Государственное автономное нетиповое образовательное учреждение
Свердловской области «Дворец молодёжи»
Центр цифрового образования «IT-КУБ»

Принята на заседании
научно-методического совета
ГАНОУ СО «Дворец молодёжи»
Протокол № 4 от 03.06.2021 г.

Утверждена директором
ГАНОУ СО «Дворец молодёжи»
А. Н. Слизько
Приказ № 464-д от 04.06.2021г.

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 02487D860079AC25AD4017CA3416FD0841
Владелец: Слизько Александр Николаевич
Действителен: с 20.11.2020 до 08.01.2022

Дополнительная общеобразовательная общеразвивающая программа
технической направленности

«Мобильная разработка»

Стартовый, базовый уровень

Возраст обучающихся: 13–17 лет
Срок реализации: 1 год

СОГЛАСОВАНО:
Начальник центра цифрового
образования «IT-куб»
В. П. Фёдоров

Авторы-составители:
Шанин М. М.,
педагог дополнительного
образования,
Петракова Т.В.,
методист

г. Екатеринбург, 2021 г.

I.Комплекс основных характеристик программы

1. Пояснительная записка

На сегодня разработка программного обеспечения является наиболее востребованным направлением в любых сферах применения. Кроме того, большое развитие мобильных платформ даёт более широкий выбор направлений разработки.

В современном мире Java как платформа является наиболее популярной в связи с тем, что не имеет требований к операционной системе для запуска своих приложений. Кроме того, мобильные устройства на самой популярной ОС Android в большинстве случаев используют приложения, написанные именно на этой платформе. Изучение языка программирования Java по данной программе обучения даёт возможность пользователю мобильного устройства с ОС Android создавать программы в среде разработки, взаимодействующие с элементами графики, аудио и видеофайлами, тестовыми форматами.

Дополнительная общеобразовательная общеразвивающая программа реализуется в сетевой форме. ГАНОУ СО «Дворец молодёжи» является базовой организацией, организация-участник определяется на основании заключенного договор о сетевой форме реализации программ.

Программа «Мобильная разработка» имеет ***техническую направленность***, ориентирована на развитие навыков программирования и проектирования программ под платформу Android.

Основанием для проектирования и реализации данной общеразвивающей программы служит ***перечень следующих нормативных правовых актов и государственных программных документов***:

Федеральный закон «Об образовании в Российской Федерации» от 29.12.2012 N 273-ФЗ;

Приказ Министерства просвещения России от 09.11.2018 г. № 196 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам».

Приказ Министерства труда и социальной защиты Российской Федерации от 05.05.2018 г. № 298н «Об утверждении профессионального стандарта «Педагог дополнительного образования детей и взрослых»»

Постановление Главного государственного санитарного врача РФ от 28.09.2020 г. № 28 «Об утверждении СанПиН 2.4.3648-20 «Санитарно-эпидемиологические требования к организациям воспитания и обучения, отдыха и оздоровления детей и молодежи»

Стратегия развития воспитания в Российской Федерации на период до 2025 года. Распоряжение Правительства Российской Федерации от 29 мая 2015 г. № 996-р;

Письмо Министерства образования и науки РФ от 18.11.2015г. № 09-3242. «О направлении Методических рекомендаций по проектированию дополнительных общеразвивающих программ (включая разноуровневые)»;

Распоряжение правительства РФ от 04.09. 2014 № 1726-р «Об утверждении Концепции развития дополнительного образования детей»;

Федеральный закон от 24.07.1998 № 124-ФЗ «Об основных гарантиях прав ребёнка в РФ».

Приказ Министерства Просвещения Российской Федерации от 03.09.2019 г. №467 «Об утверждении Целевой модели развития региональных систем дополнительного образования детей»

Распоряжение Правительства Свердловской области № 646-РП от 26.10.2018 «О создании в Свердловской области целевой модели развития региональной системы дополнительного образования детей.

Методические рекомендациями для субъектов Российской Федерации по вопросам реализации основных и дополнительных общеобразовательных программ в сетевой форме, утвержденных Министерством просвещения России 28.06.2019г №МР-81/02вн;

Положением об организации реализации дополнительных общеобразовательных общеразвивающих программ в сфере информационных и телекоммуникационных технологий в сетевой форме, а также с применением

электронного обучения и дистанционных образовательных технологий при реализации мероприятий региональных проектов «Цифровая образовательная среда», «Современная школа», «Успех каждого ребенка», обеспечивающих достижение результатов соответствующих федеральных проектов национального проекта «Образование», утвержденное приказом Министерства образования и молодежной политики Свердловской области от 01.04.2020 № 333-Д.

Актуальность программы образовательной программы связана с тем, что в настоящее время широкое распространение получили мобильные устройства: планшеты, смартфоны, и др. Количество мобильных устройств значительно превысило количество настольных компьютеров и ноутбуков, их возможности уже приближаются к возможностям современных компьютеров по быстродействию и объему памяти. Значительное число новых информационных систем и программных продуктов разрабатывается с учетом возможности работы на мобильных устройствах.

Прогностичность программы «Мобильная разработка» заключается в том, что она отражает требования и актуальные тенденции не только сегодняшнего, но и завтрашнего дня, а также имеет междисциплинарный характер, что полностью отражает современные тенденции построения как дополнительных общеобразовательных программ, так и образования в целом.

Также данная программа является базой для перехода на более сложные программы обучения. Обучающиеся приобретают знания по основам IT, которые будут востребованы для дальнейшего обучения в профильных средних специальных и высших учебных заведениях.

Адресат программы: Дополнительная общеразвивающая программа «Мобильная разработка» предназначена для детей в возрасте 13–17 лет, без ограничений возможностей здоровья. Количество обучающихся в группе – 10–14 человек. Состав групп постоянный.

Место проведения занятий: г. Екатеринбург, ул. Красных командиров, 11а.

Условия реализации: Дополнительная общеразвивающая программа «Мобильная разработка» ориентирована на учащихся в возрасте 13–17 лет, которые:

– имеют склонность к алгоритмическому мышлению;

– увлекаются ИТ-технологиями;

– владеют хотя бы одним языком программирования;

– имеют устойчивые знания по школьному курсу математики за 1–8 класс;

– уверенно владеют двоичной системой счисления, переводом чисел между десятичной, двоичной, восьмеричной и шестнадцатеричной системами счисления, сложением и вычитанием в них;

– знают основы логики, теории множеств и операций над ними.

Возрастные особенности группы

Содержание программы учитывает возрастные и психологические особенности детей 13–17 лет, которые определяют выбор форм проведения занятий с обучающимися. Дети этого возраста отличаются внутренней уравновешенностью, стремлением к активной практической деятельности, поэтому основной формой проведения занятий выбраны практические занятия. Ребят также увлекает совместная, коллективная деятельность, так как резко возрастает значение коллектива, общественного мнения, отношений со сверстниками, оценки поступков и действий ребёнка со стороны не только старших, но и сверстников. Ребёнок стремится завоевать в их глазах авторитет, занять достойное место в коллективе. Поэтому в программу включены практические занятия соревновательного характера, которые позволяют каждому проявить себя и найти своё место в детском коллективе.

Также следует отметить, что дети данной возрастной группы характеризуются такими психическими процессами, как изменение структуры личности и возникновение интереса к ней, развитие абстрактных форм мышления, становление более осознанного и целенаправленного характера деятельности, проявление стремления к самостоятельности и независимости, формирование самооценки. Эти

процессы позволяют положить начало формированию начального профессионального самоопределения обучающихся.

Режим занятий, объём общеразвивающей программы: длительность одного занятия для предметных модулей составляет 3 академических часа, периодичность занятий – 1 раз в неделю.

Срок освоения общеразвивающей программы определяется содержанием программы и составляет 1 год.

Формы обучения очная, возможна реализация очно с применением электронного обучения и дистанционных образовательных технологий (Закон №273-ФЗ, гл.2, ст.17, п.2.).

Формы обучения и виды занятий:

Учебный процесс строится таким образом, чтобы экспериментальная и практическая работа преобладала над теоретической подготовкой. Необходимые для работы теоретические сведения находятся на каждом персональном компьютере в специальной папке, даются педагогом перед началом практических занятий. Индивидуальная работа проводится во время практических занятий – при выполнении задания у каждого учащегося возникают свои вопросы. Групповая работа проводится во время теоретических занятий. Каждая тема по программированию сопровождается наглядной демонстрацией работы алгоритма для того, чтобы учащиеся представляли работоспособность алгоритма, а также к чему им нужно стремиться при выполнении поставленной задачи. Учебный процесс организуется на основе постепенного усложнения учебного материала, как теоретического, так и практического.

Программой предусмотрены следующие виды деятельности обучающихся:

- освоение теоретического и практического материала на занятиях;
- разработка индивидуального проекта;
- участие в вебинарах;
- промежуточная аттестация в форме электронного тестирования;

– самостоятельная практическая работа: выполнение домашних заданий, мини-проектов (небольшие приложения, которые реализуются учениками преимущественно на занятиях совместно с учителем с небольшими самостоятельными доработками в качестве домашнего задания).

По одному из вариантов тестов по каждому модулю представлены в приложениях 1, 2, 3, 4.

По типу организации взаимодействия педагогов с обучающимися при реализации программы используются личностно-ориентированные технологии, технологии сотрудничества.

Реализация программы предполагает использование здоровьесберегающих технологий.

Здоровьесберегающая деятельность реализуется:

– через создание безопасных материально-технических условий;

– включением в занятие динамических пауз, периодической смены деятельности обучающихся;

– контролем соблюдения обучающимися правил работы на ПК;

– через создание благоприятного психологического климата в учебной группе в целом.

Объём общеразвивающей программы: 114 часов

Форма организации образовательной деятельности – групповая.

По уровню освоения программа общеразвивающая, разноуровневая (стартовый, базовый уровни). Она обеспечивает возможность обучения детей с любым уровнем подготовки.

«Стартовый уровень» рассчитан на детей в возрасте с 13 лет, проявляющих интерес к аналитической и исследовательской деятельности, IT-технологиям, приобретению навыков программирования.

Зачисление детей на первый год обучения производится без предварительного отбора (свободный набор). К концу стартового уровня обучающиеся приобретут навыки поиска, анализа и использования информации,

а также безопасного поведения в сети Интернет; получат навыки программирования в среде разработки IntelliJ IDEA IDE.

«Базовый уровень» рассчитан на детей, проявляющих интерес к созданию программ на языке Java для решения прикладных задач, желающих совершенствовать свои навыки программирования, имеющих опыт программирования в различных интегрированных средах разработки на языке Java.

Зачисление детей на базовый уровень после завершения стартового уровня производится по результатам успешной сдачи итогового тестирования.

Модуль – структурная единица образовательной программы, имеющая определённую логическую завершённость по отношению к результатам обучения. (Словарь рабочих терминов по предпрофильной подготовке). Каждый модуль состоит из кейсов (не менее двух), направленных на формирование определённых компетенций (hard и soft). Результатом каждого кейса является «продукт» (групповой, индивидуальный), демонстрирующий сформированность компетенций.

Кейс – история, описывающая реальную ситуацию, которая требует проведения анализа, выработки и принятия обоснованных решений (Высшая школа экономики).

Кейс включает в себя набор специально разработанных учебно-методических материалов. Кейсовые «продукты» могут быть самостоятельным проектом по результатам освоения модуля или общего проекта по результатам всей образовательной программы.

Модули и кейсы различаются по сложности и реализуются по принципу «от простого к сложному».

Педагогическая целесообразность программы

Программа реализует профориентационные задачи, обеспечивает знакомство с современными профессиями в сфере IT.

Стартовый уровень предполагает использование и реализацию общедоступных и универсальных форм организации материала, минимальную сложность предлагаемого материала для освоения содержания программы.

Базовый уровень предполагает использование и реализацию таких форм организации материала, которые допускают освоение специализированных знаний и языка, гарантированно обеспечивают трансляцию общей и целостной картины в рамках содержательно-тематического направления программы.

Осваивая данную программу, обучающиеся будут овладевать навыками востребованных на рынке труда. Практически для каждой перспективной профессии будут полезны знания и навыки, рассматриваемые в данной программе.

2. Цели и задачи программы

Цель программы: формирование технической грамотности средствами приобщения обучающихся к разработке программ под современную платформу Android.

Для успешной реализации поставленной цели необходимо решить ряд педагогических, развивающих и воспитательных задач:

Обучающие:

– расширить знания о современных и популярных платформах;

– обучить языку программирования Java, языку разметки XML;

– обучить объектно-ориентированному подходу в проектировании и разработке программного обеспечения;

– познакомить с архитектурой приложения под Android;

– обучить программированию технических устройств.

Развивающие:

– сформировать алгоритмическое мышление;

– развить логическое и техническое мышление;

– сформировать навыки работы с информацией;

– сформировать умение самостоятельно решать поставленную задачу;

– сформировать умение излагать мысли в чёткой логической последовательности, отстаивать свою точку зрения, анализировать ситуацию и самостоятельно находить ответы на вопросы путём логических рассуждений;

– сформировать умение планировать свои действия с учётом фактора времени, в обстановке с элементами конкуренции, предвидеть результат и достигать его, при необходимости вносить коррективы в первоначальный замысел.

Воспитательные:

– воспитать этику групповой работы, отношений делового сотрудничества, взаимоуважения;

– развивать основы коммуникативных отношений внутри проектных групп и в коллективе в целом;

– воспитать упорство в достижении результата;

– сформировать целеустремлённость, организованность, ответственное отношение к труду, толерантности и уважительное отношение к окружающим.

3. Содержание общеразвивающей программы

II. Стартовый уровень

Таблица 1

№ п/п	Название раздела, темы	Кол-во часов			Формы аттестации/ контроля
		Всего	Теория	Практика	
Модуль 1. Компьютерная грамотность		**6**	**2**	**4**	
1.1	Разрабатывается и утверждается организацией-участником				
Модуль 2. Основы программирования.		**24**	**7**	**17**	
2.1	Вводное занятие. Инструктаж по ТБ. Введение в программу. Среда разработки	3	1	2	Знакомство. Опрос. Инструктаж по ТБ
2.2	Арифметика. Примитивные типы данных. Логика. Операции отношения и логические операции.	6	2	4	Опрос
2.3	Условные конструкции. Блоки	3	1	2	Опрос
2.4	Итеративные конструкции. Массивы. Списки.	6	2	4	Опрос. Практическая работа
2.5	Методы (функции). Видимость переменных. Рекурсия.	3	1	2	Опрос
2.6	Практикум. Контрольное тестирование по модулю	3	–	3	Тест № 1
Модуль 3. Объектно-ориентированное программирование		**30**	**10**	**20**	
3.1	Классы и объекты	3	1	2	Опрос
3.2	Классы: конструкторы, статические методы. Начальные приёмы тестирования и отладки	3	1	2	Опрос
3.3	Android. Структура. Активности. Интерфейс пользователя. Язык разметки XML.ООП.	15	6	9	Опрос. Практическая работа
3.4	Намерения. Фрагменты.	6	2	4	Опрос. Практическая работа
3.5	Практикум. Контрольное тестирование по модулю	3	0	3	Тест № 2
Итог стартовый уровень		**60**	**19**	**41**	

III. Базовый уровень

Модуль 4. Основы программирования Android-приложений		**24**	**6**	**18**	
4.1	Ввод, вывод и исключения	3	1	2	Опрос
4.2	Внутренние классы в обработке событий	3	1	2	Опрос
4.3	Параллелизм и синхронизация. Потоки	6	2	4	Опрос. Практическая работа
4.4	Двумерная графика в Android-приложениях	3	1	2	Опрос
4.5	Реализация графики на основе SurfaceView	6	1	5	Опрос. Практическая работа
4.6	Практикум. Контрольное тестирование по модулю	3	0	3	Тест № 3
Модуль 5. Алгоритмы и структуры данных		**21**	**6**	**15**	
5.1	Массивы. Списки. Алгоритмы сортировки. Алгоритм двоичного поиска. Деревья	6	2	4	Опрос. Практическая работа
5.2	Адаптеры в Android	3	1	2	Практическая работа
5.3	Ассоциативные массивы	3	1	2	
5.4	Реляционная модель данных. СУБД. Введение в SQL	6	2	4	Опрос. Практическая работа
5.5	Практикум. Контрольное тестирование по модулю	3	0	3	Тест № 4
6.	**Консультации по ИП**	**6**	**0**	**6**	
7.	**Итоговая защита ИП**	**3**	**0**	**3**	
Итог базовый уровень		**54**	**12**	**42**	
Итого		**114**	**31**	**83**	

Содержание учебного плана

Стартовый уровень

Модуль 1. Компьютерная грамотность

Разрабатывается и утверждается организацией-участником

Модуль 2. Основы программирования

Тема 2.1. Среда разработки

Цель. Закрепление на практике базовых навыков работы с Java-программой и основ синтаксиса Java.

Необходимые знания. Представление о том, что такое программа и как она выполняется на компьютере, навыки пользователя ПК.

Среда разработки IntelliJ IDEA/Eclipse. Шаблон программы на Java с функцией main(). О среде разработки IntelliJ IDEA и Eclipse. Понятие проекта. Порядок создания, компиляции, сборки и запуска приложения. Порядок установки среды разработки на домашнем компьютере.

Упражнение 1.1.1. Выполнить первую Java-программу «Здравствуй, мир»:

```
public static void main(String[] args) {
println(«Hello, user!»);        }
```

На её примере объяснить порядок создания, компиляции и сборки проекта на языке Java, порядок запуска проекта на выполнение.

Разбор Java-проекта. Уяснение факта, что имя класса должно совпадать с именем файла. Объяснение понятий, связанных с ООП, не проводится. Это будет рассмотрено в рамках 2-го модуля программы. На данный момент ученикам предлагается сконцентрироваться на содержимом метода main(). Понятие пакета (package). Роль метода main(). Комментарии. Демонстрация типичных ошибок, возникающих при компиляции и выполнении Java-программ.

Упражнение 2.1.2. Написать и отладить программу, которая выводила бы информацию об ученике в следующем виде: Ф.И.О., школа, класс.

Тема 2.2. Примитивные типы данных. Арифметика. Операции отношения и логические операции.

Цель. Рассмотреть примитивные типы данных, арифметические выражения и

операторы, операторы присваивания, преобразования типов, поразрядные операции, логические выражения.

Необходимые знания. Понятия «бит» и «байт»; двоичная, восьмеричная, шестнадцатеричная системы счисления; перевод чисел из одной системы счисления в другую, понятие переменной.

Переменные и **константы**. Данные, обрабатываемые компьютером; описание переменной; задание начального значения переменной; имя переменной. Понятие «тип переменной». Что относят к примитивным типам. Понятие константы. Ключевое слово final.

Целочисленные типы данных. Разновидности типа «целое»: int, short, long, byte. Размер каждого из типов (количество байт), диапазон представления (минимальное и максимальное значения).

Как задать значение константы в десятичной, двоичной, восьмеричной, шестнадцатеричной системе счисления. Как указать, что константа относится к типу long. Вывод на печать данных целого типа. Ввод данных целого типа.

Упражнение 2.2.1. Написание простейших программ, объявляющих переменные целого типа, присваивающих им значения. Вывод этих значений на печать. Наблюдение за поведением компилятора, когда переменной присваивается заведомо некорректное значение или выходящее за пределы диапазона для данного типа.

Программа, манипулирующая представлением целого числа в различных системах счисления. Например, задается число в тексте программы в одной системе счисления, а выводится на печать – в другой.

Типы с плавающей точкой. Типы float, double. Применение суффиксов для указания типа числовых констант.

Упражнение 2.2.2. Написание простейших программ, объявляющих переменные вещественного типа, присваивающих им значения. Вывод этих значений на печать. Наблюдение за поведением компилятора, когда переменной присваивается заведомо некорректное значение или выходящее за пределы диапазона для данного типа.

Арифметические выражения и операторы, операторы присваивания. Сложение, вычитание, умножение, деление, остаток от деления (%), инкремент (++), декремент (--). Префиксная и постфиксная запись инкремента и декремента, уяснение отличия между ними. Операторы присваивания (=, +=, -= и т.д.). Скобки. Рассмотрение подробной таблицы со столбцами: название оператора, форма записи, порядок выполнения. Примеры программ, демонстрирующих применение приоритетов.

Данные типа char. Дать общее представление о кодировке Unicode и UTF-16. Дать рекомендацию по возможности пользоваться не символами, а символьными строками.

Упражнение 2.2.3. Написание простейших программ, демонстрирующих выполнение каждого изученного оператора. Объяснение результатов её выполнения.

Задание 2.2.1. Задачи на определение по значению числа, к какому целочисленному типу оно относится. К какому типу относятся целочисленные константы: 120, 21L, 015, 0b101, 1_000_000, 0x84? Чему равны значения этих констант в десятичной системе счисления?

Задание 2.2.2. Пусть a=7, b=5, c=4. Какие значения будут иметь эти переменные в результате выполнения последовательности операторов? Подсчитать «вручную» и затем проверить результат, написав и запустив программу:

- c=a*b; b=b+1; a=c-b; b=(b+c)/2; c=++b;
- c=a%b; b=a; a*=b; c=a*(b-3); a+=c+b;
- a=b*c; b%=c; b=(a+c)%2; a+=b; c=a*b.

Задание 2.2.3. Определить приоритет операторов и порядок их выполнения:

- 5*6-8/2/2;
- 4-3*3/10;
- 19%6/2*7;
- 1<<2*3+5^4 (в таких выражениях лучше ставить скобки);
- a=5; b=2; a=b*a++.

Тип данных boolean. Логические значения true и false. Несовместимость типа boolean с int. Отметить, что приведение логических значений к целым и наоборот

невозможно.

Логические операции и операции отношения. Операторы отношения: >, <, >=, <=, !=, ==. Уяснение понятия значения операции отношения как ИСТИННО или ЛОЖНО. Логические операции: логическое И, логическое ИЛИ, логическое НЕ. Тернарная операция?:.

Выражения и **операции**. По итогу изучения различных операций рассмотрение понятия выражения в языке программирования; знаки операций; знаки-разделители. Классификация операций по количеству операндов: унарные и бинарные. Классификация операций по типу: арифметические, логические, присваивания, отношения и др.

Упражнение 2.2.4. Программа, демонстрирующая выполнение логических операций и операций отношений. Объяснение результатов её выполнения.

Задание 2.2.4. Задачи на «ручное» написание логических выражений средствами языка Java:

- x лежит вне отрезка [a, b];
- x принадлежит отрезку [a, b] или отрезку [c, d];
- x лежит вне отрезков [a, b] и [c, d];
- целое a является нечётным числом;
- целое a является трёхзначным числом, кратным пяти;
- из чисел a, b, c меньшим является c, а большим b;
- среди чисел a, b, c, d есть взаимно противоположные;
- среди целых чисел a, b и c есть хотя бы два чётных;
- из отрезков с длинами a, b, c можно построить треугольник;
- год, заданный числом a, является високосным;
- год, заданный числом a, не является високосным;
- число a является простым;
- среди целых чисел a, b, c есть хотя бы два нечётных;
- отрезки длиной a, b и c могут образовать прямоугольный треугольник.

Тема 2.3. Условные конструкции. Блоки

Цель. Изучить внутреннюю логику работы условных конструкций; приобрести

навыки их использования в различных формах, предусмотренных синтаксисом языка. Закрепить навыки написания всех ранее изученных операторов путём написания и вычисления выражений.

Необходимые знания. Здесь и далее необходимы знания в объёме предыдущих тем.

Область действия блоков. Фигурные скобки для выделения блока. Вложенность блоков. На данный момент рассмотреть только ограничение на объявление переменных с одинаковым именем в одном и том же или вложенных блоках.

Конструкция if-else.Синтаксис оператора:

if (cond_expression) TRUE_statement;

или

if (cond_expression) TRUE_statement;

else FALSE_statement;

Разъяснить, что statement – это только один оператор или блок. Фундаментальное правило для сложных ветвлений, реализуемых с помощью вложенных конструкций if-else: else относится к ближайшему if, не имеющему else.

Конструкция switch-case. Синтаксис. Что может быть в качестве метки case.

Мотивировка использования конструкции как упрощение сложных ветвлений. Логика выполнения, объяснение роли ключевых слов break и default в конструкции switch-case.

Упражнение 2.3.1. Небольшие фрагменты кода, иллюстрирующие использование операторов ветвления, приоритетов вычисления операторов в выражении. Ускоренное вычисление логических выражений – прекращение вычислений, когда результат уже ясен.

Задание 2.3.1. Написать собственный пример на использование операторов ветвления. Например, нахождение максимума, минимума среди нескольких введённых переменных.

Подготовка заданий. При подготовке заданий на написание учеником программ можно использовать автоматическую систему тестирования (например,

http://informatics.mccme.ru). Так же можно подготовить задачу самостоятельно с использованием модульного тестирования.

Рассмотрим пример тестов для задачи минимума трёх чисел:

```
public void testEasy() {
runTest(“4 5 7\n”, “4”);
runTest(“5 7 3\n”, “3”);
runTest(“4 1 7\n”, “1”);}
```

Можно добавить много тестов, сгенерированных циклом:

```
public void testEasy() {
for (int i = 0; i < 30; ++i) {
for (int j = i; j < 30; ++j) {
for (int t = j; t < 30; ++t) {
runTest(“” + i + “ ” + j + “ “ + t “ +\n”, “” + i);
runTest(“” + j + “ ” + i + “ “ + t “ +\n”, “” + i);
runTest(“” + j + “ ” + t + “ “ + i “ +\n”, “” + i);
}}        }        }
```

Тема 2.4. Итеративные конструкции. Массивы. Списки.

Цель. Изучить внутреннюю логику работы итеративных конструкций; приобрести навыки их использования в различных формах, предусмотренных синтаксисом языка. Изучить оператор for, for each, одномерные массивы. Освоение понятия «список» как структуры данных, изучение библиотечного класса LinkedList.

Цикл с предусловием while. Синтаксис. Объяснение логики работы, пример использования.

Цикл с постусловием do-while. Синтаксис. Объяснение логики работы, пример использования. Уяснение ключевого отличия от цикла while с предусловием: цикл с постусловием выполняется хотя бы один раз.

Операторы прерывания логики управления программой. Безусловные операторы перехода break, continue.

Упражнение 2.4.1. Небольшие фрагменты кода, иллюстрирующие использование операторов цикла (без использования массивов). Например,

вычисление НОД по алгоритму Евклида.

Задание 2.4.1. Написать собственный пример на использование операторов цикла и операторов безусловного перехода. Например, проверка числа на то, что оно является простым.

Массивы. Определение массива как совокупности элементов одного и того же типа, расположенных вплотную друг за другом в памяти. Объявление массива двумя способами. Подчеркнуть необходимость создания массива с помощью new(). Значения, инициализируемые массивом по умолчанию.

Инициализация массива без new() – инициализация массива при объявлении. Доступ к отдельным элементам массива. Определение количества элементов в массиве через свойство length.

Цикл for. Синтаксис. Логика работы, роль каждой из составных частей. Частные формы записи оператора for: отсутствует инкрементальное выражение; отсутствует инкрементальное выражение и начальное выражение. Уяснение связи между for и while, эквивалентная запись for через while. Примеры некорректного использования операторов цикла, приводящего к зацикливанию. Вложенные циклы for.

Цикл for each. Синтаксис. Преимущества его применения при работе с массивами в сравнении с обычным for. Отметить, что переменная в цикле for each перебирает не индексы массива, а сами элементы массива.

Упражнение 2.4.2. Фрагменты кода, иллюстрирующие на одномерном массиве решение задач нахождения максимального, минимального.

Задание 2.4.2. Написать программу по обработке массива с выводом на экран полученного результата:

- поиск заданного элемента простым перебором;
- переворот массива «задом наперёд» без использования вспомогательного массива;
- вычисление суммы элементов массива;
- нахождение самого часто повторяющегося числа среди элементов массива;
- нахождение среднего арифметического числа элементов массива;
- заполнить массив числами арифметической прогрессии по заданной

учителем формуле.

Понятие Список. Разновидности структур данных, основанных на списках. Список как базовая структура данных. Классификация структур данных, основанных на списках: по дисциплине обслуживания (стеки, очереди), по структуре (односвязные и двусвязные списки).

Очередь как реализация принципа FIFO (первым пришёл – первым вышел). Работа с очередью осуществляется с обоих концов.

Стек как реализация принципа LIFO (последним пришёл – первым вышел). Работа со стеком осуществляется с одного конца. Обратить внимание на следующее различие: работать с очередью записывающий и считывающий процессы, как правило, могут асинхронно, т. е. один процесс пишет, другой видит, что что-то появилось, и забирает что бы то ни было. Считывание же вершины стека может происходить не в любой ситуации, а только если вершина стека находится в определённом состоянии. Если это состояние не достигнуто, продолжается запись в вершину стека.

Двусвязные и **односвязные** списки. Мотивация введения второго указателя: облегчение навигации по списку в обратном направлении, т. к. сдвиг в односвязном списке назад требует больших трудозатрат, в частности, перемещение назад от конца списка требует прохода от начала по всему списку.

Библиотечный класс LinkedList, Queue, Stack. Практическое занятие по библиотечному классу LinkedList, реализующему связные списки.

Класс LinkedList как реализация связного списка. Методы, специфичные для связного списка: addFirst, addLast,getFirst,getLast,removeFirst, removeLast.

Класс Stack и **интерфейс Queue**. Объяснение, что LinkedList – это одна из реализаций интерфейса очередь, привести примеры других реализаций.

Упражнение 2.4.3. Пример применения классов LinkedList, Queue и Stack.

Задание 2.4.3. Реализовать интерфейс очереди с ограничением на количество. Очередь должна игнорировать добавление элемента, если её размер достиг максимума. Требуется реализовать методы poll (извлечь первый элемент из очереди), peek (считать первый элемент из очереди) и add (добавить элемент в конец очереди).

Тема 2.5. Методы (функции). Видимость переменных. Рекурсия.

Цель. Усвоить фундаментальное понятие функции в программировании и проектировании программного обеспечения на примере методов Java; приобрести навыки их использования. Рассмотреть видимость переменных. Приобрести представление о рекурсивных алгоритмах.

Основные понятия. Определение функции как логически самостоятельной именованной части программы, которой могут передаваться параметры, и которая может возвращать какое-то значение.

Определение функции. На примере объяснить понятия: тело метода, тип возвращаемого значения. Список формальных аргументов, список фактических аргументов. Методы с типом void и методы с пустым списком аргументов.

Упражнение 2.6.1. На примере продемонстрировать ситуации, когда функции необходимы. Привести в качестве примера функции println и readInt.

Реализовать собственную функцию для считывания и вывода массива (int[] readIntArray(int length) и void printArray(int[] a, char delimiter)) с использованием уже существующих функций.

Область видимости переменных. Обзорная классификация переменных по области видимости: область класса, область метода, область блока.

Понятие рекурсивного вызова в программировании. Умение видеть в задаче ситуацию «часть подобна целому» и осмыслить тот факт, что единовременно работает множество экземпляров рекурсивной функции. Формальные и фактические параметры, локальные и глобальные переменные при перемещении по стеку рекурсивных вызовов.

Линейная и **ветвящаяся** рекурсия. Рассмотрение примеров функций с линейной и ветвящейся рекурсией. Иллюстрация «вручную» работы рекурсивной программы (для каждого экземпляра функции подписать передаваемые аргументы и возвращаемые результаты):

- с линейной рекурсией в виде цепочки;
- с ветвящейся рекурсией в виде дерева рекурсивных.

Демонстрация на полученных иллюстрациях необходимости нерекурсивной

заглушки.

Упражнение 2.5.2. Пример линейной рекурсии. Вычисление степени n числа x. Описание рекурсивной функции. Формулировка окончания рекурсии. Подробная иллюстрация процесса вычисления. Демонстрация ограничений рекурсивных программ (ошибка переполнения стека).

Упражнение 2.5.3. Пример ветвящейся рекурсии – поиск файла в дереве директорий.

Задание 2.5.1. Для заданных программ определить вид рекурсии, нарисовать иллюстрацию рекурсивных вызовов, «вручную» определить количество рекурсивных вызовов.

Практикум

Практическое занятие по темам модуля. Конкретное содержание определяется учителем.

Контрольное тестирование по модулю

Электронный тест охватывает все темы и в большей части направлен на оценку практических знаний и навыков учеников.

Модуль 3. Объектно-ориентированное программирование

Тема 3.1. Классы и объекты.

Цель. Уяснить различие между процедурным (структурным) и объектным подходом к программированию; освоить понятия «класс», «объект».

Основные понятия. Взгляд на ООП как более естественное по сравнению с процедурным подходом отражение в программировании реалий окружающего мира и отношений между ними. Интуитивно-понятное объяснение понятий: класс, объект (экземпляр класса), переменные (поля) класса, метод класса. Иллюстрация на примерах окружающего мира и примерах школьной математики. Сопоставление каждому понятию некоего утверждения, афористично и емко раскрывающего его суть:

Объект – «всё есть объект», класс – «каждый объект имеет тип», переменные – «каждый объект имеет собственную память, отличную от других объектов», метод – «все объекты определенного типа могут принимать одинаковые сообщения»,

программа на ОО-языке – «связка объектов, говорящих друг другу что делать, посылаю сообщения или, иными словами, вызывая методы».

Описание протокола класса. Ключевое слово class как начало описания нового типа данных.

Описание полей класса. Метод класса, его аргументы и возвращаемое значение. Описание метода в протоколе класса.

Создание объекта класса с помощью оператора new. Обращение к полям класса через.

Вызов метода через переменную – объект собственного класса. Интерпретация метода как посылки сообщения объекту.

Инкапсуляция, **наследование**, **полиморфизм**. Общее понятие парадигм ООП инкапсуляция, полиморфизм и наследование на примерах из жизни.

Более подробно остановиться на инкапсуляции: уяснение целесообразности скрытия внутреннего строения объекта и ограничения доступа к его полям – взаимодействие с объектом организуется только через его методы. Взгляд на инкапсуляцию как на средство защиты целостности данных объекта: объект «Интервал» (левая граница не должна стать больше правой).

Упражнение 3.1.1. Проектирование и реализация простейшего класса, описывающего рациональную дробь. Описание полей (числитель, знаменатель). Объяснение, почему эти поля должны быть закрытыми (исключение деления на ноль). Автоматическая генерация getter/setter в классе с помощью среды разработки.

Обзор классов, соответствующих примитивным типам. Знакомство с классами Integer, Character, Float и т. д.

Упражнение 3.1.2. Разбор примера с применением классов Integer, Character, Float и т. д.

Упражнение 3.1.3. Проектирование классов и их взаимодействия для проведения урока. Например, Учитель готовит Задание, Ученик выполняет Задание и получает Решение, Учитель проверяет Решение и ставит Оценку и т. п.

Задание 3.1.1. Придумать примеры классов и соответствующие им примеры объектов, полей и методов.

Тема 3.2. Классы: конструкторы, деструкторы и статические методы. Начальные приемы тестирования и отладки.

Цель. Познакомиться с примерами java-кода, описывающего классы, приобрести первый опыт проектирования и реализации полноценного, логически завершённого класса, освоить понятие перегрузки методов, способы инициализации данных в программе на Java. Ознакомиться с приёмами, позволяющими найти ошибку в программе и предотвратить ошибки при внесении изменений в программу на примерах со строками.

Конструкторы и **деструкторы**. Конструкторы и деструкторы в Java и их использование. Разновидности конструкторов: без аргументов; с аргументами. Понятие конструктора по умолчанию. Особенности автоматической генерации конструктора по умолчанию.

Перегрузка методов. Уяснение возможности иметь несколько методов с одним и тем же именем, но разными сигнатурами (наборами аргументов) на примере конструктора класса. Распространение концепции перегрузки на любой метод Java. Пример перегрузки конструктора и обычного метода. Необходимость уникального списка типов аргументов для различения перегруженных методов. Уникальность как совокупность количества аргументов, их типов и порядка следования. По типу возвращаемого значения метод не может быть перегружен!

Ключевое слово this. Уяснение смысла ключевого слова this как ссылки на текущий объект. Примеры типового использования: возврат в return ссылки на текущий объект; различение имени локальной переменной и имени поля класса при их совпадении.

Спецификаторы доступа. Ключевое слово public и интерфейсный доступ к объектам класса. Ключевое слово private для описания закрытых полей и методов класса. Понятие доступа класса. Ключевое слово public по отношению к классу (а не члену класса) и его смысл.

Упражнение 3.2.1. Продолжение разработки класса, описывающего рациональную дробь. Реализация конструкторов. Реализация методов экземпляра: модификация числителя; модификация знаменателя; проверка дроби на

правильность; выделение целой части; выделение дробной части; вывод дроби на печать; результат деления в виде десятичной дроби; сложение с дробью; умножение на дробь; деление на дробь; вычитание дроби.

Статические методы класса. Ключевое слово static и статические члены класса. Интерпретация статических методов как методов класса в отличие от остальных методов – методов экземпляра. Обращение к статическим членам класса через имя класса (а не переменную – объект).

Инициализация различных **типов данных**. Инициализация полей класса в конструкторе. Явная и неявная инициализация примитивных типов и объектных ссылок. Инициализация статических данных. Инициализация массива, примеры.

Упражнение 3.2.2. Разбор примера: класс для демонстрации значений типов по умолчанию, показать, что поля класса инициализируются еще до вызова конструктора, в каком бы порядке не стояли в классе описания полей и тело конструктора. Разбор примера демонстрации создания и инициализации многомерных массивов.

Задание 3.2.1. Написать функцию run(), тестирующую класс «Рациональная дробь». Функция должна создавать экземпляры класса, выполнять реализованные в классе методы и выводить результат. Реализация статических методов класса: сложение дробей; вычитание дробей; умножение дробей; деление дробей.

Модифицируйте функцию print, чтобы вывод при необходимости был в виде смешанной дроби, убедитесь в корректности работы с отрицательными числами.

Модифицируйте конструктор дроби, чтобы все хранимые дроби были несократимы.

Отладочный вывод и логирование. Вывод значений переменных на экран для последующего анализа их значений. Понятие логирования и его области применения. Вывод отладочных данных с помощью android.util.Log и просмотр этих логов в (Window | Show View | Other... | Android | LogCat.).

Использование отладчика. Пошаговое выполнение программы и просмотр переменных. В отличие от отладочного вывода не требует изменения программы. Однако не позволяет легко просматривать сложные конструкции (сумма элементов

массива, диагональ матрицы).

Breakpoint для перехода на конкретную строчку программы (в том числе с указанием условия остановки).

Использование макроса (функции) assert. Проверка инвариантов в программе с помощью макроса assert. Самопроверка программы при каждом её запуске. Проверка входных параметров функции (на примере sqrt или log). Для включения функций assert необходимо установить параметр -ea в Run Configurations -> Arguments -> VM arguments (по умолчанию assert игнорируются, чтобы не замедлять программу).

Тестирование отдельных функций (модульное тестирование). Реализация отдельной функции, проверяющей правильность работы некоторой части программы. Отличие от assert. Выделение легко тестируемой (например, не читающей из входного потока) части программы. Написание модульного теста с помощью библиотеки JUnit.

Работа со строками. Краткий обзор классов String, StringBuffer, StringBuilder. Обратить внимание на то, что объекты класса String являются неизменяемыми.

Упражнение 3.2.3. Разобрать примеры программ, продемонстрировать изученные приёмы тестирования.

Задание 3.2.2. Написать программу по обработке строк. При отладке использовать отладчик и изученные приёмы тестирования.

Тема 3.3. Архитектура приложений Андроид. Активности. Интерфейс пользователя. Язык разметки XML.

Цель. Познакомиться со средой разработки Android-приложений, рассмотреть общую структуру Android-приложения; создать первое приложение.

Рассмотреть способы задания расположения элементов управления на экране устройства; уяснение необходимости задания расположения универсально для многих устройств. Изучить общую структуру языка XML: понятие тэга, опций тэга, вложенных тэгов, сокращенных тэгов (без закрытия), комментариев.

Освоить приемы ООП-проектирования Android-приложений, изучение простейших диаграмм UML.

Знакомство со средой разработки приложений под Android. О среде разработки Android Studio. Порядок создания, компиляции, сборки и запуска в среде. Порядок

установки IDE и эмулятора для разработки приложений под Android на домашнем компьютере.

Общая структурная **схема** приложения под Android. Разбор и комментирование схемы (используется схема из developer.android.com). Жизненный цикл Android-приложения.

Активности (Activity) и их жизненный цикл в Android. Создание Activity. Жизненный цикл Activity. Стеки Activity. Состояния Activity.

Отслеживание изменений состояния Activity. Класс Activity, методы onCreate, onStart, onPause, onStop, onRestart, onResume, onDestroy.

Упражнение 3.3.1. Разбор кода простейшего Android-приложения, иллюстрирующего общую схему, его запуск. В материалах приведён пример приложения, решающего линейное уравнение.

Задание 3.3.1. Модифицировать приложение так, чтобы оно решало квадратное уравнение, а также корректно обрабатывало ситуацию нулевых коэффициентов.

Примеры использования XML. XML-формат, одновременно понятный человеку и компьютеру, используется для записи конфигурации программ, для передачи данных по сети, хранения данных и другого.

Привести аналогию языка разметки с описанием полей класса. Использование XML в программировании Android-приложений.

Структура документа и комментарии. Пролог, корневой элемент, остальные элементы и их разметка, секция CDATA. Способ записи комментария в XML-документе.

Упражнение 3.3.2. Разобрать пример задания информации в xml (например, описание рецепта или геометрической фигуры).

Описание ресурсов Android с помощью XML. Описание, назначение файлов res/layout/fragment_main.xml и res/values/strings. Обоснование, для чего используется описание всех строк в отдельном файле (удобный поиск и редактирование всех строк, локализация продукта). Основные элементы интерфейса и их описание в файле fragment_main.xml: EditText, TextView.

Разметки (Layouts). LinearLayout и его ориентации, TableLayout. Описание

Layout в xml-файле. Вложенные Layout.

Упражнение 3.3.3. Расположить кнопки в форме буквы П с помощью LinearLayout.

Представления (Views). Примеры представлений: TextView, EditText и ProgressBar. Описание представлений в конфигурационном файле. Поиск представлений по их идентификатору findViewById(R.id.name).

Упражнение 3.3.4. Разобрать работу функции println в классе Program в testbed. Разобрать первые две строки processButton для получения входных и выходных данных в testbed.

Задание 3.3.2. Создать xml, описывающий список вещей, которые ученик носит в школу. Каждой вещи должен соответствовать отдельный тэг с её дополнительными свойствами (цвет, название, автор и т. п.).

Задание 3.3.3. Добавить progress bar на экран и передать экземпляр в конструктор класса Program. Расширить Program приватным методом, устанавливающим прогресс обработки. Воспользуйтесь классом BigInteger для поиска первой цифры числа n! (факториал n). В процессе вычисления обновляйте progressbar.

Упражнение 3.3.5. Разбор примера проектирования класса Дробь. Демонстрация преимуществ ОО-подхода.

Принципы SOLID. Принцип единственной обязанности (Single Responsibility Principle). Принцип открытости/закрытости (Open-Closed Principle). Принцип подстановки Барбары Лисков (Liskov Substitution Principle, кратко – LSP). Принцип разделения интерфейса (Interface Segregation Principle). Принцип инверсии зависимостей (Dependency Inversion Principle).

Диаграммы UML. UML как язык графического описания для объектного моделирования в области разработки программного обеспечения. Знакомство с инструментальной средой для создания UML-диаграмм. Диаграмма классов как подмножество диаграмм UML: основные элементы и синтаксис.

Упражнение 3.3.6. Разбор примера проектирования игры-квеста. Демонстрация на примере выделения сущностей, методов и полей классов.

Минипроект 3.1. Самостоятельное проектирование UML-диаграммы классов

приложения согласно заданию. Выдача заготовки проекта для дальнейшей реализации мини-проекта.

Тема 3.4. Намерения

Цель. Изучить понятие интерфейса, возможности наследования классов и приобрести навыки их использования; уяснить различие между отношениями наследования и вложенности.

Наследование. Наследование классов как создание новых классов на основе уже существующих. Отношение «является» между классами. Расширение базового класса новыми полями и методами в классе наследнике, характерными именно для производного класса. Примеры наследования (человек – студент, четырёхугольник – ромб и пр.). Взгляд на наследование классов как на естественную возможность повторного использования кода и независимого расширения библиотек классов. Интерфейс как задание набора методов всех классов, его реализующих (задание шаблона). Более подробное использование будет рассмотрено в теме «Полиморфизм».

Инициализация базового класса. Синтаксическое описание наследования классов и реализации интерфейсов, ключевое слово extends и implements. Подобъект базового класса внутри объекта производного класса и необходимость его инициализации. Вызов конструктора базового класса как часть конструктора производного класса. Гибкое манипулирование вызовами конструкторов с помощью ключевого слова super.

Защищенные члены класса. Ключевое слово protected и пример мотивации его использования (приведён в приложении): открытый член класса для доступа из производных классов и закрытый для доступа извне.

Упражнение 3.6.1. Разобрать пример с описанием классов, наследованием, переопределением метода, доступами и т. д.

Приведение к **базовому типу**. Уяснение ключевого правила: объект производного класса является объектом базового класса, но не наоборот. Пример использования этого правила в Java: объект неявно приводится к типу своего родителя. Стоит сообщить об этом, более подробно это будет разобрано в теме

«полиморфизм».

Ключевое слово final. Применение final к примитивным типам данных: значение переменной не может быть изменено. Применение final к объектам: объектная ссылка не может быть изменена, но содержимое объекта – может. Применение final к аргументу метода. Применение final к методу: метод не может быть переопределён в производном классе. Применение final к классу: у этого класса не может быть наследников.

Контекст. **Намерения** (Intents). Что такое Context и его использование. Явные и неявные намерения.

Создание нескольких Activity (и экранов) в одном приложении. Регистрация их в AndroidManifest.xml. Создание xml файла с описанием вида экрана в папке res/layout. Метода setContentView для загрузки xml файла из res/layout. Создание Intent для перехода на другой экран (метод startActivity).

Упражнение 3.4.2. Разобрать аналогичный пример, где одно из наследований заменено вложенностью классов. Объяснить различие в реализации.

Задание 3.4.1. В соответствии с заданным вариантом задания разработать собственный класс, аналог которого общеупотребителен в окружающем мире или в математике и имеет хорошо знакомое поведение. Для школьника задание должно быть сформулировано учителем.

Возможные варианты заданий:

Реализовать класс «Билет на общественный транспорт». Реализовать классы-наследники: автобусный билет, билет на метро, единый билет (на несколько видов транспорта одновременно). Должны быть реализованы методы для прохода в автобус и метро.

Реализовать класс «Переводчик» для перевода из десятичной системы счисления в другую. Его наследники: в двоичную, в римскую.

Реализовать класс «Нота». Поля: перечислимый тип для ноты (до, ре, ми, фа, соль, ля, си); перечислимый тип для тональности (контроктава, большая, малая, первая, вторая, третья, четвертая); перечислимый тип для знака альтерации (отсутствует, диез, бемоль). Конструктор проверяет корректность задания ноты – не

может быть до-бемоль, фа-бемоль, ми-диез, си-диез. Класс Нота обязательно сделать реализующем интерфейс Comparable и написать метод compareTo для сравнения нот: нота с меньшей высотой предшествует (меньше) ноты с большей высотой. Реализовать класс «Музыкальный интервал» как контейнер, содержащий две ноты. Конструктор должен проверять, что первая нота ниже (меньше) второй ноты. Метод: вычисление длины интервала в тонах. Реализовать и как метод класса, и как метод экземпляра.

Реализовать иерархию классов: «Шахматная фигура», «Пешка», «Слон», «Конь», «Ладья», «Король», «Ферзь». Для каждого производного класса должна быть реализована своя функция «Сделать ход». Члены класса: поле, на котором стоит Фигура (реализовать приватный класс «Поле»); цвет фигуры. Если ход возможен, поле фигуры меняется. Если невозможен – не меняется. Поле характеризуется буквой (a-h и цифрой 1–8). В конструкторах обеспечить контроль ввода (чтобы не поставить, например, белую пешку на первую горизонталь).

Реализовать иерархию классов: «Шашка», «Простая шашка» (производный класс), «Дамка» (другой производный класс). Для каждого производного класса должна быть реализована своя функция «Сделать ход». Члены класса: поле, на котором стоит Шашка (реализовать приватный класс «Поле»); цвет шашки. Если ход возможен, поле шашки меняется. Если невозможен – не меняется. Поле характеризуется буквой (a-h и цифрой 1–8). В конструкторах обеспечить контроль ввода (чтобы не поставить шашку на белое поле).

Практикум

Практическое занятие по темам модуля. Конкретное содержание определяется учителем.

Контрольное тестирование по модулю

Электронный тест охватывает все темы модуля и в большей части направлен на оценку практических знаний и навыков учеников, полученных в ходе изучения модуля.

Базовый уровень

Модуль 4. Основы программирования Android приложений

Тема 4.1. Ввод, вывод и исключения

Цель. Изучить файловый ввод вывод и механизм обработки исключений в Java и освоить понятие «исключение» как объект.

Обработка исключений как средство создания надёжного, помехоустойчивого кода. Основные понятия: исключительная ситуация; обработчик исключения; выбрасывание исключения с аргументом с помощью throw; передача управления из текущего контекста наверх.

Обработка исключения с помощью конструкции try-catch. Возможность соответствия одному блоку try нескольких блоков catch. Основные методы класса Exception. Стратегии обработки исключений: прерывание и возобновление. Пример целесообразности возобновления.

Класс File и его методы: exists(), renameTo(), getAbsolutePath(), canRead(), canWrite(), getName(), length(), isDirectory(), lastModified(). Примеры возникновения и обработки исключений, возникающих при выполнении методов класса.

Упражнение 4.1.1. Разбор примера кода с типовым использованием потоков ввода/вывода: чтение из файла и из памяти; вывод в файл. Использовать классы Scanner и PrintWriter.

Минипроект 4.1. Продолжение. Реализация обработки исключений в заготовке согласно заданию.

Задание 4.1.1. Реализация посимвольного сравнения двух файлов или страниц в интернете. Выводить требуется все отличающиеся символы в произвольном формате. Если символов очень много нужно вывести только часть и количество различий.

Тема 4.2. Внутренние классы в обработке событий

Цель. Освоить применение внутренних анонимных классов для обработки событий интерфейса Android-приложений.

Внутренние (вложенные) **классы**. Понятие внутреннего класса. Отличие от наследования. Назначение. Доступ к состоянию объекта с помощью внутреннего класса.

Локальные и **анонимные** внутренние классы. Сущность, синтаксис,

назначение.

Обработка событий пользовательского интерфейса. Краткий обзор классов и интерфейсов для обработки событий. Классы Listeners. Использование анонимных классов для реализации обработчиков событий.

Упражнение 4.2.1. Разбор примера кода с обработчиками событий.

Минипроект 4.1. Завершение. Реализация обработчиков событий с использованием анонимных классов согласно заданию.

Тема 4.3. Параллелизм и синхронизация. Потоки

Цель. Освоить понятия потока, назначения многопоточности и структуры многопоточной программы, получить базовые навыки реализации многопоточности в Java.

Общие понятия. Введение в естественный параллелизм алгоритмов. Оценка улучшения производительности при параллельном выполнении программы. Пример нарушения целостности данных при неаккуратной реализации параллелизма и необходимость синхронизации параллельных ветвей.

Потоки (threads) как средство реализации параллелизма в рамках одного процесса (программы). Общее описание того, как работает многопоточная программа. Некоторые «подводные камни» реализации многопоточности; возможная неопределенность при определении порядка доступа к общему ресурсу.

Процессы и **потоки** в Android. Реализация и запуск AsyncTask. Альтернативные способы создания потокового класса. Наследование от класса Thread и реализация интерфейса Runnable. Предпочтения использования того или иного способа.

Реализация логики потока. Метод run () как реализация логики потока. Запуск потока на выполнение с помощью метода start () и смысл его аргумента как ссылки на объект класса, чей метод run () должен выполнять данный поток. Уяснение фундаментального факта, что вызов start () является асинхронным.

Синхронизация потоков. Объявление метода с помощью ключевого слова syncronized. Применение synchronized к отдельным блокам кода внутри run (). Методы взаимодействия потоков: suspend () – resume (), wait () -notify (). Метод join () как команда одному потоку дождаться завершения выполнения другого.

Блокировки. Понятие мёртвой блокировки (deadlock), механизм её возникновения.

Упражнение 4.3.1. Разобрать пример программы, совершающей загрузку картинки из интернета и устанавливающей её на экран.

Тема 4.4. Двумерная графика в Android приложениях

Цель. Получить навыки использования двумерной графики в Android. Если графика уже была изучена в рамках тем первого модуля, то можно следующей теме 3.9. посвятить больше времени.

Класс Canvas. Обзор методов и полей класса. Создание собственного View с методом onDraw. Вызов setContentView с вновь созданным классом. Класс Paint. Закраска экрана фоновым цветом. Рисование на canvas круга, прямоугольника. Отрисовка текста. Поворот полотна canvas.rotate и convas.resotre. Отрисовка текста под наклоном.

Двумерная анимация. Обзор методов и полей класса. Создание собственного View с методом onDraw. Вызов setContentView с вновь созданным классом. Класс Paint. Закраска экрана фоновым цветом. Рисование на canvas круга, прямоугольника. Отрисовка текста. Поворот полотна canvas.rotate и canvas.restore. Отрисовка текста под наклоном. Создание массива абсцисс, ординат и скоростей для движения различных элементов.

Упражнение 4.4.1. Разбор предоставленного кода игрового приложения «Крестики – нолики». Внесение изменений в код, пересборка проекта и просмотр влияния изменений на поведение приложения.

Реализовать пример добавления в программу ещё одного экрана с графикой и описанием правил игры.

Проект состоит из трех java-файлов:

- TicTacToe.java – обработка начального экрана игры;
- Game.java – обработка процесса игры;
- GameView.java – отрисовка игрового поля.

В папке res находятся xml-файлы, описывающие параметры отображения (res/layout), строковые константы (res/values/strings), описание массивов

(res/values/arrays).

В файле AndroidManifest.xml описываются основные параметры приложения.

Задание 4.4.1. Реализуйте три экрана:

- считает количество нажатий на кнопку;
- загадывает случайное число, которое нужно угадать. Программа может сообщать больше или меньше введённое число загаданного;
- показывает две дроби и нужно определить, какая из них больше.

На каждом экране сделать возможность перейти на следующий экран. Кнопка назад должна возвращать на предыдущий. В качестве доп. задания можно сохранять информацию между экранами с помощью методов putExtra и getIntent().getExtras().get(KEY).

Тема 4.5. Разработка игровых приложений. Реализация графики на основе SurfaceView

Цель. Получить навыки разработки простейших игровых приложений под Android.

Общие подходы для реализации игровых приложений. Последовательные этапы проектирования и реализации игрового приложения. Профессии в мире индустрии игр. Понятие игрового движка и его использование при разработке игры.

Класс SurfaceView. Обзор. Отличие View от SurfaceView. Особенность класса SurfaceView – предоставляет отдельную область для рисования, действия с которой должны быть вынесены в отдельный вспомогательный поток приложения. Методы getHolder(), lockCanvas(), unlockCanvasAndPost().

Упражнение 4.5.1. Разбор примера простейшей игры с анимацией.

Практикум

Практическое занятие по темам модуля. Конкретное содержание определяется учителем.

Контрольное тестирование по модулю

Электронный тест охватывает все темы модуля и в большей части направлен на оценку практических знаний и навыков учеников, полученных в ходе изучения модуля.

Модуль 5. Алгоритмы и структуры данных

Тема 5.1. 6 часов. Массивы. Списки. Алгоритмы сортировки. Алгоритм двоичного поиска. Деревья.

Тема 5.2. 3 часа. Адаптеры в Android

Тема 5.3. 3 часа. Ассоциативные массивы

Тема 5.4. 6 часов. Реляционная модель данных. СУБД. Введение в SQL

Практикум. Контрольное тестирование по модулю. 3 часа.

Тема 5.1. Массивы. Алгоритм двоичного поиска. Алгоритмы сортировки.

Цель. Изучение массивов на примере библиотечных классов Arrays, ArrayList. Знакомство с идеей, применением и реализацией алгоритма двоичного поиска. Разобрать алгоритмы сортировки и подходы к их реализации.

Повторение основных сведений о массивах в Java. Синтаксис объявления массива. Явное возвращение массива в качестве результата функции: здесь следует подчеркнуть принципиальное отличие от C, где массив может быть возвращён только неявным образом – по адресу, переданному в одном из параметров функции.

Библиотечный класс Arrays. Класс java.utils.Arrays как набор статических методов, полезных для работы с массивами. Заполнение массива с помощью метода Arrays.fill(). Копирование массива с помощью метода System.arraycopy(). Сравнение массивов на равенство с помощью метода Arrays.equals(): уяснение того факта, что программа должна знать, как же ей сравнивать отдельные элементы массивов на равенство, в связи с чем необходимо иметь реализованным метод equals() для класса, к которому принадлежат элементы массива, если они не относятся к одному из примитивных типов.

Упражнение 5.1.1. Практическое занятие по библиотечному классу Arrays, реализующему массивы: заполнение, копирование, сравнение, печать, другие общие методы.

Библиотечный класс ArrayList как реализация массива без ограничений на количество элементов. Создание списка, метод add(), метод get(), метод size(). Параметризация списка типом объектов при создании: *ArrayList<Person> list = new*

ArrayList<Person>().

Понятие итератора и его использование для обхода контейнера. Итератор как объект. Фундаментальные методы любого итератора: next() – получить следующий объект контейнера; hasNext() – проверить, если ли ещё объекты в контейнере, т.е. не завершен ли обход.

Упражнение 5.1.2. Разобрать пример использования ArrayList<integer>. Пояснить, что в <> указывается тип, хранимый в списке. Данный тип обязан быть объектом. Разобрать две схемы прохода по контейнеру.

Последовательный и **двоичный** поиск. Последовательный поиск в произвольном линейном массиве: трудоёмкость n. Двоичный поиск в известной игре угадывания числа. Определение его трудоёмкости как двоичный логарифм n. Поиск в отсортированном массиве с помощью метода Arrays.binarySearch().

Упражнение 5.1.3. Практическое занятие по использованию методов класса Arrays, реализующих поиск.

Задание 5.1.1. Разработка собственного метода, например, вывод массива на печать. Продемонстрировать объектно-ориентированный подход к решению задаче, если тип элементов массива произвольный: необходимость наличия «печатающего» метода для класса, к которому принадлежат элементы массива.

Задание 5.1.2. Задана матрица, в которой элементы упорядочены при просмотре слева направо сверху вниз. Реализовать класс, который находит в такой матрице строку и столбец, в котором находится заданный элемент. Необходимо обработать ситуацию, когда элемент отсутствует.

Значение алгоритмов сортировки. Важность сортировки как самой по себе, так и для повышения быстродействия других алгоритмов: поиска, вставки, слияния.

Сортировка вставкой как наиболее простой алгоритм сортировки. Пошаговая реализация на примере. Программа, реализующая сортировку вставкой для массива целых чисел. Оценка трудоёмкости как n2.

Быстрая сортировка. Идея алгоритма быстрой сортировки (рекурсивное разбиение массива на два). Пошаговая реализация на примере. Структура программы, реализующей быструю сортировку. Оценка трудоёмкости как n*log(n).

Сортировка массивов с помощью метода Arrays.sort(). Уяснение того факта, что способ сравнения для класса, к которому относятся элементы массива, должен быть известен (для двух элементов нужно как-то определить, кто из них больше). Мотивация наследования интерфейса Comparable и реализации метода compareTo(), чтобы массив мог быть отсортирован.

Упражнение 5.1.4. Практическое занятие по использованию методов класса Arrays и Collections, реализующих сортировку.

Сортировка и **поиск** в List-контейнерах. Важно уяснить, что методы sort и binarySearch для списков являются статическими методами базового класса Collections, а не класса Arrays.

Задание 5.1.3. Реализовать функцию, которая проверяет является ли массив отсортированным.

Тема 5.2. Адаптеры в Android

Цель. Освоить применение адаптеров при разработке Android-приложений на Java.

Адаптеры. Для чего нужны адаптеры. Готовые адаптеры в Android: SimpleAdapter, ArrayAdapter. Абстрактный класс BaseAdapter.

Применение адаптеров для обработки событий пользовательского интерфейса. Отображение ListView через адаптер. Методы getView(), getListAdapter().

Упражнение 5.2.1. Разбор примера кода с реализацией ListView через ArrayAdapter.

Упражнение 5.2.2. С помощью SimpleAdapter реализовать более сложный список, в котором строка формируется из двух текстов.

Тема 5.3. Ассоциативные массивы

Цель. Освоить понятие «ассоциативный массив» и его реализацию в Java.

Ассоциативный массив как набор пар «ключ-значение». Требование уникальности ключа. Примеры, когда естественным индексом выступает не целое число, а произвольная символьная строка (например, «государство – столица»). Ознакомление учащихся с фактом, что многомерный ассоциативный массив с произвольным набором индексов, называемый глобал, выступает в качестве основной

структуры данных в нереляционных БД, и для работы с ним существует набор специальных функций.

Общие методы Map: put(), get(), containsKey(), containsValue().

Контейнер HashMap. Контейнер HashMap и пример его использования. Идея контейнера: поиск ключей с помощью хэш-кодов значений.

Контейнер TreeMap. Контейнер TreeMap, хранение данных, упорядоченных по ключу. Метод subMap(), возвращающий часть дерева.

Потоко-безопасность объектов. Напоминание понятия потоко-безопасности объекта. Возможность сделать контейнер потоко-безопасным. Статические методы класса Collections: synchronizedCollection, synchronizedList, synchronizedSet, synchronizedMap.

Упражнение 5.3.1. Разобрать пример использования TreeMap и HashMap.

Задание 5.3.1. Реализовать многопоточную программу, которая добавляет в synchronizedMap элементы с ключами от 100 * THREAD_NUM до 100 * (THREAD_NUM + 1), где THREAD_NUM – номер потока. Изучить, что происходит, если использовать обычный TreeMap.

Тема 5.4. Реляционная модель данных. СУБД. Введение в SQL

Цель. Рассказать учащимся, что же представляет собой в общих чертах предмет изучения, зачем он нужен и какова история его развития. Понять представление реалий окружающего мира с помощью модели «сущность-связь».

Получить представление о реляционной модели данных и получить представление о записи связей в реляционной базе.

На примере локальной СУБД SQLite изучить средства SQL, используемые для создания и модификации содержимого таблиц, изменения их структуры, удаления.

Важность БД и СУБД. Использование БД во всех направлениях современной деятельности человека на примерах из повседневной жизни. Необходимо объяснить, почему хранение данных в обычных текстовых файлах не может считаться приемлемым решением для эффективной работы с данными и какие задачи в этой связи возлагаются на СУБД. История развития и классификация СУБД.

Характеристика СУБД как соединения программного обеспечения и данных и

неотъемлемо включающей следующие составные части:

- набор файлов, содержащих данные – непосредственно физическая база данных;
- спецификация информационного содержимого физической базы данных – схема данных;
- программное обеспечение, поддерживающее доступ и изменение содержимого базы данных – механизм СУБД;
- язык программирования СУБД, используемый для задания схемы и осуществления доступа к базе данных.

Хранение данных в таблицах. Привести примеры хранения данных в таблице. Описание типов связи и отношений: один к одному, один ко многим, многие ко многим. Примеры отношений: муж – жена (в России в каждый момент времени может быть только один супруг), сын – отец (отец всегда один, детей много), брат – сестра (каждый может иметь много братьев и сестер).

Объяснить способы хранения таких данных в таблицах. Способы хранения дополнительных данных в отношениях.

Локальная СУБД SQLite. Знакомство с локальной СУБД SQLite.

Минипроект 5.4. Записная книжка. Создать БД SQLite «Записная книжка» по спроектированной ранее структуре. На её примере разобрать все изучаемые далее инструкции SQL, создание простейшего Android-приложения. Самостоятельно закончить мини-проект.

Язык запросов SQL. О языке запросов SQL. Создание таблиц. Синтаксис запроса CREATE TABLE, используемого для создания таблицы. Команды ALTER TABLE и DROP TABLE, SHOW TABLES.

Для создания базы данных SQLite проще всего воспользоваться утилитой SQLiteStudio.

Вставка, **изменение** и **удаление** данных из таблицы. Краткое разъяснение и синтаксис команды (INSERT INTO table VALUES …).

Обновление таблиц. Синтаксис запроса UPDATE, используемого для изменения записей таблиц. Ключевое слово WHERE и простейшие фильтры.

Выборка данных. Краткое разъяснение назначения и синтаксис команды SELECT. Форма SELECT * для просмотра всей таблицы. Форма SELECT для просмотра только определённых столбцов таблицы. Ключевое слово FROM; имя таблицы, которая является источником информации для запроса. Команда DELETE FROM table WHERE.

Булевы операции в запросах. Более сложные запросы SELECT. Ключевые слова ORDER BY и DISTINCT.

Агрегация. Вариант SELECT для подсчёта количества, суммы, максимума и минимума, среднего и других агрегаций.

Практикум

Практическое занятие по темам модуля. Конкретное содержание определяется учителем.

Тема 5.5 Контрольное тестирование по модулю

Электронный тест охватывает все темы модуля и в большей части направлен на оценку практических знаний и навыков учеников, полученных в ходе изучения Модуля.

6. Консультации по ИП

Цель: консультация по вопросам индивидуального проекта.

7. Итоговая защита ИП

Презентация и защита итогового проекта.

4. Планируемые результаты

Предметные результаты:

– знание и соблюдение требований техники безопасност;

– знание основ языка программирования Java и языка разметки XML;

– понимание принципа работы баз данных и клиент-серверных протоколов;

– умение использовать разные алгоритмы в приёмах программирования;

– умение пользоваться ПК и IDE-разработки для программирования устройства;

– умение читать готовую программу и находить ошибки в готовых программах.

Личностные результаты:

– сформированное ответственное отношение к учению, готовность и способность обучающихся к саморазвитию и самообразованию, средствами информационных технологий;

– сформированы универсальные способы мыслительной деятельности (абстрактно-логического мышления, памяти, внимания, творческого воображения, умения производить логические операции);

– развит опыт участия в социально значимых проектах, повышение уровня самооценки благодаря реализованным проектам;

– сформированы коммуникативные компетентности в общении и сотрудничестве со сверстниками в процессе образовательной, учебно-исследовательской и проектной деятельности;

– сформировано целостное мировоззрение, соответствующее современному уровню развития информационных технологий;

– сформировано осознанное позитивное отношение к другому человеку, его мнению, результату его деятельности;

– усвоены правила индивидуального и коллективного безопасного поведения при работе с компьютерной техникой.

Метапредметные результаты:

– умение самостоятельно ставить и формулировать для себя новые задачи, развивать мотивы своей познавательной деятельности;

– умение самостоятельно планировать пути решения поставленной проблемы для получения эффективного результата;

– умение критически оценивать правильность решения учебно-исследовательской задачи;

– умение формулировать, аргументировать и отстаивать своё мнение;

– владение основами самоконтроля, способность к принятию решений;

– умение извлекать нужную информацию из открытых источников;

– умение организовывать учебное сотрудничество и совместную деятельность с педагогом и сверстниками в процессе проектной и учебно-исследовательской деятельности.

5. Комплекс организационно-педагогических условий реализации общеразвивающей программы

1. Календарный учебный график на 2021-2022 учебный год

Таблица 2

№ п/п	Основные характеристики образовательного процесса	
1.	Количество учебных недель	38
1.1	Количество учебных недель, реализуемых организацией-участником	2
1.2	Количество учебных недель, реализуемых базовой организации	36
2.	Количество учебных дней	38
2.1	Количество учебных дней, реализуемых организацией-участником	2
2.2	Количество учебных дней, реализуемых базовой организации	36
3.	Количество часов в неделю	3
4.	Количество часов на учебный год	114
4.1	Количество часов на учебный год, реализуемых организацией-участником	6
4.2	Количество часов на учебный год, реализуемых базовой организации	108
5.	Недель в I полугодии	16
5.1	Количество учебных недель, реализуемых организацией-участником	2
5.2	Количество учебных недель, реализуемых базовой организации	16
6.	Недель во II полугодии	20
7.	Начало занятий	1сентября
	Начало занятий, реализуемых организацией-участником	1сентября
	Начало занятий, реализуемых базовой организации	13 сентября
8.	Выходные дни	31 декабря – 9 января
9.	Окончание учебного года	31 ая

2. Условия реализации программы

Материально-техническое обеспечение:

Первый модуль программы реализуется организацией – участником в соответствии с условиями договора о сетевой форме реализации программ.

Требования к помещению:

– помещение для занятий, отвечающие требованиям СанПин для учреждений дополнительного образования;

– качественное освещение;

– столы, стулья по количеству обучающихся и 1 рабочим местом для педагога;

Оборудование:

– компьютеры и ноутбуки на каждого обучающегося и преподавателя;

– проекционное оборудование – 2 шт.

– маркерная доска – 1 шт.

Расходные материалы:

– whiteboard маркеры;

– бумага писчая;

– шариковые ручки;

– permanent маркеры.

Информационное обеспечение:

– операционная система (желательно Windows);

– программное обеспечение Eclips, Android Studio, объединенные в локальную сеть;

– планшет (для отладки);

– ПК для педагога, объединённый с функцией сервера.

Кадровое обеспечение:

Программа реализуется Шаниным М. М., Князевым А.В., педагогами дополнительного образования.

3. Формы аттестации и оценочные материалы

Система отслеживания результатов, обучающихся выстроена

следующим образом:

- входная диагностика (Приложение 1);
- промежуточный контроль по первому модулю (Приложение 2);
- промежуточный контроль по второму модулю (Приложение 3);
- промежуточный контроль по третьему модулю (Приложение 4);
- промежуточный контроль по четвертому модулю (Приложение 5);
- защита итогового проекта (Приложение 6).

Система контроля знаний и умений обучающихся представляется в виде учёта результатов по итогам выполнения контрольных тестов по модулям. Метод педагогического наблюдения помогает отслеживать динамику развития обучающегося. В конце обучения подростки проходят защиту индивидуальных/групповых проектов.

Итоговый проект оценивается формируемой комиссией по 10-бальной шкале. Состав комиссии (не менее 3 человек): в обязательном порядке входит педагог; приветствуется привлечение ИТ-профессионалов, представителей высших и других учебных заведений, администрации учебной организации.

Компонентами оценки индивидуального проекта являются (по мере убывания значимости): качество ИП, отзыв руководителя проекта, уровень презентации и защиты проекта. Если проект выполнен группой обучающихся, то при оценивании учитывается не только уровень исполнения проекта в целом, но и личный вклад каждого из авторов. Решение принимается коллегиально.

Формы проведения итогов по каждой теме и каждому разделу общеразвивающей программы соответствуют целям и задачам ДООП.

Минимальное количество баллов, которое возможно получить по результатам промежуточного контроля по первому модулю - 1 балл, максимальное – 10 баллов.

Минимальное количество баллов, которое возможно получить по результатам промежуточного контроля по второму модулю - 1 балл, максимальное – 14 баллов.

Минимальное количество баллов, которое возможно получить по

результатам промежуточного контроля по третьему модулю - 1 балл, максимальное – 11 баллов.

Минимальное количество баллов, которое возможно получить по результатам промежуточного контроля по четвертому модулю - 1 балл, максимальное – 14 баллов.

Минимальное количество баллов, которое возможно получить по результатам защиты годового проекта - 1 балл, максимальное – 50 баллов.

Сумма баллов результатов промежуточного контроля и защиты итогового годового проекта переводится в один из уровней освоения образовательной программы согласно таблице 4:

Таблица 4

Баллы, набранные учащимся.	Уровень освоения
1-39	Низкий
40-79	Средний
80-100	Высокий

3.1. Перечень диагностического материала для осуществления мониторинга личностных и метапредметных планируемых результатов

1. Шкала поведенческих характеристик одаренных школьников (Приложение №7);

2. Анкета «Оценка уровня учебной мотивации», автор Лусканова Н.Г. (Приложение №8);

3. Мониторинг достижения обучающимися метапредметных результатов (Приложение №9);

5. Мониторинг достижения обучающимися личностных результатов (Приложение №10);

6. Мониторинг успешности обучающихся в рамках реализации программы в сетевой форме (Приложение №11)

Методические материалы

В образовательном процессе используются следующие методы обучения:

– устные (беседы, объяснение);

– поисковые (изменение программы для приобретения устройством новых свойств);

– демонстрационные (демонстрация возможностей устройства);

– практические (написание программы, проведение мини-соревнований).

Программой предусмотрены следующие виды деятельности обучающихся:

– работа с технической и справочной литературой;

– программирование;

– эксперимент, испытание.

Выбор методов обучения осуществляется исходя из анализа уровня готовности обучающихся к освоению содержания модуля, степени сложности материала, типа учебного занятия.

Формы обучения:

– ***фронтальная*** – предполагает работу педагога сразу со всеми обучающимися в едином темпе и с общими задачами. Для реализации обучения используется компьютер педагога с мультимедиа проектором, посредством которых учебный материал демонстрируется на общий экран. Активно используются Интернет-ресурсы;

– ***групповая*** – предполагает, что занятия проводятся с подгруппой. Для этого группа распределяется на подгруппы не более 6 человек, работа в которых регулируется педагогом;

– ***индивидуальная*** – подразумевает взаимодействие преподавателя с одним обучающимся. Как правило данная форма используется в сочетании с фронтальной. Часть занятия (объяснение новой темы) проводится фронтально, затем обучающийся выполняют индивидуальные задания или общие задания в индивидуальном темпе;

– ***дистанционная*** – взаимодействие педагога и обучающихся между собой на расстоянии, отражающее все присущие учебному процессу компоненты. Для

реализации дистанционной формы обучения весь дидактический материал размещается в свободном доступе в сети Интернет, происходит свободное общение педагога и обучающихся в социальных сетях, по электронной почте, посредством видеоконференции или в общем чате. Кроме того, дистанционное обучение позволяет проводить консультации обучающегося при самостоятельной работе дома. Налаженная система сетевого взаимодействия подростка и педагога, позволяет не ограничивать процесс обучения нахождением в учебной аудитории, обеспечить возможность непрерывного обучения в том числе, для часто болеющих детей или всех детей в период сезонных карантинов (например, по гриппу) и температурных ограничениях посещения занятий.

Формы организации учебного занятия:

В образовательном процессе помимо традиционного учебного занятия используются многообразные формы, которые несут учебную нагрузку и могут использоваться как активные способы освоения детьми образовательной программы, в соответствии с возрастом обучающихся, составом группы, содержанием учебного модуля: беседа, лекция, мастер-класс, практическое занятие, защита проектов, конкурс, викторина, диспут, круглый стол, «мозговой штурм», воркшоп, квиз.

Некоторые формы проведения занятий могут объединять несколько учебных групп или весь состав объединения, например, экскурсия, викторина, конкурс и т. д.

Дидактические материалы:

Методические пособия, разрабатываемые преподавателем с учётом конкретных условий. Техническая библиотека объединения, содержащая справочный материал, учебную и техническую литературу. Индивидуальные задания.

Методическое обеспечение учебного процесса включает разработку преподавателем методических пособий, вариантов демонстрационных программ и справочного материала:

– демонстрационные программы;
– инструкции по настройке среды разработки;
– справочные материалы по терминам ПО;
– дополнительный учебный материал по теме (портал информатикс);
– инструкции по настройке среды разработки.

Приложение 1

Пример вводного тестирования

г. Екатеринбург Дата_____

ФИО _____ **Группа**_____

1) Набор средств программирования, который содержит инструменты, необходимые для создания, компиляции и сборки мобильного приложения называется:

а) Android SDK

б) JDK

в) плагин ADT

г) Android NDK

2) С какой целью был создан Open Handset Alliance?

а) писать историю развития ОС Android

б) продавать смартфоны под управлением Android

в) рекламировать смартфоны под управлением Android

г) разрабатывать открытые стандарты для мобильных устройств

3) С какой целью инструмент Intel* Graphics Performance Analyzers (Intel* GPA) System Analyzer используется в среде разработки Intel* Beacon Mountain?

а) позволить разработчикам оптимизировать загруженность системы при использовании процедур OpenGL

б) для ускорения работы эмулятора в среде разработки

в) для оптимизированной обработки данных и изображений

г) позволить разработчикам эффективно распараллелить C++ мобильные приложения

3) Библиотеки, реализованные на базе PacketVideo OpenCORE:

а) Media Framework

б) SQLite

в) FreeType

г) 3D библиотеки

4) **Какой движок баз данных используется в ОС Android?**

а) InnoDB

б) DBM

в) MyISAM

г) SQLite

5) С какой целью инструмент Intel* Integrated Performance Primitives (Intel* IPP) используется в среде разработки Intel* Beacon Mountain?

а) для оптимизированной обработки данных и изображений

б) позволить разработчикам оптимизировать загруженность системы при использовании процедур OpenGL

в) для ускорения работы эмулятора в среде разработки

г) позволить разработчикам эффективно распараллелить C++ мобильные приложения

6) Intel XDK поддерживает разработку под:

а) JavaFX Mobile

б) Apple iOS, BlackBerry OS

в) MtkOS, Symbian OS, Microsoft Windows 8

г) Android, Apple iOS, Microsoft Windows 8, Tizen

7) Каждый приемник широковещательных сообщений является наследником класса …

а) ViewReceiver

б) IntentReceiver

в) ContentProvider

г) BroadcastReceiver

8) **Какой класс является основным строительным блоком для компонентов пользовательского интерфейса (UI), определяет прямоугольную область экрана и отвечает за прорисовку и обработку событий?**

а) GUI

б) View

в) UIComponent

г) Widget

9) Какой слушатель используется для отслеживания события касания экрана устройства?

а) OnPressListener

б) OnTouchListener

в) OnClickListener

г) OnInputListener

10) Фоновые приложения …

а) после настройки не предполагают взаимодействия с пользователем, большую часть времени находятся и работают в скрытом состоянии

б) выполняют свои функции и когда видимы на экране, и когда скрыты другими приложениями

в) небольшие приложения, отображаемые в виде графического объекта на рабочем столе

г) большую часть времени работают в фоновом режиме, однако допускают взаимодействие с пользователем и после настройки

Приложение 2

Модуль 1. Основы программирования

1. Типы данных Java. Фрагменты, которые не вызовут сообщение компилятора об ошибке преобразования типа: *(1 балл)*

- int x = 21; double y = x;
- short z = 13; int x = z;
- long x = 15; int y = x;
- int x = 14; short z = x;

2. Операции и выражения в Java. Выражения, результат которых имеет тип float или double: *(1 балл)*

- 5.0 * 3
- 5 / 2
- 5 * 0.5
- 2.5 * 2

3. Условные операторы в Java. Аналогом фрагмента кода *if (a == 2 || b == 3) { f();}* является : *(1 балл)*

- if (a == 2) { f();}

else if (b == 3) { f();}

- if (a == 2 && b == 3) { f();}
- if (a != 2 && b != 3) { f();}
- if (a == 2) if (b == 3) { f();}
- if (a != 2) {}

else if (b != 3) { f();}

4. Циклы в Java. Значение переменной *a* после окончания выполнения цикла равно: *(1 балл)*

- int a = 10;

  for (int i = 1; i < 3; ++i) a *= 4;

5. Массивы в Java. Правильное обращение к среднему элементу массива *a* нечетной длины приведено в: *(1 балл)*

- a[a.length / 2]

- a[a.length / 2 + 1]
- a[a.length %2]
- a.middle
- a[a.length/2-0.5]

6. Видимость переменных. Фрагмент кода, который вызовет ошибку компиляции: *(1 балл)*

- static int a = 7;

```
static int f() {
        int a = 3, b = 4;
        return a + b;}
```

- static int a = 7;

```
static int f() {
        int b = 4;
        return a + b;}
```

- static int a = 7;

```
static int f(int a) {
        int b = 4;
        return a + b;}
```

- static int a = 7;

```
static int f(int a, int b) {
        int b = 4;
        return a + b;}
```

- static int f(int a) {

```
        int b = 4;
        return a + b;}
```

7. Передача параметров в методы Java. В результате работы приведенного ниже фрагмента кода будет выведено: *(2 балла)*

```
static void swap(int a, int b) {
        int tmp = a;
        b = a;
        a = tmp;
}
public static void main(String[] args) {
        int a = 5;
```

```
        int b = 10;
        swap(a, b);
        System.out.print(a + ", " + b);
    }
```

8. Многомерные массивы. Пропущенный фрагмент функции, подсчитывающей сумму элементов, находящихся под побочной диагональю квадратной матрицы n*n (побочной диагональю матрицы называют линию, соединяющую левый нижний и правый верхний угол матрицы) это: *(2 балла)*

```
int SumUnderSecondaryDiagonal(int n, int[][] matrix) {
    int sum = 0;
    if (n<2) return 0;
    return sum;
```

- for(int i=0;i<n;i++)

```
    for(int j=n-i-1;j<n;j++)
        sum=sum+matrix[i][j];
```

- for(int i=n-1;i>0;i--)

```
    for(int j=i;j>0;j--)
        sum+=matrix[i][j];
```

- for(int i=0;i<n;i++)

```
    for(int j=0;j<i;j++)
        sum=sum+matrix[i][j];
```

- for(int i=n-1;i>0;i--)

```
    for(int j=n-i;j<n;j++)
        sum+=matrix[i][j];
```

Модуль 2. Объектно-ориентированное программирование.

1. Передача параметров строк. В результате работы приведенного ниже фрагмента кода будет выведено: *(1 балл)*

```
public class Example{
    static String str = "ABC";
    public static void changeStr(String s) { s = "abc"; }
    public static void main(String[] args) {
        System.out.print(str);
        changeStr(str);
        System.out.print(str);
    }
```

2. Про приемы тестирования и отладки. Виды тестирования, которые могут заменить отладчик с применением точек останова с условием: *(1 балл)*

- Модульное тестирование;
- Стресс тестирование;
- Логирование;
- Проверка утверждения (assert);
- Интеграционное тестирование.

3. Язык разметки XML. Верные утверждения: *(1 балл)*

- XML-документ всегда должен содержать ровно один корневой элемент;
- В XML-документе не могут присутствовать цифры;
- В XML-документе могут присутствовать только латинские символы;
- XML-документ может содержать произвольное количество корневых элементов;
- Ни один из вариантов.

4. Layout. Свойство, позволяющее назначить элементу важность его размера относительно других элементов в контейнере: *(1 балл)*

- layout_weight

- layout_width
- layout_height
- layout_gravity
- layout_wrap

5. Основные понятия объектно-ориентированного подхода в программировании (теория). Создание нового класса на основе уже существующего называют: *(1 балл)*

- наследование;
- полиморфизм;
- декомпозиция программы;
- инкапсуляция.

6. Классы. Правильное имя публичного класса в файле “Point.java” -*(1 балл)*

- Point
- Point.java
- point.java
- PointClass
- должно начинаться со слова Point

7. Перегрузка методов. Инициализация данных класса. Ошибка, к которой приводит отсутствие инициализации локальной переменной: *(1 балл)*

- происходит ошибка компиляции;
- переменная инициализируется случайными данными;
- переменная инициализируется null;
- происходит инициация исключения.

8. Программы на конструкторы и перегрузку. *(1 балл)*

- class Point {

```
        float x, y;
        // Конструктор
}
public class Example {
```

```
        public static void main(String[] args) {
            Point x = new Point(4.0f, 5.0f);
            System.out.print(x.x + x.y);
        }
    }
```

9. Анализ программ. *(1 балл)*

```
    public class MyDate {
        private int d; // День
        private int m; // Месяц
        private int y; // Год
        static int[] dayInMonth = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };
    public MyDate(int d, int m, int y) {
            if (d < 1 || m < 1 || m > 12 || y < 1) {
                System.out.println("Wrong date!");    return;        }
            If (d > whatIs2(m, y)) {   System.out.println("Wrong          date!");
return;        }
            this.d = d;
            this.m = m;
            this.y = y;
    }
    public static boolean isLeap(int y) {
            return (y % 400 == 0) || ((y % 4 == 0) && (y % 100 != 0));
    }

    public static int whatIs2(int m, int y) {
            if (m == 2 && isLeap(y))        return dayInMonth[m - 1] + 1;
            return dayInMonth[m - 1];
    }

    public boolean whatIs(MyDate dt) {
```

```
            if (this.y == dt.y) {
                if (this.m == dt.m)        return this.d > dt.d;
                else                       return this.m > dt.m;
            }
    else                            return this.y > dt.y;
    }
    }
```

10. Метод класса MyDate whatIs() возвращает ложь, если: *(1 балл)*

- дата в параметре метода позже либо равна дате объекта, вызвавшего метод;
- дата в параметре метода строго позже, чем дата объекта, вызвавшего метод;
- дата в параметре метода строго раньше, чем дата объекта, вызвавшего метод;
- дата в параметре метода раньше либо равна дате объекта, вызвавшего метод;
- ни один из вариантов.

11. Наследование. Количество различных методов (из определенных в приведенном коде), которое может быть вызвано в методе *main* у переменной x без преобразования ее типа равно: *(1 балл)*

```
class Base {
        public void f1() {}
        private void f2() {}
        private void f3() {}
}
        class Subclass extends Base {
            public void f1() {}
            public void f4() {}
        }
        class EntryPoint {
```

```
        public static void main(String[] argv) {
            Subclass x = new Subclass();
        }
    }
```

12. Наследование. Правильный вызов конструктора базового класса Base из конструктора его наследника приведен в: *(1 балл)*

- super(args);
- new Base(args);
- Base(args);
- new super(args);
- ни один из вариантов.

13. Полиморфизм. Методы переменной x, которые могут быть вызваны без преобразования типа переменной x: *(2 балла)*

```
class Base {
        public int f() { … }
        public int g() { … }
}
class Derived extends Base {
        public int g() { … }
        public int h() { … }
}
Base x = new Derived();
```

- f;
- g;
- h;
- ни один из вариантов.

Приложение 4

Модуль 3. Основы Android.

1. ОО проектирование. Какой из вариантов лучше описывает отношение "Человек (man) проживает (live) по адресу (address)"? *(1 балл)*

- class Man extends Address{ };
- class Man { private Live address; };
- class Man implements Address{ };
- class Man { private Address live; };
- class Address{ private Man live; }.

2. Создание Андроид проекта. Основным классом Android приложения, имеющего интерфейс пользователя, является: *(1 балл)*

- Activity;
- KeyEvent;
- View;
- Fragment;
- Canvas.

3. Ввод, вывод и исключения. Ключевое слово, показывающее, что в процессе работы метода могут произойти не обработанные исключения: *(1 балл)*

- try;
- catch;
- finally;
- throws.

4. Класс(ы), используемый(е) для чтения файла. *(1 балл)*

- FileInputStream;
- FileOutputStream;
- FileInputStream, FileOutputStream;
- ни один из перечисленных.

5. Внутренние и анонимные классы. Пропущенный фрагмент, после вставки которого программа выведет на экран значение 50 – это: *(2 балла)*

```
public class OuterClass {
```

```
        private int x = 0;
        private class InnerClass {
            InnerClass() {
                OuterClass.this.x = 50;   }       }
        private InnerClass xy;
        public OuterClass() {____}
        public static void main(String[] args) {
            OuterClass obj = new OuterClass();
            System.out.println(obj.x);
        }
    }
```

- xy = new InnerClass();
- x = 50;
- this.xy = new InnerClass();
- xy = InnerClass();
- xy = 50.

6. Слушатели. Необходимо, чтобы когда пользователь отпускал кнопку myButton (класс android.widget.Button), запускался метод doIt(). Возможные способы реализации... *(1 балл)*

- Назначить в XML layout аттрибуту android:onClick значение “doIt”;
- Назначить кнопке слушателя onClickListener и вызвать doIt() из метода onClick();
- Назначить кнопке слушателя OnTouchListener и вызвать doIt() из метода onTouch();
- Вызвать startActivity с параметром myButton из myButton.onClick();
- ни один из вариантов.

7. Параллелизм и синхронизация. Потоки. Операции, которые не должны выполняться одновременно в разных потоках: *(1 балл)*

- Два чтения одной и той же переменной;
- Запись и чтение одной и той же переменной;

- Две записи в одну и ту же переменную;
- Изменение двух разных экземпляров одного и того же класса.

8. Context и Intent. Верные утверждения: *(1 балл)*
   - Класс android.context.Context используется для доступа к глобальной информации об окружении Android приложения;
   - Для передачи намерения в Activity можно использовать метод startActivity();
   - Метод putExtra() предназначен для сохранения данных в Intent;
   - Явным (explicit) называется Intent, который содержит флаг Intent.FLAG_EXPLICIT;
   - Для передачи намерения в Activity можно использовать любой метод.
9. Анализ программ с Canvas. *(1 балл)*

```
paint.setStyle(Paint.Style.FILL);
paint.setColor(Color.GREEN);
canvas.drawRect(50, 80, 400, 280, paint);
paint.setColor(Color.YELLOW);
canvas.drawCircle(150, 150, 50, paint);
```

10. Приведенный фрагмент рисует: *(1 балл)*
    - желтый круг в верхнем левом углу зеленого прямоугольника;
    - желтый круг в верхнем правом углу зеленого прямоугольника;
    - желтый круг в нижнем правом углу зеленого прямоугольника;
    - зеленый круг в верхнем левом углу желтого прямоугольника;
    - ни один из вариантов.

Приложение 5

Модуль 4. Алгоритмы и структуры данных.

1. В поле arr переменной a после выполнения операции: a = new A(); будет хранится ... *(1 балл)*

- null;
- массив из 0 элементов;
- не определено;
- массив случайной длины из случайных чисел.

2. Списки (общие вопросы, теория). В двусвязном списке необходимо поменять два соседних элемента местами. Количество ссылок, которое будет изменено (если ни один из элементов не является ни первым ни последним) - *(1 балл)*

3. Списки: LinkedList, Queue, Stack. Операция взятия с удалением элемента из Queue: *(1 балл)*

- push;
- pop;
- poll;
- peek.

4. Деревья и двоичный поиск (общие вопросы, теория). Двоичное сбалансированное дерево обладает следующим свойством: *(1 балл)*

- Высота левого поддерева отличается от высоты правого поддерева не более, чем на 1;
- Все значения в вершинах левого поддерева не больше значений в вершинах правого поддерева;
- Все значения в вершинах левого поддерева больше значения в корне поддерева;
- Высота левого поддерева не меньше высоты правого поддерева.

5. Деревья: TreeSet. В результате работы ниже приведенного фрагмента на экран будет выведено: *(1 балл)*

```
TreeSet<Integer> tree = new TreeSet<Integer>();
```

```
tree.add(15);
tree.add(25);
tree.add(15);
tree.add(11);
Iterator<Integer> iterator = tree.iterator();
while (iterator.hasNext())  System.out.print(iterator.next() + " ");
```

6. Рекурсия. Для данной функции количество рекурсивных вызовов (т. е. вызовов функции из самой себя) при n=5 равно…*(1 балл)*

```
int fib(int n) {
    if (n < 3) {   return 1;}
    return fib(n - 1) + fib(n - 2);
}
```

7. Алгоритм двоичного поиска (класс Arrays и Collections). Arrays.binarySearch(new int[]{-1, 3, 5, 5, 19}, -1) возвратит: *(1 балл)*

- 0;
- 1;
- -1;
- true;
- false.

8. Алгоритмы сортировки (класс Arrays). При вызове метода Arrays.sort(a); переменная a может быть следующего типа: *(1 балл)*

- int[];
- String[];
- ArrayList<Integer>;
- String;
- TreeMap<Integer, Integer>.

9. Хэш-таблица и функция хэширования (практика). Заданная программа выведет на экран число: *(1 балл)*

```
class Point {
    int x, y;
```

```
        public Point(int x, int y) {        this.x = x;    this.y = y; }
        @Override
        public int hashCode() {
                final int prime = 31;
                return prime + prime * x + prime * prime * y;
        }
}public class Test {
        public static void main(String[] args) {
                HashSet<Point> set = new HashSet<Point>();
                set.add(new Point(12, 13));
                set.add(new Point(9, 0));
                set.add(new Point(12, 13));
                set.add(new Point(9, 30));
                System.out.println(set.size());
}       }
```

10. Ассоциативные массивы: HashMap, TreeMap. Порядок, в котором появляются элементы при итерации по классу HashMap: *(1 балл)*

- зависит от реализации библиотеки;
- по неубыванию ключей;
- по невозрастанию ключей;
- по неубыванию значений;
- по невозрастанию значений.

11. Анализ программ. *(1 балл)*

```
class WhoAmI {
        public int whatAmIDoing(int x) {
                set.add(x);
                return set.headSet(x).size();
        }
        public static long howDoIProcessThisArray(int[] a) {
                WhoAmI obj = new WhoAmI();
```

```
        long res = 0;
        for (int i = a.length - 1; i >= 0; --i) {
            res += obj.whatAmIDoing(a[i]);
        }
        return res;
    }
    private SortedSet<Integer> set = new TreeSet<Integer>();
}
```

12. Вызов WhoAmI.howDoIProcessThisArray(new int[]{2, 4, 7, 2, 5, 3, 1}) возвратит? *(1 балл)*

13. Реляционная модель, СУБД – понятия. Для отдельно взятой таблицы в реляционной модели баз данных верно утверждение: *(1 балл)*

- Каждая строка может иметь собственное количество столбцов;
- В одном столбце в разных строках могут быть записаны элементы разного типа;
- Все строки имеют одинаковое количество столбцов и в каждом столбце хранятся данные одного типа;
- Каждая строка содержит произвольное количество однотипных данных.

14. CREATE 2. Выполнен следующий запрос: *(1 балл)*

```
CREATE TABLE pupil (
        name varchar(30),       surname varchar(30),
        age integer,            birthday date,
        height float,           weight float ).
```

Количество колонок, которые будут иметь строковой тип в созданной таблице - ?

Запросы CREATE. Выберите таблицу, содержание которой соответствует запросу: *CREATE TABLE pupil (name varchar(3), age integer)?*

-

name	age

Ivan	13
Wu	10

-

name	age
Hu	13
Wu	10

-

name	age
Ha	13.5
Wu	10

-

name	age
Ivan	twelve
Wu	eleven

Приложение 6

Лист оценки итогового годового проекта.

№	ФИО	Актуальность проекта и его проработанность в рамках выбранной темы	Портфолио и освоенные навыки	Качество презентационных материалов, единая стилистика презентации	Выступление обучающихся на защите проекта.	Владение темой, свободное ориентирование в проекте, ответы на вопросы комиссии	Р Е З У Л Ь Т А Т
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							

Каждый показатель соответствует числу от 1 до 10, где 1 – результат не удовлетворителен, 10 – отличный результат. Итоговый результат выставляется путем сложения всех показателей. Максимальное количество баллов-50.

Приложение 7

ШКАЛА ПОВЕДЕНЧЕСКИХ ХАРАКТЕРИСТИК ОДАРЕННЫХ ШКОЛЬНИКОВ

Авторы: Дж. Рензулли и соавторы, в адаптации Л.В. Поповой

Возраст детей: 12-17 лет

Цель: Эта шкала составлена для того, чтобы педагог мог оценить характеристики, обучающихся в познавательной, мотивационной, творческой и лидерской областях. Каждый пункт шкалы следует оценивать безотносительно к другим пунктам. Ваша оценка должна отражать, насколько часто вы наблюдали у обучающегося проявление каждой из характеристик. Так как четыре шкалы представляют относительно разные стороны поведения, оценки по разным шкалам не суммируются.

Инструкция: пожалуйста, внимательно прочитайте каждое утверждение и обведите соответствующую цифру согласно следующему описанию:

1 – если вы почти никогда не наблюдали этой характеристики;

2 – если вы наблюдаете эту характеристику время от времени;

3 – если вы наблюдаете эту характеристику довольно часто;

4 – если вы наблюдаете эту характеристику почти все время.

Каждый пункт шкалы следует оценивать безотносительно к другим пунктам. Ваша оценка должна отражать, насколько часто Вы наблюдали проявление каждой из характеристик. Так как четыре шкалы представляют относительно разные стороны поведения, оценки по разным шкалам не суммируются. Далее следуйте инструкции в таблице, чтобы узнать результаты.

Пример расчета результатов:

№	Утверждение	Выберите цифру			
1.	Обладает необычно большим для этого возраста запасом слов; использует термины с пониманием; речь отличается богатством выражений, беглостью, сложностью.	**1**	**②**	**3**	**4**
2.	Обладает большим запасом информации по разнообразным темам (выходящим за пределы обычных интересов детей этого возраста).	**1**	**2**	**③**	**4**

3.	Быстро запоминает и воспроизводит фактическую информацию.	**1**	**2**	**3**	**④**
4.	Легко схватывает причинно-следственные связи; пытается понять «как» и «почему»; задает много стимулирующих мысль вопросов (в отличие от вопросов, направленных на получение фактов); хочет знать, что лежит в основе явлений или действий людей.	**1**	**2**	**③**	**4**
5.	Чуткий и сметливый наблюдатель; обычно "видит больше" или "извлекает больше", чем другие, из рассказа, фильма, из того, что происходит	**1**	**②**	**3**	**4**
Шаг 1. Подсчитайте число обведенных цифр по каждой колонке:		**-**	**4**	**6**	**4**
Шаг 2. Полученные значения умножьте на соответствующий коэффициент:		**x1**	**x2**	**x3**	**x4**
Шаг 3. Сложите полученные числа:		**-**	**8**	**18**	**16**
Шаг 4. Общий показатель равен:		**42**			

! После каждой шкалы есть интерпретация выраженности характеристик у обучающегося.

ШКАЛА I. ПОЗНАВАТЕЛЬНЫЕ ХАРАКТЕРИСТИКА УЧЕНИКА.

№	Утверждение	Выберите цифру			
1.	Обладает необычно большим для этого возраста запасом слов; использует термины с пониманием; речь отличается богатством выражений, беглостью, сложностью.	**1**	**2**	**3**	**4**
2.	Обладает большим запасом информации по разнообразным темам (выходящим за пределы обычных интересов детей этого возраста).	**1**	**2**	**3**	**4**
3.	Быстро запоминает и воспроизводит фактическую информацию.	**1**	**2**	**3**	**4**
4.	Легко схватывает причинно-следственные связи; пытается понять «как» и «почему»; задает много стимулирующих мысль вопросов (в отличие от вопросов, направленных на получение фактов); хочет знать, что лежит в основе явлений или действий людей.	**1**	**2**	**3**	**4**
5.	Чуткий и сметливый наблюдатель; обычно "видит больше" или "извлекает больше", чем другие, из рассказа, фильма, из того, что происходит.	**1**	**2**	**3**	**4**
Шаг 1. Подсчитайте число обведенных цифр по каждой колонке:					
Шаг 2. Полученные значения умножьте на соответствующий коэффициент:		**x1**	**x2**	**x3**	**x4**
Шаг 3. Сложите полученные числа:					

Шаг 4. Общий показатель равен:	

Интерпретация Шкалы I:

0 - 10 баллов – низкий показатель

11 – 20 баллов – пониженный показатель

21 – 40 баллов – средний показатель

41-65 баллов – повышенный показатель

66-80 баллов – высокий показатель

ШКАЛА II. МОТИВАЦИОННЫЕ ХАРАКТЕРИСТИКИ

№	Утверждение	Выберите цифру			
1.	Полностью уходит» в определенные темы, проблемы; настойчиво стремиться к завершению начатого (трудно привлечь к другой теме, заданию).	**1**	**2**	**3**	**4**
2.	Легко впадает в скуку от обычных заданий.	**1**	**2**	**3**	**4**
3.	Стремиться к совершенству; отличается самокритичностью.	**1**	**2**	**3**	**4**
4.	Предпочитает работать самостоятельно; требует лишь минимального направления от педагога.	**1**	**2**	**3**	**4**
5.	Имеет склонность организовывать людей, предметы, ситуацию.	**1**	**2**	**3**	**4**
Шаг 1. Подсчитайте число обведенных цифр по каждой колонке:					
Шаг 2. Полученные значения умножьте на соответствующий коэффициент:		**x1**	**x2**	**x3**	**x4**
Шаг 3. Сложите полученные числа:					
Шаг 4. Общий показатель равен:					

Интерпретация Шкалы II:

0 - 10 баллов – низкий показатель

11 – 20 баллов – пониженный показатель

21 – 40 баллов – средний показатель

41-65 баллов – повышенный показатель

66-80 баллов – высокий показатель

ШКАЛА III. ЛИДЕРСКИЕ ХАРАКТЕРИСТИКИ.

№	Утверждение	Выберите цифру			
1.	Проявляет ответственность; делает то, что обещает и обычно делает хорошо.	**1**	**2**	**3**	**4**
2.	Уверенно чувствует себя как с ровесниками, так и со взрослыми; хорошо себя чувствует, когда его просят показать свою работу группе.	**1**	**2**	**3**	**4**
3.	Ясно выражает свои мысли и чувства; хорошо и обычно понятно говорит.	**1**	**2**	**3**	**4**
4.	Любит находится с людьми, общителен и предпочитает не оставаться в одиночестве.	**1**	**2**	**3**	**4**
5.	Имеет склонность доминировать среди других; как правило, руководит деятельностью, в которой участвует.	**1**	**2**	**3**	**4**
Шаг 1. Подсчитайте число обведенных цифр по каждой колонке:					
Шаг 2. Полученные значения умножьте на соответствующий коэффициент:		**x1**	**x2**	**x3**	**x4**
Шаг 3. Сложите полученные числа:					
Шаг 4. Общий показатель равен:					

Интерпретация Шкалы III:

0 - 10 баллов – низкий показатель

11 – 20 баллов – пониженный показатель

21 – 40 баллов – средний показатель

41-65 баллов – повышенный показатель

66-80 баллов – высокий показатель

ШКАЛА IV. ТВОРЧЕСКИЕ ХАРАКТЕРИСТИКИ.

№	Утверждение	Выберите цифру			
1.	Проявляет большую любознательность в отношении многого; постоянно задает обо всем вопросы.	**1**	**2**	**3**	**4**
2.	Выдвигает большое количество идей или решений проблем и ответов на вопросы; предлагает необычные, оригинальные, умные ответы.	**1**	**2**	**3**	**4**
3.	Выражает свое мнение без колебаний; иногда радикален и горяч в дискуссиях, настойчив.	**1**	**2**	**3**	**4**
4.	Любит рисковать, имеет склонность к приключениям.	**1**	**2**	**3**	**4**

5.	Склонность к игре с идеями; фантазирует, придумывает («Интересно, что будет, если…»), занят приспособлением, улучшением и изменением общественных институтов, предметов и систем.	**1**	**2**	**3**	**4**
6.	Проявляет тонкое чувство юмора и видит юмор в таких ситуациях, которые не кажутся смешными остальным.	**1**	**2**	**3**	**4**
7.	Необычно чувствителен к внутренним импульсам и более открыт к иррациональному в себе (более свободное выражение «девчоночьих» интересов у мальчиков, большая независимость у девочек); эмоциональная чувствительность.	**1**	**2**	**3**	**4**
8.	Чувствителен к прекрасному; обращает внимание на эстетические стороны жизни.	**1**	**2**	**3**	**4**
9.	Не подвержен влиянию группы; приемлет беспорядок; не интересуется деталями; не боится быть отличным от других.	**1**	**2**	**3**	**4**
10.	Дает конструктивную критику; не склонен принимать авторитеты без критического изучения.	**1**	**2**	**3**	**4**
Шаг 1. Подсчитайте число обведенных цифр по каждой колонке:					
Шаг 2. Полученные значения умножьте на соответствующий коэффициент:		**x1**	**x2**	**x3**	**x4**
Шаг 3. Сложите полученные числа:					
Шаг 4. Общий показатель равен:					

Интерпретация Шкалы IV:

0 - 30 баллов – низкий показатель

31 – 45 баллов – пониженный показатель

46 – 65 баллов – средний показатель

76 - 100 баллов – повышенный показатель

101 - 160 баллов – высокий показатель

Приложение 8

Анкета «Оценка уровня учебной мотивации

Автор: Лусканова Н.Г.

Возраст детей: неограничен

Цель: определение школьной мотивации.

Анкета «Оценка уровня школьной мотивации» может быть использована при индивидуальной диагностике, а также применяться для групповой диагностики. Вопросы адаптированы для использования в организациях дополнительного образования. При этом допустимы два варианта предъявления:

• Вопросы читаются экспериментатором вслух, предлагаются варианты ответов, а дети должны написать те ответы, которые им подходят.

• Анкеты в напечатанном виде раздаются всем ученикам, и экспериментатор просит их отметить все подходящие ответы.

Каждый вариант имеет свои преимущества и недостатки. При первом варианте выше фактор лжи, так как дети видят перед собой взрослого, задающего вопросы. Второй вариант предъявления позволяет получить более искренние ответы, но такой способ затруднен для детей 7-8 лет, так как дети еще плохо читают.

Анкета допускает повторные вопросы, что позволяет оценить динамику мотивации. Снижение уровня мотивации может служить критерием дезадаптации ребенка в группе, а ее повышение – положительной динамике в обучении и развитии.

Инструкция:

Инструкция для индивидуальной формы работы: «Сначала послушай вопрос и три варианта ответа на этот вопрос, а затем выбери один из трёх ответов, который выражает твоё мнение»

Инструкция для групповой формы работы: «Прочитайте вопрос и из предложенных вариантов ответа выберите один и отметьте его буквенное значение на бланке ответов».

Вопросы анкеты:

1) Тебе нравится заниматься _____ (название обучающей программы)?
   1. не очень
   2. нравится
   3. не нравится
2) Ты с радостью идешь на дополнительные занятия (название кружка/программы) или тебе часто хочется остаться дома?
   1. чаще хочется остаться дома
   2. бывает по-разному
   3. иду с радостью
3) Если бы педагог сказал, что завтра на занятия не обязательно приходить всем ребятам, что желающие могут остаться дома, ты бы пошел на занятия или предпочел остаться дома?
   1. не знаю
   2. остался бы дома
   3. пошел бы в школу
4) Тебе нравится, когда у вас отменяют занятия по _____ (название программы)?
   1. не нравится
   2. бывает по-разному
   3. нравится
5) Ты хотел бы, чтобы педагог не давал самостоятельной работы домой?
   1. хотел бы
   2. не хотел бы
   3. не знаю
6) Ты хотел бы, чтобы занятия стали короче?
   1. не знаю
   2. не хотел бы
   3. хотел бы
7) Ты часто рассказываешь родителям о том, что происходит на занятиях?

1. часто
2. редко
3. не рассказываю

8) Ты хотел бы, чтобы твой педагог был менее строгим?

1. точно не знаю
2. хотел бы
3. не хотел бы

9) У тебя в группе много друзей?

1. мало
2. много
3. нет друзей

10) Тебе нравятся ребята, с которыми ты посещаешь занятия _____?

1. нравятся
2. не очень
3. не нравятся

Обработка и интерпретация результатов:

Ключ

Количество баллов, которые можно получить за каждый из трех ответов на вопросы анкеты.

№ вопроса	**оценка за 1-й ответ**	**оценка за 2-й ответ**	**оценка за 3-й ответ**
1	1	3	0
2	0	1	3
3	1	0	3
4	3	1	0
5	0	3	1
6	1	3	0
7	3	1	0
8	1	0	3
9	1	3	0
10	3	1	0

Первый уровень. 25-30 баллов – высокий уровень мотивации, учебной активности.

У таких детей есть познавательный мотив, стремление наиболее успешно выполнять все предъявляемые педагогом требования. Ребята четко следуют всем указаниям педагога, добросовестны и ответственны, сильно переживают, если получают неудовлетворительные результаты.

Второй уровень. 20-24 балла – хорошая мотивация.

Подобные показатели имеют большинство ребят 7-8 лет, успешно справляющихся с образовательной деятельностью. Подобный уровень мотивации является средней нормой.

Третий уровень. 15-19 баллов – положительное отношение к дополнительным занятиям, но их больше привлекает проектная или похожая деятельность, нежели сами занятия.

Такие дети достаточно благополучно чувствуют себя на дополнительных занятиях, однако посещают их, чтобы общаться с друзьями, с педагогом. Познавательные мотивы у таких детей сформированы в меньшей степени, и учебный процесс их мало привлекает.

Четвертый уровень. 10-14 баллов – низкая мотивация.

Эти дети посещают программы дополнительного образования неохотно, предпочитают пропускать занятия. На занятиях часто занимаются посторонними делами, играми. Находятся в состоянии неустойчивой адаптации, испытывают серьезные затруднения в учебной деятельности.

Пятый уровень. Ниже 10 баллов – негативное отношение к дополнительным занятиям, учебная дезадаптация.

Такие дети испытывают серьезные трудности в обучении: они не справляются с учебной деятельностью, испытывают проблемы в общении с ребятами, во взаимоотношениях с педагогом. Занятия в объединении/секции, нередко воспринимается ими как враждебная среда, пребывание в которой для них невыносимо. Маленькие дети (5-6 лет) часто плачут, просятся домой. В других случаях ребята могут проявлять агрессию, отказываться выполнять задания, следовать тем или иным нормам и правилам. Часто у подобных детей отмечаются нервно-психические нарушения.

Приложение 9

Мониторинг достижения обучающимися личностных результатов

за 20__-20_ учебный год

№ п/п	Ф.И. обучающегося	возраст	Ответственное отношение к обучению			коммуникативная компетентность в общении и сотрудничестве со сверстниками			ответственное отношение к обучению, способность довести до конца начатое дело			целостное мировоззрение, соответствующее современному уровню развития информационных технологий			Усвоение правил индивидуального и коллективного безопасного поведения при работе с компьютерной техникой.		
			входящий	промежуточный	итоговый	входящий	промежуточный	итоговый	входящий	промежуточный	итоговый	входящий	промежуточный	итоговый	входящий	промежуточный	итоговый
1																	
2																	
3																	
4																	
5																	
6																	
7																	
8																	
9																	
10																	

1 балл – низкий уровень;
2 балла – средний уровень;
3 балла – высокий уровень

Приложение 10

Мониторинг достижения обучающимися метапредметных результатов

за 20__-20_ учебный год

№ п/п	Ф.И. обучающегося	возраст	умение самостоятельно планировать пути решения поставленной проблемы для получения эффективного результата;			умение самостоятельно ставить и формулировать для себя новые задачи, развивать мотивы своей познавательной деятельности;			умение критически оценивать правильность решения учебно-исследовательской задачи;			умение формулировать, аргументировать и отстаивать своё мнение;			умение извлекать нужную информацию из открытых источников;			умение организовывать учебное сотрудничество и совместную деятельность с педагогом и сверстниками в процессе проектной и учебно-исследовательской деятельности		
			входящий	промежуточный	итоговый	входящий	промежуточный	итоговый	входящий	промежуточный	итоговый	входящий	промежуточный	итоговый	входящий	промежуточный	итоговый	входящий	промежуточный	итоговый
1																				
2																				
3																				
4																				
5																				
6																				
7																				

1 балл – низкий уровень;
2 балла – средний уровень;
3 балла – высокий уровень

Приложение 11

Мониторинг успешности обучающихся

п/п	Ф.И. обучающегося	возраст	результаты участия обучающегося в очных конкурсах (результат, название конкурса)					результаты участия обучающегося в дистанционных конкурсах (результат, название конкурса)					результаты участия обучающегося во Всероссийской олимпиаде школьников			количественный анализ участия обучающегося в конкурсах и олимпиадах					
			Муниципальный уровень	Региональный уровень	Межрегиональный уровень	Федеральный уровень	Международный уровень	Муниципальный уровень	Региональный уровень	Межрегиональный уровень	Федеральный уровень	Международный уровень	муниципальный уровень	региональный уровень	всероссийский уровень	муниципальный уровень	Региональный уровень	межрегиональный уровень	федеральный уровень	международный уровень	всего

Список литературы

Нормативные документы:

1. Федеральный закон «Об образовании в Российской Федерации» от 29.12.2012 N 273-ФЗ.

2. Стратегия развития воспитания в Российской Федерации на период до 2025 года. Распоряжение Правительства Российской Федерации от 29 мая 2015 г. № 996-р

3. Письмо Министерства образования и науки РФ от 18.11.2015г. № 09-3242. «О направлении Методических рекомендаций по проектированию дополнительных общеразвивающих программ (включая разноуровневые)»

4. Распоряжение правительства РФ от 04.09. 2014 № 1726-р «Об утверждении Концепции развития дополнительного образования детей»

5. «Основы законодательств РФ об охране здоровья граждан», утвержденные Верховным советом РФ от 22.07.1993 № 5487 - (ред. от 25.11.2009);

6. Федеральный закон от 24.07.1998 № 124-ФЗ «Об основных гарантиях прав ребёнка в РФ»;

7. Федеральный закон от «Об основах охраны здоровья граждан в Российской Федерации», 2011г.

8. Приказ Министерства просвещения России от 09.11.2018 г. № 196 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам» (Приказ №1008 отменен).

9. Санитарно-эпидемиологические правила и нормативы СанПиН 2.4.4.3172-14.

Список литературы, использованной при написании программы:

1. Блох Джошуа. Java. Эффективное программирование. Effective Java. Programming Language Guide. изд. «Лори». 2014. – 310 с. ISBN 978-5-85582-347-9.

2. Гослинг Джеймс, Билл Джой, Гай Л. Стил, Гилад Брача, Алекс Бакли. Язык программирования Java SE 8. Подробное описание. The Java Language Specification: Java SE8 Edition.изд. «Вильямс». 2015. – 672 с. ISBN 978-5-8459-1875-8, 978-0-13-

390069-9.

3. Зигард Медникс, Лайрд Дорнин, Блейк Мик, Масуми Накамура. Программирование под Android. Programming Android. изд. Питер. 2012. – 496 с. ISBN 978-5-459-01115-9, 978-1-449-38969-7.

4. Майер Рето. Android 2. Программирование приложений для планшетных компьютеров и смартфонов. Professional Android 2: Application Developmentecond Edition. изд. Эксмо. 2011 г. 672 стр. ISBN 978-5-699-50323-0.

5. Харди Брайн, Билл Филлипс.Программирование под Android. Android Programming: The Big Nerd Ranch Guide. изд. Питер. 2014. – 592 с. ISBN 978-5-496-00502-9, 978-0-321-80433-4.

6. Эльконин Д.Б. Детская психология: учеб. пособие для студ. высш. учеб. заведений / Д. Б. Эльконин; ред.-сост. Б. Д. Эльконин. — 4-е изд., стер. — М.: Издательский центр «Академия», 2007. — 384 с

Электронные ресурсы:

1. Портал обучения (https://informatics.msk.ru/)

2. Портал разработчика Android (https://developer.android.com/)

Аннотация

Задача инновационного развития программного обеспечения требует соответствующей образовательной среды, в том числе создания оптимальных условий детского технического творчества. Одной из наиболее инновационных областей в сфере детского технического творчества является мобильная разработка.

Для дальнейшего развития мобильных приложений существует широкий выбор направлений разработки. Каждому ребёнку интересно, как устроена Java платформа, как работает Java приложение на любой платформе и на смартфоне в том числе.

Изучение языка программирования Java по данной программе обучения даёт возможность пользователю освоить базовые навыки использования языка программирования, понять его особенности использования и выполнения на различных платформах.

Разработка мобильных приложений на базе Android на сегодняшний день. очень востребована ввиду высокой популярности данной ОС. Поэтому обучение по данной программе – это самый первый, но важный шаг в изучении основ программирования на языке Java, для создания проектов и простейших программ в среде разработки на его основе.

Программа рассчитана на обучающихся 13–17 лет.

Дополнительная общеобразовательная общеразвивающая программа реализуется в сетевой форме. ГАНОУ СО «Дворец молодёжи» является базовой организацией, организация-участник определяется на основании заключенного договор о сетевой форме реализации программ.