**МУНИЦИПАЛЬНОЕ КАЗЁННОЕ**
**ОБЩЕОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**
**« АЛАДАШСКАЯ СРЕДНЯЯ ОБЩЕОБРАЗОВАТЕЛЬНАЯ ШКОЛА»**


# Рабочая программа внеурочной деятельности
# Основы программирования на языке Python

**(8-10 кл.)**

**Составитель**: учитель информатики
Магамедова М.А.

Аладаш 2021 г.

## Пояснительная записка

Элективный курс по информатике "Основы программирования на Python" представляет собой вводный курс по программированию, дающий представление о базовых понятиях структурного программирования (данных, переменных, ветвлениях, циклах и функциях). Python – это язык, обладающий рядом преимуществ перед другими языками для начинающих изучать программирование (ясность кода, быстрота реализации).

Курс рассчитан примерно на 35 часов.

Изменение взглядов на предмет информатики как науки, её место в системе научного знания требует существенных изменений в содержании образования по информатике. В связи с этим особую актуальность приобретают раскрытие личностных резервов учащихся и создание соответствующей среды.

Никакая система задач, какой бы хорошей она ни была, никакие тренинги памяти, внимания и т. п. не дают того эффекта, который возникает в случае, если учащиеся осознают необходимость решения тех или иных задач, если у них появляется острая необходимость к преодолению интеллектуальных трудностей, связанных с познанием, если они видят смысл в сотрудничестве с одноклассниками и учителем.

Содержание обучения, представленное в программе «Основы программирования на языке Python», позволяет вести обучение школьников в режиме актуального познания. Практическая направленность курса на создание внешних образовательных продуктов — блок-схем, алгоритмов, программ — способствует выявлению фактов, которые невозможно объяснить на основе имеющихся у школьников знаний. Возникающие при этом познавательные переживания обусловливают сознательное отношение к изучению основных теоретических положений информатики.

Проявления трудолюбия, целеустремленности, возникающие при воплощении замыслов учащихся в рамках курса «Основы программирования на языке Python», стимулируют развитие индивидуально-личностных качеств школьников.

Активизация познавательного процесса позволяет учащимся более полно выражать свой творческий потенциал и реализовывать собственные идеи в изучаемой области знаний, создаёт предпосылки по применению освоенных навыков программирования в других учебных курсах, а также способствует возникновению дальней мотивации, направленной на освоение профессий, связанных с разработкой программного обеспечения.
Курс служит средством внутрипрофильной специализации в области новых информационных технологий, что способствует созданию дополнительных условий для проявления индивидуальных образовательных интересов учащихся.

### Концепция курса

Ключевой особенностью курса является его направленность на формирование у учащихся навыков поиска собственного решения поставленной задачи, составления алгоритма решения и реализации алгоритма с помощью средств программирования.

В рамках предлагаемого курса «Основы программирования на языке Python» изучение основ программирования на языке Python — это не столько средство подготовки к будущей профессиональной деятельности, сколько формирование новых общеинтеллектуальных умений и навыков: разделение задачи на этапы решения, построение алгоритма и др. Исключительно велика роль программирования для формирования мышления школьников, приёмов умственных действий, умения строить модели, самостоятельного нахождения и составления алгоритмов решения задач, умения чётко и лаконично

реализовывать этапы решения задач. Использование этих возможностей для формирования общеинтеллектуальных и общеучебных умений школьников активизирует процесс индивидуально-личностного становления учащихся. Общепедагогическая направленность занятий – гармонизация индивидуальных и социальных аспектов обучения по отношению к информационным технологиям. Умение составлять алгоритмы решения и навыки программирования являются элементами информационной компетенции — одной из ключевых компетенций современной школы. Умение находить решение, составлять алгоритм решения и реализовать его с помощью языков программирования — необходимое условие подготовки современных школьников. Особая роль отводится широко представленной в курсе системе рефлексивных заданий. Освоение рефлексии направлено на осознание учащимися того важного обстоятельства, что наряду с разрабатываемыми ими продуктами в виде программ на компьютере рождается основополагающий образовательный продукт: освоенный инструментарий. Именно этот образовательный продукт станет базой для творческого самовыражения учащихся в форме различных программ.

**Цели изучения курса:**

- понять значение алгоритмизации как метода познания окружающего мира, принципы структурной алгоритмизации;
- овладеть базовыми понятиями теории алгоритмов;
- научиться разрабатывать эффективные алгоритмы и реализовывать их в виде программы, написанной на языке программирования Python.

**Задачи курса:**

· познакомить с понятиями алгоритма, вычислимой функции, языка программирования;

· научить составлять и читать блок-схемы;

· сформировать навыки выполнения технологической цепочки разработки программ средствами языка программирования Python;

· изучить основные конструкции языка программирования Python, позволяющие работать с простыми и составными типами данных (строками, списками, кортежами, словарями, множествами);

· научить применять функции при написании программ на языке программирования Python;

· научить отлаживать и тестировать программы, делать выводы о работе этих программ.

**Методы обучения**

Отбор методов обучения обусловлен необходимостью формировать информационную и коммуникативную компетентности учащихся, реализовывать личностно-ориентированное обучение, направлять их на самостоятельное решение разнообразных проблем, развивать исследовательские и творческие способности. Решение данных задач кроется в организации деятельностного подхода к обучению, в проблемном изложении материала учителем, в переходе от репродуктивного вида работ к самостоятельным, поисково-исследовательским видам деятельности. Поэтому основная методическая установка в данном курсе — обучение учащихся навыкам самостоятельной творческой деятельности.

**Формы организации учебных занятий**

Организация учебного процесса предусматривает дистанционной формы деятельности, когда учащийся вне уроков самостоятельно выполняет на компьютере практические задания.

**Планируемые результаты курса**

В рамках курса «Основы программирования на языке Python» учащиеся овладевают следующими знаниями, умениями и способами деятельности:

- умеют составлять алгоритмы для решения задач;
- умеют реализовывать алгоритмы на компьютере в виде программ, написанных на языке Python;
- владеют основными навыками программирования на языке Python;
- умеют отлаживать и тестировать программы, написанные на языке Python.

**Способы оценивания уровня достижений учащихся**

Предметом диагностики и контроля в курсе «Основы программирования на языке Python» являются внешние образовательные продукты учащихся (созданные блок-схемы, программы), а также их внутренние личностные качества (освоенные способы деятельности, знания, умения), которые относятся к целям и задачам курса.

Качество внешней образовательной продукции желательно оценивать по следующим параметрам:

- алгоритм должен быть оптимальным по скорости выполнения и максимально простым в реализации на языке программирования;
- программа должна выполнять поставленные задачи;
- по степени «читаемости кода» (должны быть соблюдены отступы, обязательное наличие комментариев к коду программы и т. д.).

Проверка достигаемых учащимися результатов производится в следующих формах:

- текущий рефлексивный самоанализ, контроль и самооценка учащимися выполняемых заданий;
- текущая диагностика и оценка учителем деятельности школьников;
- итоговая оценка деятельности и образовательной продукции ученика в соответствии с его индивидуальной образовательной программой освоения курса; Итоговый контроль проводится в конце всего курса. и организуется тестированием.

**Программное обеспечение:**

1. Операционная система: Windows XP (или выше).
2. Среда разработки: Python 3.3 (или выше),

## Учебно-тематический план

| Наименование тем | Практ. занятия |
|---|---|
| **Тема 1. Знакомство с языком Python** | **2** |
| Урок 1. Общие сведения о языке | 1 |
| Практическая работа 1.1. Установка программы Python | |
| Урок 2. Режимы работы | 1 |
| Практическая работа 1.2. Режимы работы с Python | |
| Тест № 1. Знакомство с языком Python | |

| | |
|---|---|
| **Тема 2. Переменные и выражения** | **8** |
| Урок 3. Переменные | 1 |
| Практическая работа 2.1. Работа со справочной системой | 1 |
| Практическая работа 2.2. Переменные | 1 |
| Урок 4. Выражения | 1 |
| Практическая работа 2.3. Выражения | 1 |
| Урок 5. Ввод и вывод | 1 |
| Урок 6. Задачи на элементарные действия с числами | 1 |
| Практическая работа 2.5. Задачи на элементарные действия с числами | 1 |
| Тест № 2. Выражения и операции. | |
| **Тема 3. Условные предложения** | **10** |
| Урок 7. Логические выражения и операторы | 1 |
| Практическая работа 3.1. Логические выражения | 1 |
| Урок 8. Условный оператор | 1 |
| Практическая работа 3.2. "Условный оператор" | 1 |
| Урок 9. Множественное ветвление | 1 |
| Практическая работа 3.3. Множественное ветвление | 1 |
| Урок 10. Реализация ветвления в языке Python | 1 |
| Практическая работа 3.4. "Условные операторы" | 1 |
| Самостоятельная работа № 1 по теме "Условные операторы". | 1 |
| Урок 11. Зачетная работа № 1. "Составление программ с ветвлением". | 1 |
| Тест № 3. "Условные операторы". | |
| **Тема 4. Циклы** | **11** |
| Урок 12. Оператор цикла с условием | 1 |
| Практическая работа 4.1. "Числа Фибоначчи" | 1 |
| Урок 13. Оператор цикла for | 1 |
| Практическая работа 4.2. Решение задачи с циклом for. | 1 |
| Урок 14. Вложенные циклы | 1 |
| Практическая работа 4.3. Реализация циклических алгоритмов | 1 |
| Урок 15. Случайные числа | 1 |
| Практическая работа 4.4. Случайные числа | 1 |
| Урок 16. Примеры решения задач с циклом | 1 |
| Практическая работа 4.5. Решение задач с циклом. | 1 |

| | |
|---|---|
| Самостоятельная работа № 2 "Составление программ с циклом" | 1 |
| Тест № 4. Циклы | |
| Урок 17. Творческая работа № 1. "Циклы" | 1 |
| **Тема 5. Функции** | **9** |
| Урок 18. Создание функций | 1 |
| Практическая работа 5.1. Создание функций | 1 |
| Урок 19. Локальные переменные | 1 |
| Практическая работа 5.2. Локальные переменные | 1 |
| Урок 20. Примеры решения задач с использованием функций | 1 |
| Практическая работа 5.3. Решение задач с использованием функций | 1 |
| Самостоятельная работа № 3 по теме "Функции" | 1 |
| Урок 21. Рекурсивные функции | 1 |
| Практическая работа 5.4. Рекурсивные функции | 1 |
| Тест № 5. Функции | |
| **Тема 6. Строки - последовательности символов** | **5** |
| Урок 22. Строки | 1 |
| Урок 23. Срезы строк | 1 |
| Практическая работа 6.1. Строки | 1 |
| Урок 24. Примеры решения задач со строками | 1 |
| Практическая работа 6.2. Решение задач со строками. | 1 |
| **Тема 7. Сложные типы данных** | **12** |
| Урок 25. Списки | 1 |
| Урок 26. Срезы списков | 1 |
| Практическая работа 7.1. Списки | 1 |
| Урок 27. Списки:  примеры решения задач | 1 |
| Практическая работа 7.2. Решение задач со списками | 1 |
| Урок 28. Матрицы | 2 |
| Тест № 7. Списки | |
| Урок 29. Кортежи | 1 |
| Урок 30. Введение в словари | 2 |
| Урок 31. Множества в языке Python | 2 |
| **Тема 8.  Стиль программирования и отладка программ** | **7** |
| Урок 32. Стиль программирования | 1 |
| Урок 33. Отладка программ | 2 |

| | |
|---|---|
| Урок 34. Зачет по курсу «Программирование на языке Python» | 3 |
| Урок 35. Что дальше? | 1 |
| **ВСЕГО** | **64 ч** |

## Содержание тем учебного курса

### Тема 1. Знакомство с языком Python

Общие сведения о языке Python. Установка Python на компьютер. Режимы работы Python. Что такое программа. Первая программа. Структура программы на языке Python. Комментарии.

Практическая работа 1.1. Установка программы Python

Практическая работа 1.2. Режимы работы с Python

Тест № 1. Знакомство с языком Python

Учащиеся должны знать / понимать:

· понятие программы;

· структура программы на Python;

· режимы работы с Python.

Учащиеся должны уметь:

· выполнить установку программы;

· выполнить простейшую программу в интерактивной среде; · написать комментарии в программе.

### Тема 2. Переменные и выражения

Типы данных. Преобразование типов. Переменные. Оператор присваивания. Имена переменных и ключевые слова.

Выражения. Операции. Порядок выполнения операций. Математические функции. Композиция.

Ввод и вывод. Ввод данных с клавиатуры. Вывод данных на экран. Пример скрипта, использующего ввод и вывод данных. Задачи на элементарные действия с числами. Решение задач на элементарные действия с числами.

Практическая работа 2.1. Работа со справочной системой

Практическая работа 2.2. Переменные

Практическая работа 2.3. Выражения

Практическая работа 2.5. Задачи на элементарные действия с числами

Тест № 2. Выражения и операции.

Учащиеся должны знать / понимать:

- общую структуру программы;
- типы данных;
- целые, вещественные типы данных и операции над ними;
- оператор присваивания;
- операторы ввода-вывода.

Учащиеся должны уметь:

- пользоваться интерфейсом среды программирования Python;

- использовать команды редактора; -
- организовывать ввод и вывод данных; - записывать арифметические выражения.

**Тема 3. Условные предложения**

Логический тип данных. Логические выражения и операторы. Сложные условные выражения (логические операции and, or, not). Условный оператор. Альтернативное выполнение. Примеры решения задач с условным оператором. Множественное ветвление. Реализация ветвления в языке Python.

Практическая работа 3.1. Логические выражения

Практическая работа 3.2. "Условный оператор"

Практическая работа 3.3. Множественное ветвление

Практическая работа 3.4. "Условные операторы"

Самостоятельная работа № 1. Решение задач по теме "Условные операторы".

Зачетная работа № 1. "Составление программ с ветвлением".

Тест № 3. "Условные операторы".

Учащиеся должны знать / понимать:

- назначение условного оператора;
- способ записи условного оператора;
- логический тип данных;
- логические операторы or, and, not; Учащиеся должны уметь:
- использовать условный оператор;
- создавать сложные условия с помощью логических операторов.

**Тема 4. Циклы**

Понятие цикла. Тело цикла. Условия выполнения тела цикла. Оператор цикла с условием. Оператор цикла while. Бесконечные циклы. Альтернативная ветка цикла while. Обновление переменной. Краткая форма записи обновления. Примеры использования циклов.

Оператор цикла с параметром for. Операторы управления циклом. Пример задачи с использованием цикла for. Вложенные циклы. Циклы в циклах. Случайные числа. Функция randrange. Функция random. Примеры решения задач с циклом.

Практическая работа 4.1. "Числа Фибоначчи"

Практическая работа 4.2. Решение задачи с циклом for.

Практическая работа 4.3. Реализация циклических алгоритмов

Практическая работа 4.4. Случайные числа

Практическая работа 4.5. Решение задач с циклом.

Самостоятельная работа № 2 "Составление программ с циклом"

Тест № 4. Циклы

Творческая работа № 1. "Циклы"

Учащиеся должны знать / понимать: - циклы с условием и их виды;

- правила записи циклов условием;
- назначение и особенности использования цикла с параметром;
- формат записи цикла с параметром;

- примеры использования циклов различных типов.

Учащиеся должны уметь:

- определять вид цикла, наиболее удобный для решения поставленной задачи;
- использовать цикл с условием;
- определять целесообразность применения и использовать цикл с параметром для решения поставленной задачи;

**Тема 5. Функции**

Создание функций. Параметры и аргументы. Локальные и глобальные переменные. Поток выполнения. Функции, возвращающие результат. Анонимные функции, инструкция lambda. Примеры решения задач с использованием функций. Рекурсивные функции. Вычисление факториала. Числа Фибоначчи.

Практическая работа 5.1. Создание функций

Практическая работа 5.2. Локальные переменные

Практическая работа 5.3. Решение задач с использованием функций

Практическая работа 5.4. Рекурсивные функции

Самостоятельная работа № 3 по теме "Функции"

Тест № 5. Функции

Учащиеся должны знать / понимать: - понятие функции;

- способы описания функции;
- принципы структурного программирования;
- понятие локальных переменных подпрограмм;
- понятие формальных и фактических параметров подпрограмм;
- способ передачи параметров.

Учащиеся должны уметь:

- создавать и использовать функции;
- использовать механизм параметров для передачи значений.

**Тема 6. Строки - последовательности символов**

Составной тип данных - строка. Доступ по индексу. Длина строки и отрицательные индексы. Преобразование типов. Применение цикла для обхода строки. Срезы строк. Строки нельзя изменить. Сравнение строк. Оператор in. Модуль string. Операторы для всех типов последовательностей (строки, списки, кортежи). Примеры решения задач со строками.

Практическая работа 6.1. Строки

Практическая работа 6.2. Решение задач со строками.

Учащиеся должны знать / понимать:

- назначение строкового типа данных;
- операторы для работы со строками;
- процедуры и функции для работы со строками;
- операции со строками.

Учащиеся должны уметь:

- описывать строки;
- соединять строки;
- находить длину строки;
- вырезать часть строки;
- находить подстроку в строке;
- находить количество слов в строке.

**Тема 7. Сложные типы данных**

Списки. Тип список (list). Индексы. Обход списка. Проверка вхождения в список. Добавление в список. Суммирование или изменение списка. Операторы для списков. Срезы списков. Удаление списка. Клонирование списков. Списочные параметры. Функция range. Списки: примеры решения задач.

Матрицы. Вложенные списки. Матрицы. Строки и списки. Генераторы списков в Python.

Кортежи. Присваивание кортежей. Кортежи как возвращаемые значения

Введение в словари. Тип словарь (dict). Словарные операции. Словарные методы. Множества в языке Python. Множества. Множественный тип данных. Описание множеств. Операции, допустимые над множествами: объединение, пересечение, разность, включение. Оператор определения принадлежности элемента множеству.

Практическая работа 7.1. Списки.

Практическая работа 7.2. Решение задач со списками.

Тест № 7. Списки

Учащиеся должны знать / понимать:

- сложные типы данных;
- способ описания списка;
- способ доступа к элементам списка;
- способ описания кортежа;
- способ описания словаря;
- операции, выполняемые со списками, кортежами и словарями;
- понятие множества;
- способы описания множества;
- операторы работы с множествами.

Учащиеся должны уметь:

- описывать списки;
- вводить элементы списка;
- выводить элементы списка;
- выполнять поиск элемента в списке, поиск минимума и максимума, нахождение суммы элементов списка;
- использовать вложенные списки;
- приводить примеры использования вложенных списков (матриц);
- описывать множества;
- определять принадлежность элемента множеству;

- вводить элементы множества;
- выводить элементы множества.

**Тема 8. Стиль программирования и отладка программ** Стиль программирования. Отладка программ.

Зачет по курсу «Программирование на языке Python»

Учащиеся должны знать / понимать:

- что такое стиль программирования;
- правила именования объектов;
- основные рекомендации при написании программ.

Учащиеся должны уметь:

- определять вид ошибок и находить ошибки в программе.
- выполнять тестирование и отладку программ.

**Литература и источники**


1. Домашняя страница Python www.python.org . Справочные материалы, официальная документация.
2. Сайт проекта Интуит: Национальный открытый университет, курс «Введение в программирование на Python», http://www.intuit.ru/studies/courses/12179/1172/info.
3. Сайт проекта Интуит: Национальный открытый университет. Курс «Язык программирования *Python*» http://www.intuit.ru/studies/courses/49/49/info.
4. Сайт проекта Open Book Project openbookproject.net содержит серию практических примеров на Python Криса Мейерса.
5. *Python. Подробный справочник* Дэвида М. Бизли — книга со справочной информацией о языке Python и модулях стандартной библиотеки.
6. *Python. Справочник* Марка Лутца. Справочник по наиболее часто использующимся функциям и модулям.