МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«ИВАНОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Шуйский филиал ИвГУ

Кафедра математики, информатики и методики обучения

ОДОБРЕНО:

Руководитель ОП

_____ А.С. Зинченко
(подпись)

«25» февраля 2025 г.

Рабочая программа дисциплины

ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

Уровень высшего образования:	бакалавриат
Квалификация выпускника:	бакалавр
Направление подготовки:	44.03.05 Педагогическое образование (с двумя профилями подготовки)
Направленность (профиль) образовательной программы:	Математика; Информатика и VR-технологии

Шуя

1. Цели освоения дисциплины

Дисциплина «Основы алгоритмизации и программирования» имеет своей целью освоение студентами навыков алгоритмизации, формирование у них умения разрабатывать алгоритмы и реализовывать их на компьютере, алгоритмически подходить к решению информационных задач, разбираться в терминологии программирования, представлять возможности современных языков разработки программного обеспечения.

2. Место дисциплины в структуре ОП

Дисциплина «Основы алгоритмизации и программирования» является обязательной для изучения; относится к обязательной части блока дисциплин Б1, входит в предметно-методический модуль.

Успешное освоение данной дисциплины будет способствовать готовности обучающихся к освоению дисциплин: «Системное программирование»; «Прикладное программирование»; «Методика обучения информатике»; «Базы данных»; «Математическая логика и теория алгоритмов»; «Компьютерное моделирование»; «Логическое программирование»; «Основы робототехники»; «Основы искусственного интеллекта»; «Основы программирования VR-приложений»; «Проектирование и разработка VR-приложений»; успешному прохождению следующих практик: «Производственная практика, научно-исследовательская работа, по предметно-методическому модулю», «Производственная практика, педагогическая (комплексная) практика, по предметно-методическому модулю», подготовке к сдаче и сдаче государственного экзамена, выполнению и защите выпускной квалификационной работы.

Студент, приступающий к изучению дисциплины, должен:

Знать: основные сведения из школьных курсов информатики и ИКТ и математики.

Уметь: применять основные сведения из школьных курсов информатики и ИКТ и математики.

Владеть: навыками, полученными ранее в ходе изучения школьных курсов информатики и ИКТ и математики.

3. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

3.1. Компетенции, формированию которых способствует данная дисциплина

При реализации дисциплины формируются следующие компетенции в соответствии с ФГОС ВО по данному направлению подготовки:

а) универсальные компетенции (УК) и индикаторы их достижения:

Наименование категории (группы) универсальных компетенций	Код и наименование универсальной компетенции выпускника	Код и наименование индикатора достижения универсальной компетенции
Системное и критическое мышление	УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	УК-1.1. Демонстрирует знание особенностей системного и критического мышления и готовность к нему. УК-1.2. Применяет логические формы и процедуры, способен к рефлексии по поводу собственной и чужой мыслительной деятельности. УК-1.3. Анализирует источник информации с точки зрения временных и пространственных условий его

		возникновения. УК-1.4. Анализирует ранее сложившиеся в науке оценки информации. УК-1.5. Сопоставляет разные источники информации с целью выявления их противоречий и поиска достоверных суждений. УК-1.6. Аргументировано формирует собственное суждение и оценку информации, принимает обоснованное решение. УК-1.7. Определяет практические последствия предложенного решения задачи.
Разработка и реализация проектов	УК-2. Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1. Определяет совокупность взаимосвязанных задач, обеспечивающих достижение поставленной цели, исходя из действующих правовых норм. УК-2.2. Определяет ресурсное обеспечение для достижения поставленной цели. УК-2.3. Оценивает вероятные риски и ограничения в решении поставленных задач. УК-2.4. Определяет ожидаемые результаты решения поставленных задач.

б) общепрофессиональные компетенции (ОПК) и индикаторы их достижения:

Наименование категории (группы) общепрофессиональных компетенций	Код и наименование общепрофессиональной компетенции выпускника	Код и наименование индикатора достижения общепрофессиональной компетенции
Научные основы педагогической деятельности	ОПК-8. Способен осуществлять педагогическую деятельность на основе специальных научных знаний	ОПК-8.1. Применяет методы анализа педагогической ситуации, профессиональной рефлексии на основе специальных научных знаний. ОПК-8.2. Проектирует и осуществляет учебно-воспитательный процесс с опорой на знания основных закономерностей возрастного развития когнитивной и личностной сфер обучающихся, научно-обоснованных закономерностей организации образовательного процесса, научных основ преподаваемых предметов, их истории и места в мировой культуре и науке. ОПК-8.3. Осуществляет мониторинг и отбор программ профессионального и личностного развития.

в) профессиональные компетенции (ПК) и индикаторы их достижения:

Код и наименование профессиональной компетенции выпускника	Код и наименование индикатора достижения профессиональной компетенции	Основание (ПС, анализ опыта)
Тип задач профессиональной деятельности: педагогический		
ПК-4. Способен выделять	ПК-4.1. Выделяет и анализирует единицы различных	Профессиональный стан-

структурные элементы, входящие в систему познания предметной области (в соответствии с профилем и уровнем обучения), анализировать их в единстве содержания, формы и выполняемых функций	уровней знаковой математической структуры в единстве их содержания, формы и функций. ПК-4.2. Выделяет и анализирует закономерности и явления информатики как науки в их структурном и функциональном единстве с информационно-коммуникационными технологиями. ПК-4.3. Знает и умеет анализировать аксиомы, определения и математические доказательства в единстве с прикладными аспектами использования.	дарт 01.001 «Педагог (педагогическая деятельность в сфере дошкольного, начального общего, основного общего, среднего общего образования) (воспитатель, учитель)»

3.2. Планируемые результаты обучения по дисциплине, соотнесенные с индикаторами достижения формируемых компетенций

В результате освоения дисциплины обучающийся должен:

Знать:

– основные этапы проектирования, их последовательность и взаимосвязь (УК-2 (индикаторы – 1, 2, 3, 4);

– научные основы преподаваемого предмета, его историю и место в мировой культуре и науке (конкретизируются применительно к изучаемой учебной дисциплине) (ОПК-8 (индикаторы – 1, 2, 3);

– этапы развития предметной области (ОПК-8 (индикаторы – 1, 2, 3);

– основные понятия информатики, их системно-структурные связи в контексте развития информационно-коммуникационных технологий (ПК-4 (индикаторы – 1, 2, 3);

– основные закономерности, типичные явления применения математических объектов и алгоритмов (ПК-4 (индикаторы – 1, 2, 3);

– основные понятия и термины, связанные с алгоритмизацией и программированием (ПК-4 (индикаторы – 1, 2, 3).

Уметь:

– анализировать источники информации с точки зрения временных и пространственных условий их возникновения (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– использовать современные информационно-коммуникационные технологии (включая пакеты прикладных программ, локальные и глобальные компьютерные сети) для сбора, обработки и анализа информации (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– осуществлять корректный подбор методов анализа, проводить обработку данных исследования и правильную интерпретацию результатов (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– определять совокупность необходимых ресурсов для реализации каждой задачи (УК-2 (индикаторы – 1, 2, 3, 4);

– оценивать уровень и качество каждого ресурса, обеспечивающего выполнение определенной задачи (УК-2 (индикаторы – 1, 2, 3, 4);

– выявлять возможности преодоления рисков и ограничений с учетом имеющихся ресурсов и резервов (УК-2 (индикаторы – 1, 2, 3, 4);

– проектировать процесс решения каждой задачи проекта, выбирая оптимальный способ ее решения, исходя из действующих правовых норм и имеющихся ресурсов и ограничений (УК-2 (индикаторы – 1, 2, 3, 4);

– совершенствовать свои профессиональные знания и умения на основе постоянного самообразования (ОПК-8 (индикаторы – 1, 2, 3);

– соотносить основные этапы развития предметной области с ее актуальными задачами, методами и концептуальными подходами, тенденциями и перспективами ее современного развития (ОПК-8 (индикаторы – 1, 2, 3);

– руководствоваться в профессиональной деятельности научными основами

преподаваемого предмета (конкретизируются применительно к изучаемой учебной дисциплине) (ОПК-8 (индикаторы – 1, 2, 3);

– устанавливать содержательные, методологические и мировоззренческие связи предметной области со смежными научными областями (ОПК-8 (индикаторы – 1, 2, 3);

– использовать различного рода прикладные программы и языки программирования для решения учебных задач (ПК-4 (индикаторы – 1, 2, 3).

Владеть:

– навыками сопоставления разных источников информации с целью выявления их противоречий и поиска достоверных суждений (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– навыками определения практических последствий предложенного решения задач (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– технологиями анализа и синтеза информации на основе системного подхода (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– основными методами математической обработки информации (УК-1 (индикаторы – 1, 2, 3, 4, 5, 6, 7);

– приемами декомпозиции цели, используя вариативные трактовки задач, конкретизирующих различные пути достижения поставленной цели (УК-2 (индикаторы – 1, 2, 3, 4);

– способами определения резервов, использование которых может компенсировать недостаток имеющихся ресурсов (УК-2 (индикаторы – 1, 2, 3, 4);

– способами решения конкретных задач проекта на уровне заявленного качества и за установленное время (УК-2 (индикаторы – 1, 2, 3, 4);

– способами совершенствования профессиональных знаний и умений путем использования информационной среды (ОПК-8 (индикаторы – 1, 2, 3);

– научными основами преподаваемого предмета (конкретизируются применительно к изучаемой учебной дисциплине) (ОПК-8 (индикаторы – 1, 2, 3);

– теоретическими и практическими знаниями для постановки и решения исследовательских задач в предметной области и в области образования (ОПК-8 (индикаторы – 1, 2, 3);

– базовыми понятиями и терминами в области математики и информатики (ПК-4 (индикаторы – 1, 2, 3);

– технологиями алгоритмизации и программирования (ПК-4 (индикаторы – 1, 2, 3).

4. Объём и содержание дисциплины

Очная форма обучения

Объем дисциплины составляет 6 зачетных единиц (162 час.)

Контактная работа 74 час.

Самостоятельная работа 88 час.

<table>
<tr><td rowspan="3">Виды учебных занятий и формы промежуточной аттестации</td><td colspan="2">1 сем.</td><td colspan="2">2 сем.</td><td colspan="2">3 сем.</td></tr>
<tr><td>Контактная работа с преподавателем</td><td>Самостоятельная работа обучающегося</td><td>Контактная работа с преподавателем</td><td>Самостоятельная работа обучающегося</td><td>Контактная работа с преподавателем</td><td>Самостоятельная работа обучающегося</td></tr>
<tr><td>Лекции</td><td>12</td><td rowspan="2">29,75</td><td>12</td><td rowspan="2">30</td><td>12</td><td rowspan="2">28,25</td></tr>
<tr><td>Лабораторные</td><td>12</td><td>12</td><td>13,5</td></tr>
</table>

работы						
Зачет	0,25	-	-	-	0,25	-
Всего:	24,25	29,75	24	30	25,75	28,25

4.1. Содержание дисциплины по разделам (темам), соотнесенное с видами и объемом учебных занятий

Очная форма обучения

№ п/п	Раздел (тема) дисциплины	Семестр	Виды учебных занятий, включая самостоятельную работу обучающихся и объем (в часах)				Формы текущего контроля успеваемости Формы промежуточной аттестации
			Занятия лекционного типа	Занятия семинарского типа		Самостоятельная работа обучающегося	
				Лаб.	Пр.		
1	Введение. Основные понятия алгоритмизации	1	1,5	1,5		3,75	лабораторная работа
2	Основы программирования в среде Scratch	1	3	4,5		6	лабораторная работа
3	Язык и среда программирования Кумир	1	3	3		8	лабораторная работа
4	Исполнители среды программирования Кумир	1	3	3		6	лабораторная работа
5	Этапы разработки приложения	1	1,5			6	лабораторная работа
Всего за 1 семестр			12	12		29,75	зачет
6	Основы языка программирования на Python	2	4,5	4,5		9	лабораторные работы
7	Системы программирования	2	1,5	1,5		3	лабораторная работа
8	Структуры данных в языке программирования Python	2	3	3		9	лабораторная работа
9	Подпрограммы. Рекурсия	2	3	3		9	лабораторная работа
Всего за 2 семестр			12	12		30	
10	Организация ввода-вывода данных при помощи файлов	3	1,5	1,5		5	лабораторные работы
11	Основные принципы объектно-ориентированного программирования (ООП)	3	1,5	3		5	лабораторные работы
12	Визуальное событийно-управляемое программирование	3	4,5	4,5		9	лабораторные работы
13	Разработка оконного	3	4,5	4,5		9,25	лабораторные

	приложения					работы
Всего за 3 семестр		12	13,5		28,25	зачет
ИТОГО:		36	37,5		88	

4.2. Развернутое описание содержания учебного материала по разделам (темам)

Введение. Основные понятия алгоритмизации. Понятие алгоритма. Свойства алгоритмов. Формы записей алгоритмов. Общие принципы построения алгоритмов. Основные алгоритмические конструкции: линейные, разветвляющиеся (полное или неполное ветвление), циклические (детерминированный цикл, цикл с предусловием, цикл с постусловием). Основные базовые типы данных и их характеристика. Основные этапы решения задач.

Основы программирования в среде Scratch. Установка среды программирования Scratch. Знакомство с официальным сайтов в сети Интернет. Написание мини-игры в среде Scratch. Основные понятия среды: спрайты, скрипты, блок, сцена, сообщение, клон. Блоки для программирования движения спрайта и изменения внешнего вида. Понятие сенсора в среде программирования Scratch. Основные инструменты встроенного графического редактора программной среды Scratch. Создание переменных, списков и подпрограмм.

Язык и среда программирования Кумир. Установка среды программирования Кумир. Алфавит языка. Типы данных языка программирования Кумир. Реализация линейных, разветвляющихся и циклических алгоритмов в Кумир. Решение алгоритмических задач. Знакомство с исполнителями среды программирования Кумир. Создание и обработка массивов. Обработка строк.

Исполнители среды программирования Кумир. Исполнитель «Кузнечик». Исполнитель «Черепаха». Исполнитель «Чертёжник». Написание программ для исполнителя «Робот». Задачи ОГЭ и ЕГЭ, связанные с написанием программ в среде программирования Кумир.

Этапы разработки приложения. Постановка задачи. Создание программы. Тестирование, отладка приложения. Создание документации (документирование программы).

Основы языка программирования на Python. Эволюция языков программирования и языка программирования Python. Общая информация о языке. Алфавит языка. Знакомство с официальным сайтом в сети Интернет. Интерпретатор языка программирования и его установка. Создание и запуск программы во встроенной среде разработки. Математические операции языка Python. Основы алгебры логики. Логические операции с высказываниями: конъюнкция, дизъюнкция, инверсия. Простой условный оператор. Сокращенный условный оператор. Оператор цикла for. Оператор цикла while.

Системы программирования. Понятие системы программирования. Обзор имеющихся систем программирования. Среды программирования: Eclipse, PyScripter, PyCharm. Преимущества и недостатки. Установка и настройка сред программирования на различных операционных системах.

Структуры данных в языке программирования Python. Понятие списка в Python. Хранение списка в памяти. Операции для работы со списками. Ввод и вывод списков. Обработка списков. Понятие словаря в Python. Хранение словаря в памяти. Операции для работы со словарями. Основные операции работы со строками в языке Python. Обработка строк.

Подпрограммы. Рекурсия. Понятие подпрограммы. Процедуры и функции, их сущность, назначение, различие. Функции: способы организации и описание в Python. Вызов функций, рекурсия. Программирование рекурсивных алгоритмов. Создание модулей.

Организация ввода-вывода данных при помощи файлов. Чтение информации из

текстового файла. Запись информации в текстовый файл. Запись и чтение информации из двоичного файла.

Основные принципы объектно-ориентированного программирования (ООП). История развития ООП. Базовые понятия ООП: класс, его свойства и методы, объект. Основные принципы ООП: инкапсуляция, наследование, полиморфизм.

Визуальное событийно-управляемое программирование. Создание интерфейса пользователя. Основы работы с Tkinter. Создание формы и виджетов: «Кнопка», «Текстовое поле», «Надпись». Создание виджета «Флажок». Создание виджета «Переключатель».

Разработка оконного приложения. Настройка внешнего вида окон. Определение объектных связей между виджетами. Различные варианты компоновки виджетов. Обеспечение взаимодействия элементов управления посредством сигналов/слотов.

5. Образовательные технологии, в том числе перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине

В соответствии с требованиями ФГОС ВО в программе данного курса рекомендуется использовать следующие образовательные технологии: лекции, лабораторные занятия, самостоятельная работа студентов.

В преподавании дисциплины существуют темы, изучение которых должно обязательно начинаться с теоретической подготовки. Чтобы повысить результативность изучения теоретического материала, применяются следующие приемы:

- лекции сопровождаются компьютерными презентациями;
- применяются методики «мозгового штурма», разбора практических ситуаций;
- используются краткие сообщения студентов по новейшим разработкам (по материалам Интернет) с использованием демонстрационных материалов.

Основной и самой результативной формой обучения являются лабораторно-практические занятия. Для преподавания дисциплины создан учебно-методический комплекс, обеспечивающий предоставление информации (электронные учебники и пособия) и управление деятельностью обучаемого.

Разработанные лабораторные задания включают в себя обучающие тексты, набор пошаговых инструкций, учебных задач и заданий, демонстрационный материал.

Такой подход к организации проведения лабораторных занятий стимулирует и организует самостоятельную работу студента, достигаются не только учебные, но и развивающие цели, т.к. формируется умения по самоорганизации учебной деятельности.

Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине: мультимедиа технологии; технологии смешанного обучения.

6. Учебно-методическое обеспечение самостоятельной работы обучающихся

Самостоятельная работа обучающихся по дисциплине осуществляется в следующих формах:

- установка и настройка программного обеспечения для работы с языком программирования;
- составление алгоритма для решения задач в словесном виде и в виде блок-схем;
- разработка проектов в среде программирования Scratch;
- написание и тестирования программ на языках Кумир и Python;
- изучение и анализ предложенных электронных образовательных ресурсов, содержащих информацию по алгоритмизации и программированию;

- повторение материала с использованием конспектов.

Полностью весь методический материал по обеспечению самостоятельной работы студентов приводится в приложении 1 к настоящей программе. Доступ к методическим материалам осуществляется через сайт Шуйского филиала ИвГУ, в том числе через ЭБС Шуйского филиала ИвГУ на платформе «Руконт».

7. Характеристика оценочных средств для текущего контроля успеваемости и промежуточной аттестации по дисциплине

Текущая аттестация осуществляется с использованием накопительного рейтингового оценивания. Критерии оценки деятельности обучающихся разрабатываются в зависимости от уровня их информационно технологической и дидактической подготовки обучающихся.

Для текущего контроля успеваемости в 1 семестре применяются:

- Лабораторные работы на компьютере: лабораторная №1 оценивается в 6 баллов, а также 10 лабораторных по 5 баллов (56 баллов);
- Практические задания для самостоятельного выполнения: 12 заданий по 2 балла (24 балла);
- Доклад по предложенной теме (10 баллов);

Помимо этого учитывается:

- степень активности и самостоятельности обучающегося на занятиях – 5 баллов,
- посещаемость (отсутствие пропусков занятий) – 5 баллов.

Для текущего контроля успеваемости в 2 и 3 семестре применяются:

- Лабораторные работы на компьютере: 9 лабораторных по 5 баллов (45 баллов);
- Практические задания для самостоятельного выполнения: 12 заданий по 2 балла (24 балла);
- Доклад по предложенной теме (10 баллов);

Помимо этого учитывается:

- степень активности и самостоятельности обучающегося на занятиях – 10 баллов,
- посещаемость (отсутствие пропусков занятий) – 11 баллов.

Промежуточная аттестация по дисциплине (зачёт) проводится с использованием рейтинговой шкалы оценки. На зачёте по дисциплине «Основы алгоритмизации и программирования» контролируемые знания, умения и владения, характеризующие этапы формирования компетенций в процессе освоения образовательной программы. Зачёт проводится в форме интегрального тестового задания, включающего вопросы по всем темам курса (максимально 30 баллов).

Общая накопительная оценка складывается из баллов текущей успеваемости в семестре по учебной дисциплине и баллов, полученных студентами на зачёте, таким образом, чтобы результирующий балл не превышал значения 100.

Оценка на зачёте выставляется по следующей шкале:

Шкала оценки результатов по дисциплине «Основы алгоритмизации и программирования» при рейтинговой системе оценивания

Повышенный уровень	Базовый уровень	Пороговый уровень	Ниже порогового уровня
зачтено	зачтено	зачтено	не зачтено
85-100 баллов	71-84 балла	61-70 баллов	60 и менее баллов

8. Учебно-методическое и информационное обеспечение дисциплины

Учебная и учебно-методическая литература:

– ***основная литература:***

1. Зюзьков, В.М. Программирование : учебное пособие / В.М. Зюзьков ; Министерство образования и науки Российской Федерации, Томский Государственный Университет Систем Управления и Радиоэлектроники (ТУСУР). - Томск : Эль Контент, 2013. - 186 с. : ил. - Библиогр. в кн. - ISBN 978-5-4332-0141-5 ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=480616
2. Калитвин, В. А. Введение в программирование на Python : учебное пособие : [16+] / В. А. Калитвин ; Липецкий государственный педагогический университет им. П. П. Семенова-Тян-Шанского. – Липецк : Липецкий государственный педагогический университет им. П.П. Семенова-Тян-Шанского, 2023. – 85 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=714538. – Библиогр. в кн. – ISBN 978-5-907655-86-7. – Текст : электронный.
3. Лубашева, Т.В. Основы алгоритмизации и программирования : учебное пособие / Т.В. Лубашева, Б.А. Железко. - Минск : РИПО, 2016. - 378 с. : ил. - Библиогр. в кн. - ISBN 978-985-503-625-9 ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=463632
4. Незнанов, А. А. Программирование и алгоритмизация : учебник / А. А. Незнанов ; науч. ред. В. П. Кутепов. - Москва : Академия, 2010. - 304 с.

– ***дополнительная литература:***

1. Голицына О. Л. Основы алгоритмизации и программирования : учебное пособие / О. Л. Голицына, И. И. Попов. - Москва : Форум : Инфра-М, 2005. - 432 с.
2. Зыков, С.В. Введение в теорию программирования. Объектно-ориентированный подход / С.В. Зыков. - 2-е изд., испр. - Москва : Национальный Открытый Университет «ИНТУИТ», 2016. - 189 с. : схем. - (Основы информационных технологий). - Библиогр. в кн. - ISBN 5-9556-0009-4 ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=429073
3. Зыков, С.В. Введение в теорию программирования. Функциональный подход / С.В. Зыков. - 2-е изд., испр. - Москва : Национальный Открытый Университет «ИНТУИТ», 2016. - 153 с. : ил. - (Основы информационных технологий). - Библиогр. в кн. - ISBN 5-9556-0009-4 ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=429119
4. Калентьев, А.А. Новые технологии в программировании : учебное пособие / А.А. Калентьев, Д.В. Гарайс, А.Е. Горяинов ; Министерство образования и науки Российской Федерации, Томский Государственный Университет Систем Управления и Радиоэлектроники (ТУСУР). - Томск : Эль Контент, 2014. - 176 с. : схем., ил. - Библиогр.: с. 166-169. - ISBN 978-5-4332-0185-9 ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=480503
5. Карякин, М. И. Технологии программирования и компьютерный практикум на языке Python : учебное пособие : [16+] / М. И. Карякин, К. А. Ватульян, Р. М. Мнухин ; Южный федеральный университет. – Ростов-на-Дону ; Таганрог : Южный федеральный университет, 2022. – 244 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=698687. – Библиогр. в кн. – ISBN 978-5-9275-4108-9. – Текст : электронный.
6. Минина, И. В. Основы современных компьютерных технологий : учебное пособие : [12+] / И. В. Минина, А. В. Прилепина, Т. Ю. Спивак ; Оренбургский государственный университет. – Оренбург : Оренбургский государственный университет, 2014. – Часть 4. –

143 с. : схем., табл., ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=492640. – Библиогр. в кн. – Текст : электронный.
7. Основы алгоритмизации и программирования : лабораторный практикум / Министерство образования и науки Российской Федерации, Федеральное государственное автономное образовательное учреждение высшего профессионального образования «Северо-Кавказский федеральный университет» ; сост. Е.И. Николаев. - Ставрополь : СКФУ, 2015. - 211 с. : ил. - Библиогр. в кн. ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=457504
8. Семакин И.Г. Основы программирования: учебник/ И.Г.Семакин, А.П.Шестаков. – М: Мастерство: НМЦ СПО: Высшая школа, 2001. – 432 с.
9. Хиценко, В.П. Основы программирования : учебное пособие / В.П. Хиценко ; Министерство образования и науки Российской Федерации, Новосибирский государственный технический университет. - Новосибирск : НГТУ, 2015. - 83 с. : схем. - Библиогр. в кн. - ISBN 978-5-7782-2706-4 ; То же [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=438365

Интернет-ресурсы:

1. Самоучитель Python 3 для начинающих. Режим доступа: https://pythonworld.ru/
2. Официальный сайт языка программирования Python. Режим доступа: https://www.python.org/
3. Страница системы программирования Кумир на сайте ФГУ ФНЦ НИИСИ РАН https://www.niisi.ru/kumir/
4. Единое окно доступа к образовательным ресурсам. http://window.edu.ru/
5. Образовательные сайты и порталы. http://school2100.com/
6. Сайты учителей. http://rus.1september.ru/
7. Профессиональный сайт учителей «Методисты». http://nsportal.ru/
8. Социальная сеть работников образования. http://nsportal.ru/shkola

Современные профессиональные базы данных и информационно-справочные системы:

- ЭБС «Университетская библиотека онлайн»;
- Коллекция Шуйского филиала ИвГУ на национальном цифровом ресурсе «Руконт».
- Консультант Плюс http://www.consultant.ru/

Программное обеспечение:

В процессе изучения дисциплины используются офисные пакеты Microsoft Office, интерпретатор языка Python 3.8 (или выше) и среды разработки IDLE и PyScripter, программа для просмотра и чтения файлов PDF Adobe Acrobat Reader DC, программа для воспроизведения флэш-анимации в браузерах Adobe Flash Player, веб-обозреватели Яндекс.Браузер, Mozilla Firefox, Антивирус Касперского, 1С университет.

9. Материально-техническое обеспечение дисциплины

1. Учебная аудитория для проведения занятий лекционного типа, укомплектованная специализированной мебелью и техническими средствами обучения, служащими для предоставления учебной информации большой аудитории.
2. Лаборатория, оснащенная лабораторным оборудованием, укомплектованная

специализированной мебелью и техническими средствами обучения.
3. Учебная аудитория для проведения текущего контроля и промежуточной аттестации, укомплектованная специализированной мебелью и техническими средствами обучения.
4. Помещение для самостоятельной работы, оснащенное специализированной мебелью, компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду организации.
5. Учебно-наглядные пособия для занятий лекционного типа, обеспечивающие тематические иллюстрации: презентации по программированию и алгоритмизации.

Автор рабочей программы дисциплины: В.А. Смирнов, старший преподаватель кафедры математики, информатики и методики обучения.

Программа рассмотрена на заседании кафедры математики, информатики и методики обучения «19» февраля 2025 г., протокол № 8

Программа обновлена
протокол заседания кафедры № _____ от «_____» _____20 __ г.
Согласовано:
Руководитель ОП _____ А.С. Зинченко
(подпись)

Программа обновлена
протокол заседания кафедры № _____ от «_____» _____20 __ г.
Согласовано:
Руководитель ОП _____ А.С. Зинченко
(подпись)

Программа обновлена
протокол заседания кафедры № _____ от «_____» _____20 __ г.
Согласовано:
Руководитель ОП _____ А.С. Зинченко
(подпись)

Приложение 1

Методические указания для обучающихся по освоению дисциплины

На самостоятельную работу студентов по курсу «Основы алгоритмизации и программирования» отводится более 60% учебного времени от общей трудоемкости курса. Сопровождение самостоятельной работы студентов по данному курсу может быть организовано в следующих формах:

– согласование индивидуальных заданий (виды и темы заданий, сроки представления результатов) самостоятельной работы студента в пределах часов, отведенных на самостоятельную работу;

– консультации (индивидуальные и групповые), в том числе с применением дистанционных технологий взаимодействия;

– промежуточный контроль хода выполнения заданий, строящихся на основе различных способов самостоятельной информационной деятельности в открытой информационной среде.

Преподаватель определяет содержание самостоятельной работы, график её выполнения; создает сетевую информационную и коммуникационную среду для организации самостоятельной работы.

В процессе организации самостоятельной работы особое внимание уделяется не только развитию информационной культуры, но и формированию готовности к кооперации, к работе в коллективе в сетевом пространстве, для чего целесообразно стимулировать студентов к совместной деятельности в малых группах.

Одной из важнейших особенностей заданий для самостоятельной работы по курсу «Основы алгоритмизации и программирования» является их ориентация на формирование способности оценивать приоритеты и ограничения при выборе средств информационных технологий для решения управленческих задач, а также на мотивирование самообразовательной деятельности с использованием ИКТ.

К типовым заданиям для самостоятельной работы студентов относятся:

– Конспектирование и реферирование литературных источников по тематике учебного курса.

– Решение задач, связанных с установкой программного обеспечения для написания программ, создание программ на языках Кумир и Python для решения прикладных задач;

– Изучение и анализ предложенных электронных образовательных ресурсов, содержащих информацию по основам алгоритмизации и программирования;

– Разработка проектов в среде программирования Scratch;

– Повторение материала с использованием конспектов.

Перечень вопросов для самостоятельного изучения

1. Определение алгоритма. Свойства алгоритма. Формы записи алгоритмов. Примеры.
2. Запись алгоритмов блок-схемами. Основные элементы блок-схем.
3. Алгоритмы с ветвлением. Пример алгоритма.
4. Алгоритм цикла с предусловием. Пример алгоритма.
5. Алгоритм цикла с постусловием. Пример алгоритма.
6. Алгоритм цикла с управляющей переменной. Пример алгоритма.
7. Логические и арифметические операции в среде программирования Scratch
8. Дополнения «Музыка», «Перо», «Текст в речь», «Перевести» среды программирования Scratch

9. Среда программирования Snap! в обучении школьников на курсах дополнительного образования
10. Среда программирования Alice в обучении школьников на курсах дополнительного образования
11. Проведение занятий по программированию с использованием учебного исполнителя «Черепашка» в среде Кумир
12. Проведение занятий по программированию с использованием учебного исполнителя «Чертежник» в среде Кумир
13. Функции для работы со строками среды программирования Кумир
14. Команды для работы с файлами среды программирования Кумир
15. Массивы. Линейный и бинарный поиск в массиве.
16. Сортировка массивов. Сортировка выбором. Сортировка методом "пузырька". Сортировка вставками. Сортировка подсчетом. Сортировка слиянием. Быстрая сортировка Хоара.
17. Переменные. Типы данных в Python.
18. Структура программы. Команда присваивания в Python.
19. Ввод-вывод данных. Формат выводимых данных в Python.
20. Алгоритм линейной структуры в Python.
21. Структура if, классификация в Python. Примеры.
22. Алгоритмы циклической итерационной структуры. Оператор цикла while в Python. Примеры использования.
23. Строки как последовательности символов в Python. Примеры использования.
24. Алгоритмы циклической итерационной структуры. Оператор цикла обхода for в Python. Примеры использования.
25. Списки – изменяемые последовательности в Python. Примеры использования.

ЗАДАНИЯ ДЛЯ ОРГАНИЗАЦИИ ЛАБОРАТОРНЫХ ЗАНЯТИЙ (1 СЕМЕСТР):

Лабораторная работа №1 «Основы алгоритмизации»

Цель работы: усвоить понятия: алгоритм как фундаментальное понятие информатики, способы описания, основные типы алгоритмов, освоить принципы решения задач с использованием основных алгоритмических конструкций.

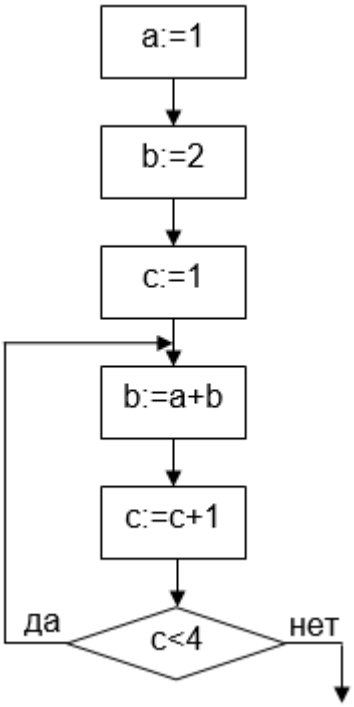
Общие сведения: Этапы решения задачи на ЭВМ. Работа по решению любой задачи с использованием компьютера делится на следующие этапы:

- Постановка задачи.
- Формализация задачи.
- Построение алгоритма.
- Составление программы на языке программирования.
- Отладка и тестирование программы.
- Проведение расчетов и анализ полученных результатов.

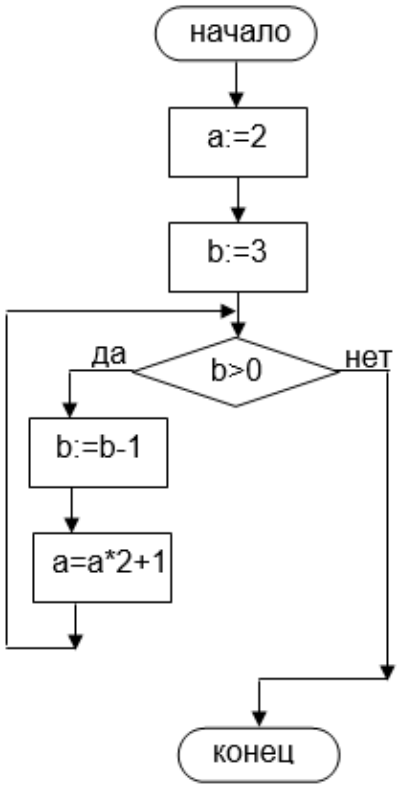
Блок-схема алгоритма — графическое изображение алгоритма в виде связанных между собой с помощью стрелок (линий перехода) и блоков — графических символов, каждый из которых соответствует одному шагу алгоритма. Внутри блока дается описание соответствующего действия.

Задачи:

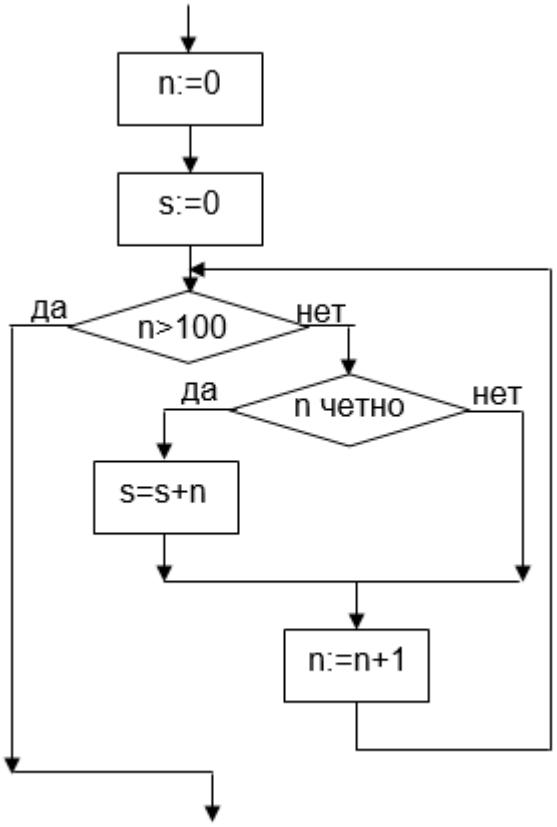
1. Составить алгоритм решения задачи с помощью блок-схем, используя конструкцию линейного алгоритма.
    a. Определить площадь трапеции по введенным значениям оснований (a и b) и высоты (h)
    b. Вычислить площадь треугольника со сторонами a,b,c по формуле Герона.
2. Определите значение переменной b после выполнения фрагмента алгоритма.



3. Определите значение переменной a после выполнения алгоритма.



4. Определите значение переменной s после выполнения фрагмента алгоритма.



5. Составить алгоритм решения задачи с помощью блок-схем, используя конструкцию алгоритма с ветвлением:
    a. Определить максимальное четное число из двух введенных;
    b. Определить, можно ли из отрезков с длинами x, y и z построить треугольник.
    c. Определить, в каком квадранте находится точка с координатами x и y.
6. Составить алгоритм решения задачи с помощью блок-схем, используя конструкцию циклического алгоритма.
    a. Найти сумму чисел, кратных трем, в диапазоне от 0 до 50.
    b. Найти сумму первых десяти чисел, кратных пяти.
    c. Определить количество целых чисел, кратных 3 (от 3 и далее), дающих в сумме число, превышающее 200.
7. Составить алгоритм решения практической задачи с помощью блок-схем:
    a. Вы положили в банк 1500 рублей. Определить, сколько денег будет на Вашем вкладе через 1 год, если каждый месяц вклад увеличивается на 0.76 % от суммы предыдущего месяца.
    b. Решив заняться легкой атлетикой, Вы пробежали в первый день 2 км. Сколько километров Вы пробежите за 2 недели, если каждый день Вы увеличиваете дистанцию на 10% от предыдущего дня?

Критерии оценки:
Учащийся выполняет все задачи лабораторной работы. Ответы на вопросы 2-4 оцениваются в 1 балл, решение задач в каждом из остальных вопросов (1, 5-7) оценивается в 1 балл.
Дополнительно: 1 балл выставляется за написание отчёта по проделанной работе.
Итого: 6 баллов.

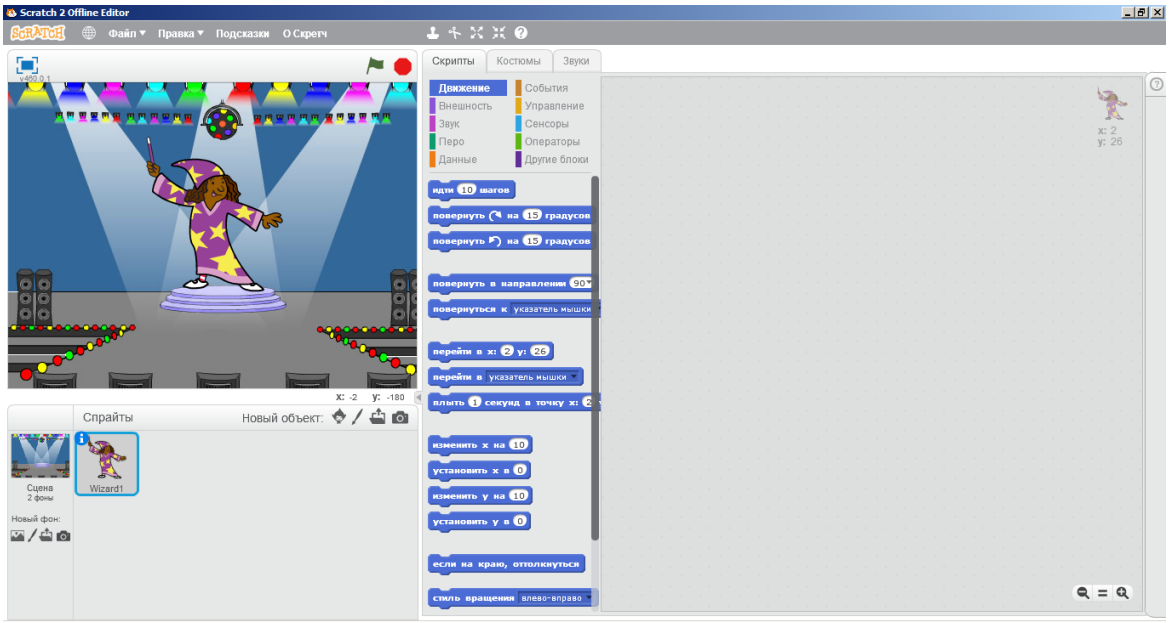
Лабораторная работа №2 «Простые проекты в среде Scratch»

Часть 1 «Сообщения»

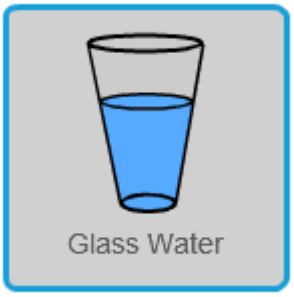
Факт передачи сообщения является событием, на которое могут реагировать спрайты. При помощи этого механизма можно организовать запуск скриптов одного спрайта из другого.

Передача сообщений происходит невидимо для пользователя, но на каждое из сообщений может среагировать любой спрайт.

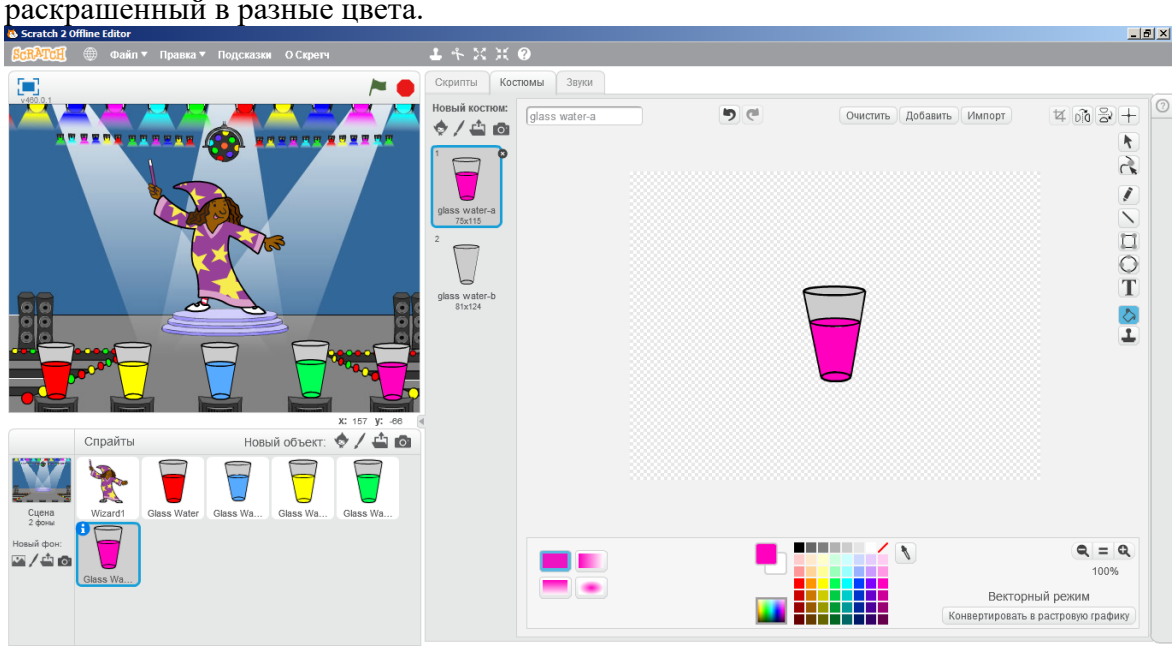
Шаг 1: Выбрать фон и персонажа.



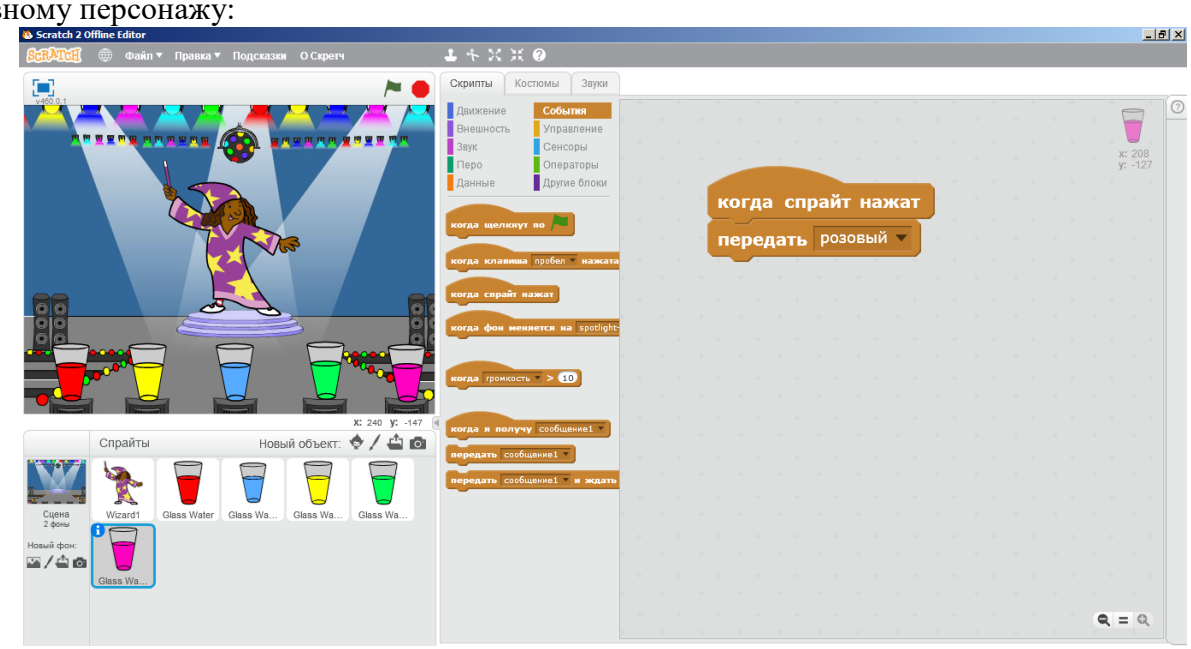
Шаг 2: Выбираем предметы, который будет добавлять эффекты персонажу. Рекомендуем использовать спрайт

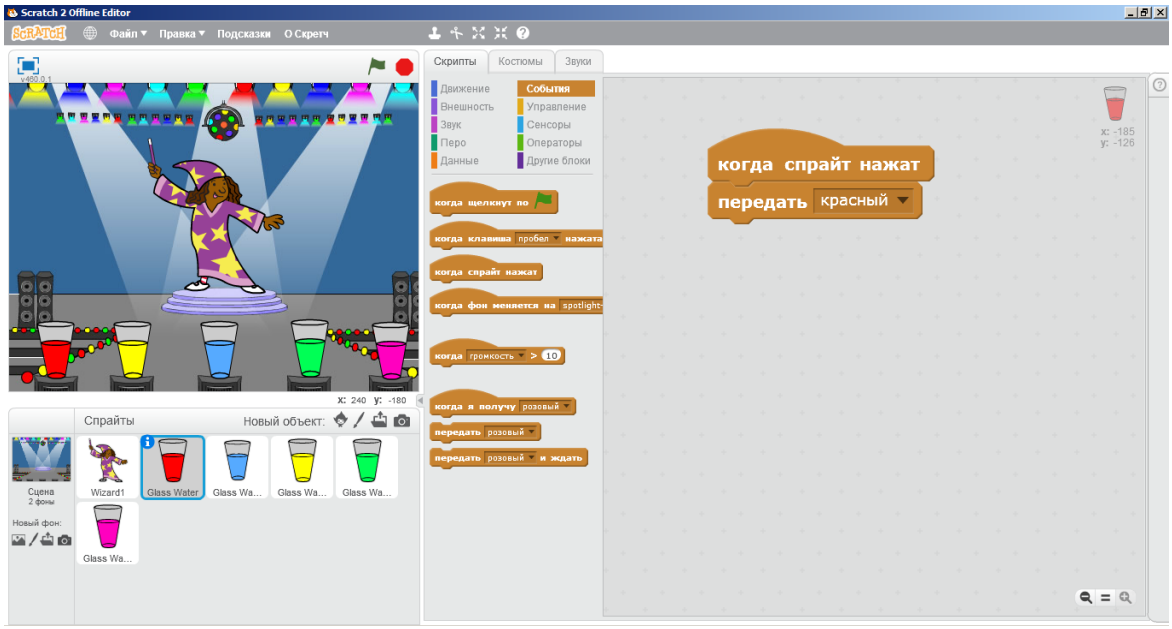


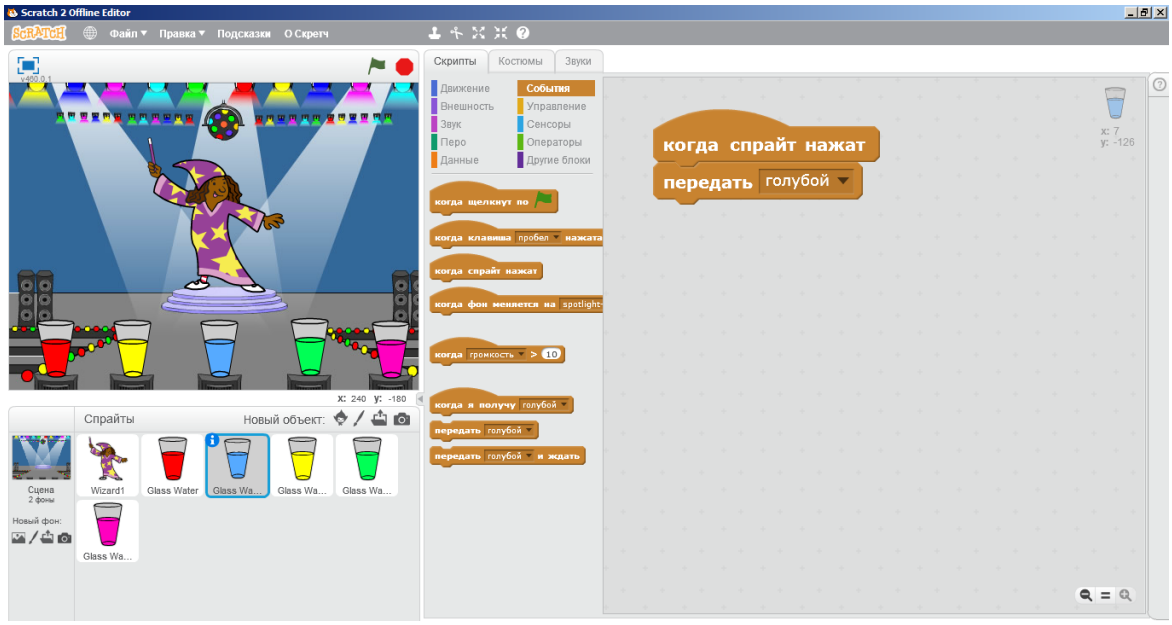
раскрашенный в разные цвета.



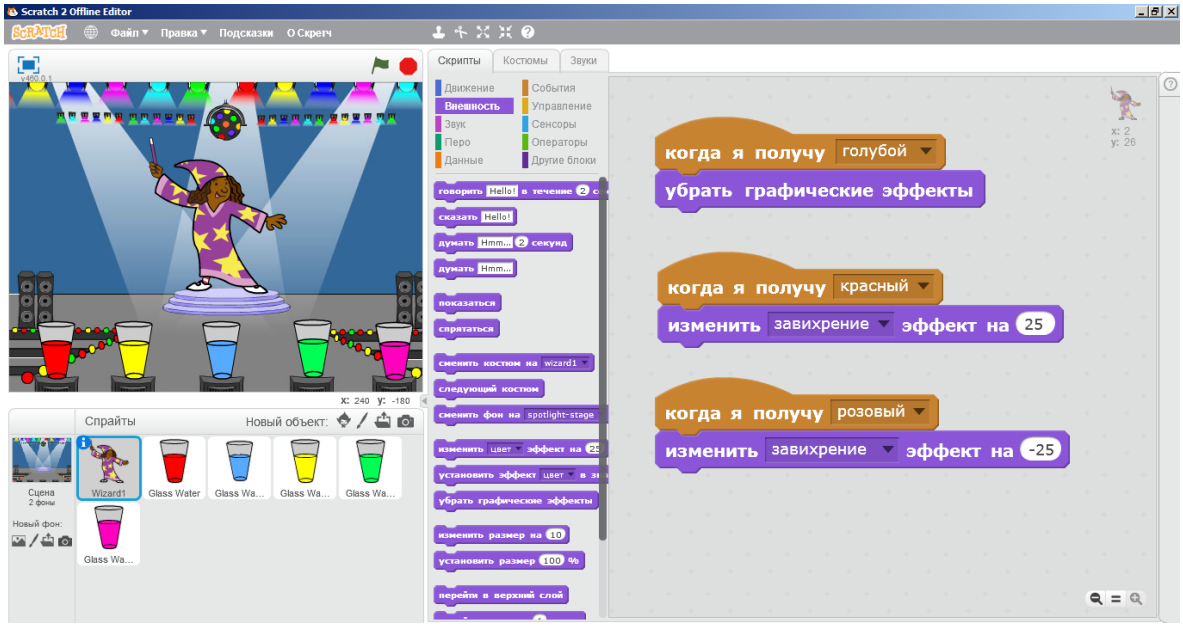
Каждому «стакану» назначаем соответствующий скрипт о передаче сообщения нашему главному персонажу:







Шаг 3: В персонаже добавляем обработку сообщений:



Шаг 4: Проверить работу скриптов. Красный стакан увеличивает эффект завихрения, а розовый – уменьшает. Нажатие на голубой стакан возвращает спрайт в первоначальное состояние.

Задание:
1) Сменить эффект «завихрение» на любой другой.
2) Самостоятельно добавить эффекты по двум оставшимся стаканам.

Сохранить проект под именем «ФамилияИО_Сообщения».

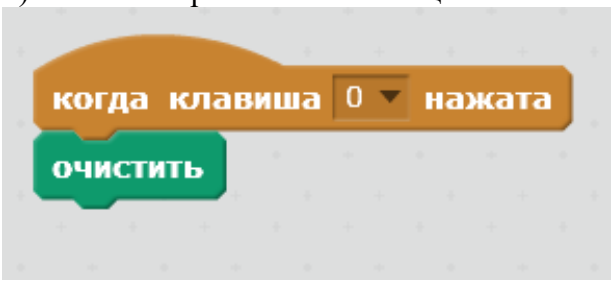
Часть 2 «Перо»

Перо необходимо для рисования линий на сцене. Все нарисованные линии могут быть удалены при выполнении соответствующей команды («очистить»).

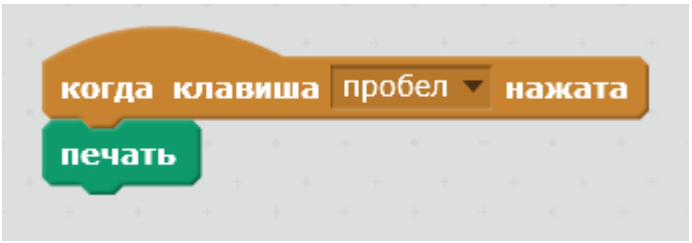
Перо может находиться в двух состояниях: опущено (когда при перемещении спрайта на сцене отрисовывается проделанный им путь), поднято (когда рисования не происходит).

В новом проекте запрограммировать у кота следующие действия:

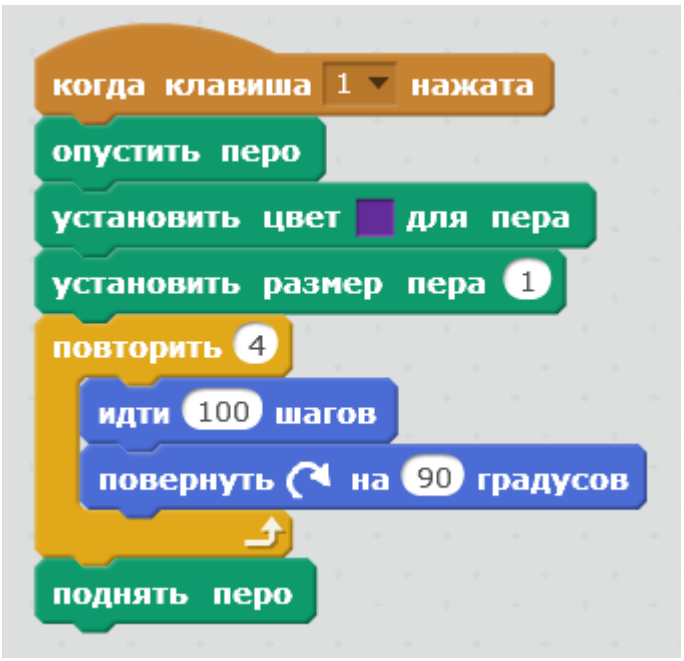
1) Очистка нарисованного на сцене по нажатию на клавишу 0 на клавиатуре.



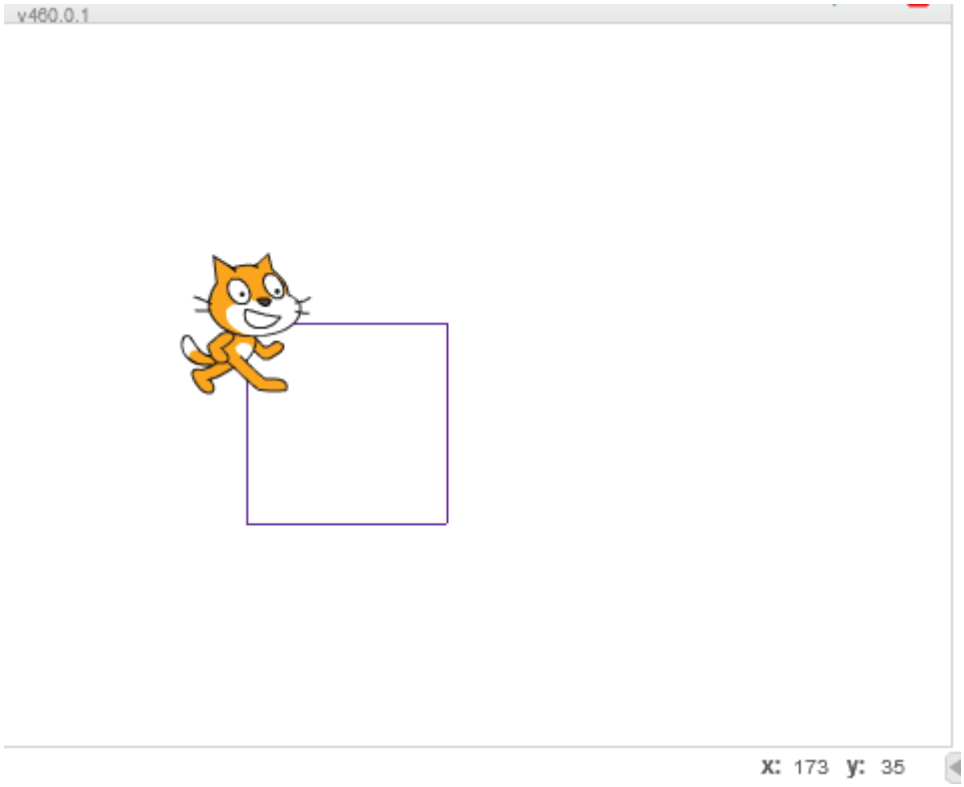
2) Оттиска спрайта по нажатию на клавишу «пробел» на клавиатуре.



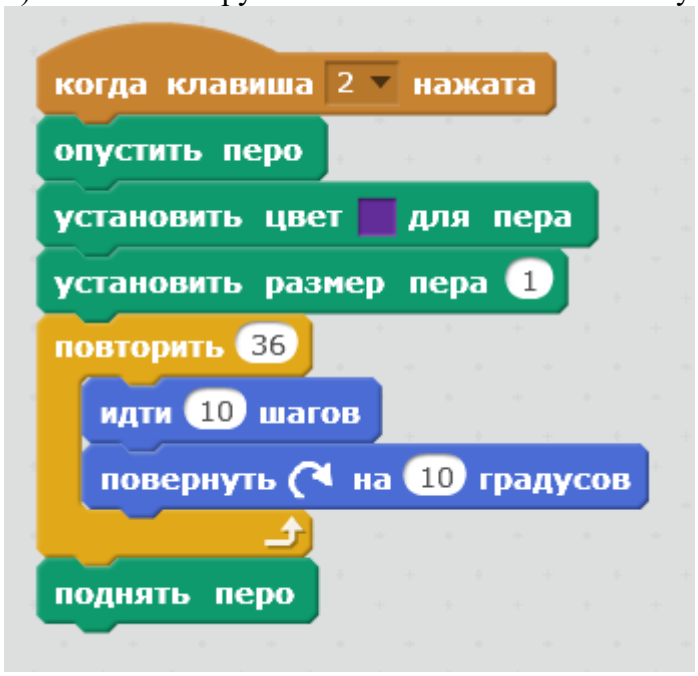
3) Рисование квадрата по нажатию на клавишу 1 на клавиатуре.



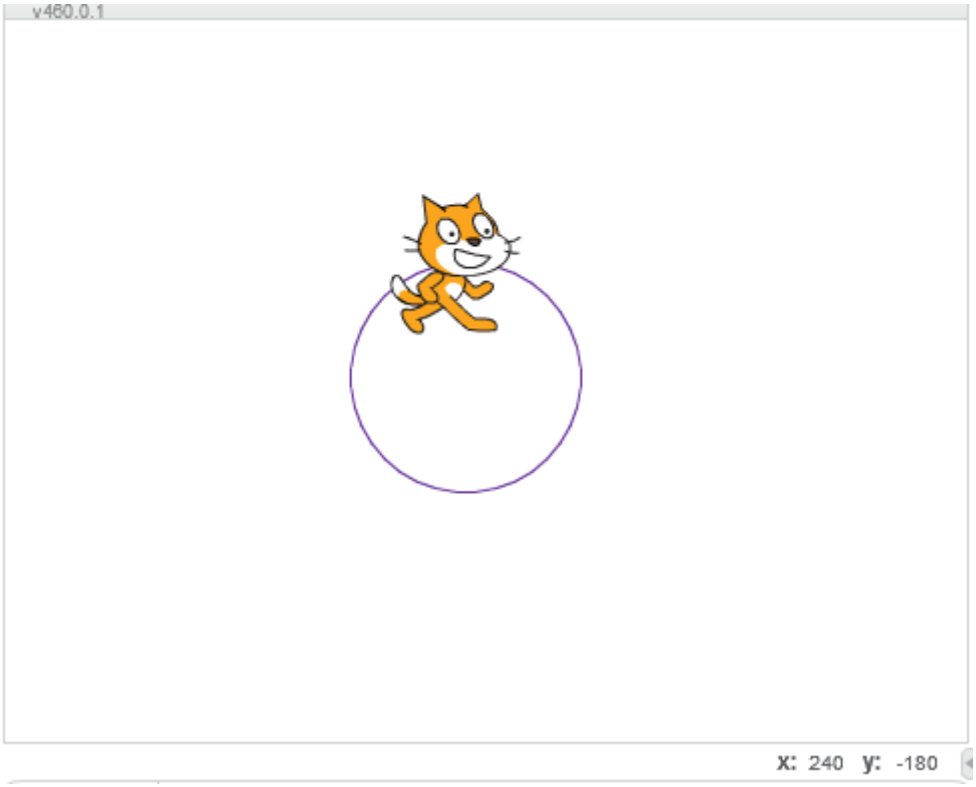
Результат нажатия на кнопку:



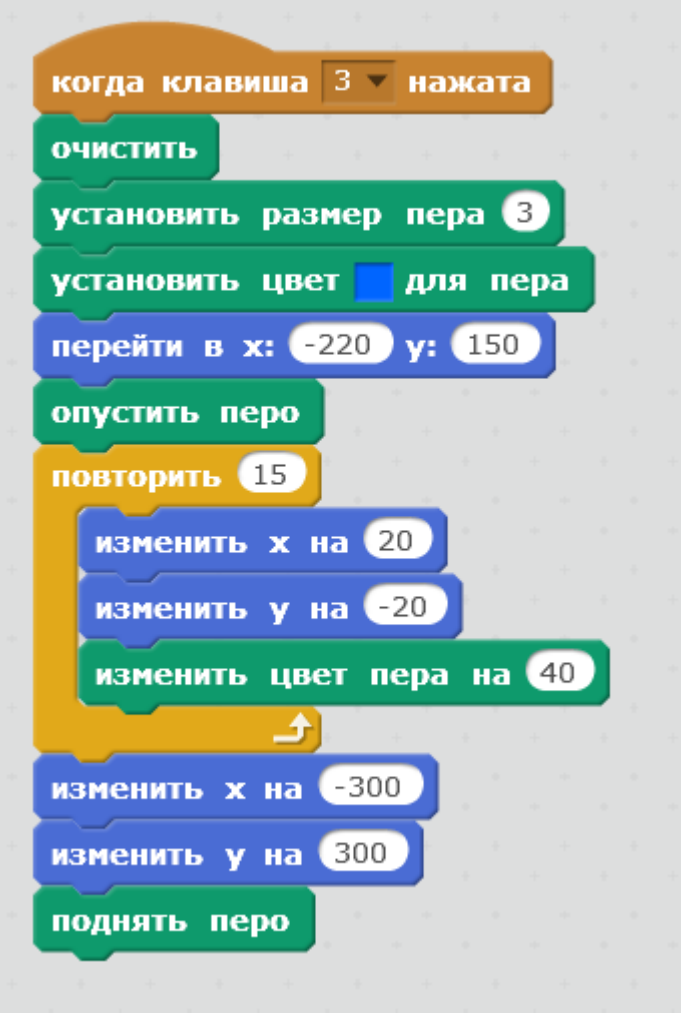
4) Рисование окружности по нажатию на клавишу 2 на клавиатуре.



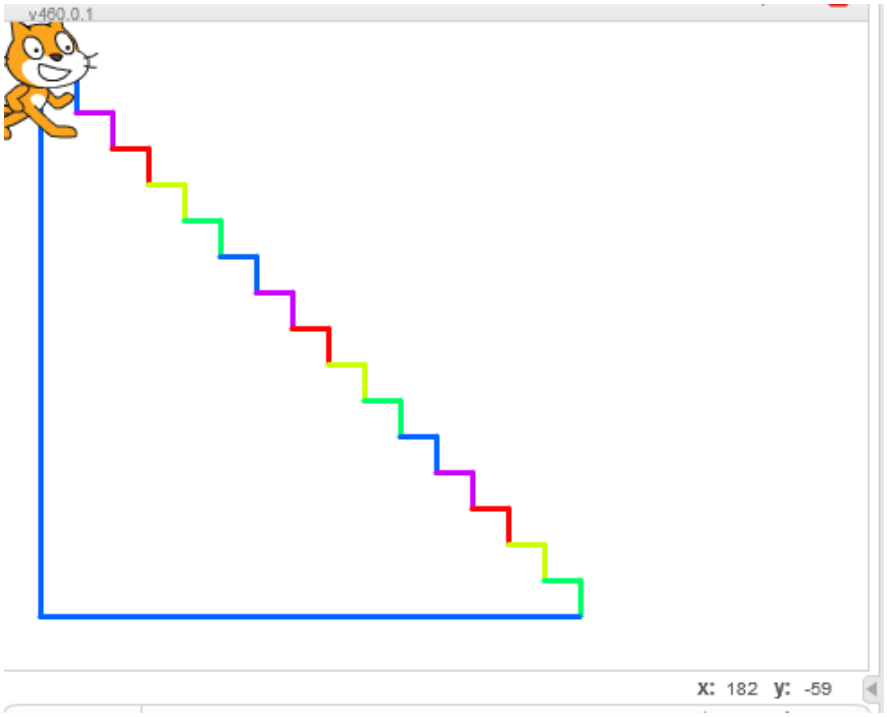
Результат нажатия на кнопку:



5) Рисование лестницы по нажатию на клавишу 3 на клавиатуре.



Результат нажатия на кнопку:



Сохранить проект под именем «ФамилияИО_Перо».

Критерии оценки:
Верное самостоятельное выполнение всех указанных в лабораторной работе шагов – 3 балла
Составление отчёта – 2 балла
Итого за лабораторную работу: 5 баллов.

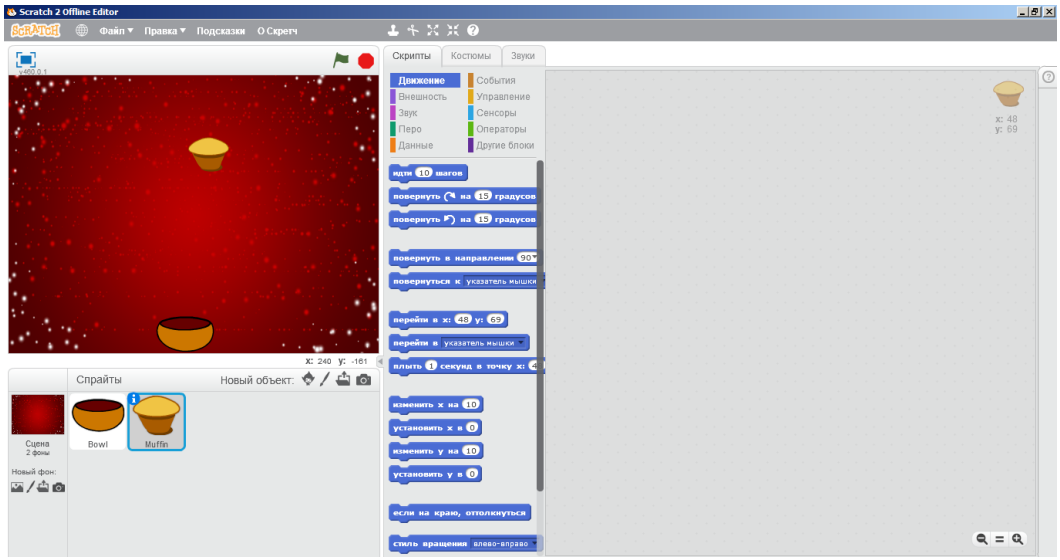
Лабораторная работа №3 «Простые проекты в среде Scratch»

Часть 1 «Подготовка игрового поля»

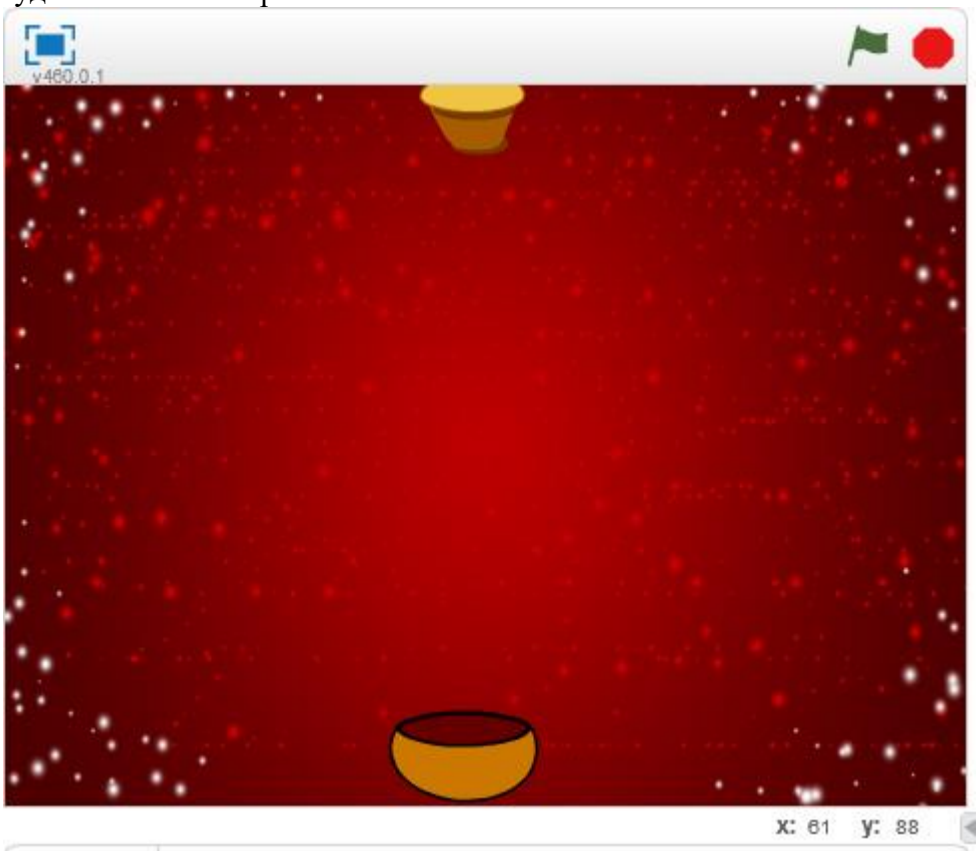
Шаг 1: Выбрать фон и спрайты:

1) Спрайт, который будет основой для падающих объектов (на скриншоте ниже таким спрайтом является кекс);

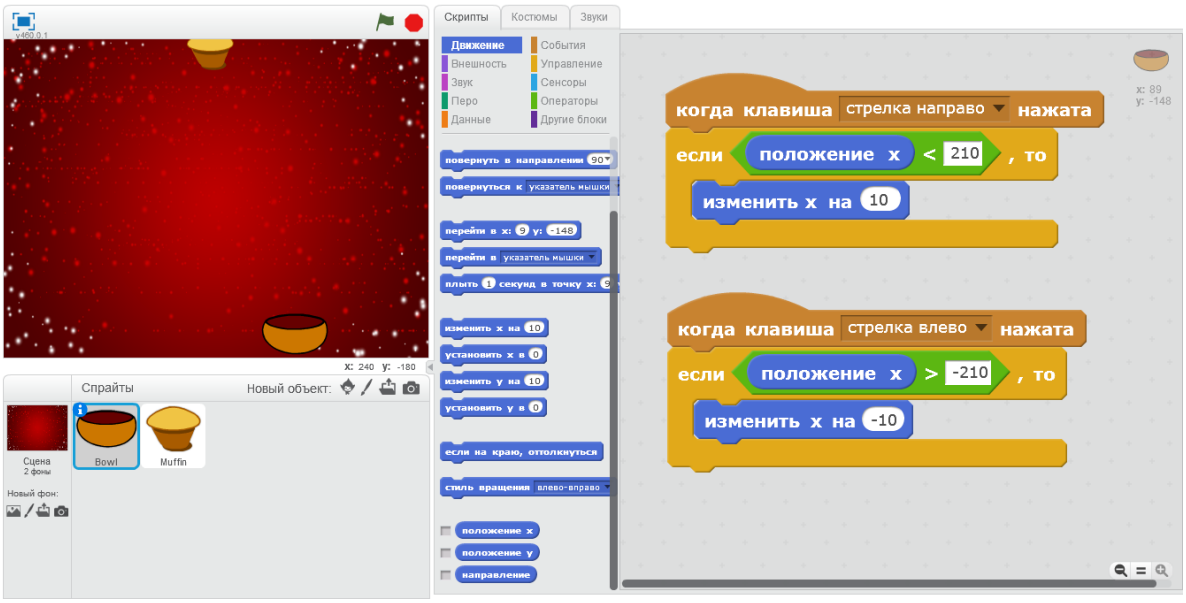
2) Спрайт, который будет их ловить.



Шаг 2: Расположить предмет для ловли объектов в нижней части сцены. Предмет, который нужно будет ловить – в верхней.



Шаг 3: Добавить нижнему спрайту команды:



Шаг 4: Проверить работу скриптов. Объект должен перемещаться нажатием стрелок на клавиатуре, но не уходить за край сцены.

Часть 2 «Клоны»

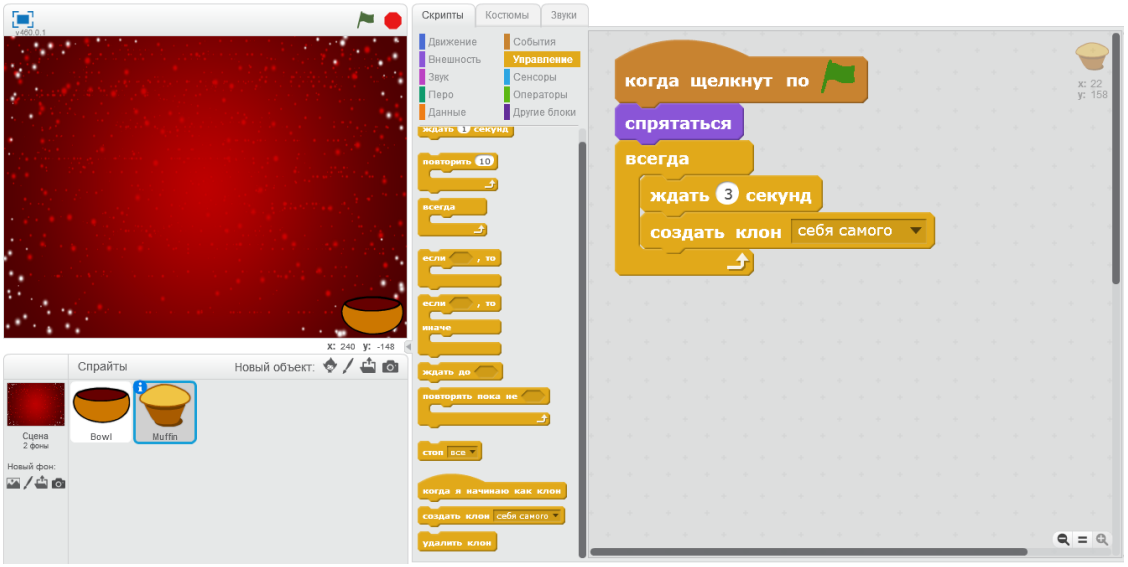
Помимо механизма дублирования объектов из раздела «Спрайты», в разделе «Управление» есть команда, которая называется «Создать клон «себя самого»».

В отличие от процесса дублирования при клонировании новый спрайт не создаётся. Клонирование приводит к возникновению на сцене точной копии клонируемого спрайта, обладающей такими же свойствами и скриптами.

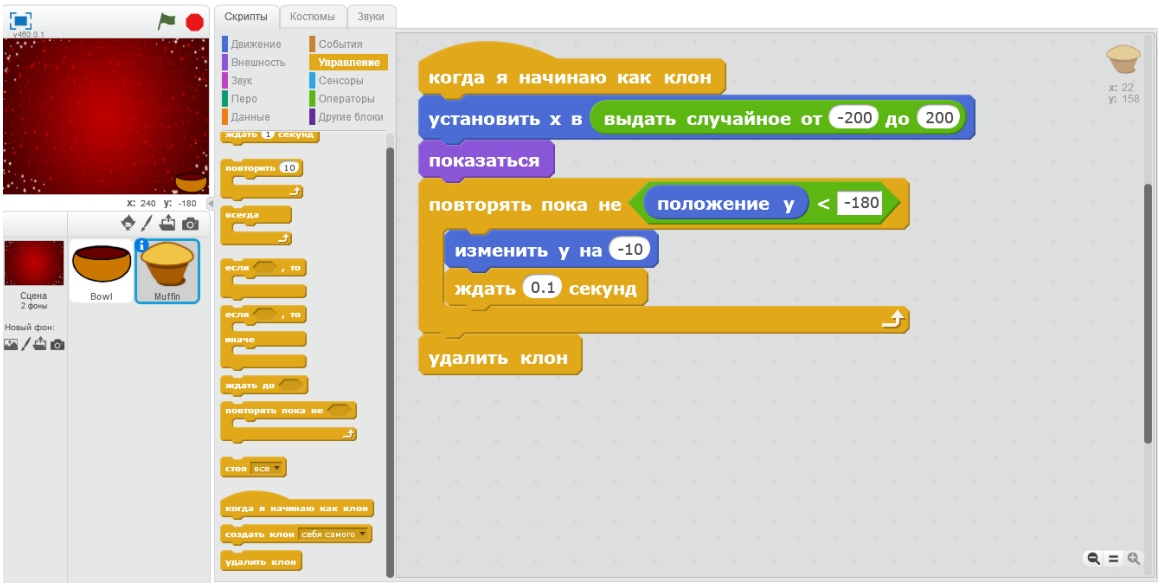
Клонировать спрайт можно только с помощью специальной команды в процессе работы программы. При её остановке с помощью кнопки  все клоны уничтожаются.

В рамках работы программы не рекомендуется допускать создание более **300** клонов для спрайтов.

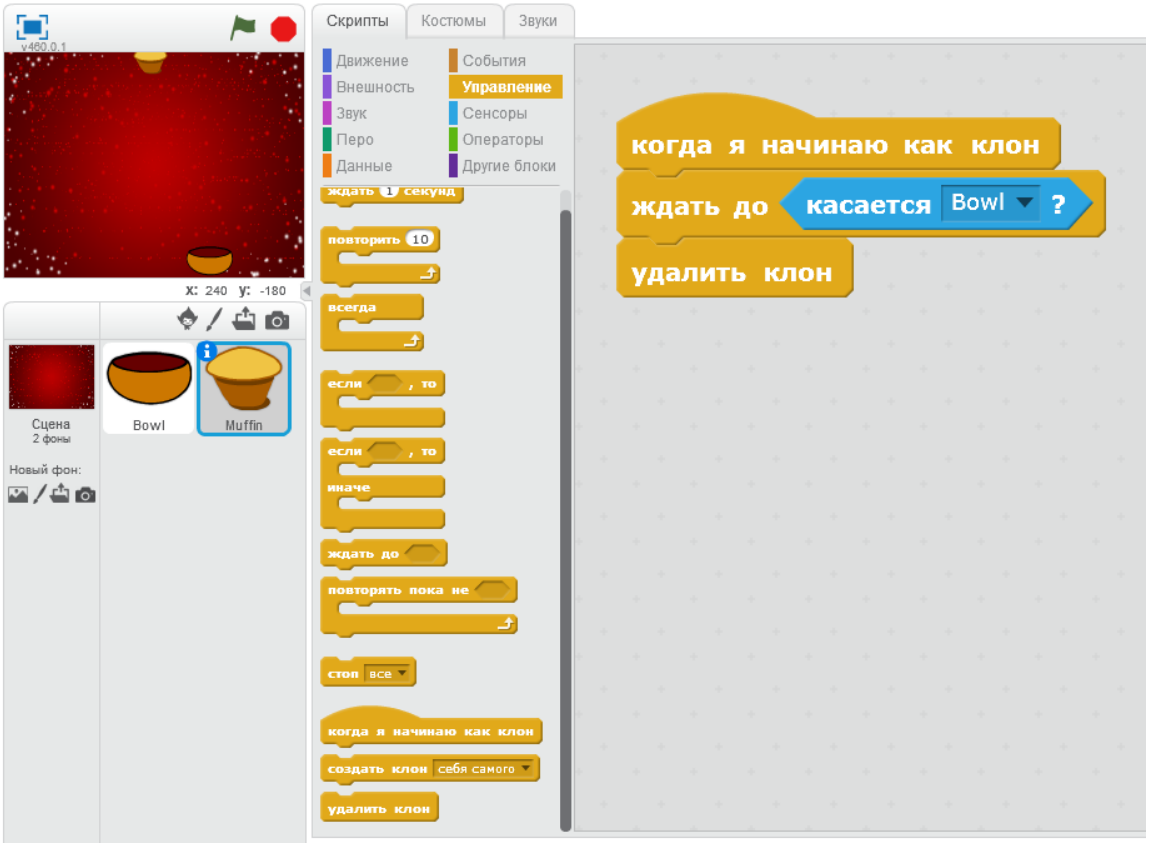
У падающего предмета программируем следующие действия:



В результате этих команд каждые 4 секунды будет появляться новый клон спрайта. Далее добавляем команды, чтобы этот клон падал вниз:



Остаётся добавить команды, отвечающие за ловлю этого объекта нижним спрайтом:

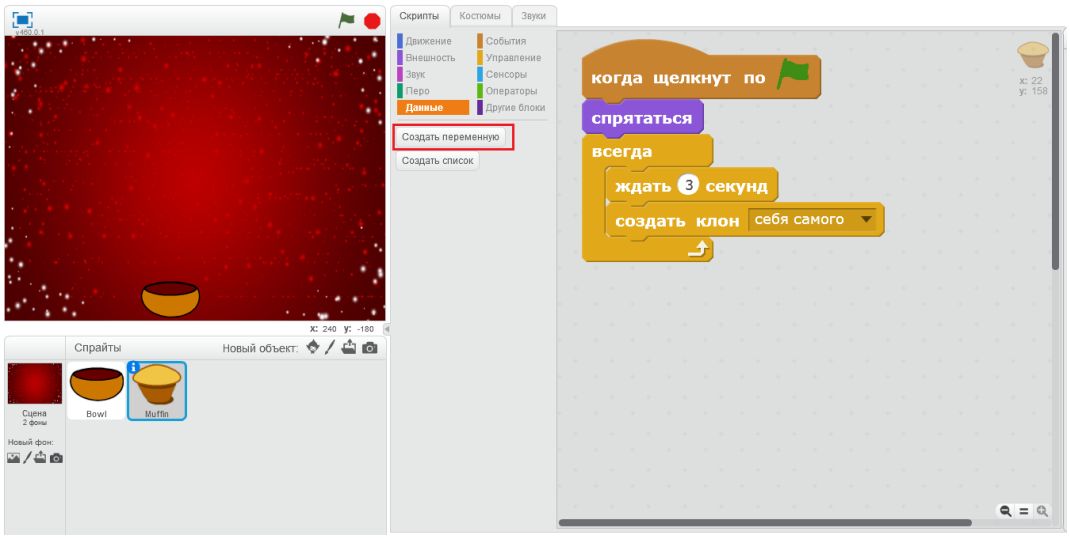


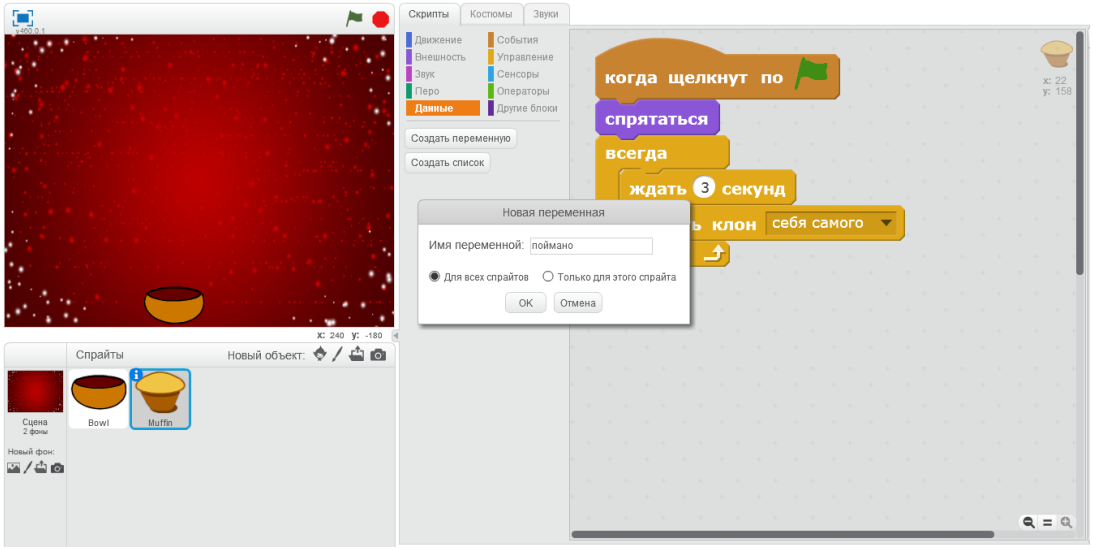
Часть 3 «Переменные»

Для создания полноценной игры осталось добавить подсчёт количества пойманных предметов. Чтобы хранить это число и впоследствии его изменять, в Scratch, как и во многих языках программирования, используются переменные.

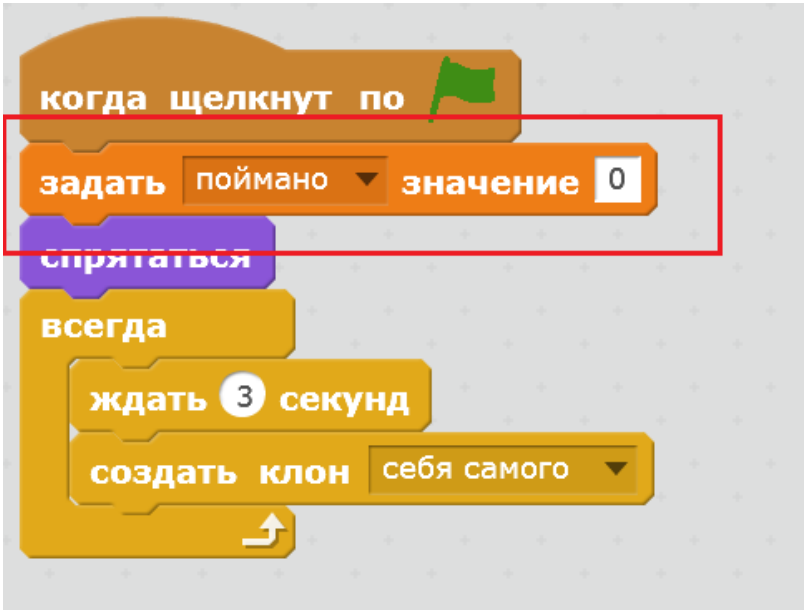
Переменная состоит из имени и значения. Значение – это и есть какая-либо сохраненная информация, чем в нашем случае является количество пойманных предметов. Через имя переменной мы обращаемся к ячейкам памяти компьютера, чтобы получить оттуда данные или изменить их.

В разделе «Данные» создаём переменную:

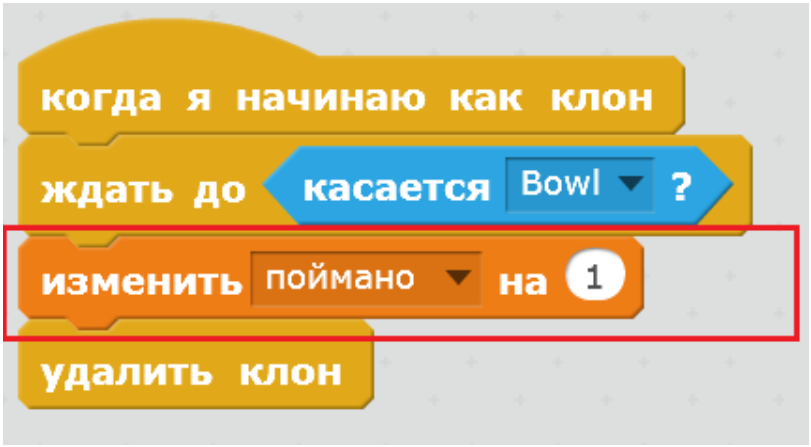




Падающему спрайту в первый блок добавляем команду:



А в последний блок:



Сохранить проект под именем «ФамилияИО_Ловля предметов».

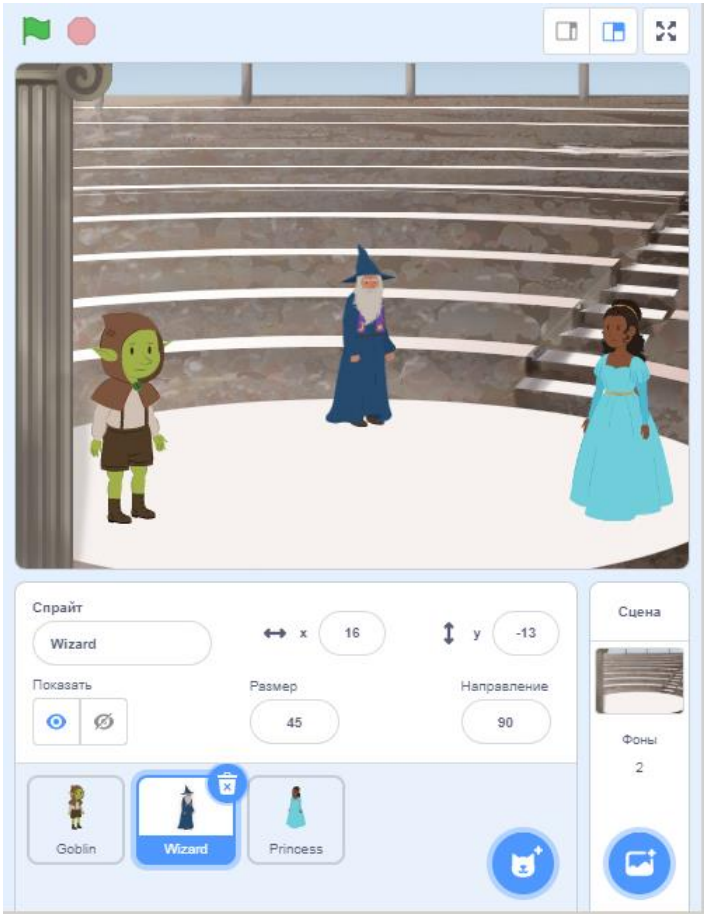
Критерии оценки:
Верное самостоятельное выполнение всех указанных в лабораторной работе шагов – 3 балла
Составление отчёта – 2 балла
Итого за лабораторную работу: 5 баллов.

Лабораторная работа №4 «Ввод и обработка данных в Scratch»

Часть 1 «Подготовка игрового поля»

Выбрать произвольный фон и любых трёх персонажей, с которыми игрок будет взаимодействовать.

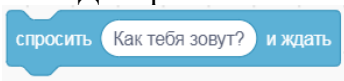
Расположить персонажей можно произвольно.



Часть 2 «Программирование первого персонажа»

Конечный результат: при клике персонаж спрашивает у игрока два числа. После этого говорит их сумму.

Для решения этой задачи в Scratch используется переменная 

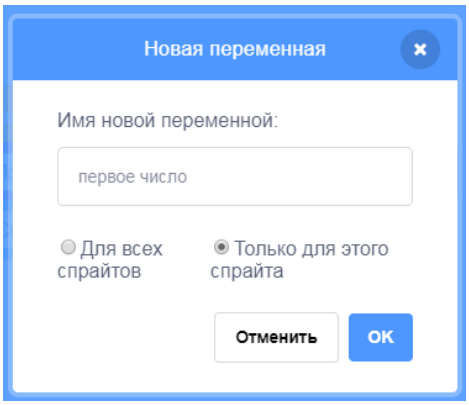
и команда 

, которые находятся в разделе «сенсоры».

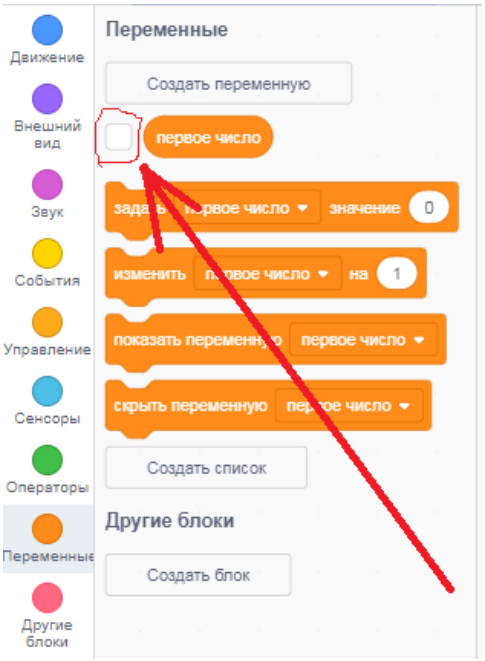
Переменная 

может запоминать только одно значение. Поэтому для хранения двух чисел нам необходимо создать ещё одну переменную. Напоминаем, что сделать это можно в разделе «Переменные».

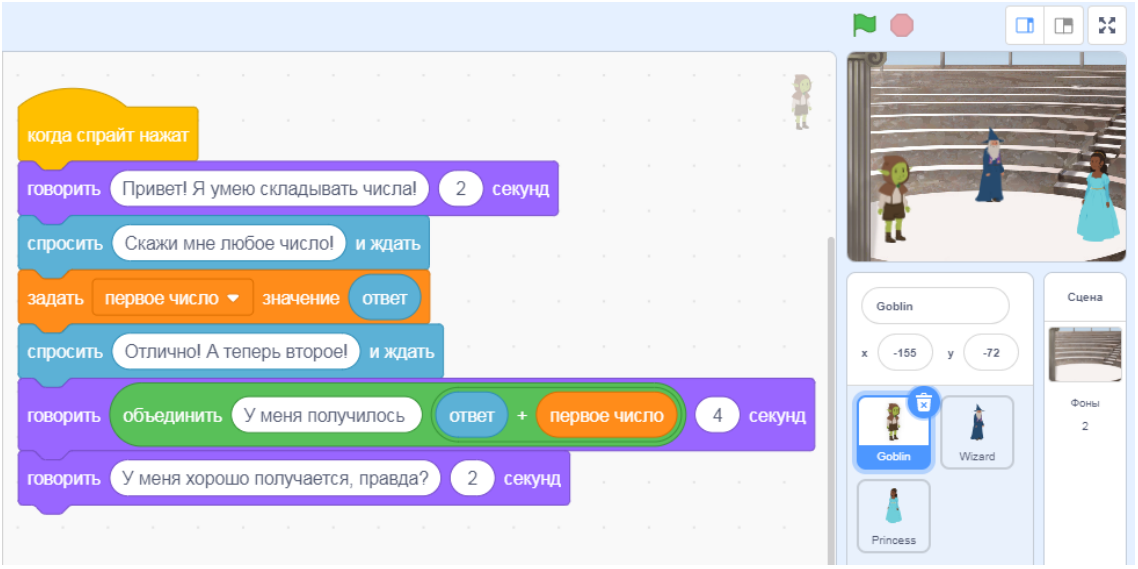
В этой работе, в отличие от прошлой, нам будет необходимо использовать несколько переменных. Поскольку использовать созданную нами переменную будет только тот персонаж, который занимается сложением чисел, необходимо выбрать пункт «Только для этого спрайта». В ином случае данная переменная будет отображаться и у других спрайтов, что будет мешать найти другие переменные.



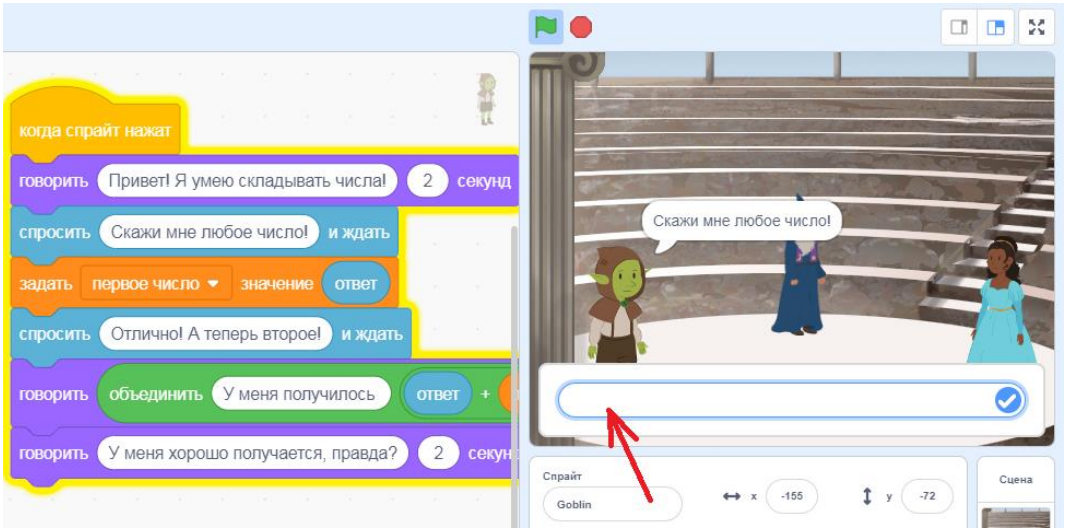
Необходимости отображать переменную на экране нет. Поэтому галочка с поля, сигнализирующая о её отображении на сцене, должна быть снята.



Программируем спрайт на выполнение вычислений:



Проверяем работу скрипта, кликнув на персонажа. Вводить число необходимо в появившееся поле для ввода.



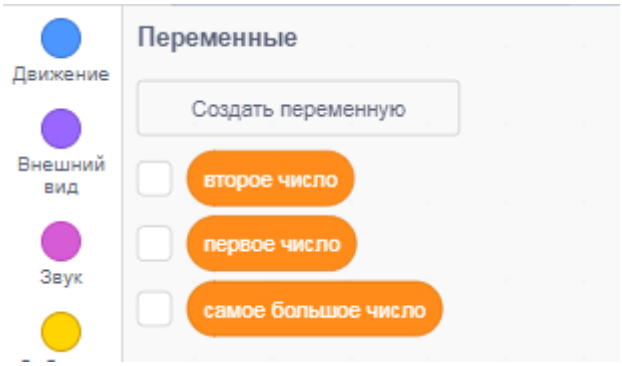
После ввода числа можно нажать «Enter» или на синюю кнопку справа. В дробных числах для разделения дробной и целой частей используется точка.

Самостоятельно замените сложение умножением.

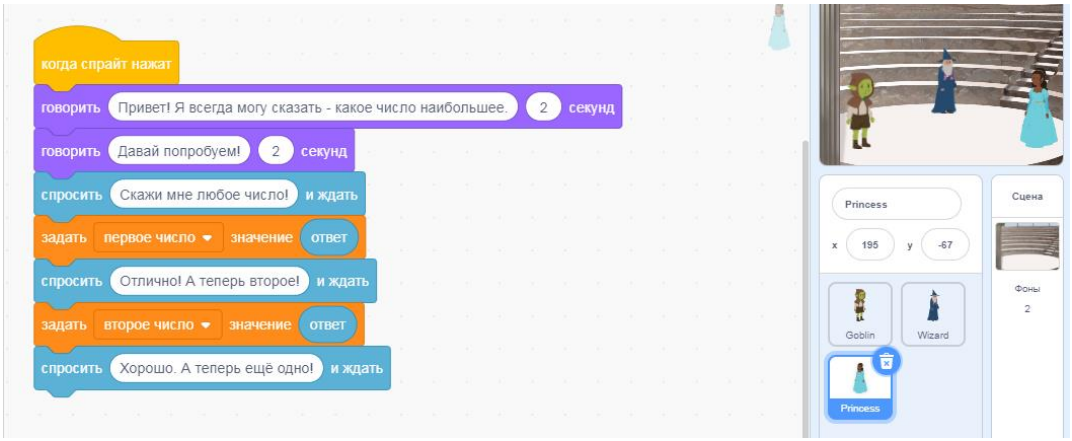
Часть 3 «Программирование второго персонажа»

Конечный результат: при клике персонаж спрашивает у игрока три числа. После этого говорит максимальное из них.

Для этого персонажа необходимо создать три переменных: «первое число», «второе число» и «самое большое число». Все переменные будут задействованы только в данном спрайте.

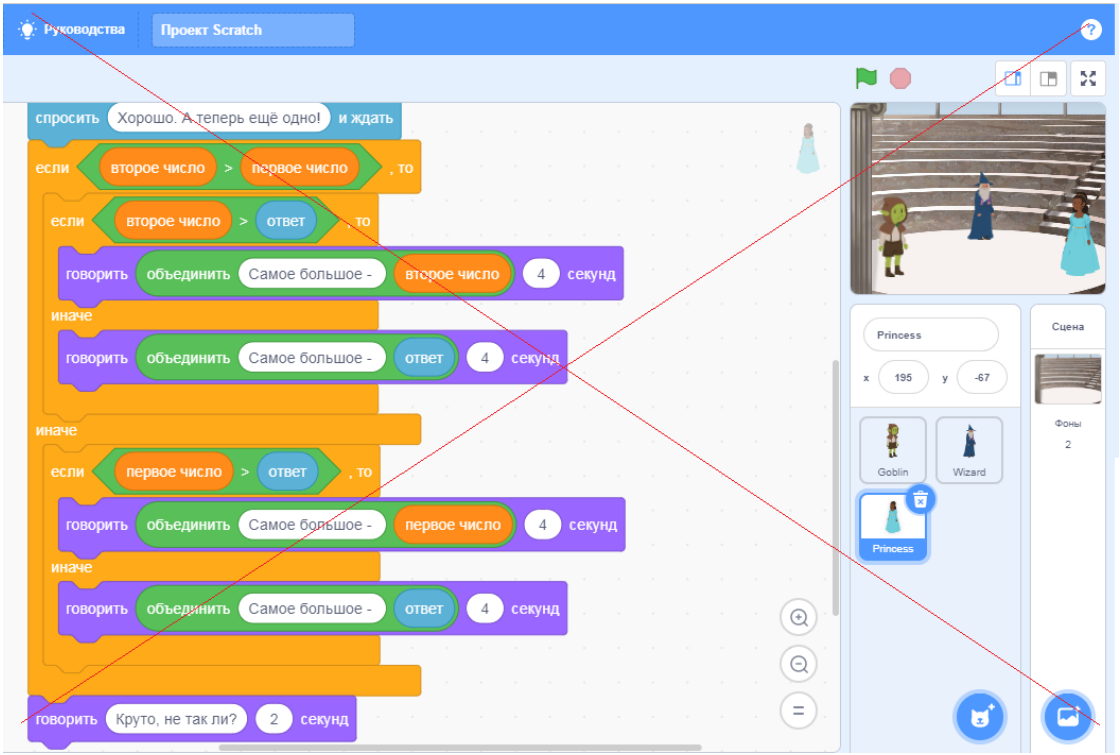


Для персонажа начало скрипта будет выглядеть следующим образом:



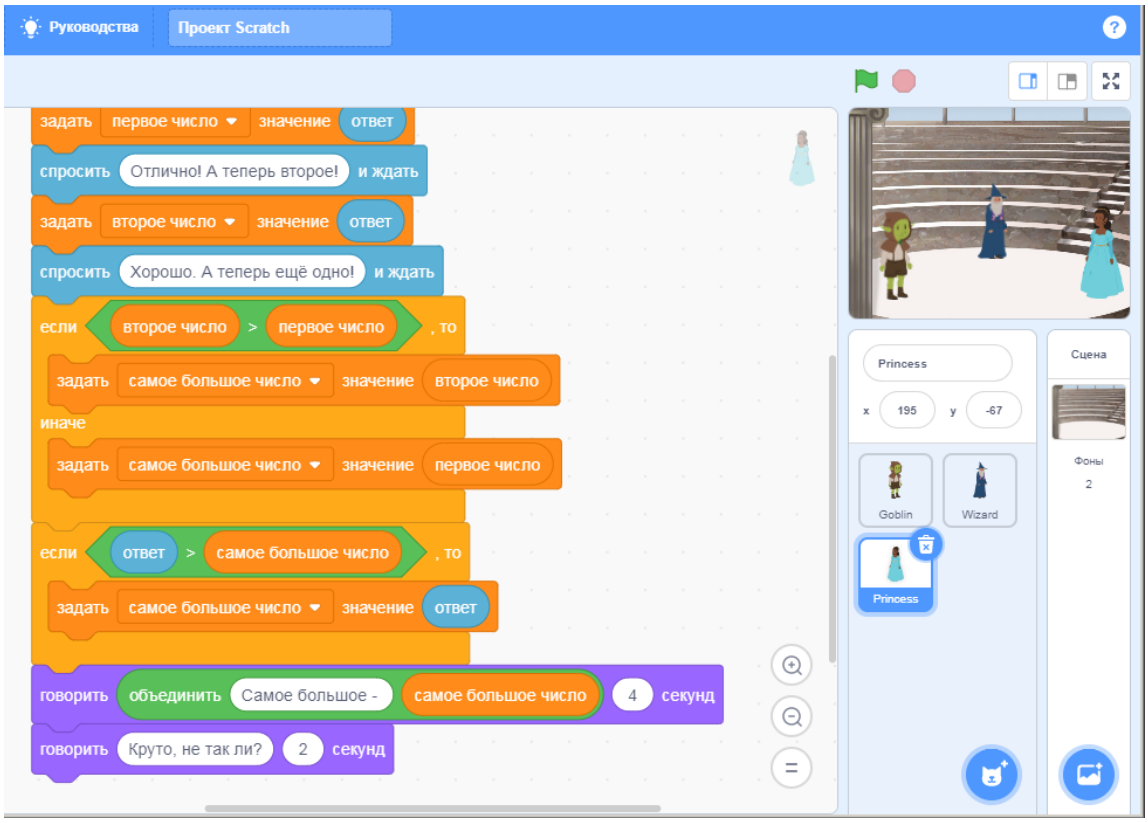
После этого у программиста есть два пути написания скрипта. Первый вариант – неоптимальный. Можно найти с использованием конструкции «если ..., то ... иначе ...» максимальное из первых двух чисел, а затем сравнивать его с третьим с помощью ещё одной конструкции. Переменная «самое большое число» в этом случае не понадобится.

Использовать этот вариант не стоит.



Второй вариант – мы ищем сначала наибольшее только из чисел «первое число» и «второе число». А затем помещаем его в переменную «самое большое число».

После этого мы забываем о существовании переменных «первое число» и «второе число», а сравниваем между собой только переменные «самое большое число» и «ответ». Максимальное из этих двух чисел и будет ответом.



До такого варианта сложнее догадаться, но именно в этом случае скрипт стоит считать написанным более оптимально. В олимпиадах по программированию и конкурсах проектов оценивается не только количество команд, но и качество программы, которая с их помощью составлена.

Часть 4 «Программирование третьего персонажа»

Конечный результат:

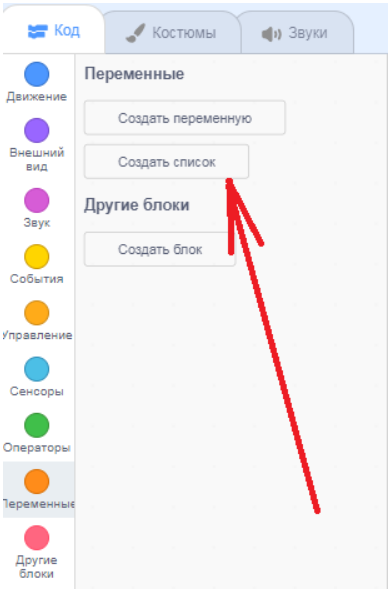
Шаг 1. При клике персонаж спрашивает у игрока количество чисел, которые он будет вводить (обозначим это значение как N).

Шаг 2. Персонаж спрашивает у игрока значение каждого из этих чисел (то есть задаст вопрос о вводе числа N раз).

Шаг 3. Персонаж даёт ответ на вопрос: сколько чисел, которые ввел пользователь на шаге 2, являются чётными.

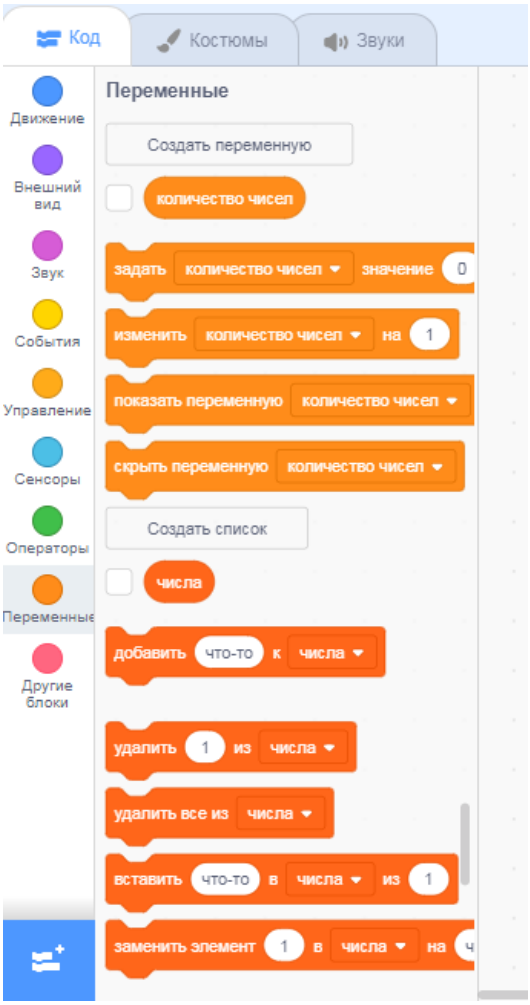
В этом скрипте мы заранее не знаем количество чисел, которые потребуется запомнить персонажу. Поэтому создать нужное количество переменных заранее не получится.

Для таких ситуаций в Scratch используется список. Кнопка для его создания находится в разделе «Переменные».

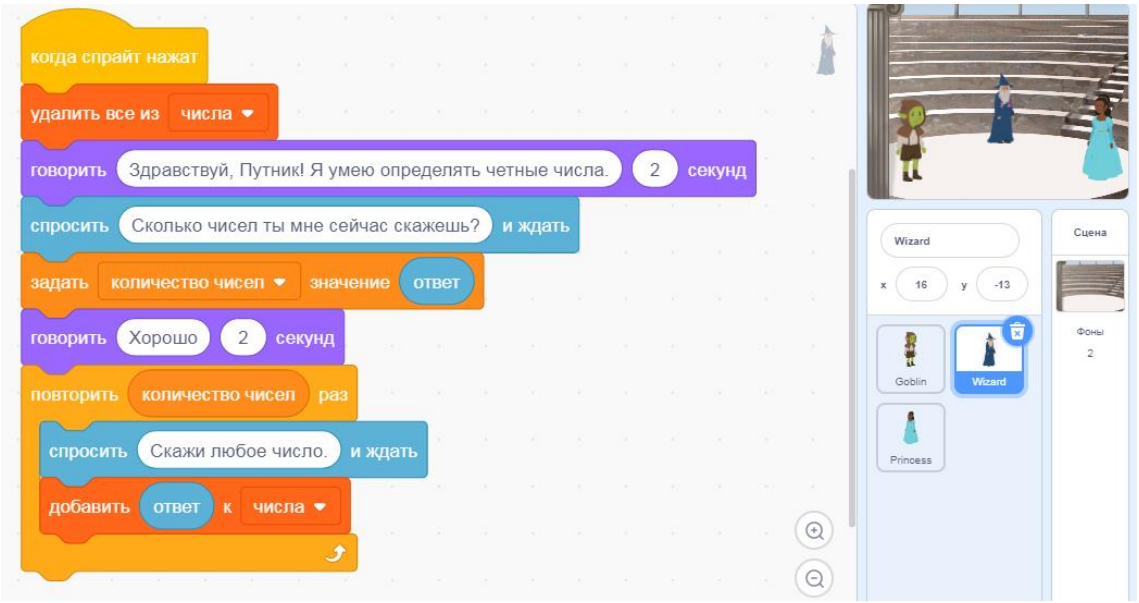


Кроме этого потребуется переменная для хранения числа, которое введет пользователь.

В итоге имеем:

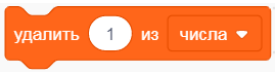


В начале скрипта, как и в предыдущих частях, будет ввод данных со стороны игрока.

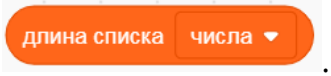


При добавлении чисел в список стоит учитывать, что в нём уже могут присутствовать числа, оставшиеся после предыдущих запусков скрипта. Именно с этой целью в качестве первой команды здесь вставлено «Удалить все из "числа"».

После этого нам необходимо проанализировать все элементы, добавленные в список и каждый из них проверить на четность. В любом списке Scratch, длина которого составляет N элементов, элементы пронумерованы от 1 до N. Используя номер элемента, можно получить из списка его значение (

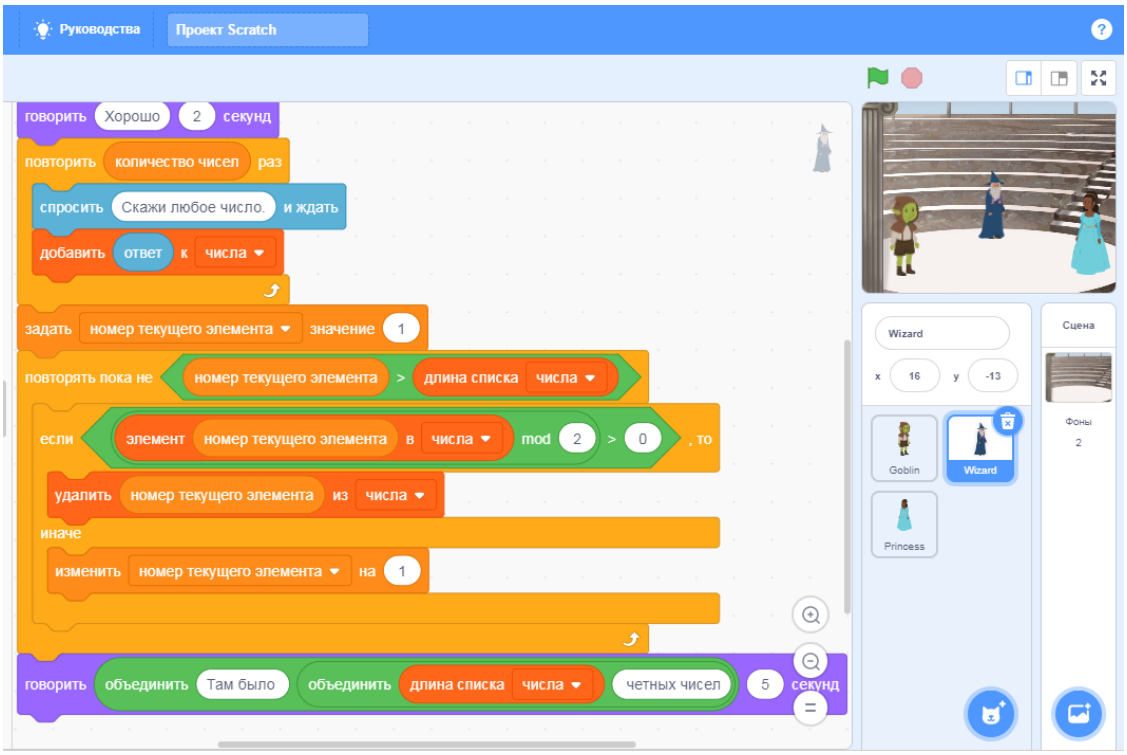
), а также удалить его (

).

В любой момент можно узнать количество элементов в списке с помощью команды 

.

Для выбора каждого элемента из списка будем использовать новую переменную «номер текущего элемента».



Оператор mod позволяет найти остаток от деления числа слева от оператора на число справа от оператора. В скрипте мы удаляем из списка те элементы, которые не являются четными (то есть их остаток от деления на 2 больше нуля).

Самостоятельно доделать проект так, чтобы в случае, если игрок ввел только нечетные числа, персонаж сообщал «Четных чисел не было».

Сохранить проект под именем «ФамилияИО_Математики».

Критерии оценки:
Верное самостоятельное выполнение всех указанных в лабораторной работе шагов – 3 балла
Составление отчёта – 2 балла
Итого за лабораторную работу: 5 баллов.

Лабораторная работа №5 «Реализация линейных, разветвляющихся и циклических алгоритмов в Кумир»

1. Напишите программу, которая вычисляет периметр прямоугольника. Протестируйте ее работу на своих примерах.

2. Напишите программу, которая вычисляет периметр треугольника.

3. Напишите программу, которая вычисляет сумму цифр введенного трехзначного числа.

4. Напишите программу, которая меняет местами первую и третью цифры введенного трехзначного числа.

5. По введенным трем положительным числам определить, можно ли построить треугольник с этими сторонами.

6. По введенным трем положительным числам определить, будут ли эти три числа составлять стороны прямоугольного треугольника.

7. Написать программу для решения линейного уравнения вида **ax=b**. Проверить решение на числах: а) 0 и 0; б) 1 и 0; в) 10 и 15; г) 2 и 10. Объяснить подбор тестовых примеров.

8. При помощи программы с циклом найти сумму квадратов чисел от 1 до 5.

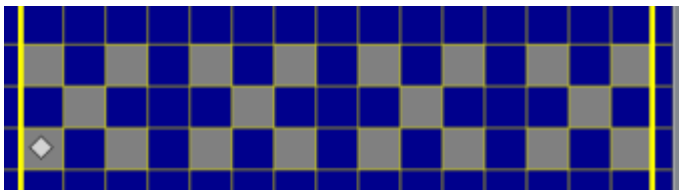
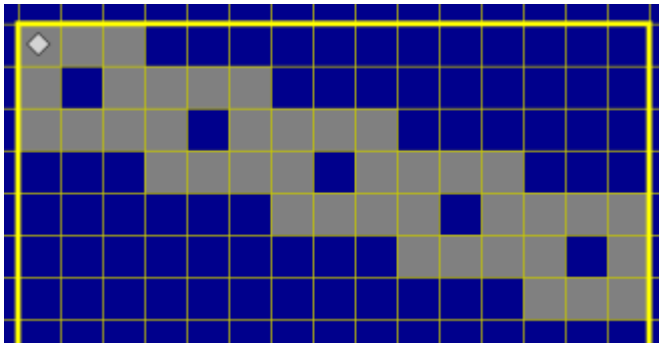
9. При помощи программы с циклом найти сумму дробей: 1+1/2+1/3+1/4+1/5.

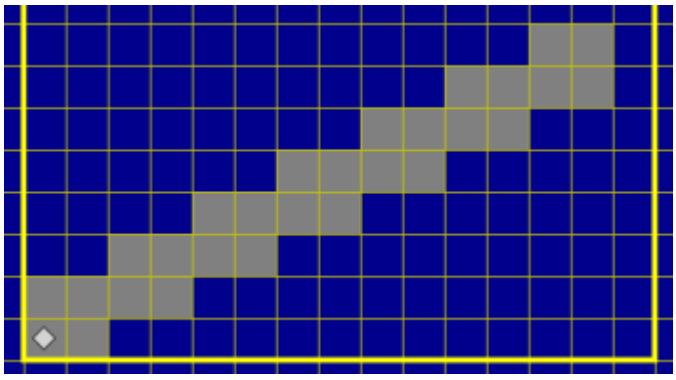
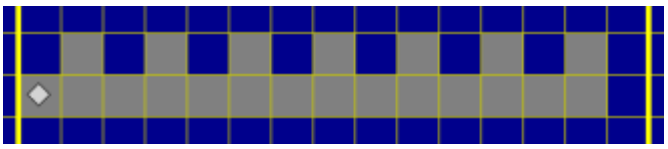
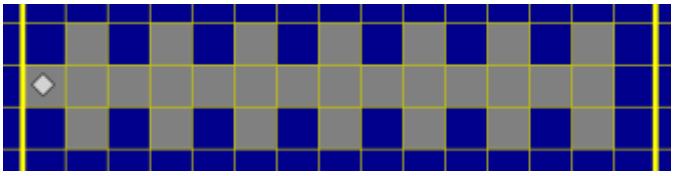
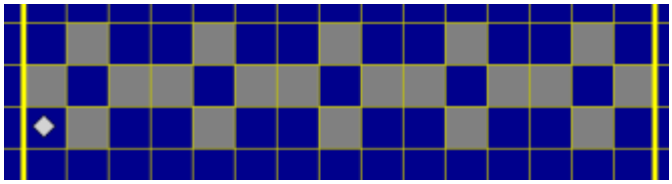
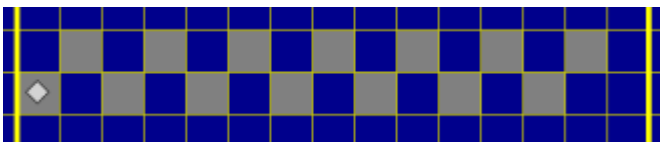
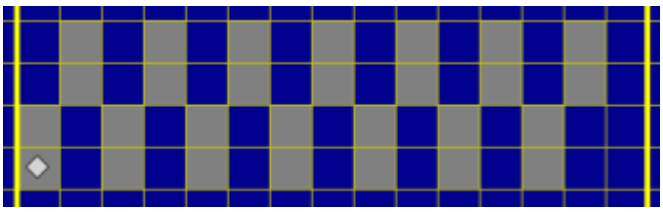
10. Напишите программу, которая определяет количество делителей введенного числа ***a***.

Критерии оценки:
Верное самостоятельное выполнение всех указанных в лабораторной работе шагов – 3 балла
Составление отчёта – 2 балла
Итого за лабораторную работу: 5 баллов.

Лабораторная работа №6 «Исполнитель "Робот" в среде программирования Кумир»

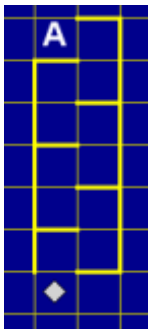
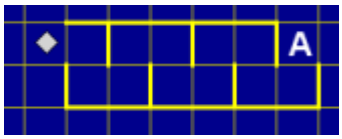
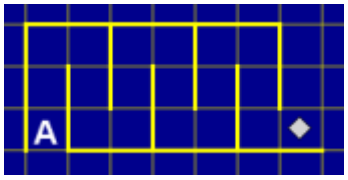
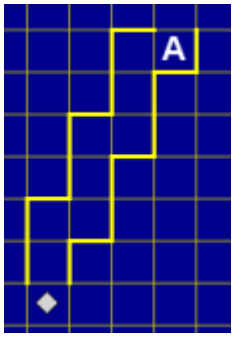
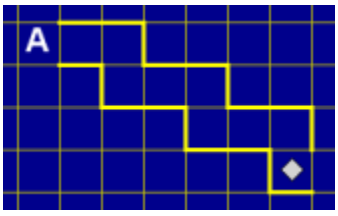
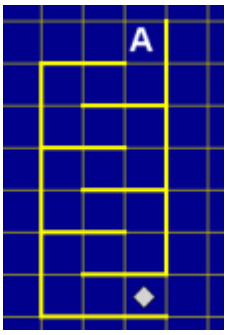
Задание №1: Составьте программу рисования узора. Начальное положение Робота отмечено символом ◊.

1	
2	

3	
4	
5	
6	
7	
8	

В качестве ответа представить файлы с кодом (.kum).

Задание №2: Необходимо провести Робота по лабиринту из начального положения (◊) в точку А.

1		2	
3		4	
5		6	

В качестве ответа представить файлы с кодом (.kum) и стартовой позицией робота (.fil).

Лабораторная работа №7 «Массивы в Кумир»

Массив — это пронумерованная последовательность величин одинакового типа, обозначаемая одним именем. Элементы массива располагаются в последовательных ячейках памяти, обозначаются именем массива и индексом. Каждое из значений, составляющих массив, называется его компонентой (или элементом массива).

С использованием имени массива можно ссылаться как на массив данных в целом, так и на любую из его компонент. Например, A[7] —элемент массива A с индексом 7; D[6] — элемент массива D с индексом 6.

В среде КуМир массив объявляют как `целтаб a[1:100]` (в квадратных скобках указаны индекс первого и последнего элементов массива, а `целтаб` означает, что числа в массиве будут целые).

Пример задачи №1: Дан целочисленный массив из N элементов. Увеличить все элементы в 2 раза.

Решение в среде КуМир:

```
алг пример_10_1
нач
  целтаб a[1:100], цел пер_цикла, кол_элем
  вывод "Введите количество элементов массива"
  ввод кол_элем
  вывод "Введите элементы:"
  нц для пер_цикла от 1 до кол_элем
    ввод a[пер_цикла]
  кц
  вывод "Полученный массив:"
  нц для пер_цикла от 1 до кол_элем
    a[пер_цикла] := a[пер_цикла] * 2
    вывод a[пер_цикла]
    вывод нс
  кц
кон
```

Пример задачи №2: Дан целочисленный массив из N элементов. Найти максимальный элемент массива.

Решение в среде КуМир:

```
алг пример_10_2
нач
  целтаб a[1:100], цел пер_цикла, кол_элем, макс
  вывод "Введите количество элементов массива"
  ввод кол_элем
  вывод "Введите элементы:"
  нц для пер_цикла от 1 до кол_элем
    ввод a[пер_цикла]
  кц
  | Предположим, что первый элемент
  | является максимальным
  макс := a[1]
  | Сравниваем его с остальными элементами
  нц для пер_цикла от 2 до кол_элем
    если a[пер_цикла] > макс то
      макс := a[пер_цикла]
    все
  кц
  вывод "Максимальный элемент: ", макс
кон
```

Решить задачи:

1. Дан целочисленный массив. Возвести в квадрат отрицательные элементы.
2. Дан целочисленный массив. Найти максимум среди четных элементов массива.
3. Создать одномерный целочисленный массив, состоящий из n случайных чисел в диапазоне от -50 до 50. Заменить все элементы массива на противоположные по знаку.

4. Создать одномерный целочисленный массив, состоящий из n случайных чисел в диапазоне от 1 до 20. Найти номера четных элементов, стоящих на нечетных местах.

Критерии оценки:
Верное самостоятельное выполнение всех указанных в лабораторной работе шагов – 3 балла
Составление отчёта – 2 балла
Итого за лабораторную работу: 5 баллов.

Лабораторная работа №8 «Символьные величины в Кумир»

Часто при программировании надо решать не только вычислительные задачи, но и обрабатывать текстовую информацию. Например, требуется найти слово или часть его, сформировать из нескольких отдельных слов предложение или узнать длину текста. Для этого используют символьные и литерные величины.

Символьные величины при написании программ уже встречались ранее для ввода или вывода соответствующих значений. Однако с символьными величинами можно производить много полезных и интересных действий.

Разберем некоторые операции, которые можно производить с символьными данными.

Первая операция похожа на самое простое арифметическое действие – сложение и обозначается знаком « + ». Это операция **сцепление**. В результате сцепления к символьной величине, находящейся слева от знака « + », будет присоединена символьная величина, находящаяся справа от этого знака. Результат такого действия может быть присвоен только величине типа **лит**, которую **нужно** задать в разделе описания переменных.

Например, в результате действия следующего фрагмента программы, в которой все величины имеют тип литерный.

```
x := "пар"
y := "воз"
z := x + "о" + y
вывод z
```

величина z получит значение "паровоз" и соответствующее слово появится на экране.

Разберем пример:

```
алг
нач
  лит x, y, z
  x := "план"
  y := "выполнен"
  z := x + "не" + y
  вывод z
```

Раздел описания переменных,
x,y,z имеют литерный тип данных
Присвоение переменной x значения
Присвоение переменной y значения
Выполнение операции сцепления
Вывод значения литерной переменной z

```
кон                                  | Конец алгоритма
```

В результате работы алгоритма переменная величина z получит значение "план не выполнен" и соответствующее словосочетание появится на экране.

С выведенной фразой что-то не так? Подумайте, почему.

С помощью другой операции, операции сравнения " = ", можно определить тождественность двух символьных величин.

Символьные величины считаются равными при абсолютном совпадении всех знаков символьных величин. Если же они хоть в одном знаке будут отличаться, то значит, равенство не выполняется.

Часто для написания программ с символьными значениями надо определять длину величины, то есть количество символов в ней. Для этой цели в языке КуМир имеется функция **длин**

Разберем пример:

```
алг                                  |
нач                                  |
  лит x, цел n, k                    | Начало алгоритма
  x := "информатика"                 | Описание переменных
                                     | Присваивание значения величине x
  n := длин(x)                       | Определение длины символьной строки и сохранение результата в переменную n
                                     |
                                     |
  k := длин("программа работает")    | Определение длины символьной строки и сохранение результата в переменную k
                                     |
                                     |
  вывод n, нс                        | Вывод значений переменных
  вывод k                            | Конец алгоритма
кон                                  |
```

переменная n получит значение 11, а k будет равна 18 (не забудьте, что пробел тоже полноправный символ).

Обратите внимание, что аргументом функции **длин** может быть только символьная величина, а результатом – число.

Иногда приходится выделять фрагменты из символьной строки и работать с одним или с группой знаков. Для этого существует специальная конструкция **вырезка**.

Общий вид вырезки из символьной переменной А таков:
А [*с какого*: *по какой*], где

с какого – номер символа, с которого начинается вырезка

по какой символ – номер символа, которым заканчивается вырезка

Эта конструкция может использоваться для вырезки части текста символьной строки или ее замены

Предположим A="ИНФОРМАТИКА"

B := A[3:7] - данная запись предполагает, что в символьную переменную B, переписываются из символьной переменной A символы со 3 по 7 включительно. И ее значение будет равно «ФОРМА»

A[8:11] := "ЦИЯ" – данная запись оператора заменяет с 8 по 11 символы в символьной строке A. После этой операции символьная переменная A будет иметь значение равное слову “ИНФОРМАЦИЯ”

Разберем пример.

Сколько раз встречается буква «а» во введенной с клавиатуры строке?

Для того чтобы сосчитать количество букв “а”, надо из строки x поочередно выделять по одной букве в переменную y, сравнивать с образцом и в случае совпадения, увеличивать переменную - счетчик k на единицу (k := k + 1). Чтобы проверить всю строку нужно сначала определить ее длину l. Переменные x и y должны быть описаны как строковые (тип *лит*), а k и l – как целые (*цел*).

```
алг определение длины строки
нач
  цел i, k, l, лит x, y

  вывод "Введите строку символов"
  ввод x

  l := длин(x)



  k := 0

  нц для i от 1 до l

    y := x[ i: i]


    если y="a" то



      k := k + 1
```

Заголовок программы
Начало алгоритма
Раздел описания переменных

Ввод символьной строки

Определение длины символьной строки и сохранение результата в переменную l
Обнуление счетчика k

Начало цикла
Выделение одной буквы из строки на позиции номер i

Определение: является ли выделенная буква буквой «а»

Увеличение значения счетчика на 1 в случае выполненного условия

Конец цикла

```
    все
  кц


  вывод нс, "k=", k

кон
```

Вывод количества букв «а» в строке

Конец алгоритма

Поскольку выделение одного символа встречается довольно часто, в КуМире разрешено вместо A[i:i] писать просто A[i].

Решить задачи:

1. С клавиатуры вводятся 4 слова. Определите, какое из них максимальной длины.
2. Используя операции соединения, сконструируйте сложные слова из введенных корней: пар, ход, воз, лет, вод (Используйте соединительную гласную «о», пример такого соединения приведен в тексте практической работы).
3. Определите, сколько раз встречается во введенной строке слог «ба». (Для проверки работы программы можете использовать такой пример: баран упал на барабан – слог «ба» встречается 3 раза).
4. Замените все пробелы во введенном с клавиатуры предложении на знаки подчеркивания.

Критерии оценки:
Верное самостоятельное выполнение всех указанных в лабораторной работе шагов – 3 балла
Составление отчёта – 2 балла
Итого за лабораторную работу: 5 баллов.

Лабораторная работа №9 «Программирование линейных алгоритмов на языке программирования Python»

Цель работы: выработать практические навыки работы с системой программирования, научиться создавать, вводить в компьютер, выполнять и исправлять простейшие программы на языке Python в режиме диалога, познакомиться с диагностическими сообщениями компилятора об ошибках при выполнении программ, реализующих линейные алгоритмы.

Общие сведения: Линейным называется алгоритм, в котором результат получается путем однократного выполнения заданной последовательности действий при любых значениях исходных данных. Операторы программы выполняются последовательно, один за другим, в соответствии с их расположением в программе.

Задачи:

8. Дана длина ребра куба. Найти объем куба и площадь его боковой поверхности.
9. Три сопротивления R1, R2, R3 соединены параллельно. Найти сопротивление соединения.
10. Определить время падения камня на поверхность земли с высоты h.

11. Известна длина окружности. Найти площадь круга, ограниченного этой окружностью.
12. Треугольник задан координатами своих вершин. Найти:
    a. периметр треугольника;
    b. площадь треугольника.
13. Вычислить высоту треугольника, опущенную на сторону a, по известным значениям длин его сторон a, b, c.
14. Вычислить объем цилиндра с радиусом основания r и высотой h.
15. Определить расстояние, пройденное физическим телом за время t, если тело движется с постоянным ускорением a и имеет в начальный момент времени скорость V0
16. Вычислить площадь треугольника по формуле Герона, если заданы его стороны.
17. Определить координаты вершины параболы $y=ax^2+bx+c$. Коэффициенты a,b,c заданы пользователем.
18. По данным сторонам прямоугольника вычислить его периметр, площадь и длину диагонали.

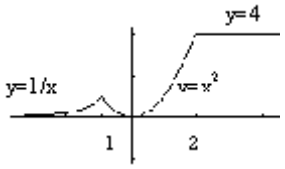
Критерии оценки:
Учащийся решает одну любую задачу из задач 1-4, 5-11. Каждая задача оценивается в 2 балла.
2 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа на языке программирования Python;
1,5 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа, но написанная программа на языке программирования Python содержит синтаксические ошибки;
1 балл выставляется, если составлена верная математическая модель, построена блок-схема, но программа отсутствует;
0,5 балла выставляется за верную математическую модель, если блок-схемы и программа содержат существенные ошибки.
0 баллов выставляется, если работа не была выполнена или составлена только математическая модель.
Дополнительно: 1 балл выставляется за написание отчёта по проделанной работе.
Итого: 5 баллов.

Лабораторная работа №10 «Программирование разветвляющихся алгоритмов на языке программирования Python»

Цель работы: научиться правильно использовать условный оператор if; научиться составлять программы решения задач на разветвляющиеся алгоритмы.

Общие сведения: Алгоритм называется разветвляющимся, если он содержит несколько ветвей, отличающихся друг от друга содержанием вычислений. Выход вычислительного процесса на ту или иную ветвь алгоритма определяется исходными данными задачи. Перед выполнением работы необходимо ознакомится с правилами записи логических выражений, операций сравнения.

Задачи:
1. Даны действительные положительные числа x, y, z. Выяснить, существует ли треугольник с длинами сторон x, y, z.
2. Дано действительное a. Для функции f(a), график которой представлен на рисунке, вычислить f(a).



3. Даны три действительных числа. Выбрать из них те, которые принадлежат интервалу (1, 3).
4. Даны действительные числа x, y. Если x, y отрицательны, то каждое значение заменить его модулем; если отрицательное только одно из них, то оба значения увеличить на 0.5; если оба значения не отрицательны и ни одно из них не принадлежит отрезку [0.5, 2.0], то оба значения уменьшить в 10 раз; в остальных случаях x, y оставить без изменения.
5. Определить, какая из двух фигур (круг или квадрат) имеет большую площадь. Известно, что сторона квадрата равна a, радиус круга r. Вывести на экран название и значение площади большей фигуры.
6. Определить, попадает ли точка M(x,y) в круг радиусом r с центром в точке (x0,y0)
7. Даны действительные числа a, b, c, x, y. Выяснить, пройдет ли кирпич с ребрами a,b,c в прямоугольное отверстие со сторонами x и y. Просовывать кирпич в отверстие разрешается только так, чтобы каждое из его ребер было параллельно или перпендикулярно каждой из сторон отверстия.
8. Сможет ли шар радиуса R пройти в ромбообразное отверстие со стороной P и острым углом Q?
9. Две точки заданы на плоскости своими координатами, которые могут быть как декартовыми, так и полярными. Требуется вычислить расстояние между этими двумя точками.

Критерии оценки:
Учащийся решает одну любую задачу из задач 1-3, 4-9. Каждая задача оценивается в 2 балла.
2 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа на языке программирования Python;
1,5 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа, но написанная программа на языке программирования Python содержит синтаксические ошибки;
1 балл выставляется, если составлена верная математическая модель, построена блок-схема, но программа отсутствует;
0,5 балла выставляется за верную математическую модель, если блок-схемы и программа содержат существенные ошибки.
0 баллов выставляется, если работа не была выполнена или составлена только математическая модель.
Дополнительно: 1 балл выставляется за написание отчёта по проделанной работе.
Итого: 5 баллов.

Лабораторная работа №11 «Программирование циклических алгоритмов»

Цель работы: закрепить практические навыки работы с системой программирования, научиться правильно использовать различные операторы циклов; научиться составлять программы решения задач с использование циклических структур.

Общие сведения: Алгоритм называется циклическим, если он содержит многократное выполнение одних и тех же операторов при различных значениях промежуточных данных. Число повторений этих операторов может быть задано в явной (цикл с известным заранее числом повторений) или неявной (цикл с неизвестным заранее числом повторений) форме.

Задачи:

1. Найти все двузначные числа, сумма цифр которых не меняется при умножении числа на 2,3,4,5,6,7,8,9.
2. Найти все трехзначные числа, сумма цифр которых равна данному целому числу.
3. Найти все трехзначные числа, средняя цифра которых равна сумме первой и второй цифр.
4. Найти все трехзначные числа, которые можно представить разностью между квадратом числа, образованного первыми двумя цифрами и квадратом третьей цифры.
5. Найти все двузначные числа, сумма квадратов цифр которых делится на 17.
6. Найти все трехзначные числа, представимые в виде сумм факториалов своих цифр.
7. Найти двузначное число, обладающее тем свойством, что куб суммы его цифр равен квадрату самого числа.
8. Найти двузначное число, равное утроенному произведению его цифр.
9. Определить, является ли заданное число совершенным, т.е. равным сумме всех своих (положительных) делителей, кроме самого этого числа (например, число 6 совершенно: 6=1+2+3).
10. Дано натуральное k. Напечатать k-ю цифру последовательности 1234567891011121314..., в которой выписаны подряд все натуральные числа.

Критерии оценки:
Учащийся решает одну любую задачу из задач 1-4, 5-8 и 9-10. Каждая задача оценивается в 2 балла.
2 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа на языке программирования Python;
1,5 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа, но написанная программа на языке программирования Python содержит синтаксические ошибки;
1 балл выставляется, если составлена верная математическая модель, построена блок-схема, но программа отсутствует;
0,5 балла выставляется за верную математическую модель, если блок-схемы и программа содержат существенные ошибки.
0 баллов выставляется, если работа не была выполнена или составлена только математическая модель.
Дополнительно: 2 балла выставляется за написание отчёта по проделанной работе.
Итого: 5 баллов.

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ (1 СЕМЕСТР):

1. Мальчик купил в магазине n порций мороженного по цене 1200 руб. и k плиток шоколада по цене 3800 руб. Сколько всего потратил мальчик?

2. Составьте программу нахождения площади равнобедренной трапеции через стороны a,

b, c.

3. Студент начал решать задачи данного урока программирования, когда электронные часы показывали h1 часов и min1 минут, а закончил, когда было h2 часов и min2 минут. Составьте программу, позволяющую определить, сколько времени студент решал эти задачи. (Будем считать, что задачи решались не дольше суток).

4. Дано действительное число a. Не пользуясь никакими другими операциями, кроме умножения, получить: а) a^4 за две операции; б) a^6 за три операции; в) a^7 за четыре операции; г) a^8 за три операции; д) a^9 за четыре операции; е) a^{10} за четыре операции; ж) a^{13} за пять операций; з) a^{15} за пять операций; и) a^{21} за шесть операций; к) a^{28} за шесть операций; л) a^{64} за шесть операций;

5. Найти максимум и минимум двух натуральных чисел не используя ветвления алгоритма.

6. Написать программу, которая печатает True или False в зависимости от того, выполняются или нет заданные условия:

- квадрат заданного трехзначного числа равен кубу суммы цифр этого числа;
- сумма двух первых цифр заданного четырехзначного числа равна сумме двух его последних цифр;
- среди цифр заданного трехзначного числа есть одинаковые;
- среди первых трех цифр из дробной части заданного положительного вещественного числа есть цифра 0.

7. Проверить, можно ли из четырех данных отрезков составить параллелограмм.

8. Найдите все значения натуральных чисел X, Y, Z из интервала от 1 до 50, удовлетворяющих равенству: $X^2 + Y^2 = Z^2$.

9. Вывести 15 первых чисел (начиная с 1) так, чтобы каждое последующее было в 2 раза больше предыдущего.

10. Дано вещественное число A и целое число N (> 0). Вывести $1 + A + A^2 + A^3 + ... + A^N$.

11. Ввести натуральное число с клавиатуры. Найти количество его цифр и их сумму

12. Дано натуральное n. Вычислить: 1 + 1/2 + 1/3 + 1/4 + ... + 1/n.

Критерии оценки:
Всего баллов – 24 (Максимально)
По 2 балла за каждое правильное самостоятельно решённое задание.

ЗАДАНИЯ ДЛЯ ОРГАНИЗАЦИИ ЛАБОРАТОРНЫХ ЗАНЯТИЙ (2 СЕМЕСТР):

Лабораторная работа №1 «Программирование с использованием списков»

Цель работы: научиться правильно использовать списки при написании программ на

языке Python; научиться составлять программы решения задач, требующих хранения последовательно введённых чисел.

Общие сведения: Списки в Python - упорядоченные изменяемые коллекции объектов произвольных типов (почти как массив в других языках, но количество элементов и их типы могут отличаться).

Задачи:

1. Введите с клавиатуры в список пять целочисленных значений, введите их в одну сторону через запятую и найдите среднее арифметическое элементов списка.
2. Дан список из 10 целых чисел. Подсчитайте наибольшее число одинаковых идущих подряд в нем чисел.
3. Подсчитать сколько дней в январе было дней с температурой ниже, выше и равной средней температуре месяца (пользователь вводит по одной температуре, соответствующей определённому дню).
4. По фигурному катанию N судей выставляют оценки. Наибольшая и наименьшая отбрасываются, а по остальным выставляется средний бал. Напишите программу, вычисляющую среднюю оценку и выдайте номер судей, которые поставили наибольшую и наименьшую оценки.
5. Дан список целых чисел. Найти сумму максимума среди чётных и минимума среди нечётных.
6. Дана последовательность n-чисел. Заменить все её члены, больше заданного числа z, этим числом. Посчитать количество замен. Вывести изменённую последовательность.

Критерии оценки:
Учащийся решает одну любую задачу из задач 1-3, а также из задач 4-6. Каждая задача оценивается в 2 балла.
2 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа на языке программирования Python;
1,5 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа, но написанная программа на языке программирования Python содержит синтаксические ошибки;
1 балл выставляется, если составлена верная математическая модель, построена блок-схема, но программа отсутствует;
0,5 балла выставляется за верную математическую модель, если блок-схемы и программа содержат существенные ошибки.
0 баллов выставляется, если работа не была выполнена или составлена только математическая модель.
Дополнительно: 1 балл выставляется за написание отчёта по проделанной работе.
Итого: 5 баллов.

Лабораторная работа №2 «Программирование с использованием словарей»

Цель работы: научиться правильно использовать словари при написании программ на языке Python.

Общие сведения: Словари в Python - неупорядоченные коллекции произвольных объектов с доступом по ключу.

Задача: Если записать числа от 1 до 5 английскими словами (one, two, three, four, five), то

используется 3 + 3 + 5 + 4 + 4 = 19 букв. Сколько букв понадобится для записи всех чисел от 1 до 1000 (one thousand) включительно? Напишите программу, реализующую данные вычисления.

Пояснение:
Для решения удобно использовать словарь, заданный следующим образом:

```
nums={
    1:'one',
    2:'two',
    3:'three',
    4:'four',
    5:'five',
    6:'six',
    7:'seven',
    8:'eight',
    9:'nine',
    10:'ten',
    11:'eleven',
    12:'twelve',
    13:'thirteen',
    15:'fifteen',
    18:'eighteen',
    20:'twenty',
    30:'thirty',
    40:'forty',
    50:'fifty',
    60:'sixty',
    70:'seventy',
    80:'eighty',
    90:'ninety',
    }
```

Критерии оценки:
4 балла выставляется, если составлена верная модель, построена блок-схема и написана программа на языке программирования Python;
3 балла выставляется, если составлена верная модель, построена блок-схема и написана программа, но написанная программа на языке программирования Python содержит синтаксические ошибки;
2 балла выставляется, если составлена верная модель, построена блок-схема, но программа отсутствует;
1 балл выставляется за верную модель, если блок-схемы и программа имеются, но содержат существенные ошибки.
0 баллов выставляется, если работа не была выполнена или составлена только модель.
Дополнительно: 1 балл выставляется за написание отчёта по проделанной работе.
Итого: 5 баллов.

Лабораторная работа №3 «Программирование строк и строковых функций»

Цель работы: научиться правильно использовать строковые функции при написании

программ на языке Python.

Задачи:

1. Во введённой строке заменить пробелы или несколько пробелов символом *.
2. Вывести все числа и цифры из введённой строки.
3. Определить длину самого короткого слова в строке (строка состоит только из латинских символов и пробелов).
4. Определить процент строчных и прописных букв в строке от их общего числа.
5. Во введённой строке найти все слова, которые входят в неё только один раз.

Критерии оценки:
Учащийся решает одну любую задачу из задач 1-4, а также задачу 5. Каждая задача оценивается в 2 балла.
2 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа на языке программирования Python;
1,5 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа, но написанная программа на языке программирования Python содержит синтаксические ошибки;
1 балл выставляется, если составлена верная математическая модель, построена блок-схема, но программа отсутствует;
0,5 балла выставляется за верную математическую модель, если блок-схемы и программа содержат существенные ошибки.
0 баллов выставляется, если работа не была выполнена или составлена только математическая модель.
Дополнительно: 1 балл выставляется за написание отчёта по проделанной работе.
Итого: 5 баллов.

Лабораторная работа №4 «Программирование функций»

Цель работы: изучить понятие подпрограмм, рекурсии; научиться правильно создавать и использовать собственные функции при написании программ на языке Python.

Задачи:

1. Даны действительные числа a, b, c, d, e - стороны выпуклого пятиугольника. Найти площадь пятиугольника. (Определить функцию вычисления площади треугольника по его сторонам.)
2. Даны действительные числа x1, y1, x2, y2, …,x10, y10. Найти периметр десятиугольника, вершины которого имеют соответственно координаты (x1, y1), (x2, y2), …, (x10, y10). (Определить функцию вычисления расстояния между двумя точками, заданными своими координатами.)
3. Даны координаты вершин двух треугольников. Определить, какой из них имеет большую площадь. (Определить функцию вычисления площади треугольника по его вершинам.)
4. Написать программу, которая выводит в порядке возрастания все правильные несократимые дроби, знаменатели которых не превосходят N (2<=N<=500).
5. Написать функцию сложения двух чисел, используя только прибавление единицы.
6. Написать функцию умножения двух чисел, используя только операцию сложения.
7. Найти первые N чисел Фибоначчи двумя способами: с помощью рекурсии и с

помощью итерации. Сравнить эффективность алгоритмов.
8. Рекурсивно подсчитать количество цифр в заданном числе.

Критерии оценки:
Учащийся решает одну любую задачу из задач 1-4, а также из задач 5-8. Каждая задача оценивается в 2 балла.
2 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа на языке программирования Python;
1,5 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа, но написанная программа на языке программирования Python содержит синтаксические ошибки;
1 балл выставляется, если составлена верная математическая модель, построена блок-схема, но программа отсутствует;
0,5 балла выставляется за верную математическую модель, если блок-схемы и программа содержат существенные ошибки.
0 баллов выставляется, если работа не была выполнена или составлена только математическая модель.
Дополнительно: 1 балл выставляется за написание отчёта по проделанной работе.
Итого: 5 баллов.

Лабораторная работа №5 «Работа с файлами»

Цель работы: познакомить с понятием файлового типа данных (бинарные и текстовые); выработать навыки работы с файловым типом данных в языке программирования Python. научиться считывать информацию из файлов, записывать информацию в файл; научится решать задачи с использованием файлов.

Общие сведения: Файл представляет собой структурированный тип данных, содержащий последовательность компонентов одного типа и одной длины. Число элементов в файле (длина файла) не фиксировано. Это является основным отличием файла от списка или массива. Файл можно представить как ленту, у которой есть начало, а конец не фиксирован. Элементы файла записываются на эту ленту последовательно, друг за другом с помощью некоторого устройства - указателя файла. При чтении или записи этот указатель перемещается к следующему элементу и делает его доступным для обработки. В каждый момент доступен для чтения или записи только тот элемент файла, на который установлен указатель.

Задачи:
1. Дан текстовый файл f. Группы символов, разделенные пробелами (одним или несколькими) и не содержащие пробелов внутри себя, будем называть словами. Удалить из файла все однобуквенные слова и лишние пробелы. Результат записать в файл g.
2. Дан файл f, в котором записаны целые числа. Записать в файл g наибольшее значение первых пяти чисел из файла f, затем - следующих пяти и т.д. Если в последней группе окажется менее пяти чисел, то последнее число файла g должно быть равно наибольшему из чисел файла f, образующих последнюю (неполную) группу.
3. Дан файл f, в котором записаны целые числа. Никакое из них не равно нулю. Числа в файле идут в следующем порядке: десять положительных, десять отрицательных,

десять положительных, десять отрицательных и т.д. Переписать числа файла f в файл g так, чтобы в файле g числа шли в следующем порядке: а) пять положительных, пять отрицательных, пять положительных, пять отрицательных и т.д.; б) двадцать положительных, двадцать отрицательных, двадцать положительных, двадцать отрицательных и т.д. (предполагается, что количество чисел в файле f делится на 40).

4. Творческое задание. Написать программы для сжатия и распаковки фалов.
5. Задание повышенной сложности. Дан текстовый фал, содержащий программу на языке Python. Проверить эту программу на соответствие числа открывающихся и закрывающихся скобок (любых).

Критерии оценки:
Учащийся решает одну любую задачу из задач 1-3, которая оценивается в 2 балла.
2 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа на языке программирования Python;
1,5 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа, но написанная программа на языке программирования Python содержит синтаксические ошибки;
1 балл выставляется, если составлена верная математическая модель, построена блок-схема, но программа отсутствует;
0,5 балла выставляется за верную математическую модель, если блок-схемы и программа содержат существенные ошибки.
0 баллов выставляется, если работа не была выполнена или составлена только математическая модель.

Также учащийся решает творческую задачу 4 или задачу повышенной сложности 5. Эта задача оценивается по критериям:
Работоспособность программы – от 0 до 2 баллов
Правильность оформления кода – от 0 до 1 балла
Всего за задачу можно получить до 3 баллов.

Итого за лабораторную работу: 5 баллов.

ЗАДАНИЯ ДЛЯ ОРГАНИЗАЦИИ ЛАБОРАТОРНЫХ ЗАНЯТИЙ (3 СЕМЕСТР):

Лабораторная работа №6 «Основы объектно-ориентированного программирования»

Цель работы: познакомить с понятием "класс"; выработать навыки работы со структурами данных в языке программирования Python. Научится решать задачи на использование структурированных типов данных.

Задачи:

1. Багаж пассажира характеризуется количеством вещей и общим весом вещей. Сведения о багаже каждого пассажира представляют собой класс с двумя полями: одно поле целого типа (количество вещей) и одно - действительное (вес в килограммах).
    a. Найти багаж, средний вес одной вещи в котором отличается не более, чем на 0.3 кг от общего среднего веса одной вещи.
    b. Найти число пассажиров, имеющих более двух вещей и число пассажиров,

количество вещей которых превосходит среднее число вещей.
   c. Определить, имеются ли два пассажира, багажи которых совпадают по числу вещей и различаются по весу не более чем на 0,5 кг.
   d. Выяснить, имеется ли пассажир, багаж которого превышает багаж каждого из остальных пассажиров и по числу вещей, и по весу.
   e. Выяснить, имеется ли пассажир, багаж которого состоит из одной вещи весом менее 30 кг.
2. После поступления в ВУЗ о студентах собрана информация: фамилия, нуждается ли в общежитии, стаж, работал ли учителем, что окончил, какой язык изучал. Составить программу, определяющую:
   a. Сколько человек нуждаются в общежитии;
   b. Списки студентов, проработавших 2 и более лет учителем;
   c. Списки окончивших педучилище;
   d. Списки языковых групп.
3. В библиотеке для каждого заказывающего книгу читателя заполняется карточка: фамилия, дата заказа, дата выдачи книги. Определить:
   a. Самый маленький срок, за который нашли книгу;
   b. Сколько заказов было не удовлетворено;
   c. Кто чаще всего берет книги;
   d. Кому выдали книги 15.09.19;
   e. Сколько человек заказывали книги 25.04.19.
4. В школе было три 9 класса, в августе каждый классный руководитель имел сведения о своих учениках: фамилия, куда поступал, поступил или нет. Определить сколько учеников хотели пойти в 10 класс, кто хотел поступать в училище и техникум, кто поступил в училище или техникум, сколько учеников будет учится в 10 классе, сколько необходимо создать 10 классов и по сколько человек.
5. На олимпиаде по информатике на школьников заполнялись анкеты: фамилия, номер школы, класс, занятое место. Напечатать:
   a. Списки школ, занявших призовые места (их количество вводит пользователь);
   b. Какая из школ заняла больше всех призовых мест;
   c. Списки учеников занявших первое место, указать их класс.
6. В деканате хранится информация о зимней сессии на 1 курсе (фамилия, номер группы, оценка 1 по геометрии, оценка 2 по алгебре, оценка 3 по информатике). Составить программу, печатающую фамилии студентов, имеющих задолженность хотя бы по одному предмету, качество успеваемости (т.е. процент студентов, сдавших экзамены на 4 и 5), название предмета, который был сдан лучше всего, номера групп в порядке убывания средней успеваемости их студентов.
7. В отделе кадров студентов хранится следующая информация о каждом студенте: фамилия, имя, отчество, пол, возраст, курс. Составить программу которая печатает номер курса, на котором наибольший процент мужчин, самые распространенные мужские и женские имена, фамилии в алфавитном порядке и инициалы всех студенток, отчество и возраст которых являются одновременно самыми распространенными.
8. В счете указано: название игрушки, стоимость в копейках, возрастные границы. Получить следующие сведения:
   a. Название игрушек, цена которых не превышает 4 руб. и которые подходят детям до пяти лет;
   b. Цену самого дорогого конструктора;

c. Названия наиболее дорогих игрушек, цена которых отличается от цены самой дорогой игрушки не более, чем на 1 руб.;
d. Название игрушек, которые подходят как детям 4 лет, так и детям 10 лет;
e. Можно ли подобрать игрушку, любую, кроме мяча, подходящую ребенку 3 лет, и дополнительно мяч так, чтобы стоимость игрушек не превосходила 5 руб.

9. Хранятся сведения о лесе: вид дерева, общая численность, численность здоровых деревьев. Составить программу вычисления:
    a. Суммарного числа деревьев на контрольном участке;
    b. Суммарного числа здоровых деревьев;
    c. Относительную численность (%) больных деревьев;
    d. Относительную численность (%) различных видов, в том числе больных (%) для каждого вида.
10. Написать программу, осуществляющую приведение подобных членов многочлена. Для хранения данных создать класс одночлена.

Критерии оценки:
Учащийся решает одну любую задачу из задач 1-7, а также из задач 8-10. Каждая задача оценивается в 2 балла.
2 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа на языке программирования Python;
1,5 балла выставляется, если составлена верная математическая модель, построена блок-схема и написана программа, но написанная программа на языке программирования Python содержит синтаксические ошибки;
1 балл выставляется, если составлена верная математическая модель, построена блок-схема, но программа отсутствует;
0,5 балла выставляется за верную математическую модель, если блок-схемы и программа содержат существенные ошибки.
0 баллов выставляется, если работа не была выполнена или составлена только математическая модель.
Дополнительно: 1 балл выставляется за написание отчёта по проделанной работе.
Итого: 5 баллов.

Лабораторная работа №7 «Разработка графического интерфейса»

Цель работы: рассмотрение способов создания классов, объектов и их использования; обучение созданию графического приложения с использованием стандартного средства tkinter.

Ход работы:

Графический пользовательский интерфейс GUI (Graphical User Interface) – это разновидность пользовательского интерфейса, в котором элементы интерфейса (меню, кнопки, значки, списки и т. п.), представленные пользователю на дисплее, исполнены в виде графических изображений. В отличие от интерфейса командной строки (типа старого DOS системы Windows), в GUI пользователь имеет свободный доступ ко всем элементам интерфейса экрана и может ими управлять с помощью клавиатуры или мыши.

Библиотека tkinter (от англ. Tk interface) – это встроенная графическая библиотека на основе средств Tk, широко распространенная в мире GNU/Linux и других UNIX-подобных

систем, которая портирована (переведена), в частности, на Microsoft Windows и Apple Mac OS). Tk (от англ. Toolkit – «набор инструментов», «инструментарий») – кроссплатформенная библиотека базовых элементов графического интерфейса, распространяемая с открытыми исходными текстами.

Библиотека tkinter входит в стандартный дистрибутив языка Python (до версии Python 3.0 именовалась Tkinter). В tkinter стандартизированные компоненты графического интерфейса, с которыми взаимодействует пользователь, называются виджетами (widget, от англ. window gadget).

Создание окна

Для подключения модуля tkinter используется форма

```
import tkinter from *
```

позволяющая использовать все имена модуля без указания префикса tkinter.

Tk() является базовым классом любого tkinter приложения. При создании объекта этого класса запускается интерпретатор tcl/tk и создается виджетами наивысшего уровня. Обычно им является основное окно приложения. tkinter является событийно-ориентированной библиотекой. В приложениях такого типа имеется главный цикл обработки событий. В tkinter такой цикл запускается методом mainloop(). Для явного выхода из интерпретатора и завершения цикла обработки событий используется метод quit().

Таким образом минимальное приложение на Tkinter будет таким:

```
import tkinter from *
root = Tk()
root.mainloop()
```

Создание окна, рамки и кнопок

В качестве первого примера создания пользовательского приложения с использоваением графической библиотеки tkinter рассмотрим программу gui1.py, в которой создается пользовательское окно, а в нем – рамка с тремя кнопками:

```
from tkinter import *
class My_frame (Frame):
    ''' Рамка с тремя кнопками '''
    def __init__ (self,master):
        super ().__init__(master)
        self.grid()
        self.create_widgets()
    def create_widgets (self):
        ''' Создает три кнопки '''
        self.bt1=Button(self, text='Кнопка №1', fg='blue',
                        font='12' )
        self.bt1.grid()
        self.bt2=Button(self)
        self.bt2.grid()
        self.bt2['text']= 'Кнопка №2'
        self.bt2['fg']= 'blue'
        self.bt2['font']= '12'
        self.bt3=Button(self)
        self.bt3.grid()
        self.bt3.config(text='Кнопка №3', fg='blue',
                        font='12' )
```

```
root=Tk() #Создание базового окна
root.title( 'Рамка с кнопками' )
root.geometry( '230x95+800+200' )
app=My_frame(root)
root.mainloop()
```

В программе gui1.py подключение модуля tkinter осуществляется с использованием формы from tkinter import *. При этом имена всех классов и методов модуля помещаются в пространство имен программы и обращение к ним происходит без добавления префикса tkinter.

На базе класса Frame модуля tkinter создается пользовательский класс My_frame, который описывает рамку, содержащую три кнопки. При создании конструктора этого класса используется встроенная функция super(), которая предназначена для наследования методов родителького класса при их перегрузке. Таким образом, к методам конструктора класса Frame() объекта master добавляются методы: grid() – для размещения рамки, и create_widgets() – для создания кнопок в рамке.

Кнопки в виде объектов bt1, bt2 и bt3 создаются с использованием класса Button модуля tkinter. Хотя кнопки создаются в этом примере одинаковые, способы задания параметров кнопки: text – указывающего название кнопки, fg и font – задающих соответственно цвет (blue) и размеры (12 пикселей) символов, используются разные:

- для кнопки №1 – непосредственно при ее создании;
- для кнопки №2 – задаются с помощью ключей словаря экземпляра класса Button;
- для кнопки №3 – с помощью метода config().

После описания класса в программе gui1.py:

- создается пользовательское окно, обычно именуемое root;
- указываются имя этого окна – 'Рамка с кнопками' и его размеры и расположение на экране:
  - длина – 230 пикселей;
  - ширина – 85 пикселей;
  - смещение по горизонтали – 800 пикселей;
  - смещение по вертикали – 200 пикселей;
- в этом окне создается рамка, свойства которой описаны в классе My_frame();
- запускается цикл обработки событий (хотя в этом примере события не обрабатываются).

Обработка событий

В качестве примера обработки событий, т.е. действий пользователя над виджетами, приведена программа gui2.py, в которой также, как и в программе gui1.py, создается пользовательский класс My_frame2, описывающий рамку, которая, однако, имеет следующие отличия:

- содержит только одну кнопку
- для этой кнопки кроме параметра text, указывающего ее начальное название, задаются также следующие параметры:
  - command – указывает метод – обработчик события для кнопки (self.count);
  - width – задает длину кнопки (30);
  - bg – задает цвет фона (#efffef');
  - fg – задает цвет символов;(blue)
  - font – задает размер символов (6mm);

- содержит метод count(), который по каждому щелчку по кнопке увеличивает значение переменной bt_clicks на единицу и изменяет в названии кнопки число выполненных по ней щелчков.

```
from tkinter import *
class My_frame2 (Frame):
    '''Рамка с кнопкой, подсчитывающей события '''
    def __init__ (self,master):
        super ().__init__(master)
        self.grid()
        self.bt_clicks=0
        self.create_widgets()
    def create_widgets (self):
        ''' Создает кнопку '''
        self.bt1=Button(self,text= 'Число щелчков: 0' ,
                        command=self.count,
                        width= '20' ,
                        height= '2' ,
                        bg= '#efffef' ,
                        fg= 'blue' ,
                        font= 'arial 14' )
        self.bt1.grid()
    def count (self):
        self.bt_clicks+=1
        self.bt1['text'] = 'Число щелчков: ' \
                            + str (self.bt_clicks)

root=Tk()
root.title( 'Обработка событий' )
root.geometry( '240x60+800+300' )
app2=My_frame2(root)
root.mainloop()
```

Работа с текстом

```
from tkinter import *
class My_frame3 (Frame):
    ''' Рамка для работы с текстом '''
    def __init__ (self,master):
        super ().__init__(master)
        self.grid()
        self.create_widgets()
    def create_widgets(self):
        ''' Создает 2 метки, текстовое поле,
            текстовую область и кнопку '''
        #метка-инструкция
        self.ins_lb=Label(self,
                          text='Как прожить до 100 лет?',
                          fg= 'blue' , font= 'arial 14' )
```

```
            self.ins_lb.grid(row=0, column=0,
                             columnspan=2,sticky=W)
            #метка-пароль
            self.ps_lb=Label(self, text= 'Пароль',
                             fg='blue' , font='arial 10' )
            self.ps_lb.grid(row=1, column=0, sticky=W)
            #текстовое поле для ввода пароля
            self.ps_ent=Entry(self, bg='#f0f0ff',
                              font='arial 10' )
            self.ps_ent.grid(row=1, column=1, sticky=E)
            #создание текстовой области, куда будет помещен ответ
            self.txt=Text(self, width=30, height=4, wrap=WORD,
                          bg='#f0f0ff' , font='arial 10' )
            self.txt.grid(row=2, column=0, columnspan=2)
            #кнопка отправки сообщения
            self.bt=Button(self, text='Узнать секрет', fg='blue',
                           bd='5' , font='arial 12',
                           command=self.reveal)
            self.bt.grid(row=3, column=0, sticky=W)
        def reveal(self):
            contents=self.ps_ent.get()
            if (contents== 'secret'):
                message='Чтобы прожить до 100 лет, надо ' \
                'сначала прожить 99 лет и 11 месяцев, а потом ' \
                'вести себя очень-очень осторожно.'
            else:
                message= 'Вы ввели неправильный пароль.'
            self.secret_txt.delete(0.0,END)
            self.secret_txt.insert(0.0,message)

root=Tk()
root.title( 'Работа с текстом' )
root.geometry( ''250x160+850+300' )
app3=My_frame3(root)
root.mainloop()
```

В программе gui3.py описан класс My_frame3, в котором с помощью метода create_widgets() создаются следующие виджеты:

- неизменяемые сообщения (метки) на базе класса Label:
    - 'Чтобы узнать секрет долголетия введите пароль' (объект ins_lb);
    - 'Пароль' (объект ps_lb);
- текстовое поле для ввода пароля на базе класса Entry (объект ps_ent);
- кнопка с сообщением (надписью) 'Узнать секрет' на базе класса Button (объект submit_bt);
- текстовая область для помещения ответа на базе класса Text (объект secret_txt), которая имеет следующие параметры:
    - ширина (width) – 35 символов;
    - высота (height) – 5 рядков;

- перенос по границе слева (WORD))

- Помимо метода create_widgets() в классе My_frame3 также описан обработчик щелчка по кнопке 'Узнать секрет' – метод reveal(), который:
  - получает с помощью метода get() из текстового поля ps_ent введенный пользователем пароль;
  - сравнивает его с паролем 'secret' и формирует сообщение, соответствующее результату сравнения: либо приводится "секрет долголения", либо сообщается о том, что пароль введен неправильно;
  - очищает текстовую область ответа (secret_txt);
  - помещает туда сформированное сообщение.

Отметим использование в этой программе метода grid(), который осуществляет размещение (упаковку) графических элементов на рамке. В отличие от программ gui1.py и gui2.py, где метод grid() использовался без параметров и, следовательно, виджеты располагались по усматрению интерпретатора, в программе gui3.py явным образом указано расположение элементов. При использовании метода grid() поле рамки изображается в виде решетки, состоящей из рядков и колонок. Для указания местоположения виджета использованы следующие параметры:

- row – указывает номер рядка;
- column – указывает номер колонки;
- columnspan – указывает размер ячейки в колонках (по умолчанию – 1);
- sticky – указывает выравнивание данных относительно границ ячейки:
  - N – по верхней границе;
  - E – по правой границе;
  - S – по нижней границе;
  - W – по левой границе.

Использование флажков

```
from tkinter import *
class My_frame4 (Frame):
    ''' Использование флажков '''
    def __init__ (self,master):
        super (My_frame4, self).__init__(master)
        self.grid()
        self.create_widgets()
    def create_widgets (self):
        ''' Создает три флажка '''
        #метка-описание
        Label(self, text='Укажите любимые телепередачи',
              font='arial 13' , fg='blue'
              ).grid(row=0, column=0,sticky=W)
        #метка-инструкция
        Label(self, text='Выберете все, что нравится',
              font='arial 11' , fg='blue'
              ).grid(row=1, column=0,sticky=W)
        #флажок 'Футбол'
        self.likes_football=BooleanVar()
        Checkbutton(self, text='Футбол',
```

```
                    variable=self.likes_football,
                    command=self.update_text
                    ).grid(row=2, column=0,sticky=W)
        #флажок 'История'
        self.likes_history=BooleanVar()
        Checkbutton(self, text='История',
                    variable=self.likes_history,
                    command=self.update_text
                    ).grid(row=3, column=0,sticky=W)
        #флажок 'Природа'
        self.likes_nature=BooleanVar()
        Checkbutton(self, text='Природа',
                    variable=self.likes_nature,
                    command=self.update_text
                    ).grid(row=4, column=0,sticky=W)
        #текстовая область с результатом
        self.results_txt=Text(self, fg='blue',
                              font='arial 11', width=40,
                              height=5, wrap=WORD)
        self.results_txt.grid(row=5, column=0, columnspan=3)
    def update_text (self):
        likes= ''
        if self.likes_football.get():
            likes += 'Вам нравится футбол\n'
        if self.likes_history.get():
            likes += 'Вас привлекает история\n'
        if self.likes_nature.get():
            likes += 'Вам небезразлична природа\n'
        self.results_txt.delete(0.0,END)
        self.results_txt.insert(0.0,likes)

root=Tk()
root.title( 'Использование флажков' )
root.geometry( '270x185+800+200' )
app4=My_frame4(root)
root.mainloop()
```

В отличие от программ gui1.py, gui2.py и gui3.py в программе gui4.py при создании объектов меток (класс Label) используется более короткая форма – без указания имен объектов. Это допускается делать в тех случаях, когда имена этих объектов в дальнейшем не используются в описании класса.

Другой особенностью программы gui4.py является использование флажков – графических элементов класса Checkbutton, при помощи которых осуществляется выбор (от одной до трех) телепередач:

- о футболе;
- об истории;
- о природе.

При этом для каждого флажка:

- указывается переменная класса BooleanVar, соответственно:

  - likes_football;
  - likes_history;
  - likes_nature.
- создается с использованием короткой формы (т.е. без указания имени) объект класса Checkbutton и задаются следующие значения его параметров:
  - для text – наименование флажка (соответственно, "Футбол", "История" и "Природа");
  - для command – наименование функции (update_text), которая обрабатывает события, связанные с изменением состояния флажка (т.е. выбран флажок или нет);
  - для variable, отражающего текущее состояние флажка, – имя переменной, с которой этот параметр связывается(соответственно likes_football, likes_history или likes_nature).

Обработчик update_text() выполняет следующие действия:

- последовательно анализирует состояние переменных likes_football, likes_history и likes_nature и в зависимости от значения переменной (True или False) добавляет или не добавляет к переменной likes значение, соответственно, 'Вам нравится футбол', 'Вас привлекает история' или 'Вам небезразлична природа';
- очищает текстовое поле results_txt;
- выводит в текстовое поле results_txt значение переменной likes.

При выполнении программы gui4.pyw на экране появляется окно со следующими виджетами:

- меткой-описанием "Укажите любимые телепередачи";
- меткой-инструкцией "Выберете все, что нравится";
- флажками:
  - "Футбол";
  - "История";
  - "Природа";
- текстовой областью для вывода результатов выбора.

Использование переключателей

```
from tkinter import *
class My_frame5 (Frame):
    ''' Использование переключателей '''
    def __init__ (self,master):
        super (My_frame5, self).__init__(master)
        self.grid()
        self.create_widgets()
    def create_widgets (self):
        ''' Создает переключатель '''
        #метка-инструкция
        Label(self, text='Укажите любимую телепередачу',
              fg= 'blue' , font= 'arial 13'
              ).grid(row=0, column=0,sticky=W)
        self.favorite=StringVar()
        self.favorite.set( None )
        #положение переключателя 'Футбол'
        Radiobutton(self, text='Футбол',
```

```
                    variable=self.favorite,
                    value= 'о футболе',
                    command=self.update_text
                    ).grid(row=1, column=0,sticky=W)
        #положение переключателя 'История'
        Radiobutton(self, text= 'История',
                    variable=self.favorite,
                    value= 'об истории',
                    command=self.update_text
                    ).grid(row=2, column=0,sticky=W)
        #положение переключателя 'Природа'
        Radiobutton(self, text= 'Природа',
                    variable=self.favorite,
                    value= 'о природе',
                    command=self.update_text
                    ).grid(row=3, column=0,sticky=W)
        #текстовая область для результата
        self.results_txt=Text(self, fg='blue',
                              font='arial 10', width=40,
                              height=5, wrap=WORD)
        self.results_txt.grid(row=4, column=0, columnspan=3)
    def update_text (self):
        message= "Ваша любимая телепередача - "
        message += self.favorite.get()
        self.results_txt.delete(0.0, END)
        self.results_txt.insert(0.0, message)

root=Tk()
root.title( 'Использование переключателя' )
root.geometry( '310x120+800+200' )
app=My_frame5(root)
root.mainloop()
```

В отличие от программы gui4.py, в которой может осуществляться выбор нескольких телепередач, в программе gui5.py можно выбрать только одну телепередачу. Поэтому вместо трех флажков класса Checkbutton используется один переключатель с тремя положениями, наименования которых соответствуют наименованиям флажков.

При этом выполняются следующие действия:

- указывается атрибут self.favorite класса StringVar для хранения наименования выбранного положения переключателя (при его создании указывается значение None);
- для каждого положения переключателя создается объект класса Radiobutton и указываются значения его параметров:
    - для text – наименование положения переключателя (соответственно, "Футбол", "История" и "Природа");
    - для command – наименование функции (update_text), которая обрабатывает события, связанные с изменением состояния переключателя;
    - для variable, отражающего текущее состояние переключателя, –

self.favorite (имя атрибута, с которыи этот параметр связывается;
- для value – значение выбранного положения переключателя, соответственно, 'о футболе', 'об истории' или 'о природе'.

Обработчик update_text() выполняет следующие действия:
- задает начальное значение переменной message ("Ваша любимая телепередача - ");
- используя атрибут self.favorite и метод get() получает значение выбранного положения переключателя и добавляет его к значению переменной message;
- очищает текстовое поле results_txt;
- выводит в текстовое поле results_txt значение переменной message.

При выполнении программы IE: gui5.pyw на экране появляется окно со следующими виджетами:
- меткой-инструкцией "Укажите любимую телепередачу";
- переключателем, имеющем три состояния:
  - "Футбол";
  - "История";
  - "Природа";
- текстовой областью для вывода результата выбора.

Индивидуальные задания

Разработать программу на языке Python, в которой на базе модуля tkinter разработан графический интерфейс, имеющий несколько окон:
- окно с кнопкой (объект Button), надпись на которой меняется по каждому щелчку и содержит следующую информацию – номер щелчка и число, которое определяется согласно колонке "Число" табл. №1:
  - 1 – очередное простое число (начиная с 1);
  - 2 – очередное число Фибоначчи (начиная с 1);
  - 3 – очередное простое число вида 2 n - 1 (начиная с n = 1);
- окно, содержащее:
  - текстовое поле (объект Text), куда выводится начало некоторого небольшого сообщения (рассказа и т.п.);
  - поле ввода пароля (объект Entry);
  - кнопку для продолжения вывода сообщения (объект Button);
  - второе текстовое поле (объект Text), куда выводится неожиданное (смешное) завершение сообщения (при правильном вводе пароля) или сообщение о неправильном вводе пароле;
  - метки (объект Label) – при необходимости;
- окно, содержащее:
  - пять флажков (объекты Checkbutton), задающие согласно колонке "Флажки" табл. №1:
    - 1 – числа 1, 2, 4, 8 и 16;
    - 2 – 5 букв кириллицы (на усмотрение студента);
    - 2 – 5 английских букв (на усмотрение студента);
  - текстовое поле, куда помещается результат:
    - для чисел – сумма заданных чисел;
    - для букв – слово, состоящее из выбранных букв;
- окно, содержащее:
  - переключатели (объекты Radiobutton), задающие согласно колонке

"Переключатели" табл. №1:
- 1 – изменяемые типы;
- 2 – неизменяемые типы;
- 3 – контейнеры;
- 4 – последовательности;

- текстовое поле, куда помещается результат – пример литерала для выбранного класса.

Таблица 1 – Перечень индивидуальных заданий

Номер п/п	Число	Флажки	Переключатели
1	1	3	1
2	2	1	2
3	3	2	3
4	1	3	4
5	2	1	1
6	3	2	2
7	1	3	3
8	2	1	4
9	3	2	1
10	1	3	2
11	2	1	3
12	3	2	4
13	1	3	1
14	2	1	2
15	3	2	3
16	1	3	4
17	2	1	1
18	3	2	2
19	1	3	3
20	2	1	4

Критерии оценки:
Верное самостоятельное выполнение всех указанных в лабораторной работе шагов – 1 балл
Выполнение индивидуального задания – 3 балла
Составление отчёта – 1 балл
Итого за лабораторную работу: 5 баллов.

Лабораторная работа №8 «Многострочное текстовое поле»

1. Настроить для программы собственную цветовую схему (отредактировать параметры `bg` и `fg` у виджетов).

2. Доработать программу так, чтобы в ней присутствовала кнопка «Backup» для быстрого сохранения файла в папке с программой. Имя файла задаётся текущей датой и временем.

3. Добавить на форму надпись (Label) с информацией об авторе программы (ФИО, курс, группа).

```
from tkinter import *
import datetime

root = Tk()
root.title("Блокнот")

fr1 = Frame(root)
fr1.pack(fill=BOTH, expand=True)

# Виджет - полоса прокрутки
scrollbar = Scrollbar(fr1)
scrollbar.pack(side=RIGHT, fill=Y)

# Виджет - многострочное текстовое поле
# wrap=WORD – автоматический перенос слов на следующую строчку
# wrap=CHAR – посимвольный перенос на следующую строчку
# wrap=NONE – без переносов (в этом случае желательно
# также подключение полосы прокрутки через xscrollcommand)
# yscrollcommand - установка вертикальной прокрутки
text = Text(fr1, wrap=WORD,
            yscrollcommand=scrollbar.set)
text.pack(side=LEFT, fill=BOTH, expand=True)

scrollbar.config(command=text.yview)

fr2 = Frame(root)
fr2.pack(expand=True)

def create_new():
    # 1.0 – координаты начала фрагмента текста, на который бу-
дет
    # действовать команда. Формат: y.x, где
    # y (в команде: 1) – номер строки, первая строка имеет но-
мер 1
    # x (в команде: 0) – номер столбца, первый имеет номер 0
    text.delete(1.0, END)

def open_file():
    # Вызов диалога открытия файла
    # и получение информации об адресе открытого
    # файла в переменную file_name
    file_name = filedialog.askopenfilename()
    # Если выбран файл
    if file_name != '':
        # Открываем файл с помощью стандартной команды
        f = open(file_name)
```

```
            # Чтение текста из файл
            s = f.read()
            # Очистка предыдущего текста
            text.delete(1.0, END)
            # Вставка считанного текста
            text.insert(1.0, s)
            # Закрытие файла
            f.close()

    def save_file():
        file_name = filedialog.asksaveasfilename(
            defaultextension='.txt')
        if file_name != '':
            f = open(file_name, 'w')
            s = text.get(1.0, END)
            f.write(s)
            f.close()

    def insert_date():
        paste = str(datetime.datetime.now().year)
        # Получение номера месяца
        # с добавлением нуля слева до двух символов
        paste += str(datetime.datetime.now().month).rjust(2, "0")
        paste += str(datetime.datetime.now().day).rjust(2, "0")
        paste += " "
        paste += str(datetime.datetime.now().hour).rjust(2, "0")
        paste += str(datetime.datetime.now().minute).rjust(2, "0")
        paste += str(datetime.datetime.now().second).rjust(2, "0")
        # Вставка сформированной строки paste
        # в позицию курсора - text.index("insert")
        text.insert(text.index("insert"), paste)

    btnNew = Button(fr2, text='Новый', command=create_new)
    btnNew.pack(side=LEFT, padx=10, pady=5)

    btnOpen = Button(fr2, text='Открыть', command=open_file)
    btnOpen.pack(side=LEFT, padx=10, pady=5)

    btnSave = Button(fr2, text='Сохранить', command=save_file)
    btnSave.pack(side=LEFT, padx=10, pady=5)

    btnDate = Button(fr2, text='Вставить дату', com-
mand=insert_date)
    btnDate.pack(side=LEFT, padx=10, pady=5)

    root.mainloop()
```

Критерии оценки:
Верное самостоятельное выполнение всех указанных в лабораторной работе шагов – 3 балла

Составление отчёта – 2 балла
Итого за лабораторную работу: 5 баллов.

Лабораторная работа №9 «Работа с графикой в tkinter»

1. Ознакомиться с программой для отрисовки примитивов.

```
    # Подключение модуля для создания окна
    from tkinter import *

    root = Tk()
    root.geometry('500x600+100+100')

    # Создание холста для рисования
    canv = Canvas(root, bg='grey')
    # размещение виджета с помощью «упаковщика»
    # Виджет будет занимать весь размер формы
    # и расширяться при масштабировании
    canv.pack(fill=BOTH, expand=True)

    def keypressed(event):
        # Получение доступа к переменным y, x и rect
        # из основного скрипта
        global y
        global x
        global rect
        # удаление всех нарисованных фигур
        canv.delete("all")
        # проверка кода нажатой клавиши
        # стрелки: 38 – вверх, 40 – вниз, 37 – влево, 39 - вправо
        if event.keycode == 38:
            y -= 5
        elif event.keycode == 37:
            x -= 5
        elif event.keycode == 40:
            y += 5
        elif event.keycode == 39:
            x += 5
        # Создание прямоугольника
        rect = canv.create_rectangle(x - r, y - r,
                                     x + r, y + r, fill='#FF0000')
        # Создание овала
        #  canv.create_oval(x  -  r,  y  -  r,  x  +  r,  y  +  r,
fill='blue')
        # Создание крестика
        # canv.create_line(x - r, y - r, x + r,
        #                  y + r, fill='red', width=3)
        # canv.create_line(x + r, y - r, x - r,
        #                  y + r, fill='red', width=3)
```

```
    x = 200
    y = 300
    r = 100

    rect = canv.create_rectangle(x - r, y - r,
                                 x + r, y + r, fill='#FF0000')
    # canv.create_oval(x - r, y - r, x + r, y + r, fill='blue')
    # canv.create_line(x - r, y - r, x + r, y + r, fill='red',
width=3)
    # canv.create_line(x + r, y - r, x - r, y + r, fill='red',
width=3)

    # Обработка нажатия на кнопку на клавиатуре подпрограммой key-
pressed
    root.bind('<Key>', keypressed)

    root.mainloop()
```

2. Доработать программу так, чтобы в ней перемещался смайлик.



Критерии оценки:
Верное самостоятельное выполнение всех указанных в лабораторной работе шагов – 3 балла
Составление отчёта – 2 балла
Итого за лабораторную работу: 5 баллов.

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ 2-3 СЕМЕСТР:

1. Дана дата (число, месяц, год и день недели). Определить правильность заданной даты, вычислить последний день данного месяца, сколько дней прошло от 1-го января 1-го года нашей эры, определить день недели, приходящийся на заданную дату, если 1-е января 1-го года нашей эры было понедельником.

2. Написать программу вычисления длин сторон, углов, площади, радиусов вписанной и описанной окружностей для треугольника, заданного координатами вершин на плоскости в декартовых или полярных координатах.

3. Дана непустая последовательность слов из строчных русских букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке:

- все гласные буквы, которые входят в каждое слово;
- все согласные буквы, которые не входят ни в одно слово;
- все звонкие согласные буквы, которые входят хотя бы в одно слово;
- все глухие согласные буквы, которые не входят хотя бы в одно слово;
- все согласные буквы, которые входят только в одно слово;
- все глухие согласные буквы, которые не входят только в одно слово;
- все звонкие согласные буквы, которые входят более чем в одно слово;
- все гласные буквы, которые не входят более чем в одно слово;

4. Подсчитать общее количество цифр и знаков '+', '-', и '*', входящих в строку s.

5. Написать программу раздачи карт при игре в дурака, количество игроков задается с клавиатуры.

6. Составить функцию создания прямоугольника из текста в консоли, окаймленного рамкой из псевдографических символов. В параметры функции ввести символ заполнения рамки и символ заполнения содержимого прямоугольника, а также его ширину и высоту.

Пример для параметров 'x', '$', 8, 5:

```
xxxxxxxx
x$$$$$$x
x$$$$$$x
x$$$$$$x
xxxxxxxx
```

7. Составить программу случайного выбора дежурного из списка, в котором 4 мальчика и 4 девочки, причем для девочек вероятность выбора в два раза ниже, чем для мальчиков.

8. Необходимо составить программу проверки равенства треугольников по их элементам. Пользователь поочерёдно вводит данные об углах и сторонах треугольников, а программа сохраняет введённые данные или выдаёт сообщение о том, что такой же треугольник уже был введён. Предусмотреть защиту от неверного ввода данных.

9. Аналогично задаче 8, сравнивать трапеции.

10. Составить программу перемещения по экрану вводимой предварительно с клавиатуры текстовой строки. Перемещение осуществляется по восьми направлениям, направление должно меняться через определённое количество шагов. Причем при смене направления движения должен меняться цвет надписи.

11. Создать окно с тестом по информатике. Тест должен включать как минимум один вопрос с выбором варианта ответа и вопрос с числовым ответом. Реализовать проверку правильности ответов при нажатии на кнопку «ОК».

12. Создать оконное приложение с кроссвордом по информатике. Кроссворд должен включать как минимум четыре слова. Ввод слова осуществляется по одной букве в каждое текстовое поле. Проверка правильности и корректности (ровно одна буква в поле) – при нажатии на кнопку «ОК». Тексты вопросов размещать в надписях (Label).

Критерии оценки:
Всего баллов – 24 (Максимально)
По 2 балла за каждое правильное самостоятельно выполненное задание.

ПРИМЕРНАЯ ТЕМАТИКА ДОКЛАДОВ (1 СЕМЕСТР):

1. История формирования понятия "алгоритм".
2. История языков программирования.
3. Известнейшие алгоритмы в истории математики. Проблема существования алгоритмов в математике.
4. Современные ученые и языки программирования.
5. История языка программирования Python.
6. Сферы применения языка программирования Python.
7. Логические элементы и их история.
8. Система программирования Eclipse. История развития, основные характеристики и особенности программирования на языке Python в данной среде.
9. Система программирования PyScripter. История развития, основные характеристики.
10. Система программирования PyCharm. История развития, основные характеристики.
11. Принципы Джона Фон Неймана.
12. Машина Тьюринга.

Критерии оценки

Оценки в баллах	Критерии
2	Самостоятельность в подготовке материала
3	Аргументированность и полнота материала
2	Использование современной и актуальной литературы
3	Качество презентации
10	**Итого**

ПРИМЕРНАЯ ТЕМАТИКА ДОКЛАДОВ (2-3 СЕМЕСТР):

1. Функции в языке программирования Python.
2. Основные определения и теоремы теории рекурсивных функций.
3. Определение объекта и класса. Причины возникновения. Способы использования.
4. Классы в Python.
5. Жизненный цикл программных систем.
6. Методы управления проектами при разработке программных систем.
7. Модульный подход к программированию.
8. Структурный подход к программированию.
9. Экстремальное программирование.
10. Параллельное программирование.

11. Создание интерфейса приложения.
12. Разработка функциональной схемы работы приложения.

Критерии оценки

Оценки в баллах	Критерии
2	Самостоятельность в подготовке материала
3	Аргументированность и полнота материала
2	Использование современной и актуальной литературы
3	Качество презентации
10	**Итого**