Муниципальное бюджетное общеобразовательное учреждение
«Каменская средняя общеобразовательная школа»

Центр образования цифрового и гуманитарного
профилей «Точка роста»

РАССМОТРЕНО на педагогическом Совете протокол № 3 от 28.10.2019 г.	СОГЛАСОВАНО руководитель Центра _____ Я.С. Бандурист 30.10.2019 г.	УТВЕРЖДЕНО Директор МБОУ «Каменская СОШ» _____ Ю.А. Ребзон Приказ №167/2 от 31.10.2019 г.

ДОПОЛНИТЕЛЬНАЯ ОБЩЕОБРАЗОВАТЕЛЬНАЯ (ОБЩЕРАЗВИВАЮЩАЯ) ПРОГРАММА

«PYTHON. Программирование для любознательных»

НАПРАВЛЕННОСТЬ: ТЕХНИЧЕСКАЯ
Уровень: базовый
Возраст обучающихся: 12 - 17 лет
Срок реализации: 1 год

Автор-составитель:
Ребзон Сергей Леонидович
педагог дополнительного
образования

2019

СОДЕРЖАНИЕ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Дополнительная общеобразовательная (общеразвивающая) программа **«PYTHON. Программирование для любознательных»** разработана на основе следующих нормативно-правовых документов:

- федеральный закон от 29.12.2012 г. № 273-ФЗ «Об образовании в Российской Федерации»;
- приказ Министерства просвещения Российской Федерации от 09.11.23019 г. № 196 «Об утверждении Порядка организации и осуществления образовательной деятельности по дополнительным общеобразовательным программам»;
- профессиональный стандарт педагога дополнительного образования детей и взрослых, утвержденный приказом министерства труда и социальной защиты Российской Федерации от 5 мая 2018 г. № 298н;
- концепция развития дополнительного образования детей, утвержденная Распоряжением правительства Российской Федерации от 04.09.2014 г. « 1726-р;
- указ Президента Российской Федерации от 7 мая 2018 года № 204 «О национальных целях и стратегических задачах развития Российской Федерации на период до 2024 года;
- национальный проект «Образование», утвержденный президиумом Совета при Президенте РФ по стратегическому развитию и национальным проектам, протокол от 24.12.2018 г. № 16.

Дополнительная общеобразовательная (общеразвивающая) программа **«PYTHON. Программирование для любознательных»** имеет **техническую** направленность.

В современную жизнь человека все шире внедряются компьютеры и информационные технологии. Поэтому все большее значение приобретает компьютерная грамотность.

Курс изучения компьютерной грамотности состоит из двух разделов: пользовательского курса и программирования. Раздел «Программирование» в школьном курсе представлен языком программирования Pascal, а многим учащимся хочется познакомиться с другими языками программирования, самим попробовать разработать программы, которые можно использовать на уроках и во внеурочной деятельности. Данная Программа позволяет реализовать эти желания, так как уделяется большое внимание практической работе учащихся на компьютере, самостоятельной разработке ими программ для решения практических задач.

Данная Программа разработана на основании программы «Программирование на языке Python» (разработчик Киселева Н.Н., педагог дополнительного образования ГБОУ Школа № 1375).

Новизна программы

Новизна Программы заключается в том, что Python дает более широкие возможности в области программирования, чем Pascal, который входит в школьный курс информатики. На языке Python можно легко и быстро создавать простые компьютерные игры, трёхмерные модели и программировать роботов. Этот язык быстрее и легче усваивается, чем Pascal. Многие мировые компании такие, как Intel, Cisco, Hewlett-Packard, используют этот язык при реализации своих проектов. Крупнейшие интернет-ресурсы такие, как Google, YouTube, также разработаны с помощью языка программирования Python.

Актуальность программы

С развитием современных информационных технологий сегодня любой учащийся под руководством опытного педагога может с лёгкостью научиться программировать.

Компьютеры и компьютерные системы – неотъемлемая часть жизни нашего общества. Научившись программировать, мы можем быть не только пользователями информационных технологий, но и активными их создателями.

Языки программирования можно сравнить с иностранными языками, овладеть ими может каждый. Учиться программировать очень интересно. Результат программирования очень часто виден сразу. Кроме того, создание компьютерных игр и обучающих программ способствует развитию логики и креативного мышления. Ещё одной значимой стороной обучения программированию является спрос на рынке труда на специалистов данного направления деятельности.

Педагогическая целесообразность Программы

Педагогическая целесообразность программы заключается в привлечении учащихся к занятиям техническим творчеством, что способствует развитию логического мышления, творческих способностей и навыков решения задач программирования. Программирование мотивирует к занятиям в различных научных областях (физики, информатики, алгебры, геометрии и др.), развивает воображение и способствует ранней профориентации подростков. Для достижения поставленных задач занятия проводятся в формате «от простого к сложному». Учащиеся вспоминают свои знания по основам алгоритмизации и программирования и на их основе, углубляя их, учатся составлять простые и сложные программы.

Цель Программы: обучение учащихся программированию посредством языка Python, развитие инженерного мышления, воспитание конкурентно способной личности.

Задачи Программы

Обучающие:

- обучить языку программирования Python и созданию программ на его основе;
- научить создавать прикладное программное обеспечение;
- расширять кругозор обучающихся в области программирования;
- научить дизайнерскому оформлению созданного ПО.

Развивающие:

- развивать память и внимание, познавательную и творческую активность;
- развивать творческие способности, эстетическое и эргономическое восприятие объектов труда;
- развивать логическое мышление.

Воспитательные:

- прививать интерес к активному творческому самовыражению, культуре труда;
- воспитывать упорство в достижении желаемого результата;
- воспитывать эстетический вкус;
- воспитывать чувство взаимопомощи, доверия, коллективизма.

Отличительные особенности данной программы

Основное количество часов отводится практическому написанию программ. Каждый обучающийся реализует индивидуальный проект в результате освоения программы. Продукт, полученный в результате освоения программы, имеет прикладной характер и может быть использован по необходимости.

Возрастная категория обучающихся по программе от 12 до17 лет.

Срок реализации программы составляет 1 год. Общее количество часов в год составляет 144 часа.

Формы и режим занятий - групповые – для всей группы при изучении общих практических и теоретических вопросов. Наполняемость группы до 15 человек.

В ходе реализации программы применяется дифференцированный, индивидуальный подход к каждому обучающемуся.

Занятия проводятся 2 раза в неделю по 2 часа (2 раза по 45 минут с перерывом 15

минут).

Ожидаемые результаты и способы определения их результативности

По итогам реализации программы, учащиеся

будут знать:
- принципы программирования на языке Python;
- основы дизайнерского оформления созданных программ.

будут уметь:
- производить чтение и запись программ на языке Python;
- запускать и отлаживать программу.

Формы определения результативности обучения

Результаты освоения программы отслеживаются по итогам опросов, выполнения практических заданий.

Формы аттестации и оценочные материалы

В процессе реализации программы предусмотрены следующие виды контроля:

- ***входной контроль проводится*** с целью определения уровня знаний учащихся (Приложение № 1);

- ***промежуточный контроль*** проводится регулярно на занятиях с целью определения степени усвоения материала в форме опроса, решения задач и практических заданий;

- ***итоговый контроль*** – защита проекта.

Формы подведения итогов реализации программы

Результаты обучения по программе выявляются по итогам проведения олимпиад, соревнований по программированию, защиты проекта.

Содержание программы.
Учебный (тематический) план

№ п/п	Наименование раздела, темы	Количество часов			Формы аттестации/ контроля
		Всего	Теория	Практика	
1.	**Введение в Python**	**14**	**6**	**8**	
1.1.	Язык Python. Среда программирования. Особенности ввода-вывода	4	2	2	Практическое задание
1.2.	Типы данных, операции. Оператор присваивания	4	2	2	Выполнение упражнений
1.3.	Числа. Стандартные операции	6	2	4	Практическое задание
2.	**Алгоритмические инструкции**	**22**	**6**	**16**	
2.1.	Условный оператор	2	2	-	Опрос
2.2.	Цикл while. Теория чисел	10	2	8	Практическое задание

2.3.	Цикл for	10	2	8	Практическое задание
3.	**Строки**	**20**	**8**	**12**	
3.1.	Литералы строк	4	2	2	Практическое задание
3.2.	Срезы строк	4	2	2	Практическое задание
3.3.	Методы строк	12	4	8	Практическое задание
4.	**Функции**	**20**	**8**	**12**	
4.1.	Парадигма и преимущества структурного программирования	2	2	-	Опрос
4.2.	Граф и стек вызовов функций. Области видимости переменных	4	2	2	Практическое задание
4.3.	Прямая рекурсия	10	2	8	Практическое задание
4.4.	Косвенная рекурсия	4	2	2	Решение задач повышенной трудности
5.	**Списки и кортежи**	**20**	**8**	**12**	
5.1.	Списки и кортежи в Python. Сходства и различия	2	2	-	Опрос
5.2.	Операции со списками	10	2	8	Практическое задание
5.3.	Срезы списков	4	2	2	Практическое задание
5.4.	Матрицы. Операции над матрицами	4	2	2	Решение задач повышенной трудности
6.	**Словари и множества**	**8**	**4**	**4**	
6.1.	Словари	4	2	2	Практическое задание
6.2.	Множества	4	2	2	Практическое задание

7.	**Объектно - ориентированное программирование**	**38**	**4**	**34**	
7.1.	Классы в Python	2	2	-	Опрос
7.2.	Разработка собственного класса	6	2	4	Практическое задание
7.3.	Разработка и программирование собственного проекта	30	-	30	
8.	**Заключительное занятие**	**2**	**-**	**2**	
8.1.	Подведение итогов. Индивидуальный проект.	2		2	Защита проекта
	Всего	**144**	**44**	**100**	

Содержание учебного (тематического) плана

1. Введение в Python

Основные понятия: трансляция, интерпретация, компиляция, синтаксис, семантика, прагматика, переменная, динамическая типизация, служебные слова, идентификаторы, простые типы данных, приоритеты операций, литералы чисел, операция присваивания, PEP 8.

1.1. Язык Python. Среда программирования. Особенности ввода-вывода

Теория. Язык программирования Python. Достоинства и недостатки. Области применения. Интерактивный режим работы программы.

Практика. Установка языка программирования Python 3.5 и среды программирования WinglDE 100. Регистрация на Интернет-ресурсах.

1.2. Типы данных, операции. Оператор присваивания

Теория. Ввод и вывод числовой информации.

Практика. Тренировочное задание на ввод и вывод числовой информации.

1.3. Числа. Стандартные операции

Теория. Стандартные операции с целыми и действительными числами. Стиль программирования Python (PEP 8).

Практика. Решение простых задач в интерактивном режиме.

2. Алгоритмические инструкции

Основные понятия: логический тип данных, логические операции (and, or, not, ^{A}), условный оператор, условное и альтернативное исполнение алгоритма, операторы сравнения, вложенность операторов, оператор цикла, переменная-флаг, генерация псевдослучайной последовательности, инструкции break, continue и pass.

2.1. Условный оператор

Теория. Условная и циклическая инструкции. Каскадность и вложенность алгоритмических инструкций.

2.2. Цикл while

Теория. Теория чисел. Фильтрация потока чисел. НОД и НОК. Проверка числа на простоту. Алгоритм Евклида. Нахождение максимума и минимума.

Практика. Решение задач на анализ чисел потока и целочисленной арифметики.

2.3. Цикл for

Теория. Теория чисел. Фильтрация потока чисел. НОД и НОК. Проверка числа на простоту. Алгоритм Евклида. Нахождение максимума и минимума.

Практика. Решение задач на анализ чисел потока и целочисленной арифметики повышенной трудности.

3. Строки

Основные понятия: символ, строка, литерал, таблицы кодов ASCII, UTF-8, отладка кода, неизменяемый объект, формат вывода строки, экранированные escape-последовательности, положительная и отрицательная нумерация символов в строке, срез, конкатенация, длина строки.

3.1. Литералы строк

Теория. Понятие «литералы строк».

Практика. Ввод-вывод строки. Решение задач на ввод строки, поиск подстроки.

3.2. Срезы строк

Теория. Форматирование строки.

Практика. Преобразование строки. Решение упражнений.

3.3. Методы строк

Теория. Методы работы со строкой.

Практика. Применение методов строки. Решение задач.

4. Функции

Основные понятия: подпрограмма, функция, процедура, рекурсия, глубина рекурсии, объявление, определение и вызов функции, возврат значений, глобальные и локальные переменные, передача параметров, работа с памятью, граф вызовов, стек вызовов, полиморфизм функций, утиная типизация, lambda-функции.

4.1. Парадигма и преимущества структурного программирования

Теория. Обзор парадигм программирования. Особенности применения языков программирования.

4.2. Граф и стек вызовов функций. Области видимости переменных

Теория. Применение стек и граф вызовов функций.

Практика. Выполнение тренировочных упражнений.

4.3. Прямая рекурсия

Теория. Понятие прямой рекурсии и ее применение.

Практика. Выполнение тренировочных упражнений.

4.4. Косвенная рекурсия

Теория. Понятие косвенной рекурсии и ее применение.

Практика. Выполнение тренировочных упражнений.

Практические занятия к темам 4.2., 4.3., 4.4.: Нахождение суммы чисел. Числа Фибоначчи. Вычисление степени. Ханойские башни.

Использование библиотеки математических функций. Решение задач повышенной трудности.

5. Списки и кортежи

Основные понятия: список, кортеж, элемент списка и кортежа, индекс, срез списка, матрица, многомерный список, сортировка, сложность алгоритма, устойчивость сортировки,

квадратичная, быстрая, синхронная, поразрядная сортировки списка, случайное перемешивание.

5.1. Списки и кортежи в Python. Сходства и различия

Теория. Представление списка и кортежа в памяти компьютера, сходства и различия.

5.2. Операции со списками

Теория. Способы заполнения списка (с клавиатуры, из файла, случайным образом, по формуле).

Практика. Решение задач на ввод-вывод элементов одномерного и многомерного списка.

5.3. Срезы списков

Теория. Методы работы со списком и кортежем. Методы сортировки списка.

Практика. Решение задач на ввод-вывод элементов одномерного и многомерного списка и кортежа, преобразование, поиск, замену, подсчет.

5.4. Матрицы. Операции над матрицами

Теория. Вычисление сложности алгоритма. Многомерные списки.

Практика. Решение задач повышенной трудности.

6. Словари и множества

Основные понятия: *словарь, множество, ключ, кодирование.*

6.1. Словари

Теория. Понятие словаря. Способы создания. Словарь, преимущества и недостатки, методы работы со словарем. Словари со смешанными значениями. Кодирование и декодирование текста.

Практика. Решение задач на заполнение, преобразование, поиск, замену, подсчет, вывод элементов словаря.

6.2. Множества

Теория. Понятие множества. Создание множеств. Множество, преимущества и недостатки, методы работы с множеством.

Практика. Решение задач повышенной трудности.

7. Объектно-ориентированное программирование (ООП)

Основные понятия: ООП, класс, метод INIT, экземпляр, наследование, полиморфизм, исключения, виджет, интерфейс, событие, техническое задание, проект, проектная деятельность, виды проектов.

7.1. Классы в Python

Теория. Понятия «класс», «метод INIT», «экземпляр», «наследование», «полиморфизм», «исключения», «виджет», «интерфейс», «интерфейс», «событие».

7.2. Разработка собственного класса

Теория. Принципы разработки собственного класса. Обработка и генерация исключений. Виджет, методы виджета. Графическая библиотека tkinter, класс Tk. Системные методы. События.

Практика. Создание собственного класса.

7.3. Разработка и программирование собственного проекта

Практика. Выбор вида и темы проекта. Составление технического задания. Программирование. Разработка технической документации и презентации проекта.

8. Подведение итогов

8.1. Защита индивидуального проекта

Организационно-педагогические условия реализации программы

Занятия проводятся в следующих формах: лекции, семинары, практические задания.
Дидактический материал, необходимый для проведения занятий:

- краткие конспекты материалов для лекций;
- распечатки заданий для практикумов;
- презентационные материалы для объяснения;
- карточки с индивидуальными заданиями.

Техническое оснащение занятий:

- компьютер для демонстрации презентаций;
- проектор;
- рабочие компьютеры учащихся для работы с доступом в Интернет;
- принтер для распечатки заданий.

Список литературы

1. Доусен М. Программируем на Python / М. Доусен - СПб.: Питер, 2016. - 416с.
2. Лутц М. Изучаем Python, 4 издание / М. Лутц - СПб.: Символ-Плюс, 2011. - 1280 с.
3. Любанович Б. Простой Python. Современный стиль программирования / Б. Любанович. - СПб.: Питер, 2016. - 480с.
4. Прохоренок Н.А., Дронов В.А. Python 3 и PyQt 5. Разработка приложений / Н.А. Прохоренок, В.А. Дронов - СПб.: «БХВ- Петербург», 2016. - 832с.
5. Саммерфильд М. Python на практике / М. Саммерфильд, пер. А.А. Слинкин – М.: ДМК-Пресс, 2014. - 338с.

Цифровые образовательные ресурсы

1. Россум Г., Дж. Дрейк Ф.Л., Откидач Д.С., Задка М., Левис М., Монтаро С., Реймонд Э.С., Кучлинг А.М., Лембург М.-А., Йи К.-П., Ксиллаг Д., Петрилли Х.Г., Варсав Б.А., Ахлстром Дж.К., Роскинд Дж., Шеменор Н., Мулендер С. Язык программирования Python: [Электронный ресурс]. 2001. URL: https://goo.gl/8TzY8w.

Приложение № 1

1. У исполнителя Квадратор две команды, которым присвоены номера:

1. возведи в квадрат,
2. прибавь 3.

Первая из них возводит число на экране во вторую степень, вторая — прибавляет к числу 3. Составьте алгоритм получения из числа 5 числа 127, содержащий не более 5 команд. В ответе запишите только номера команд. *(Например, 12212 — это алгоритм: возведи в квадрат прибавь 3 прибавь 3 возведи в квадрат прибавь 3, который преобразует число 2 в число 103).*

Если таких алгоритмов более одного, то запишите любой из них.

2. У исполнителя Альфа две команды, которым присвоены номера:

1. прибавь 1;
2. умножь на b

(*b* — неизвестное натуральное число; $b \geq 2$).

Выполняя первую из них, Альфа увеличивает число на экране на 1, а выполняя вторую, умножает это число на *b*. Программа для исполнителя Альфа - это последовательность номеров команд. Известно, что программа 11211 переводит число 6 в число 82. Определите значение *b*.

3. Ниже приведена программа, записанная на алгоритмическом языке:

```
алг
нач
цел s, t
ввод s
ввод t
если s > 9 или t > 9
    то вывод "YES"
    иначе вывод "NO"
все
кон
```

Было проведено 9 запусков программы, при которых в качестве значений переменных *s* и *t* вводились следующие пары чисел:

(9, 9); (9, 10); (8, 5); (11, 6); (–11, 10); (–5, 9); (–10, 10); (4, 5); (8, 6).

Сколько было запусков, при которых программа напечатала «NO»?

4. Запишите значение переменной *t*, полученное в результате работы следующей программы.

```
алг
нач
цел t,i
t := 2
 нц для i от 1 до 3
  t :=t * i
 кц
вывод t
кон
```

5. Некоторый алгоритм из одной цепочки символов получает новую цепочку следующим образом. Сначала вычисляется длина исходной цепочки символов; если она чётна, то в середину цепочки символов добавляется символ А, а если нечётна, то в начало цепочки добавляется символ Б. В полученной цепочке символов каждая буква заменяется буквой, следующей за ней в русском алфавите (А — на Б, Б — на В и т. д., а Я — на А). Получившаяся таким образом цепочка является результатом работы алгоритма.

Например, если исходной была цепочка **ВРМ,** то результатом работы алгоритма будет цепочка **ВГСН,** а если исходной была цепочка **ПД,** то результатом работы алгоритма будет цепочка **РБЕ.**

Дана цепочка символов **ТОР.** Какая цепочка символов получится, если к данной цепочке применить описанный алгоритм дважды (т. е. применить алгоритм к данной цепочке, а затем к результату вновь применить алгоритм)? Русский алфавит: АБВГДЕЁЖЗИИКЛМНОПРСТУ- ФХЦЧШЩЪЫЬЭЮЯ.

6. Напишите программу, которая в последовательности натуральных чисел определяет количество чисел, оканчивающихся на 6. Программа получает на вход количество чисел в последовательности, а затем сами числа. В последовательности всегда имеется число, оканчивающееся на 6. Количество чисел не превышает 1000. Введённые числа не превышают 30000. Программа должна вывести одно число — количество чисел, оканчивающихся на 6.

Пример работы программы:

Входные данные	Выходные данные
3 16 26 24	2

7. Напишите программу для решения следующей задачи. На контрольной работе по алгебре ученикам 9 класса было предложено 10 примеров. Неудовлетворительная оценка выставляется, если правильно решено менее половины примеров. Сколько неудовлетворительных оценок было получено учениками? Если хотя бы один из учеников правильно решил все задачи, выведите YES, иначе выведите NO. Программа получает на вход количество учеников в классе N ($1 \leq N \leq 30$), затем для каждого ученика вводится количество правильно решённых примеров.

Пример работы программы:

Входные данные	**Выходные данные**
4 3 9 2 8	2 NO

8. Напишите программу, которая в последовательности натуральных чисел определяет сумму всех чисел, кратных 6 и оканчивающихся на 4. Программа получает на вход натуральные числа, количество введённых чисел неизвестно, последовательность чисел заканчивается числом 0 (0 - признак окончания ввода, не входит в последовательность). Количество чисел не превышает 1000. Введённые числа не превышают 30 000. Программа должна вывести одно число: сумму всех чисел, кратных 6 и оканчивающихся на 4.

Пример работы программы:

Входные данные	Выходные данные
14 24 144 22 12 0	168