

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ
ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ

ЦЕНТР ДЕТСКОГО (ЮНОШЕСКОГО) ТЕХНИЧЕСКОГО ТВОРЧЕСТВА
КИРОВСКОГО РАЙОНА САНКТ-ПЕТЕРБУРГА

Программа принята
на педагогическом совете
ГБУ ДО ЦДЮТТ

« 30 » августа 2016 г.



«Утверждаю»
Директор ГБУ ДО ЦДЮТТ

Ясинская Е.С.

« 05 » сентября 2016 г.

ДОПОЛНИТЕЛЬНАЯ ОБЩЕОБРАЗОВАТЕЛЬНАЯ
ОБЩЕРАЗВИВАЮЩАЯ ПРОГРАММА

«ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ JAVA»

Срок реализации программы: 3 года

Возраст учащихся: 13 – 17 лет

Автор-составитель: Боголюбов Д.А., к.т.н.,
педагог дополнительного образования ГБУ ДО ЦДЮТТ

Санкт-Петербург
2016

Пояснительная записка

Современный мир невозможно представить без программирования и его продукции во всём её многообразии. Автоматизация коренным образом изменила большую часть отраслей человеческой деятельности. Программисты, инженеры по автоматизации и системные администраторы неизменно присутствуют в списке самых востребованных специальностей на рынке образования и труда.

Российское образование в этой области не уступает в качестве зарубежным конкурентам. Российские студенты выигрывают престижные международные соревнования по программированию, высоко развит международный обмен специалистами.

Программирование на языке Java – относительно молодое и бурно развивающееся направление. На этом языке пишутся программы для стационарных компьютеров и приложения для мобильных устройств, что стало особенно актуальным в последнее десятилетие.

Программа для объединений дополнительного образования по созданию сайтов и веб-программированию составлена в соответствии с Законом Российской Федерации «Об образовании» в редакции Федерального Закона от 29.12.2012, № 273-ФЗ, Типового положения об образовательном учреждении дополнительного образования детей, нормативных документов Министерства образования РФ.

Направленность образовательной программы - техническая.

Программа содержит научно обоснованные рекомендации по структуре и организации учебного процесса по обучению школьников объектно-ориентированному программированию на различных этапах многолетней подготовки.

Актуальность и педагогическая целесообразность программы

Необходимость разработки данной программы обусловлена острым недостатком квалифицированных специалистов по объектно-ориентированному программированию и востребованностью данного направления со стороны старшеклассников.

Занятия объектно-ориентированным программированием на языке Java позволяют усовершенствовать навыки алгоритмизации, развить абстрактное мышление и приобрести востребованные на рынке труда навыки.

Отличительная особенность представляемой образовательной программы состоит в организации многогодичного процесса обучения объектно-ориентированному программированию под различные платформы. На 1 году обучения создаются приложения для стационарных компьютеров, 2 и 3 год посвящены программированию под мобильные устройства. Для начала обучения по данной программе учащемуся необходимо освоить любой язык процедурного программирования.

В учебном процессе целесообразно использовать новейшие версии компиляторов NetBeans (1 год) и Android Studio (2-3 год обучения). Важно знакомить учащихся с размещёнными в Интернете пособиями и видеоуроками, в т.ч. на английском языке.

Цель программы – развитие абстрактного мышления и информационной культуры учащихся путём вовлечения их в процесс создания современных программных продуктов средствами объектно-ориентированного программирования.

Задачи программы

Обучающие (для ОП на 72 ч):

- познакомить с синтаксисом языка Java;
- научить работать в среде компиляторов NetBeans и Android Studio;
- ознакомить учащихся с принципами объектно-ориентированного программирования;
- научить работать с готовыми библиотеками Java;
- научить разрабатывать приложения для мобильных устройств;
- ознакомить с основными течениями в разработке современных приложений;
- создать условия для достижения учащимися оптимального уровня знаний и умений в области объектно-ориентированного программирования.

Обучающие (для ОП на 144 ч):

- познакомить с синтаксисом языка Java;
- научить работать в среде компиляторов NetBeans и Android Studio;
- ознакомить учащихся с принципами объектно-ориентированного программирования;
- научить работать с готовыми библиотеками Java;

- ознакомить с основными течениями в разработке современных приложений;
- научить разрабатывать приложения для мобильных устройств;
- научить создавать читаемый программный код с использованием комментариев;
- создать условия для достижения учащимися оптимального уровня знаний и умений в области объектно-ориентированного программирования;
- познакомить с технологиями продвижения своего программного продукта.

Обучающие (для ОП на 216 ч.):

- познакомить с синтаксисом языка Java;
- научить работать в среде компиляторов NetBeans и Android Studio;
- ознакомить учащихся с принципами объектно-ориентированного программирования;
- научить работать с готовыми библиотеками Java;
- ознакомить с основными течениями в разработке современных приложений;
- научить разрабатывать приложения для мобильных устройств;
- научить создавать читаемый программный код с использованием комментариев;
- создать условия для достижения учащимися оптимального уровня знаний и умений в области объектно-ориентированного программирования;
- познакомить с технологиями продвижения своего программного продукта;
- научить использовать технологии парсинга данных.

Воспитательные (одинаковы для ОП на 72 ч., 144 ч. и 216 ч.):

- воспитывать целеустремленность в работе;
- формировать чувство ответственности за принимаемые решения;
- воспитывать коммуникативные навыки.

Развивающие (одинаковы для ОП на 72 ч., 144 ч. и 216 ч.):

- развивать информационную культуру учащихся;
- формировать творческий подход к решению задач;
- развивать способность к самообразованию;
- развивать навык адекватной самооценки своей работы.

Сроки реализации образовательной программы; режим занятий; характеристика детей, участвующих в реализации образовательной программы

Программа рассчитана на 3 года обучения для учащихся 13-17 лет (1 год – 13-15 лет, 2 год – 14-16 лет, 3 год – 15-17 лет).

Программа реализуется в объёме от 72 до 216 часов в год при режиме занятий от 2 до 6 часов в неделю соответственно (режим занятий – 1-3 раза в неделю по 2 ч).

Наполняемость группы: на 1 году обучения – 11 человек, 2 год – 11 человек, 3 год – 11 человек).

При записи в объединение приоритет отдаётся учащимся, владеющим любым языком процедурного программирования. Пол значения не имеет.

Программа второго года обучения рассчитана на учащихся, прошедших начальную подготовку по данной программе в объеме программы 1-го года обучения. Также возможен прием детей на обучение по программе 2-го года по результатам собеседования.

Программа третьего года обучения рассчитана на учащихся, прошедших подготовку по данной программе в объеме программы 1-го и 2-го года обучения.

В зависимости от уровня подготовки учащихся и заинтересованности в отдельных вопросах, количество часов, отведенное на определенные темы, может варьироваться в пределах общего количества часов настоящей программы. Также может варьироваться количество часов, отведенное на учебно-массовые мероприятия.

Ожидаемые результаты образовательной программы

По окончании обучения по программе учащийся должен:

Знать (для ОП на 216 ч, 144 ч., 72 ч.):

- структуру программы на языке Java;
- основные команды языка Java;
- парадигму объектно-ориентированного программирования;
- принципы формирования программных пакетов и приложений;
- общую структуру ОС Windows и Android;
- интерфейс программирования приложений ОС Android;
- содержимое базовых библиотек языка Java;

Знать (для ОП на 72 ч.):

- структуру программы на языке Java;

- основные команды языка Java;
- парадигму объектно-ориентированного программирования;
- принципы формирования программных пакетов и приложений;
- общую структуру ОС Windows и Android;
- общие сведения об интерфейсе программирования приложений ОС Android;

Уметь (для ОП на 72 ч):

- создавать классы, апплеты, программы, приложения;
- писать структурированный код, сопровождая его необходимыми комментариями;
- пользоваться встроенными и сетевыми библиотеками;
- работать в среде компиляторов NetBeans и Android Studio;
- организовывать пакеты приложений;
- создавать документацию к программе;
- проектировать графический интерфейс приложения;

Уметь (для ОП на 144 ч):

- создавать классы, апплеты, программы, приложения;
- писать структурированный код, сопровождая его необходимыми комментариями;
- пользоваться встроенными и сетевыми библиотеками;
- работать в среде компиляторов NetBeans и Android Studio;
- организовывать пакеты приложений;
- проводить тестирование программного продукта;
- создавать документацию к программе;
- проектировать графический интерфейс приложения;

Уметь (для ОП на 216 ч):

- создавать классы, апплеты, программы, приложения;
- писать структурированный код, сопровождая его необходимыми комментариями;
- пользоваться встроенными и сетевыми библиотеками;
- работать в среде компиляторов NetBeans и Android Studio;
- организовывать пакеты приложений;
- проводить тестирование программного продукта;
- создавать документацию к программе;
- проектировать графический интерфейс приложения;
- продвигать свой программный продукт через сетевые репозитории.

Быть:

- способным проявлять творческий подход при решении задач;
- целеустремленным, ответственным в работе;
- способным к адекватной и эффективной коммуникации в группе;
- способным к адекватной самооценке своей работы.

Формы занятий

Основными формами занятий по данной программе являются практическое и комбинированное занятия (сочетание теоретической и практической частей).

Способы и формы проверки ожидаемых результатов реализации образовательной программы

Контроль за освоением учебного материала проводится в форме практических работ, опросов.

Кроме вышесказанного, минимум три раза в год делается так называемый "срез" знаний, умений и личностных характеристик учащегося. При этом заполняется диагностическая карта результативности учебно-воспитательного процесса, в которой оцениваются уровень знаний и умений учащегося, а также его личностные характеристики. В карте отмечается степень самостоятельности при выполнении заданий, уровень сложности заданий, качество исполнения, участие в конкурсах и соревнованиях, культура поведения, дисциплинированность, степень активности учащегося и др. По данной карте прослеживается динамика формирования личности учащегося, динамика изменения образовательного уровня. Данные из карты используются для оценивания работы педагога, для оценивания результативности образовательной программы, при работе с родителями.

Формы подведения итогов реализации образовательной программы

В конце каждого года обучения учащиеся должны предоставить авторское приложение, разработанное в рамках выпускной индивидуальной работы.

Учебно-воспитательная работа

В течение каждого занятия педагог в ненавязчивой форме воспитывает патриотическое отношение к своему городу, уважение к старшему поколению, к своим товарищам, к труду.

Дети участвуют в учебно-массовых мероприятиях, проводимых в рамках ЦДЮТТ Кировского района, района и города (конкурсы и др.), что позволяет учащимся дополнительно приобрести навыки общения, еще более расширить кругозор и почувствовать значимость своей деятельности. Приоритетным является участие в ежегодном городском конкурсе по программированию и компьютерным работам.

План учебно-массовых мероприятий составляется ежегодно.

**УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН 1 ГОДА ОБУЧЕНИЯ
НА 216 ЧАСОВ**

Тема занятий	Часы		
	Теория	Практика	Всего
Введение	1	1	2
Основы синтаксиса	8	16	24
Знакомство с ООП	6	6	12
Классы и объекты	4	18	22
Работа с файлами	6	10	16
Оконные приложения	16	26	42
Обработка исключений	6	8	14
Парсинг	12	26	38
Выпускная индивидуальная работа	0	38	38
Учебно-массовые мероприятия	0	8	8
Всего часов	59	157	216

**УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН 1 ГОДА ОБУЧЕНИЯ
НА 144 ЧАСА**

Тема занятий	Часы		
	Теория	Практика	Всего
Введение	1	1	2
Основы синтаксиса	6	14	20
Знакомство с ООП	4	2	6
Классы и объекты	3	11	14
Оконные приложения	12	16	28
Обработка исключений	6	6	12
Работа с файлами	8	16	24
Выпускная индивидуальная работа	0	30	30
Учебно-массовые мероприятия	0	8	8
Всего часов	40	104	144

УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН 1 ГОДА ОБУЧЕНИЯ
НА 72 ЧАСА

Тема занятий	Часы		
	Теория	Практика	Всего
Введение	1	1	2
Основы синтаксиса	3	7	10
Знакомство с ООП	4	2	6
Классы и объекты	2	6	8
Оконные приложения	6	8	14
Обработка исключений	4	4	8
Выпускная индивидуальная работа	0	16	16
Учебно-массовые мероприятия	0	8	8
Всего часов	20	52	72

Задачи первого года обучения

Программа направлена на решение следующих задач.

Обучающие:

- обучить правилам охраны труда при работе за компьютером;
- познакомить со структурой класса и пакета в среде программирования на Java;
- обучить базовым конструкциям языка: операторами ветвления, циклами и т.д.;
- познакомить с различными типами данных;
- познакомить с парадигмой объектно-ориентированного программирования;
- обучить базовым приемам объектно-ориентированного программирования на языке Java;
- научить обработке исключений и работе с внешними файлами;
- познакомить с семантической разметкой средствами языка XML (для ОП на 216 час.)

Воспитательные:

- воспитывать целеустремленность в работе;
- формировать чувство ответственности за принимаемые решения;
- воспитывать коммуникативные навыки.

Развивающие:

- содействовать преодолению психологического барьера перед программированием как особым видом деятельности;
- формировать творческий подход к решению задач;
- развивать способность к самообразованию.

Ожидаемые результаты освоения программы первого года обучения

Учащийся должен

Знать:

- базовый синтаксис языка Java;
- структуру класса и пакета;
- понятия класса и объекта, свойства и метода
- правила обработки исключений различных типов;
- технологию проектирования оконного приложения;
- потоки файлового ввода-вывода (для ОП на 144 и 216 час.);

- способы извлечения данных из сторонних файлов средствами Java (только для ОП на 216 час.)

Уметь:

- создавать классы и программы на языке Java;
- писать грамотный, хорошо структурированный код, сопровождая его комментариями;
- создавать типовые приложения;
- работать с потоками ввода-вывода;
- проектировать простейшие графические интерфейсы оконных приложений;
- подключать и использовать в работе приложения внешние хранилища данных (для ОП на 144 и 216 час.);
- применять различные средства извлечения данных из файлов произвольного формата (для ОП на 216 час.)

Быть:

- способным проявлять творческий подход при решении задач;
- целеустремленным, ответственным в работе;
- способным к адекватной и эффективной коммуникации в группе.

СОДЕРЖАНИЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

1 год обучения

Количество учебных часов в образовательной программе на 216 часов увеличивается по сравнению с образовательной программой на 144 часа и 72 часа, в основном, за счёт практической работы, а также за счёт темы «Работа с файлами» в ОП на 144 часа и темы «Парсинг» в ОП на 216 часов. Выполняется большее количество заданий; уделяется больше времени на тщательность выполнения отдельных практических работ. Кроме того, в образовательной программе на 216 часов упор делается на формирование самостоятельности и творческого подхода при выполнении учебных задач.

Введение

Знакомство с коллективом; организационные вопросы. Первичный инструктаж по охране труда и технике безопасности.

История языка Java. Графический интерфейс среды программирования NetBeans.

Структура класса в Java

Практическая работа:

Создание рабочего пространства на компьютере: первый проект и класс.

Основы синтаксиса

Примитивные типы данных. Операторы ветвления и множественного ветвления.

Циклы со счётчиком, постусловием и предусловием. Классы и библиотеки по работе с датой и временем. Алгоритмы сортировки массивов.

Практическая работа:

Создание пакетов «Калькулятор», игры «Mastermind». Обработка простых потоков ввода-вывода.

Знакомство с ООП

Парадигма объектно-ориентированного программирования. Понятия полиморфизма, инкапсуляции. Иерархическая структура классов.

Практическая работа:

Проектирование и реализация приложения «Инженер». Создание простейших текстовых баз данных.

Классы и объекты

Классы-оболочки. Встроенные классы пакетов java.io, java.util.

Практическая работа:

Программы «Вычисление рабочих дней», «Календарь». Сортировка разноформатных данных.

Работа с файлами

Потоки файлового ввода-вывода. Буферизованные пакеты. Исключения при работе с файлами.

Практическая работа:

Создание класса «Регистрация в системе».

Проектирование комплексного пакета приложений «База данных кадров предприятия».

Оконные приложения

Элементы графического оконного интерфейса: кнопки, поля ввода, списки и пр.

Форма JFrame. Библиотека SwingGUI.

Практическая работа:

Приложение «Оконный калькулятор». Приложение «Фотобанк».

Обработка исключений

Операторы try catch, throw. Арифметические исключения, ошибки доступа.

Встраивание статических методов.

Практическая работа:

Доработка ранее разработанных приложений с использованием класса Exception и полученных сведений об исключениях.

Парсинг

Текстовые базы данных. Организация файлового поиска средствами языка Java.

Введение в язык XML. Семантическая разметка текстовых данных.

Практическая работа:

Разработка оконного приложения «Англо-русский словарь» с использованием данных, полученных из сторонних источников.

Выпускная индивидуальная работа

Практическая работа:

Разработка авторского приложения на индивидуальную тему (наиболее удачные проекты отправляются на различные конкурсы по программированию).

Учебно-массовые мероприятия

Учебно-массовые мероприятия проводятся по тематике объединения в рамках объединения: викторины, конкурсы. План мероприятий составляется ежегодно.

Учащиеся мотивируются к участию в городских и всероссийских конкурсах по программированию и компьютерным работам.

**УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН 2 ГОДА ОБУЧЕНИЯ
НА 216 ЧАСОВ**

Тема занятий	Часы		
	Теория	Практика	Всего
Введение	2	-	2
Повторение	4	8	12
Архитектура ОС Android	10	22	32
Интерфейс программирования приложений Android	6	12	18
Экран приложения	8	16	24
Графика	6	18	24
Асинхронные задачи	8	16	24
Анимация элементов экрана	12	24	36
Выпускная творческая работа	2	34	36
Учебно-массовые мероприятия	-	8	8
Всего часов	58	158	216

**УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН 2 ГОДА ОБУЧЕНИЯ
НА 144 ЧАСА**

Тема занятий	Часы		
	Теория	Практика	Всего
Введение	2	-	2
Повторение	3	7	10
Архитектура ОС Android	8	12	20
Интерфейс программирования приложений Android	4	8	12
Экран приложения	4	8	12
Графика	4	12	16
Асинхронные задачи	6	10	16
Анимация элементов экрана	8	16	24
Выпускная творческая работа	2	24	26
Учебно-массовые мероприятия		6	6
Всего часов	41	103	144

УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН 2 ГОДА ОБУЧЕНИЯ НА 72 ЧАСА

Тема занятий	Часы		
	Теория	Практика	Всего
Введение	2	-	2
Повторение	2	6	8
Архитектура ОС Android	4	6	10
Интерфейс программирования приложений Android	2	4	6
Экран приложения	2	4	6
Графика	2	4	6
Анимация элементов экрана	4	8	12
Выпускная творческая работа	2	12	14
Учебно-массовые мероприятия	-	8	8
Всего часов	20	52	72

Задачи второго года обучения

Программа направлена на решение следующих задач.

Обучающие:

- познакомить с устройством операционной системы Android;
- обучить программированию простых однопользовательских приложений под Android;
- объяснить основы композиции экрана приложения;
- подготовить к созданию авторского приложения.

Воспитательные:

- воспитывать целеустремленность в работе;
- формировать чувство ответственности за принимаемые решения;
- воспитывать коммуникативные навыки.

Развивающие:

- развивать творческий подход к решению задач;
- развивать способность к самообразованию.
- развивать информационную культуру учащихся;
- развивать навык адекватной самооценки своей работы.

Ожидаемые результаты освоения программы второго года обучения

Учащийся должен

Знать:

- общую структуру ядра операционной системы Android;
- этапы развития операционной системы Android;
- основные понятия рабочего листа приложения и интерактивных его элементов;
- правила построения асинхронных компонентов ядра приложений;

Уметь:

- создавать простейшие приложения под операционную систему Android;
- проектировать и реализовывать интерактивные элементы приложений;
- проектировать и разрабатывать авторские графические элементы приложений.

Быть:

- способным проявлять творческий подход при решении задач;
- целеустремленным, ответственным в работе;
- способным к адекватной и эффективной коммуникации в группе;
- способным к адекватной самооценке своей работы.

Содержание образовательной программы

2 год обучения

Количество учебных часов в образовательной программе на 216 часов увеличивается по сравнению с образовательной программой на 144 часа и 72 часа, в основном, за счёт практической работы. Выполняется большее количество заданий; уделяется больше времени на тщательность выполнения отдельных практических работ. Кроме того, в образовательной программе на 216 часов упор делается на формирование самостоятельности и творческого подхода при выполнении учебных задач.

Введение

Организационные вопросы. Первичный инструктаж по охране труда и технике безопасности.

Нововведения языка Java.

Повторение

Общий синтаксис языка Java. Структура пакета. Классы приложения. Пакеты java.util, java.io.

Практическая работа

Создание приложений «Морской бой», «Инженерный калькулятор».

Архитектура ОС Android

История операционной системы Android. Этапы её развития. Ядро системы и его компоненты. Общие сведения о мобильных устройствах.

Практическая работа

Исследование и анализ существующих приложений.

Интерфейс программирования приложений Android

Уровни ОС Android. Создание приложений. Реализация интерфейсов. Управление ресурсами. Хранение информации. Доступ к аппаратным возможностям.

Практическая работа

Создание приложения «Список для чтения».

Экран приложения

Расположение Layout. Меню, адаптеры данных. Активности приложения и их виды. Способы переключения.

Отладка приложений.

Практическая работа

Создание приложения «Самопрезентация», «Расписание занятий».

Графика

Создание компонентов UI. Жизненный цикл компонента UI. Графика в Android. Анимация.

Практическая работа

Создание приложений «Анимация шара», «Цветовой тестер», «Флаг».

Асинхронные задачи (для ОП на 144 и 216 часов)

Создание и отмена задачи. Параметры асинхронной задачи. Поворот экрана и его отработка приложения. Загрузка файлов из локальной сети средствами асинхронных задач.

Практическая работа

Создание приложений «Управление объектом», «Анимация фигуры». Создание игры «Тетрис» (только в ОП на 216 часов).

Анимация элементов экрана

Изображение спрайта. Свойства компонентов. Программирование управляемых событий. Программирование движения. Анимация графических примитивов. Движение по траектории. Комбинированные эффекты.

Практическая работа

Создание мультфильма в формате приложения Android.

Выпускная творческая работа

Повторение теоретических вопросов, изученных ранее.

Практическая работа

Разработка авторского приложения на индивидуальную тему.

Учебно-массовые мероприятия

Учебно-массовые мероприятия проводятся по тематике объединения в рамках объединения: викторины, конкурсы. План мероприятий составляется ежегодно.

Учащиеся мотивируются к участию в городских и всероссийских конкурсах по программированию и компьютерным работам.

УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН 3 ГОДА ОБУЧЕНИЯ
НА 216 ЧАСОВ

Тема занятий	Часы		
	Теория	Практика	Всего
Введение	2	-	2
Повторение	4	8	12
Многоэкранные приложения	8	16	24
Мультимедийные данные	6	14	20
Картография	4	14	18
Камера и рисование	12	24	36
OpenGL	20	40	60
Выпускная творческая работа	2	34	36
Учебно-массовые мероприятия	-	8	8
Всего часов	58	158	216

УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН 3 ГОДА ОБУЧЕНИЯ
НА 144 ЧАСА

Тема занятий	Часы		
	Теория	Практика	Всего
Введение	2	-	2
Повторение	3	7	10
Многоэкранные приложения	6	12	18
Мультимедийные данные	4	8	12
Картография	4	8	12
Камера и рисование	8	14	22
OpenGL	12	24	36
Выпускная творческая работа	2	24	26
Учебно-массовые мероприятия	-	6	6
Всего часов	41	103	144

УЧЕБНО-ТЕМАТИЧЕСКИЙ ПЛАН 3 ГОДА ОБУЧЕНИЯ НА 72 ЧАСА

Тема занятий	Часы		
	Теория	Практика	Всего
Введение	2	-	2
Повторение	2	4	6
Многоэкранные приложения	4	6	10
Мультимедийные данные	2	6	8
Камера и рисование	2	6	8
OpenGL	4	6	10
Выпускная творческая работа	2	18	20
Учебно-массовые мероприятия	-	8	8
Всего часов	18	54	72

Задачи третьего года обучения

Обучающие:

- познакомить с работой с аппаратным обеспечением мобильного устройства;
- обучить работе с компьютерной графикой;
- обучить работе с мультимедийными данными в режиме реального времени;
- познакомить с принципами геопозиционирования и системой электронных карт Google (только для ОП на 144 и 216 часов).

Воспитательные:

- воспитывать целеустремленность в работе;
- формировать чувство ответственности за принимаемые решения;
- воспитывать коммуникативные навыки.

Развивающие:

- развивать творческий подход к решению задач;
- развивать способность к самообразованию;
- развивать навык адекватной самооценки своей работы.

Ожидаемые результаты освоения программы третьего года обучения

Учащийся должен

Знать:

- принципы строения многоэкранных приложений;
- понятие и принцип работы виджета;
- правила обработки мультимедийной информации в среде приложений;
- графическую библиотеку OpenGL.

Уметь:

- создавать многоэкранные приложения;
- обрабатывать данные разных форматов;
- работать с аппаратным обеспечением мобильного устройства – стилус, камера, микрофон;
- создавать трёхмерные графические элементы приложения.

Быть:

- способным проявлять творческий подход при решении задач;
- целеустремленным, ответственным в работе;
- способным к адекватной и эффективной коммуникации в группе;
- способным к адекватной самооценке своей работы.

Содержание программы для групп третьего года обучения

Количество учебных часов в образовательной программе на 216 часов увеличивается по сравнению с образовательной программой на 144 часа и 72 часа, в основном, за счёт практической работы, а также за счёт темы «Картография» в ОП на 144 и 216 часов. Выполняется большее количество заданий; уделяется больше времени на тщательность выполнения отдельных практических работ. Кроме того, в образовательной программе на 216 часов упор делается на формирование самостоятельности и творческого подхода при выполнении учебных задач.

Введение

Порядок и содержание работы объединения. Краткий исторический обзор новейших тенденций в сфере веб-дизайна и программирования. Характеристика современного состояния веб-дизайна как вида технического творчества.

Техника безопасности.

Повторение

Программирование приложений под Android. Строение приложений, архитектура операционной системы, интерфейс программирования приложений.

Практическая работа

Создание приложения «Кулинарная книга», «Расписание автобусов».

Многоэкранные приложения

Слайдинг. Обработка переключения экрана различными способами. Тач-интерфейс и особенности его реализации на мобильных устройствах различных производителей.

Демонстрация распознавания жестов.

Практическая работа

Создание приложения со слайдингом по шаблону компилятора. Создание приложения «Фотогалерея»

Медиаданные

Система управления базами данных SQLite. Поддерживаемые системы анимации, основные классы, используемые для добавления анимации в приложения, способы изображения графических объектов в Android. Интеграция элементов HTML 5.

Практическая работа

Создание мобильного приложения «Крестики-нолики» с веб-интерфейсом.

Картография

Методы определения местоположения. Работа с картами Google Maps. Внедрение карт в приложение. Управление картой в приложении. Добавление точек на карту. Определение местоположения без карт.

Практическая работа

Создание приложения на базе API СПб ГКУ «Организатор перевозок» и «Яндекс.Транспорт».

Камера и рисование

Взаимодействие приложения с камерой, стилусом и другими компонентами аппаратного обеспечения мобильного устройства.

Практическая работа

Создание простейшего фоторедактора на базе шаблонов Android Studio.

OpenGL

Установка необходимых компонентов NDK. Библиотека JOGL. Формирование отображения кадра. Функции трёхмерного отображения. 3dVertex.

Практическая работа

Создание приложения «Трёхмерная игра».

Выпускная творческая работа

Повторение теоретических вопросов, изученных ранее.

Практическая работа

Разработка авторского многоэкранного приложения на индивидуальную тему.

Учебно-массовые мероприятия

Учебно-массовые мероприятия проводятся по тематике объединения в рамках объединения: викторины, конкурсы. План мероприятий составляется ежегодно.

Учащиеся мотивируются к участию в городских и всероссийских конкурсах по программированию и компьютерным работам.

Методическое обеспечение образовательной программы

1 год обучения

№ п/п	Тема	Форма занятий	Методы и приёмы	Дидактический материал	Материально- техническое оснащение	Формы подведения итогов
1	Введение	теоретическое	Фронтальная, индивидуальная в рамках фронтальной. Перцептивные, гностические, логические, производные, управленческие. Рассказ, беседа, инструктаж, устный обучающий контроль, диалог, демонстрация практических приемов работы.	Распечатка с инструкциями по охране труда	Персональные компьютеры с установленным пакетом программ NetBeans	опрос в ходе беседы.
2	Основы синтаксиса	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, машинный контроль с элементами самоконтроля, практическая работа, самостоятельная работа, проблемная постановка вопроса, творческая работа.	Листы по Java (см. приложения)	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, самостоятельная проверочная работа.
3	Знакомство с ООП	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, показ практических приемов работы, устный обучающий контроль, машинный контроль с элементами самоконтроля, проблемная постановка вопроса, творческая работа,	Листы по Java (см. приложения)	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная работа.

			самостоятельная работа.			
4	Классы и объекты	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, машинный контроль с элементами самоконтроля, показ практических приемов работы, творческая работа, самостоятельная работа.	Листы по Java (см. приложения)	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная самостоятельная работа.
5	Работа с файлами	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, машинный контроль с элементами самоконтроля, самостоятельная работа, творческая работа.	Листы по Java (см. приложения)	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная работа.
6	Оконные приложения	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, показ практических приемов работы, устный обучающий контроль, машинный контроль с элементами самоконтроля, самостоятельная работа.	Листы по Java (см. приложения)	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная работа
7	Обработка исключений	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, показ практических приемов работы, машинный контроль с элементами самоконтроля, самостоятельная работа, проблемная	Листы по Java (см. приложения)	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, выставка творческих работ.

			постановка вопроса, соревновательный элемент.			
8	Парсинг	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, показ практических приемов работы, машинный контроль с элементами самоконтроля, самостоятельная работа, проблемная постановка вопроса, соревновательный элемент.	Листы по Java (см. приложения)	Персональные компьютеры с установленным пакетом программ NetBeans	
	Выпускная творческая работа	практическое занятие.	Индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, показ практических приемов работы, машинный контроль с элементами самоконтроля, самостоятельная работа, проблемная постановка вопроса, соревновательный элемент.		Персональные компьютеры с установленным пакетом программ NetBeans	Выставка творческих работ
	Учебно-массовые мероприятия				Персональные компьютеры с установленным офисным пакетом программ	Участие в конкурсах, конференциях, выставках и др.

2 год обучения

№ п/п	Тема	Форма занятий	Методы и приёмы	Дидактический материал	Материально- техническое оснащение	Формы подведения итогов
1	Введение	теоретическое	Фронтальная, индивидуальная в рамках фронтальной. Перцептивные, гностические, логические, производные, управленческие. Рассказ, беседа, инструктаж, устный обучающий контроль, диалог, демонстрация практических приемов работы.	Распечатка с инструкциями по охране труда	Персональные компьютеры с установленным пакетом программ NetBeans	опрос в ходе беседы.
2	Повторение	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, машинный контроль с элементами самоконтроля, практическая работа, самостоятельная работа, проблемная постановка вопроса, творческая работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, самостоятельная проверочная работа.
3	Архитектура ОС Android	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, показ практических приемов работы, устный обучающий контроль, машинный контроль с элементами самоконтроля, проблемная постановка вопроса, творческая работа, самостоятельная работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная работа.

4	Интерфейс программирования приложений Android	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, машинный контроль с элементами самоконтроля, показ практических приемов работы, творческая работа, самостоятельная работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная самостоятельная работа.
5	Экран приложения	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, машинный контроль с элементами самоконтроля, самостоятельная работа, творческая работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная работа.
6	Графика	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, показ практических приемов работы, устный обучающий контроль, машинный контроль с элементами самоконтроля, самостоятельная работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная работа
7	Асинхронные задачи	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, показ практических приемов работы, машинный контроль с элементами самоконтроля, самостоятельная работа, проблемная постановка вопроса, соревновательный элемент.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, выставка творческих работ.

8	Анимация элементов экрана	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, показ практических приемов работы, машинный контроль с элементами самоконтроля, самостоятельная работа, проблемная постановка вопроса, соревновательный элемент.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	
9	Выпускная творческая работа	комбинированное, практическое (компьютерное) занятие.	Индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, показ практических приемов работы, машинный контроль с элементами самоконтроля, самостоятельная работа, проблемная постановка вопроса, соревновательный элемент.		Персональные компьютеры с установленным пакетом программ NetBeans	Выставка творческих работ
	Учебно-массовые мероприятия				Персональные компьютеры с установленным офисным пакетом программ	Участие в конкурсах, конференциях, выставках и др.

3 год обучения

№ п/п	Тема	Форма занятий	Методы и приёмы	Дидактический материал	Материально- техническое оснащение	Формы подведения итогов
1	Введение	теоретическое	Фронтальная, индивидуальная в рамках фронтальной. Перцептивные, гностические, логические, производные, управленческие. Рассказ, беседа, инструктаж, устный обучающий контроль, диалог, демонстрация практических приемов работы.	Распечатка с инструкциями по охране труда	Персональные компьютеры с установленным пакетом программ NetBeans	опрос в ходе беседы.
2	Повторение	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, машинный контроль с элементами самоконтроля, практическая работа, самостоятельная работа, проблемная постановка вопроса, творческая работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, самостоятельная проверочная работа.
3	Многоэкранные приложения	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, показ практических приемов работы, устный обучающий контроль, машинный контроль с элементами самоконтроля, проблемная постановка вопроса, творческая работа, самостоятельная работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная работа.

4	Мультимедийные данные	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, машинный контроль с элементами самоконтроля, показ практических приемов работы, творческая работа, самостоятельная работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная самостоятельная работа.
5	Картография	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, машинный контроль с элементами самоконтроля, самостоятельная работа, творческая работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная работа.
6	Камера и рисование	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, показ практических приемов работы, устный обучающий контроль, машинный контроль с элементами самоконтроля, самостоятельная работа.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, машинный контроль, проверочная работа
7	OpenGL	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, показ практических приемов работы, машинный контроль с элементами самоконтроля, самостоятельная работа, проблемная постановка вопроса, соревновательный элемент.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	устный опрос в ходе беседы, выставка творческих работ.

8	Выпускная творческая работа	комбинированное, практическое (компьютерное) занятие.	Фронтальная, индивидуальная в рамках фронтальной, групповая. Перцептивные, гностические, логические, производные, управленческие. Рассказ, диалог, пояснения, упражнения, устный обучающий контроль, показ практических приемов работы, машинный контроль с элементами самоконтроля, самостоятельная работа, проблемная постановка вопроса, соревновательный элемент.	Встроенная справка пакета программ NetBeans	Персональные компьютеры с установленным пакетом программ NetBeans	
	Учебно-массовые мероприятия				Персональные компьютеры с установленным офисным пакетом программ	Участие в конкурсах, конференциях, выставках и др.

СПИСОК ЛИТЕРАТУРЫ

Первый год обучения

Для педагога:

1. Васильев А.Н. Java: Объектно-ориентировочное программирование. СПб.: Питер, 2011. - 400 с.
2. Хорстманн Кей С., Корнелл Гари. Библиотека профессионала. Java 2. - в 2 томах. М.: Издательский дом "Вильямс", 2003. - 848 с: ил.

Для учащихся:

1. Эккель Б. Философия Java. Библиотека программиста. 4-е изд. - СПб.: Питер, 2009. - 640 с.

Второй год обучения

Для педагога:

1. Васильев А.Н. Java: Объектно-ориентировочное программирование. СПб.: Питер, 2011. - 400 с.
2. Хорстманн Кей С., Корнелл Гари. Библиотека профессионала. Java 2. - в 2 томах. М.: Издательский дом "Вильямс", 2003. - 848 с: ил.

Для учащихся:

1. Эккель Б. Философия Java. Библиотека программиста. 4-е изд. - СПб.: Питер, 2009. - 640 с.

Третий год обучения

Для педагога:

1. Васильев А.Н. Java: Объектно-ориентировочное программирование. СПб.: Питер, 2011. - 400 с.
2. Хорстманн Кей С., Корнелл Гари. Библиотека профессионала. Java 2. - в 2 томах. М.: Издательский дом "Вильямс", 2003. - 848 с: ил.

Для учащихся:

1. Виноградов А. Програмируем игры для мобильных телефонов. - М. – Триумф, 2007. – 272с.
2. Моррисон М. Создание игр для мобильных телефонов. – М.: ДМК Пресс, 2006. - 494 с.
3. Эккель Б. Философия Java. Библиотека программиста. 4-е изд. - СПб.: Питер, 2009. - 640 с.

4. Я дилетант. Мобильные приложения своими руками
<http://idilettante.ru/category/mobilnye-prilozeniya/>

Приложение. Листы по Java (примеры)

Java. Лист 15. Типы данных и их преобразование.

Классы-оболочки

Дата создания: 01.04.2016

Классы-оболочки

В Java широко используются не только базовые типы данных, но и соответствующие классы-оболочки из пакета java.lang: Boolean, Character, Integer, Byte, Short, Long, Float, Double. Объекты этих классов могут хранить те же значения, что и соответствующие им базовые типы. На самом деле классы-оболочки – это объектное представление уже знакомых нам примитивных типов данных.

Таблица соответствия классов-оболочек примитивным типам данных

Примитивный тип	Класс-оболочка	Аргументы
byte	Byte	byte или String
short	Short	short или String
int	Integer	int или String
long	Long	long или String
float	Float	float, double или String
double	Double	double или String
char	Character	char
boolean	Boolean	boolean или String

Преобразование типов

Преобразование типов из одного примитивного числового типа в другой делается следующим образом:

```
int b=(int) d;
```

Для преобразования числа в строку уже используют классы-оболочки:

```
int i = 35;
```

```
String str = Integer.toString(i);
```

Следующий листинг иллюстрирует использование класса-оболочки int для более удобного преобразования типов с помощью метода parseInt:

```
class MyClass {
```

```
public static void main(String args
[]) { int sum = 0;
  for(int cnt = 0; cnt < args.length; cnt+
    +)
    sum+=Integer.parseInt(args[cn
    t]);
    System.out.println( Sum is : +sum);
  }
}
```

Источники информации:

1. <http://prologistic.com.ua/klassy-obolochki-v-java-rukovodstvo-s-primerami.html>
2. <http://www.realcoding.net/articles/glava-4-klassy-obolochki.html>
3. <http://crypto.pp.ua/2010/06/klassy-obolochki/>
4. <http://study-java.ru/uroki-java/urok-7-preobrazovanie-tipov-v-java/>
5. <http://goloburdin.blogspot.ru/2011/03/json-java.html>

Java. Лист 17. Коллекции

Дата создания: 17.02.2017

Коллекция в общем виде — это возможность собрать объекты в некоторую группу/множество и работать с этой группой. Для коллекции создаётся свой специальный класс.

Для работы с коллекциями необходимо подключить `java.util.Collection`. Это основной интерфейс, описывающий базовые методы, которыми должна обладать любая коллекция. Т.е. если какой-то класс претендует на звание КОЛЛЕКЦИЯ — он должен реализовать те методы, которые описаны в этом интерфейсе.

Существует множество способов применение коллекции. Рассмотрим наиболее употребительные инструменты.

1. Интерфейс List и массив списков ArrayList

Список обладает следующими важными признаками:

- Список может включать одинаковые элементы.
- Элементы в списке хранятся в том порядке, в котором они туда помещались. Самопроизвольных перемещений элементов в нем не происходит — только с вашего ведома. Например, вы можете добавить элемент на какую-то позицию и тогда произойдет сдвиг других элементов.
- Можно получить доступ к любому элементу по его порядковому номеру/индексу внутри списка.

Класс `ArrayList` представляет обобщенную коллекцию, которая наследует свою функциональность от класса `AbstractList` и применяет интерфейс `List`. Проще говоря, `ArrayList` представляет простой список, аналогичный массиву, за тем исключением, что количество элементов в нем не фиксировано.

`ArrayList` имеет следующие конструкторы:

- `ArrayList()`: создает пустой список
- `ArrayList(Collection <? extends E> col)`: создает список, в который добавляются все элементы коллекции `col`.

- `ArrayList (int capacity)`: создает список, который имеет начальную емкость `capacity`

Емкость в `ArrayList` представляет размер массива, который будет использоваться для хранения объектов. При добавлении элементов фактически происходит перераспределение памяти - создание нового массива и копирование в него элементов из старого массива. Изначальное задание емкости `ArrayList` позволяет снизить подобные перераспределения памяти, тем самым повышая производительность.

Массив списков `ArrayList`:

```
private static Collection createFirstCollection() {  
    // Создать коллекцию на основе стандартного класса ArrayList  
    Collection col = new ArrayList();  
    // Добавление в коллекцию  
    col.add("1");  
}
```

2. Метод `Iterator`

Реализация интерфейса `Collection` предполагает, что с помощью вызова метода `next()` можно получить следующий элемент. С помощью метода `hasNext()` можно узнать, есть ли следующий элемент, и не достигнут ли конец коллекции. И если элементы еще имеются, то `hasNext()` вернет значение `true`. Метод `hasNext()` следует вызывать перед методом `next()`, так как при достижении конца коллекции метод `next()` выбрасывает исключение `NoSuchElementException`. И метод `remove()` удаляет текущий элемент, который был получен последним вызовом `next()`.

Используем итератор для перебора коллекции `ArrayList`:

```
System.out.println("===== Удаление через итератор");  
while(!col1.isEmpty()) {  
    Iterator it = col1.iterator();  
    Object o = it.next();  
    System.out.println("Удаляем:" + o);  
    // Удаляем элемент  
    it.remove();  
}
```

3. Отображение `Map`

Отображение, или карта `Map` — очень удобная конструкция, которая хранит данные не в виде списка значений, а в виде пары ключ-значение. Это очень востребованная форма, в которой вы получаете доступ к значению в

коллекции по ключу. Например, доступ к данным пользователя на сайте может быть осуществлен по логину. Самых данных может быть достаточно много, но для поиска можно использовать очень короткую строку-ключ.

При добавлении элемента по существующему ключу происходит запись нового элемента по ключу вместо старого. Это демонстрирует следующий пример:

```
Map<String, String> map = new HashMap<String, String>();
map.put("1", "a");
map.put("1", "2");
for( Entry<String, String> entry : map.entrySet() ){
    System.out.println( entry.getKey() + " " + entry.getValue() );
}
```

Методы `java.util.Collections`

```
// Смешивание
Collections.shuffle(list);
printCollection("Смешивание", list);

// Обратный порядок
Collections.reverse(list);
printCollection("Обратный порядок", list);

// "Проворачивание" на определенное количество
Collections.rotate(list, 2); // Число может быть отрицательным - тогда порядок будет
обратный
printCollection("Проворачивание", list);

// Обмен элементов
Collections.swap(list, 0, list.size()-1);
printCollection("Обмен элементов", list);

// Замена
Collections.replaceAll(list, new MyClass("Вася"), new MyClass("Маша"));
printCollection("Замена", list);
```

Java. Лист 14. Обработка исключений с помощью операторов try catch

Дата создания: 08.12.2015

Редакция 27.01.2017

Исключения – это ошибки, возникающие в процессе выполнения программы. Отсутствие файла для записи, деление на 0, отсутствие нужных данных в потоке – всё это исключения. В Java можно предусматривать подобные ситуации с помощью объекта «Исключение».

Рассмотрим пример. Если попытаться выполнить следующий код:

```
int a=2;  
int b=0;  
a=a/b;
```

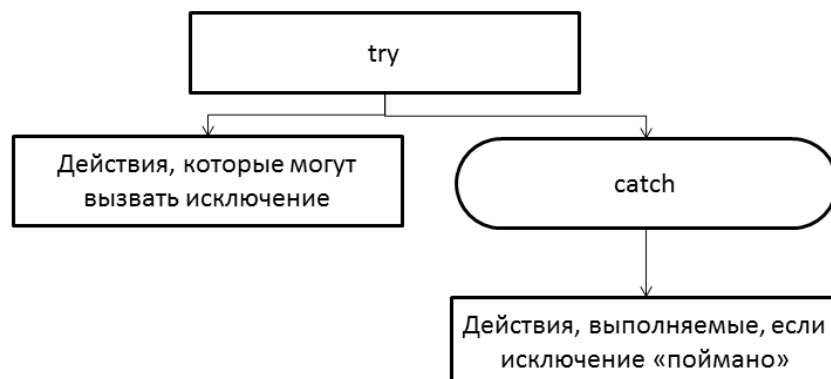
В коде ошибок нет, и компиляция пройдет успешно. Однако при попытке запуска такой программы отобразится следующая ошибка:

```
Exception in thread "main" java.lang.ArithmeticException
```

Слово «Exception» переводится с английского языка как «исключение». В данном случае компилятор сообщает нам, что программа попыталась выполнить недопустимую операцию. Более того, компилятор нам укажет конкретный номер строки, в которой имеется ошибка.

Язык Java поддерживает множество встроенных вариантов исключений.

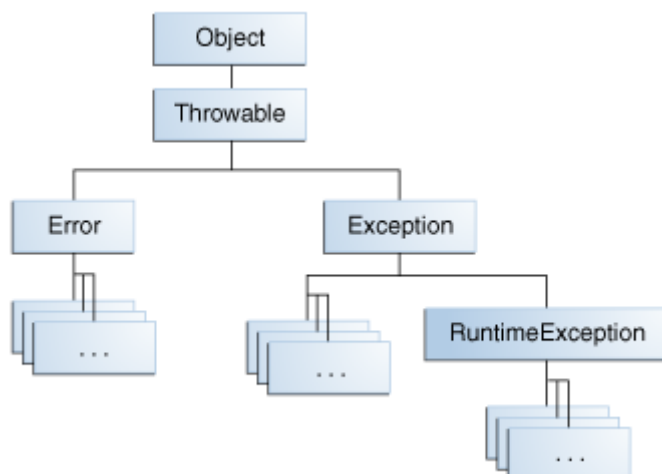
Для обработки исключений используются операторы throw и try catch. Темой данного листа является конструкция try catch. Вот общая схема работы оператора:



Приведём пример обработки арифметического исключения на примере деления на ноль:

```
class List14{
    static void division(){
        try{
            int a=2;
            int b=0;
            int c=a/b;
        } catch (ArithmeticException e){
            System.out.println("division by zero");
        }
        System.out.println("Continue");
    }
    public static void main(String[] args){
        List14.division();
    }
}
```

Иерархия исключений



Обратите внимание на то, что хотя все исключения имеют общего предка **Throwable**, иерархия сразу разделяется на две ветви: **Error** и **Exception**.

Иерархия **Error** описывает внутренние ошибки и ситуации, связанные с нехваткой ресурсов, в системе поддержки выполнения программ. Ни один объект этого типа самостоятельно сгенерировать невозможно. При возникновении внутренней ошибки возможности программиста крайне ограничены. Можно лишь уведомить пользователя и попытаться аккуратно прервать выполнения программы. Такие ситуации достаточно редки.

При программировании на Java основное внимание следует уделять иерархии **Exception**. Эта иерархия также разделяется на две ветви: исключения, производные от класса **RuntimeException**, и остальные. Исключения типа **RuntimeException** возникают вследствие ошибок программирования. Все другие исключения являются следствием непредвиденного стечения обстоятельств, например, ошибок ввода-вывода, возникающих при выполнении вполне корректных программ.

Исключения, производные от класса **RuntimeException**, связаны со следующими проблемами:

- Неверное приведение типов;
- Выход за пределы массива;
- Попытка обратиться к объекту по ссылке null.

Остальные исключения возникают в следующих случаях:

- Попытка чтения после конца файла;
- Попытка установить соединение, используя неправильный URL;
- Попытка получить объект Class, если в строке указан несуществующий класс.

Исключения RuntimeException практически всегда возникают по вине программиста. Исключения ArrayIndexOutOfBoundsException можно избежать, если всегда проверять индексы массива. Исключение NullPointerException никогда не возникает, если перед использованием переменной вы проверите, не содержит ли она значение null.

В спецификации языка Java любое исключение, производное от класса **Error** или **RuntimeException**, называется неконтролируемым (unchecked). Все остальные исключения называются контролируемыми(checked). Для всех контролируемых исключений компилятор проверяет наличие соответствующих обработчиков.

Имя RuntimeException иногда вводит в заблуждение. Разумеется, все ошибки возникают в ходе выполнения программы.

Java. Лист 20. Коллекция «Очередь»

Дата создания: 10.03.2017

Queue - коллекция, предназначенная для хранения элементов в порядке, нужном для их обработки. В дополнение к базовым операциям интерфейса Collection, очередь предоставляет дополнительные операции вставки, получения и контроля. Очереди обычно, но не обязательно, упорядочивают элементы в FIFO (first-in-first-out, "первым вошел - первым вышел") порядке.

Методы интерфейса Queue

E element() – возвращает, но не удаляет головной элемент очереди;

boolean offer(E o) – вставляет элемент в очередь, если возможно. Этот метод предпочтительнее, чем метод add(), который вы уже знаете, поскольку offer проверяет возможность вставки элемента, а add() - нет;

E peek() – возвращает, но не удаляет головной элемент очереди, возвращает null, если очередь пуста;

E poll() – возвращает и удаляет головной элемент очереди, возвращает null, если очередь пуста;

E remove() – возвращает и удаляет головной элемент очереди.

Сводная таблица методов Queue

Суть метода	Exception	Null
Добавить	add()	offer()
Удалить	remove()	poll()
Проверить	element()	peek()

ПРИМЕР 20_1.java демонстрирует применение этих методов

Для работы с очередями используются два интерфейса – Comparable и Comparator, реализующие удобные методы упорядочивания объектов.

Интерфейсы Comparable и Comparator

В интерфейсе Comparable объявлен всего один метод compareTo(Object obj), предназначенный для упорядочивания объектов класса. Это используется при сортировке упорядоченных списков или массивов объектов.

Данный метод сравнивает вызываемый объект с obj. В отличие от метода equals, который возвращает true или false, compareTo возвращает:

- 0, если значения равны;
- Отрицательное значение, если вызываемый объект меньше параметра;
- Положительное значение, если вызываемый объект больше параметра.

Метод может выбросить исключение ClassCastException, если типы объектов не совместимы при сравнении.

Необходимо помнить, что аргумент метода compareTo имеет тип сравниваемого объекта класса.

Классы Byte, Short, Integer, Long, Double, Float, Character, String уже реализуют интерфейс Comparable.

```
class Person implements Comparable<Person>{
    private String name;
    public Person(String value){
        name=value;
    }
    String getName(){return name;}
    public int compareTo(Person p){
        return name.compareTo(p.getName()); //сравнение name с
полученным значением
    }}
20_2.java содержит пример реализации интерфейса Comparable
```

В данном примере значения сортируются сначала по полю str, а затем по number.

Очередь упорядочивает элементы либо по их натуральному порядку (используя интерфейс Comparable), либо с помощью интерфейса Comparator, полученному в конструкторе.

В интерфейсе Comparator объявлено два метода compare(Object obj1, Object obj2) и equals(Object obj).

compare(Object obj1, Object obj2) – так же, как и метод compareTo интерфейса Comparable, упорядочивает объекты класса. Точно так же на выходе получает 0, положительное значение или отрицательное значение. Метод может выбросить исключение ClassCastException, если типы объектов не совместимы при сравнении.

Comparator

Основным отличием интерфейса Comparator от Comparable является то, что **вы можете создавать несколько видов независимых сортировок.**

Общий принцип использования интерфейса Comparator аналогичен Comparable, но его метод `compare` возвращает числовое значение - если оно отрицательное, то объект `a` предшествует объекту `b`, иначе - наоборот. Если метод возвращает ноль, то объекты равны.

20_3.java содержит пример реализации интерфейса Comparator

Deque

Интерфейс Deque определяет «двунаправленную» очередь и, соответственно, методы доступа к первому и последнему элементам двусторонней очереди. Методы обеспечивают удаление, вставку и обработку элементов. Каждый из этих методов существует в двух формах. Одни методы создают исключительную ситуацию в случае неудачного завершения, другие возвращают какое-либо из значений (`null` или `false` в зависимости от типа операции). Вторая форма добавления элементов в очередь сделана специально для реализаций Deque, имеющих ограничение по размеру. В большинстве реализаций операции добавления заканчиваются успешно.

Методы интерфейса Deque:

- **void addFirst(E obj):** добавляет элемент в начало очереди;
- **void addLast(E obj):** добавляет элемент `obj` в конец очереди;
- **E getFirst():** возвращает без удаления элемент из головы очереди. Если очередь пуста, генерирует исключение `NoSuchElementException`;
- **E getLast():** возвращает без удаления последний элемент очереди. Если очередь пуста, генерирует исключение `NoSuchElementException`;
- **boolean offerFirst(E obj):** добавляет элемент `obj` в самое начало очереди. Если элемент удачно добавлен, возвращает `true`, иначе – `false`;
- **boolean offerLast(E obj):** добавляет элемент `obj` в конец очереди. Если элемент удачно добавлен, возвращает `true`, иначе – `false`;
- **E peekFirst():** возвращает без удаления элемент из начала очереди.

Аналогично, `E peekLast()` возвращает без удаления последний элемент очереди. Если очередь пуста, возвращают `null`;

- **E pollFirst()**: возвращает с удалением первый элемент. Аналогично, `E pollLast()` возвращает с удалением последний элемент очереди. Если очередь пуста, возвращают `null`;
 - **E pop()**: возвращает с удалением элемент из начала очереди. Если очередь пуста, генерирует исключение `NoSuchElementException`;
 - **void push(E element)**: добавляет элемент в самое начало очереди;
 - **E removeFirst()**: возвращает с удалением элемент из начала очереди. Если очередь пуста, генерирует исключение `NoSuchElementException`;
- 20_4.java иллюстрирует реализацию двунаправленной очереди

на Java

В примере 20_4 Методы `addFirst()`, `addLast()` вставляют элементы в начало и в конец очереди соответственно. Метод `add()` унаследован от интерфейса `Queue` и абсолютно аналогичен методу `addLast()` интерфейса `Deque`.

Практическая часть

1. Запустите примеры 20_1, 20_2, 20_3, 20_4.
2. Реализуйте сортировку в обратном порядке для примера 20_2.
3. Реализуйте таймер обратного отсчета: очередь предварительно загружается со всеми целочисленными значениями от числа, полученного с консоли, чтобы обнулить в порядке убывания. Затем значения удаляются из очереди и печатаются в односекундных интервалах. Можно использовать паузу `Thread.sleep(значение в миллисекундах)`.
4. Попробуйте реализовать отмену последнего действия в вашем итоговом проекте.

Почитать отдельно

<http://java-course.ru/begin/object-queue/>

<http://javadevblog.com/primer-ispol-zovaniya-java-priority-queue-priorityqueue.html> - про очередь с приоритетом.

Java. Лист 21. Слушатели событий

Дата создания: 14.04.2017

Графический интерфейс приложения включает в себя не только разнообразные элементы Swing – кнопки, текстовые поля и прочее, но и реакцию системы на то или иное событие, происходящее в приложении. В Java реализован класс `Listener`, функционирующий в соответствии с паттерном проектирования «наблюдатель». Этот паттерн подразумевает, что компонент приложения должен оповещать остальные компоненты о смене своего состояния или каких-либо воздействиях извне. Соответственно, такому компоненту необходим специальный метод с заранее определённой реакцией на событие (т.е. должен быть упомянут тип события и инициируемые им действия).

Имена таких методов имеют форму `addTypeListener(TypeListener listener)` или `setTypeListener(TypeListener listener)`, где `Type` – тип происходящего события.

В библиотеке Swing существуют следующие «слушатели событий»:

1. **MouseListener** – интерфейс событий мыши.

В этом интерфейсе перечислены следующие методы:

- `public void mouseClicked(MouseEvent event)` — выполнен щелчок мышкой на наблюдаемом объекте.
- `public void mouseEntered(MouseEvent event)` — курсор мыши вошел в область наблюдаемого объекта.
- `public void mouseExited(MouseEvent event)` — курсор мыши вышел из области наблюдаемого объекта.
- `public void mousePressed(MouseEvent event)` — кнопка мыши нажата в момент, когда курсор находится над наблюдаемым объектом.
- `public void mouseReleased(MouseEvent event)` — кнопка мыши отпущена в момент, когда курсор находится над наблюдаемым объектом.

2. **FocusListener** – фокус на объекте.

3. **MouseWheelListener** с единственным методом `public void mouseWheelMoved(MouseWheelEvent event)`

4. **ChangeListener** – изменение состояния объекта.

5. **KeyListener** – работа с клавиатурой. При вводе с клавиатуры генерируется событие типа `KeyEvent`. Существует три типа клавиатурных событий, обозначаемых константами:

- `KEY_PRESSED` – клавиша нажата.
- `KEY_RELEASED` – клавиша отпущена.
- `KEY_TYPED` – ввод символа. При нажатии `Shift KEY_TYPED` не срабатывает!

В классе `KeyEvent` определяется целый ряд других целочисленных констант. Например, константы `VK_0-VK_9` и `VK_A-VK_Z` обозначают эквиваленты чисел и букв в коде ASCII.

Каждый раз, когда пользователь нажимает клавиши на клавиатуре и система вызывает методы `keyTyped`, `keyPressed` и `keyReleased`, в качестве параметра им передается объект `KeyEvent`, который содержит всю необходимую информацию о произошедшем событии. Отсюда можно узнать код клавиши, которая была нажата – метод `getKeyCode()`. Были ли зажаты при этом такие клавиши, как `Alt`, `Shift` или `Ctrl`. Проверить это можно, вызвав соответственно методы `isAltDown()`, `isShiftDown()` и `isControlDown()`. Класс `KeyEvent` содержит большой набор констант. Каждая константа содержит код соответствующей клавиши. Поэтому нет необходимости коды всех клавиш. Достаточно использовать какую-то из констант. По названиям констант можно легко определить, какой клавише она соответствует. Например `KeyEvent.VK_ENTER` или `KeyEvent.VK_F`.

Ниже перечислены другие константы, определяемые в классе

`KeyEvent`.

-
- `VK_ALT`
 - `VK_CANCEL`
 - `VK_CONTROL`
 - `VK_DOWN`
 - `VK_ENTER`
 - `VK_ESCAPE`
-
- `VK_LEFT`
 - `VK_PAGE_DOWN`
 - `VK_PAGE_UP`
 - `VK_RIGHT`
 - `VK_SHIFT`
 - `VK_UP`
-

Виртуальный код клавиши (например, `VK_UP`, `VK_A` и т.д.) передает в качестве параметра код, а символьный эквивалент нажатой клавиши, если таковой существует — в качестве параметра символ. В отсутствие достоверного символа параметр символ принимает значение константы `CHAR_UNDEFINED`.

6. **WindowListener** – обработчик событий окна (см. лист 18).

7. **ActionListener** – универсальный обработчик.

Среди многочисленных событий, на которые реагирует каждый элемент управления (и о которых он оповещает соответствующих слушателей, если они к нему присоединены), есть одно основное, вытекающее из самой сути компонента и

обрабатываемое значительно чаще, чем другие. Например, для кнопки это щелчок на ней, а для выпадающего списка — выбор нового элемента.

Для отслеживания и обработки такого события может быть использован особый слушатель `ActionListener`, имеющий один метод:

```
public void actionPerformed(ActionEvent event).
```

У использования `ActionListener` есть небольшое преимущество в эффективности (так, при обработке нажатия на кнопку не надо реагировать на четыре лишних события — ведь даже если методы-обработчики пустые, на вызов этих методов все равно тратятся ресурсы). Также очень удобно запомнить и постоянно использовать один класс с одним методом и обращаться к остальным лишь в тех относительно редких случаях, когда возникнет такая необходимость.

Пример: Обработка нажатия на кнопку `ok` в форме:

```
ok.addMouseListener(new ActionListener() {  
    public void actionPerformed(ActionEvent event) {  
        if (loginField.getText().equals("John"))  
            JOptionPane.showMessageDialog(null, "Login OK");  
        else JOptionPane.showMessageDialog(null, "Login failed");  
    }  
});
```

Метод `showMessageDialog()` показывает окно с предупреждением и кнопкой `OK`.

Практические задания

1. Реализовать приложение в два окна, в первом из которых происходит проверка логина и пароля, а во втором — реализуются какие-либо собственные функции приложения. Если у вас ещё нет такого приложения, пусть это будет просто отдельное окно `JFrame` с надписью `login succeeded`.
2. Реализуйте в вашем приложении обработчики закрытия окна с фразой «Уже уходите?» Если ваше приложение является тестом или контрольной работой, попробуйте реализовать вывод сообщения «Не смей переключаться в браузер, когда решаешь наш тест!» и предупреждение об аннулировании результатов.
3. Реализуйте реакцию на клавиши в вашем приложении — изменение координат объекта, закрытие / открытие / скрытие / появление элементов и пр.

Файл 21.java содержит пример отображения координат курсора!

Источники информации

1. <https://docs.oracle.com/javase/7/docs/api/java/awt/event/KeyListener.html>
2. https://javaswing.wordpress.com/2009/12/23/keylistener_using/
3. <http://math.sgu.ru/sites/chairs/prinf/materials/java/lesson10.htm>

