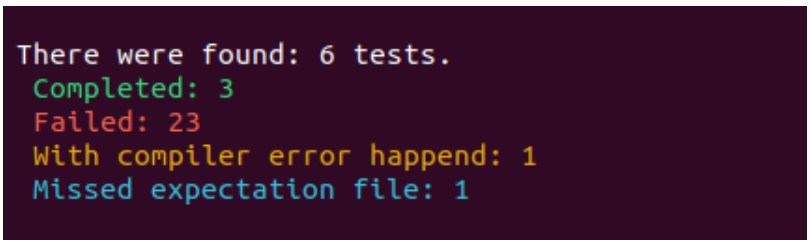

Рисунок 6 — Пример краткий вывода по окончанию тестирования

ЗАКЛЮЧЕНИЕ

В ходе работы были изучены и проанализированы материалы по методике преподавания языка программирования КуМир, и проведено исследование использования школьниками языка программирования КуМир, также были проанализрованы исследования, определяющие роль изучения робототехники в рамках обучения школьников для различных аппаратных комплектов, используемых при обучении. Были решены следующие задачи:

- был исследован набор основных команд и архитектура программ при разработке на языке программирования КуМир;
- были проанализированы результаты ОГЭ за 2019-2022 годы в разных регионах российской федерации;
- были рассмотрены учебники, рекомендованные ФГОС по предмету "Информатика"на 2022-2023 с целью анализа методики использования системы программирования Кумир в учреждениях основного и среднего образования;
- были проанализированы современные исследования, определяющие цели и задачи изучения робототехники в рамках школьной программы;
- были исследованы и анализированы используемые аппаратные платформы для изучения робототехники, а также их альтернативы;

В рамках ВКР был был разработан транслятор с языка программирования КуМир в язык C++. Программа имеет ряд недостатков, которые предстоить исправить и список улучшений, которые планируется реализовать. Среди задач по улучшению транслятора можно выделить следующие:

- создание системы тестирования разработанного транслятора;
- тестирование и поиск ошибок при трансляции программного кода с языка КуМир в C++;
- создание отдельного клиента среды программирования КуМир, специально для разработки роботов;
- добавление возможности прошивки робота из клиента;
- добавление инструмента выбора порта с подключенным роботом для прошивки;
- добавление настраиваемого алгоритма прошивки, определяющего роль результата трансляции в архитектуре программы для прошивки робота.

После реализации клиента среды программирования КуМир для разработки роботов и

исправления ошибок транслятора найденных в ходе тестирования планируется спроектировать и разработать робота на базе аппаратного комплекса Arduino для опробации разработки в школах.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 *Сармантаева, Л. С.* Исследование технологий обучения программированию в школе / Л. С. Сармантаева // *Проблемы и перспективы развития образования в России.* — 2012. — № 16. — С. 111–114.

2 *Челнокова, Е. А.* Робототехника в образовательной практике школы / Е. А. Челнокова, А. В. Хижная, Д. А. Казначеев // *Проблемы современного педагогического образования.* — 2019. — Т. 1, № 65. — С. 297–299.

3 *Сенюшкин, Н. С.* Изучение робототехники в школе - путь интеграции в инженерное образование / Н. С. Сенюшкин // *Актуальные проблемы авиации и космонавтики.* — 2014. — Т. 2, № 10. — С. 377–378.

4 *Брянцева, Р. Ф.* Занимательная робототехника в современной школе / Р. Ф. Брянцева // *Наука и перспективы.* — 2018. — № 1. — С. 3–7.

5 Образовательная робототехника дайджест актуальных материалов / Под ред. Т. Г. Попова. — Екатеринбург: ГАОУ ДПО «Институт развития образования Свердловской области»; Библиотечно-информационный центр, 2015.

6 Оффициальный сайт КуМир [Электронный ресурс]. — 2022. — URL: https://www.niisi.ru/kumir/. Загл. с экр. Яз. рус.

7 *Леонов, А. Г.* Методика преподавания основ алгоритмизации на базе системы «КуМир» [Электронный ресурс]. — 2009. — URL: https://inf.1sept.ru/view_article.php?ID=200901701. Загл. с экр. Яз. рус.

8 *Кушниренко, А. Г.* докл. ПиктоМир: пропедевтика алгоритмического языка (опыт обучения программированию старших дошкольников) // Большой московский семинар по методике раннего обуч. информатике. — М.: ИТО-РОИ, 2012.

9 *Леонов, А. Г.* Переход от непосредственного управления исполнителями к составлению программ в пропедевтическом курсе информатики // Ярославский педагогический вестник. — 3 № 3. — 2013. — С. 17–30.

10 Исходный код среды исполнения КуМир [Электронный ресурс]. — URL: https://github.com/a-a-maly/kumir2. Загл. с экр. Яз. рус.

11 *Кушниренко, А. Г.* Система программирования КуМир 2.x / А. Г. Кушниренко, М. А. Ройтберг, Д. В. Хачко, В. В. Яковлев // *Труды НИИСИ*. — 2015. — Т. 5, № 1. — С. 142–146.

12 *Кушниренко, А. Г.* Опыт интеграции цифровой образовательной среды КуМир в платформу Мирера // Объединенная конференция "СПО: от обучения до разработки": материалы конференции / Под ред. В. Л. Чёрный. — МАКС Пресс, 2022. — С. 24–30.

13 Спецификация ОГЭ [Электронный ресурс]. — 2022. — URL: https://fipi.ru/oge/demoversii-specifikacii-kodifikatory#!/tab/173801626-5. Загл. с экр. Яз. рус.

14 *Полякова, Н.* Основные итоги государственной итоговой аттестации по общеобразовательным программам основного общего образования в формах основного государственного экзамена на территории Чукотского автономного округа в 2022 году / Н. Полякова, Л. Колыханова, Д. Павлун; Под ред. А. Боленков, Т. Русина. — Анадырь: Отдел оценки и контроля качества образования Департамента образования и науки Чукотского автономного округа, 2022.

15 *Евдокимова, Л. А.* Методические рекомендации для учителей, подготовленные на основе анализа результатов основного государственного экзамена на территории калининградской области / Л. А. Евдокимова, В. А. Зеленцова, др.; Под ред. К. М.И. — Калининград: Изд-во Калининградского областного института развития образования, 2022.

16 *Таммемяги, Т. Н.* Аналитический отчет предметной комиссии о результатах государственной итоговой аттестации выпускников 9 классов по информатике и икт в 2022 году в Санкт-петербурге / Т. Н. Таммемяги, С. Б. Зеленина, Н. Н. Яковлев. — СПб.: ГБУ ДПО «СПб ЦОКОиИТ», 2022.

17 Утверждённая ДЕМО-версию ЕГЭ по информатике 2023 [Электронный ресурс]. — URL: https://doc.fipi.ru/ege/demoversii-specifikacii-kodifikatory/2023/inf_11_2023.zip. Загл. с экр. Яз. рус.

18 Список школьных учебников ФГОС на 2022-2023 [Электронный ресурс]. — 2022. — URL: https://fpu.edu.ru/. Загл. с экр. Яз. рус.

19 *Илькевич, Б. В.* Формирование творческого инженерного мышления в процессе обучения робототехнике / Б. В. Илькевич, К. Б. Илькевич, Т. Г. Илькевич // *Ученые записки университета Лесгафта*. — 2021. — Т. 3, № 193. — С. 150–157.

20 *Шабалин, К. В.* Формирование креативных способностей школьников при выполнении проектов на базе платформы Arduino / К. В. Шабалин // *Педагогическое образование в России*. — 2022. — № 2. — С. 135–140.

21 *Филимонова, Е. В.* Задачный подход к обучению робототехнике с использованием среды trik studio в школьном курсе информатики / Е. В. Филимонова // *Наука и школа*. — 2020. — № 2. — С. 109–121.

22 *Воронин, И. В.* Использование программируемого робота конструктора в образовательных целях / И. В. Воронин // *Ярославский педагогический вестник*. — 2013. — Т. 3, № 4. — С. 98–102.

23 *Бешенков, С. А.* Использование визуального программирования и виртуальной среды при изучении элементов робототехники на уроках технологии и информатики / С. А. Бешенков, М. И. Шутикова, В. Б. Лабутин, В. И. Филиппов // *Информатика и образование*. — 2018. — Т. 5, № 294. — С. 20–22.

24 *Гордиевских, В. М.* Микроконтроллеры Lego ev3 и Arduino uno как технологическая основа для курса робототехники в вузе / В. М. Гордиевских // *Вестник Шадринского государственного педагогического университета*. — 2016. — Т. 3, № 31. — С. 160–163.

25 *Самарина, А. Е.* Возможности конструктора «Scratchduino» для обеспечения занятий по робототехнике на разных ступенях школы / А. Е. Самарина // *Концепт*. — 2016. — № 10. — С. 82–88.

26 *Гейхман, Л. К.* Образовательная робототехника в работе с детьми дошкольного и младшего школьного возраста / Л. К. Гейхман // *Вестник Пермского национального исследовательского политехнического университета. Проблемы языкознания и педагогики.* — 2015. — Т. 14, № 4. — С. 115–126.

27 *Глазов, С. Ю.* Возможности применения платформы Arduino в учебном процессе педагогического вуза и общеобразовательных школ / С. Ю. Глазов, А. Н. Сергеев, В. Л. Усольцев // *Известия ВГПУ*. — 2021. — Т. 163, № 10. — С. 24–29.

28 *Серёгин, М. С.* Использование платформы Arduino в образовательной деятельности / М. С. Серёгин // *Инновационная наука*. — 2019. — № 6. — С. 62–64.

29 *Леонов, А. Г.* Тенденции объектно-ориентированного программирования в разработке системы КуМир / А. Г. Леонов // *Программные продукты и системы*. — 2012. — № 4. — С. 251–254.