**МИНИСТЕРСТВО ПРОСВЕЩЕНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
МИНИСТЕРСТВО ОБРАЗОВАНИЯ ИРКУТСКОЙ ОБЛАСТИ
ААГО УПРАВЛЕНИЯ ОБРАЗОВАНИЯ
МАОУ «АНГАРСКИЙ ЛИЦЕЙ № 2 ИМЕНИ М.К.ЯНГЕЛЯ»**

РАССМОТРЕНО

Руководитель ЭМЛ
информатики

_____ Сабина Л.Г.

Протокол № _____ 1 _____

от "30" августа 2023 г.

СОГЛАСОВАНО

Заместитель директора по УВР

_____ Баринкова Л.В..

УТВЕРЖДЕНО

Директор

_____ Е.Ю.Распопина

Приказ № _____

от " _____ " _____ 2023 г

ПРОГРАММА

« _____ Программирование на языке Python _____ »

для обучающихся 7 класса

Ангарск, 2023 год

Пояснительная записка

Программа факультатива «Программирование на языке Python» разработана в соответствии с:

- Федеральным законом «Об образовании в Российской Федерации» от 29 декабря 2012 года № 273-ФЗ (с изменениями);
- Федеральным государственным образовательным стандартом среднего общего образования, утвержденным приказом Минобрнауки России от 17.05.2012 г. № 413 (с изменениями и дополнениями);
- Приказом Министерства образования и науки РФ от 30 августа 2013 г. № 1015 «Об утверждении Порядка организации и осуществления образовательной деятельности по основным общеобразовательным программам - образовательным программам начального общего, основного общего и среднего общего образования» (с изменениями и дополнениями);
- Примерной основной образовательной Программой среднего общего образования (одобрена решением федерального учебно-методического объединения по общему образованию (протокол от 28 июня 2016 г. № 2/16-з);
- Приказом Министерства образования и науки РФ от 14 февраля 2014 г. N 115 «Об утверждении Порядка заполнения, учета и выдачи аттестатов об основном общем и среднем общем образовании и их дубликатов» (с изменениями и дополнениями);
- Основной образовательной программой СОО МАОУ «Ангарский лицей № 2 им. М.К. Янгеля»;
- Положением о рабочей программе по учебному предмету (курсу) МАОУ «Ангарский лицей № 2 им. М.К. Янгеля»;

Программа предназначена для учащихся 7-8 классов и рассчитана на 68 часов. Программа может быть реализована в течение одного учебного года (2 часа в неделю) или в течение двух учебных лет: 34 часа в 7 классе и 34 часа в 8 классе (1 час в неделю).

Программа «Программирование на языке Python» направлена на подготовку творческой, технически грамотной, гармонично развитой личности, обладающей логическим мышлением, способной анализировать и решать задачи в команде в области информационных технологий.. Занятия по данному курсу рассчитаны на общенаучную подготовку обучающихся, развитие их мышления, логики, математических способностей, исследовательских навыков. Курс направлен на изучение основ программирования на языке Python. Приобретённые знания будут применимы в творческих проектах.

Актуальность

Современные технологии развиваются с невероятной быстротой, и в особенности, развиваются информационные и компьютерные технологии, последние, как известно, напрямую связаны с программированием. Ряд отраслей науки непосредственно зависят от уровня развития научного программирования. Так, своего исполнения ждут беспилотный транспорт, «умные» системы управления городским и сельским хозяйством, фондовыми биржами; ждут робототехника, геновая инженерия, искусственный интеллект, превосходящий возможности человека.

Существует большое количество языков программирования. По итогам октябрьского рейтинга 2021 года языков программирования ТЮВЕ Python занял 1 место. Python – мощный высокоуровневый язык программирования, идеально подходящий для разработки самостоятельных программ и сценариев. Python используют такие гиганты, как Google, Intel, Cisco и Hewlett-Packard, на нем работают популярные площадки YouTube, «ВКонтакте», DropBox. Python – язык программирования, который будет востребован еще очень долго!

Выбор языка программирования Python для изучения на факультативе обусловлен тем, что это язык, обладающий рядом преимуществ перед другими языками:

- Язык программирования – средство, а не цель
- Свободная кросс-платформенная реализация
- Удобная среда разработки, адаптированная для учебного процесса
- Широкая распространенность, поддерживаемость и развиваемость языка
- Возможность автоматической проверки
- Понятность языка и легкое восприятие учащимися

Новизна программы

Успешность изучения темы «Программирование» во многом зависит от мотивации обучения, от личностных планов на будущее, наличия стремления приобрести знания. В настоящее время обучение построено на внутренней мотивации и высокой самоорганизации при изучении тем. Анализ известных программ факультативов и элективных курсов по программированию показывает, что изучение языков программирования построено по принципу классно-урочной системы, что понижает мотивацию изучения данных курсов.

Данная программа направлена на изучение языка программирования Python с использованием метода проектов. Это позволяет повысить интерес к предмету, развить коммуникабельность, дает возможность проявить индивидуальные способности в различных направлениях проектной деятельности. Использование методов проектов позволяет учащимся подготовиться к исследовательской деятельности, написанию курсовых и дипломных работ. Программа ориентирована на обучение в вузах, где чаще всего программирование изучается на языке Python.

Методологические положения программы

В основу программы положена проектная деятельность при изучении языка Python, которая представляет собой творческую деятельность, результатом которой становится решение какой-либо проблемы, представленное в виде его подробного описания и практического результата (проекта и электронного продукта). Целью проектной технологии обучения является создание условий, при которых учащиеся самостоятельно приобретают знания из различных источников; учатся пользоваться приобретёнными знаниями для решения познавательных задач; развивают коммуникативные и исследовательские умения; развивают системное мышление.

При создании программы авторы опирались на следующие принципы:

- принцип прогностичности;
- принцип пошаговости;
- принцип нормирования;
- принцип обратной связи;
- принцип саморазвития;
- принцип продуктивности.

Программа факультатива предлагает использование следующих технологий:

- интерактивной;
- проектной;
- мультимедийной.

Методологической основой программы послужили теоретические положения проектной технологии (Т. Хатчинсон).

Организационно метод проектов предусматривает сочетание индивидуальной самостоятельной работы с работой в сотрудничестве, в малых группах и в коллективе. Реализация метода проектов на практике ведет к изменению позиции учителя. Из носителя готовых знаний он превращается в организатора познавательной, исследовательской деятельности своих учеников. Изменяется и психологический климат в классе, так как учителю приходится переориентировать свою учебно-воспитательную работу и работу учащихся на разнообразные виды самостоятельной деятельности, на приоритет деятельности исследовательского, поискового, творческого характера. Обеспечение и сопровождение проектной деятельности строится на принципах сотрудничества и включает:

- помощь в определении школьником замысла проектной деятельности;
- консультирование стадий проекта: поиска информации, решений проектных задач, поощрение практического опыта непосредственной работы с текстом;
- внимание к индивидуальным формам и способам аналитического и образного мышления, рассуждений и интерпретации, инициирование навыков продумывания деятельности и

прогнозирования ее продукта.

Формы проведения занятий – лекция, практическое занятие, творческий отчет,. Формы организации деятельности - групповая, индивидуально-групповая, индивидуальная, по подгруппам.

Программой предусмотрены различные формы взаимодействия учащихся. Помимо индивидуальной, планируется работа в парах, мини - группах, что приучает учащихся к кооперации, снимает монотонность в работе, обеспечивает обучение через цепочку «учитель-ученик», «ученик-ученик».

Общая характеристика программы

Программа по факультативу «Программирование на языке Python» предназначена для изучения всех основных разделов курса программирования на базовом уровне. В рамках предлагаемого курса изучение основ программирования на языке Python — это не столько средство подготовки к будущей профессиональной деятельности, а 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. 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.

Цель изучения курса: формирование интереса обучающихся к изучению профессий, связанных с основами программирования через освоение языка Python.

Задачи:

обучающие:

- освоение основных алгоритмических конструкций;
- обучение основам алгоритмизации и программирования;
- освоение первоначальных навыков программирования на языке программирования высокого уровня Python;
- приобщение к проектно-творческой деятельности;

воспитывающие:

- воспитание интереса к информационной и коммуникационной сфере человеческой деятельности,
- воспитание потребности соблюдать этические и правовые нормы работы с информацией;

- воспитание бережного отношения к техническим устройствам;
развивающая:
- развитие творческого воображения, алгоритмического мышления учащихся;
- развитие навыков планирования проекта, умения работать в группе;
- развитие навыков ориентации в информационных потоках окружающего мира и применения точной и понятной инструкции для решения учебных задач и в повседневной жизни.

Планируемый результат реализации программы

Основным результатом обучения является формирование вектора развития обучающихся с упором на формирование у них алгоритмического мышления.

Личностные результаты:

- сформированность мировоззрения, соответствующего современному уровню развития науки и техники;
- осознанный выбор будущей профессии и возможностей реализации собственных жизненных планов;
- сформированность представлений о мире профессий, связанных с программированием, и требованиях, предъявляемых различными востребованными профессиями, такими как программист, системный администратор;
- навыки сотрудничества в образовательной, учебно-исследовательской, проектной и других видах деятельности;
- навыки взаимо- и самооценки, навыки рефлексии.

Метапредметные результаты:

- владение навыками познавательной, учебно-исследовательской и проектной деятельности, навыками разрешения проблем;
- способность и готовность к самостоятельному поиску методов решения практических задач, применению различных методов познания.

Предметные результаты:

- навыки алгоритмического мышления и понимание необходимости формального описания алгоритмов;
- владение стандартными приёмами написания программы для решения стандартной задачи с использованием основных конструкций программирования и отладки таких программ на языке Python;
- знание особенностей структуры программы, представленной на языке Python,
- представление о модулях, входящих в состав среды Python,

- возможности и ограничения использования готовых модулей,
- представление о величине, ее характеристиках,
- знание что такое операция, операнд и их характеристики,
- знание принципиальные отличия величин, структурированных и не структурированных,
- представление о таких структурах данных, как число, текст, кортеж, список, словарь,
- представление о составе арифметического выражения;
- знание математических функций, входящих в Python, представление о логических выражениях и входящих в них операндах, операциях и функциях,
- умение записывать примеры арифметических и логических выражений всех атрибутов, которые могут в них входить,
- знание основных операторов языка Python, их синтаксис,
- представление о процессе исполнения каждого из операторов,
- умение разрабатывать программы обработки числовой и символьной информации,
- умение разрабатывать программы (линейные, разветвляющиеся и с циклами),
- представление о значении полноценных процедур и функций для структурноориентированного языка высокого уровня,
- правила описания функций в Python и построение вызова,
- принципиальные отличия между формальными, локальными и глобальными переменными.

Содержание курса в 7, 8 классе

Язык Python. Простейшие программы. Реализация вычислений и ветвлений. (12 ч.)

Знакомство с Python и средами программирования. Типы данных в программировании. Определение переменной. Ввод данных с клавиатуры. Логические выражения. Условный оператор. Множественное ветвление.

Реализация циклических, вспомогательных алгоритмов. (14 ч.)

Евклида (нахождение наибольшего общего делителя). Вычисление факториала на языке программирования Python. Работа с цифрами числа.

Проектная деятельность (8 ч.)

Что такое «метод проектов». Основные понятия. Основные требования предъявляются к учебным проектам. Виды учебных проектов. Этапы проектной деятельности. Оформление и структура отчета по проектной теме. Критерии оценивая проекта.

Словари. Массивы. Обработка массивов. Создание мини проекта. (16 ч.)

Списки — изменяемые последовательности. Массивы. Основные задачи обработки массивов: поиск, сортировка, реверс, ... Замена элементов в списке. Двоичный (бинарный) поиск элемента в массиве. Сортировка выбором (поиск минимума и перестановка). Сортировка методом пузырька Введение в словари. Решето Эратосфена – алгоритм определения простых чисел.

Символьные строки. Обработка символьных строк. Создание мини проекта. (6 ч.)

Символьные строки. Функции для работы с символьными строками. Сравнение и сортировка строк.

Матрицы. Ввод, вывод, обработка матриц. Чтение и запись текстовых файлов. Создание мини проекта. (12 ч.)

Матрицы. Основные задачи обработки всех элементов матрицы. Обработка строк и/или столбцов матрицы. Файлы. Чтение текстового файла. Запись в файл.

Календарно-тематическое планирование

№ занятия	Наименование темы	Часы	Дата
Язык Python. Простейшие программы. Реализация вычислений и ветвлений. (12 ч.)			
1.	Знакомство с Python и средами программирования.	1	1 неделя
2.	Типы данных в программировании. Определение переменной.	1	2 неделя
3.	Ввод данных с клавиатуры.	1	3 неделя
4.	Простейшие программы	1	4 неделя
5.	Вычисления	1	5 неделя
6.	Ввод нескольких значений в одной строке	1	6 неделя
7.	Логические выражения.	1	7 неделя
8.	Условный оператор.	1	8 неделя
9.	Вложенные условные операторы	1	9 неделя
10.	Каскадное ветвление.	1	10 неделя
11.	Сложные условия	1	11 неделя
12.	Сложные условия	1	12 неделя
Реализация циклических (14 ч.)			
13.	Цикл с условием	1	13 неделя
14.	Цикл с условием	1	14 неделя
15.	Цикл с условием	1	15 неделя
16.	Цикл с переменной	1	16 неделя
17.	Цикл с переменной	1	17 неделя
18.	Цикл с переменной	1	18 неделя
19.	Вложенные циклы.	1	19 неделя
20.	Вложенные циклы	1	20 неделя
21.	Алгоритм Евклида (нахождение наибольшего общего делителя).	1	21 неделя
22.	Алгоритм Евклида (нахождение наибольшего общего делителя).	1	22 неделя
23.	Вычисление факториала на языке программирования Python.	1	23 неделя
24.	Работа с цифрами числа.	1	24 неделя
25.	Работа с цифрами числа.	1	25 неделя
26.	Работа с цифрами числа.	1	26 неделя
Проектная деятельность (8 ч.)			
27.	Метод проектов	1	27 неделя

28.	Виды учебных проектов	1	28 неделя
29.	Этапы проектной деятельности	1	29 неделя
30.	Оформление документации	1	30 неделя
31.	Мини проект	1	31 неделя
32.	Мини проект	1	32 неделя
33.	Защита проекта	1	33 неделя
34.	резерв	1	34 неделя
Массивы. Обработка массивов. Создание мини проекта. (16 ч.)			
35.	Списки — изменяемые последовательности.	1	35 неделя
36.	Массивы. Заполнение массива с клавиатуры.	1	36 неделя
37.	Массивы. Заполнение массива случайными числами.	1	37 неделя
38.	Массивы. Решение основных задач.	1	38 неделя
39.	Максимум и минимум в массиве, работа с индексами	1	39 неделя
40.	Максимум и минимум в массиве, работа с индексами	1	40 неделя
41.	Максимум и минимум в массиве, работа с индексами	1	41 неделя
42.	Замена элементов в списке.	1	42 неделя
43.	Сортировка выбором (поиск минимума и перестановка).	1	43 неделя
44.	Сортировка методом пузырька	1	44 неделя
45.	Введение в словари.	1	45 неделя
46.	Решето Эратосфена - алгоритм определения простых чисел.	1	46 неделя
47.	Оформление документации	1	47 неделя
48.	Мини проект	1	48 неделя
49.	Мини проект	1	49 неделя
50.	Защита проекта	1	50 неделя
Символьный тип данных (6 ч.)			
51.	Метод проектов	1	51 неделя
52.	Виды учебных проектов	1	52 неделя
53.	Этапы проектной деятельности	1	53 неделя
54.	Оформление документации	1	54 неделя
55.	Мини проект	1	55 неделя
56.	Мини проект	1	56 неделя
Матрицы. Ввод, вывод, обработка матриц. Чтение и запись текстовых файлов. Создание мини проекта. (12 ч.)			
57.	Матрицы.	1	57 неделя
58.	Основные задачи обработки всех элементов матрицы.	1	58 неделя
59.	Основные задачи обработки всех элементов матрицы.	1	59 неделя

60.	Обработка строк и/или столбцов матрицы.	1	60 неделя
61.	Обработка строк и/или столбцов матрицы.	1	61 неделя
62.	Файлы.	1	62 неделя
63.	Чтение текстового файла. Запись в файл.	1	63 неделя
64.	Мини проект	1	64 неделя
65.	Мини проект	1	65 неделя
66.	Мини проект	1	66 неделя
67.	Защита проекта	1	67 неделя
68.	Резерв	1	68 неделя

Список литературы, использованной при составлении программы

- Ресурсы сайта Л. Шапошниковой (<http://younglinux.info>) , содержащие
 - авторские программы:
 - <http://younglinux.info/python.php> (<http://window.edu.ru/resource/825/76825>)
 - <http://younglinux.info/algorithm>
 - учебные пособия:http://younglinux.info/sites/default/files/python_structured_programming.pdf
- Сайт Д.П.Кириенко. Московский институт открытого образования. Школа №179
<http://www.179.ru/~dk/python.html>
- Ресурсы сайта профессора, доктора технических наук, учителя информатики высшей категории, автора учебников по информатике К.Ю. Полякова:
 - http://kpolyakov.spb.ru/loadstat.php?f=/download/ch10-8_python.pdf
 - <http://kpolyakov.spb.ru/loadstat.php?f=/download/slides10-8py.zip>
 - <http://kpolyakov.spb.ru/download/progr1011.doc>
 - http://kpolyakov.spb.ru/download/infobr_2013-6.pdf
- Сайт дистанционная подготовка по информатике <http://informatics.mccme.ru/>
 - <http://informatics.mccme.ru/course/view.php?id=95>
- Изучаем Python Автор: Марк Лутц Издательство: Символ-Плюс ISBN 978-5-93286-159-2, 978-0-596-15806-4; 2011 г. Переводчик: А. Киселев
- Head First Python. Автор: Paul Barry. Издательство: O'Reilly, год: 2010, Язык: Английский, страниц: 494, формат: pdf, размер: 33 МБ (<http://forcoder.ru/python/head-first-python-1191>)
- Hello World! Computer Programming for Kids and Other Beginners Second edition, Warren D. Sande and Carter Sande <http://helloworldbookblog.com/>
- Python for Kids. A Playful Introduction to Programming by Jason R. Briggs 2012, 344 pp. ISBN: 978-1-59327-407-8 Full Color <http://www.nostarch.com/pythonforkids>)
- Язык Python: <http://www.python.org>
- Среда разработки Wing IDE: <http://www.wingware.com/>
 - <https://kpolyakov.spb.ru/school/ppt.htm>
- «Поколение Python»: курс для начинающих,
 - <https://stepik.org/course/58852/promo/>

- Сайт проекта Интуит: Национальный открытый университет, курс «Введение в программирование на Python»,
 - [http://www.intuit.ru/studies/courses/12179/1172/info /](http://www.intuit.ru/studies/courses/12179/1172/info/)
- Сайт проекта Интуит: Национальный открытый университет. Курс «Язык программирования Python»,
 - [http://www.intuit.ru/studies/courses/49/49/info /](http://www.intuit.ru/studies/courses/49/49/info/)

Список литературы для учащихся

Ресурсы сайта профессора, доктора технических наук, учителя информатики высшей категории, автора учебников по информатике К.Ю. Полякова:

http://kpolyakov.spb.ru/loadstat.php?f=/download/ch10-8_python.pdf

<http://kpolyakov.spb.ru/loadstat.php?f=/download/slides10-8py.zip>

<http://kpolyakov.spb.ru/download/progr1011.doc>

- http://kpolyakov.spb.ru/download/infobr_2013-6.pdf
- Сайт дистанционная подготовка по информатике <http://informatics.mccme.ru/>
 - <http://informatics.mccme.ru/course/view.php?id=95>
- Сайт Д.П.Кириенко. Московский институт открытого образования. Школа №179,
 - <http://www.179.ru/~dk/python.html/>
- <https://kpolyakov.spb.ru/school/ppt.htm/>
- «Поколение Python»: курс для начинающих,
 - <https://stepik.org/course/58852/promo/>

Приложение 1

Тема урока: ввод-вывод данных

1. Вывод данных, команда `print()`
2. Ввод данных, команда `input()`
3. Решение задач

Вывод данных, команда `print`

Для вывода данных на экран используется команда `print()`.

Внутри круглых скобок пишем, что хотим вывести на экран. Если это текст, то обязательно указываем его внутри кавычек. Кавычки могут быть одинарными или двойными. Только обязательно ставим одинаковые до и после текста.

Например, следующий код:

```
print('Мы изучаем язык Python')
```

выведет на экран текст:

```
Мы изучаем язык Python
```

Внимание! Кавычки могут быть и одинарными, и двойными. Следующие две строки делают одно и то же.

```
print('Python')
```

```
print("Python")
```

То, что мы пишем в круглых скобках у команды `print()`, называется **аргументами** или **параметрами** команды.

Команда `print()` позволяет указывать несколько аргументов, в таком случае их надо отделять запятыми.

Например, следующий код:

```
print('Скоро я', 'буду программировать', 'на языке', 'Python!')
```

выведет на экран текст:

```
Скоро я_буду программировать_на языке_Python!
```

Обратите внимание, в качестве разделителя при выводе данных между аргументами команды используется **символ пробел**. По умолчанию команда `print()` добавляет ровно один пробел между всеми своими аргументами. Например, следующая программа:

```
print('1', '2', '4', '8', '16')
```

выводит числа `1 2 4 8 16`. Числа выведены ровно через один пробел.

Внимание! При написании кода, между аргументами команды `print()` после запятой мы ставим 1 символ пробел. Это общепринятое соглашение в языке Python. Этот символ пробела не влияет на вывод данных.

Примечания

Примечание 1. Команда `print()` записывается только маленькими буквами, другое написание недопустимо, так как в Python строчные и заглавные буквы различны.

Примечание 2. Каждая последующая команда `print()` выводит указанный текст **с новой строки**. Например, следующая программа:

```
print('Какой хороший день!')
```

```
print('Работать мне не лень!')
```

выведет на экран две строки:

```
Какой хороший день!
```

```
Работать мне не лень!
```

Примечание 3. Команда `print()` с пустым списком аргументов просто вставляет новую пустую строку. Например:

```
print('Какой хороший день!')
```

```
print()
```

```
print('Работать мне не лень!')
```

выведет на экран три строки, одна из которых пустая:

```
Какой хороший день!
```

```
Работать мне не лень!
```

Примечание 4. Почему в Python можно использовать как одинарные, так и двойные кавычки для обрамления текста? Ответ очень прост — часто кавычки это часть текста. И чтобы Python мог правильно распознать такой текст, пользуемся разными:

- если в тексте нужны одинарные кавычки, то для обрамления такого текста используем двойные кавычки;
- если в тексте нужны двойные кавычки, то обрамляем его одинарными.

Результатом выполнения кода:

```
print('В тексте есть "двойные" кавычки')
```

```
print("В тексте есть 'одинарные' кавычки")
```

будет:

В тексте есть "двойные" кавычки

В тексте есть 'одинарные' кавычки

Приложение 2

Тема урока: `sep`, `end`, переменные, PEP 8

1. Необязательные параметры команды `print()`: `sep`, `end`
2. Переменные
3. Стандарт PEP 8
4. Комментарии
5. Решение задач

Аннотация. Урок посвящен изучению необязательных параметров команды `print()` и более детальному разбору понятия переменных. Также мы поговорим о комментариях и стандарте PEP 8, которого придерживаются Python-программисты.

Необязательные параметры команды `print`

По умолчанию команда `print()` принимает несколько аргументов (параметров), выводит их через один **пробел**, после чего **ставит перевод строки**. Это поведение можно изменить, используя необязательные именованные параметры `sep` (separator, разделитель) и `end` (окончание).

Параметр `sep`

Рассмотрим следующий код:

```
print('a', 'b', 'c')
```

```
print('d', 'e', 'f')
```

Результатом выполнения такого кода будет:

```
a b c
```

```
d e f
```

Рассмотрим следующий код:

```
print('a', 'b', 'c', sep='*')
```

```
print('d', 'e', 'f', sep='**')
```

Результатом выполнения такого кода будет:

```
a*b*c
```

```
d**e**f
```

При первой печати в качестве строки разделителя между аргументами команды `print()` установлена строка `sep='*'`.

При второй печати в качестве строки разделителя между аргументами команды `print()` установлена строка `sep='**'`.

Таким образом, необязательный параметр `sep` команды `print()` позволяет установить строку, с помощью которой будут разделены аргументы при печати.

Параметр `end`

Если перевод строки делать не нужно или требуется указать специальное окончание, то следует явно указать значение для параметра `end`.

Рассмотрим следующий код:

```
print('a', 'b', 'c', end='@')
print('d', 'e', 'f', end='@@')
```

Результатом выполнения такого кода будет:

```
a b c@d e f@@
```

По завершении первой печати вставлена строка `@` вместо перевода строки. Аналогично, по завершении второй печати вставлена строка `@@`.

Параметры `sep` и `end` можно использовать вместе. Рассмотрим следующий код:

```
print('a', 'b', 'c', sep='*', end='finish')
print('d', 'e', 'f', sep='**', end='^__^')
print('g', 'h', 'i', sep='+', end='%')
print('j', 'k', 'l', sep='-', end='#')
print('m', 'n', 'o', sep='/', end='!')
```

Результатом выполнения такого кода будет:

```
a*b*cfinishd**e**f^__^g+h+i%j-k-l#m/n/o!
```

Примечания

Примечание 1. Вызов команды `print()` с пустыми скобками ставит перевод строки.

Примечание 2. Последовательность символов `\n` называется управляющей последовательностью и задает перевод строки.

Примечание 3. Значения по умолчанию у параметров `sep` и `end` следующие:

```
sep=' ' # пробел
```

```
end='\n' # перевод строки
```

Примечание 4. Чтобы убрать все дополнительные выводимые символы, можно вызывать команду `print()` так:

```
print('a', 'b', 'c', sep=",", end="")
```

Примечание 5. Программный код

```
print('Python')
```

равнозначен коду

```
print('Python', end='\n')
```

Примечание 6. Если после вывода данных нужно более одного перевода строки, то необходимо использовать следующий код:

```
print('Python', end='\n\n\n')
```

Приложение 3

Тема урока: работа с целыми числами

1. Целочисленный тип данных
2. Преобразование строки к целому числу
3. Операции над целыми числами
4. Решение задач

Аннотация. Урок посвящен изучению работы с целыми числами. Изучим основные операции, а также научимся преобразовывать строки в числа.

Целые числа

Все предыдущие программы, которые мы писали, работали с текстовыми данными. Действительно, команда `input()` считывает строку текста. Однако во многих случаях нам нужно работать именно с числами. Чтобы в Python создать переменную целого типа данных, нужно опустить кавычки при объявлении переменной. Рассмотрим следующий код:

```
num1 = 7      # num1 - это число
num2 = 10     # num2 - это число
num3 = num1 + num2 # num3 - это число
print(num3)
```

В результате выполнения такой программы будет выведено число 17.

Внимание! Числа обозначаются без кавычек, а строки — с кавычками.

Основные операции с числами

В языке Python, как и в математике, над числами можно совершать 4 основные операции (+, −, *, /).

Оператор	Описание
+	сложение
-	вычитание
*	умножение
/	деление

Рассмотрим следующую программу:

```
a = 3
```

```
b = 2
```

```
print(a + b)
```

```
print(a - b)
```

```
print(a * b)
```

```
print(a / b)
```

Результатом выполнения такой программы будут числа:

```
5
```

```
1
```

```
6
```

```
1.5
```

Порядок выполнения операций

В математике существует порядок выполнения операций, определяющий, какие операции должны выполняться раньше других, даже если в выражении они написаны правее. Порядок выполнения операций в Python аналогичен порядку выполнения операций, которые вы изучали на уроках математики.

Рассмотрим следующий код:

```
num1 = 2 + 3 * 4
```

```
num2 = (2 + 3) * 4
```

```
print(num1)
```

```
print(num2)
```

Результатом выполнения такой программы будут числа 14 и 20. В переменной `num1` будет храниться число 14, поскольку первым делом выполняется умножение, а уже затем сложение. В переменной `num2` будет храниться число 20, поскольку скобки изменили приоритет выполнения операций.

Внимание! Первым делом выполняется умножение или деление, затем сложение и вычитание. Для изменения порядка выполнения операций понадобятся скобки.

Преобразование типов

Преобразование строки к целому числу. Для того, чтобы преобразовать строку к целому числу, мы используем команду `int()`. Рассмотрим следующий код:

```
s = '1992'
```

```
year = int(s)
```

Переменная `s` имеет строковый тип данных. С помощью команды `int()`, мы преобразовали строку к целому числу и записали результат в переменную с именем `year`. **Мы используем новую команду (точнее, функцию) – `int()`**. Она означает следующее: «Возьми то, что указано в качестве аргумента в скобках, и преврати это в целое число». Таким образом, переменная `year` имеет целый тип данных, а переменная `s` – строковый тип данных.

Напишем программу, которая считывает два целых числа и выводит на экран их сумму. Следующий код тут не сработает:

```
num1 = input()
```

```
num2 = input()
```

```
print(num1 + num2)
```

Python считает, что в переменных `num1` и `num2` находится **текст**, поскольку команда `input()` по умолчанию считывает именно **текст**. Для того, чтобы явно указать, что требуется работать с переменными целого типа, надо написать так:

```
num1 = int(input())
```

```
num2 = int(input())
```

```
print(num1 + num2)
```

Внимание! Для того, чтобы считать одно целое число, мы пишем следующий код:

```
num = int(input())
```

Преобразование целого числа к строке. Для того, чтобы преобразовать целое число в строку, мы используем команду `str()`. Рассмотрим следующий код:

```
num = 17
```

```
s = str(17)
```

Переменная `num` имеет целый тип данных. С помощью команды `str()` мы преобразовали целое число в строку и записали результат в переменную `s`. Таким образом, переменная `num` имеет целочисленный тип данных, а переменная `s` – строковый тип данных.

Примечания

Примечание 1. При работе с целочисленными переменными часто нужно осуществлять присвоение значению переменной целой математической формулы:

```
<имя переменной> = <математическая формула>
```

Имя переменной = математическая формула

Примечание 2. Сокращение `int` происходит от английского **integer** – целый.

Примечание 3. В большинстве языков программирования переменная целого типа имеет ограничения. Например, в языке C# переменная целого типа может принимать значения из диапазона $[-2^{63}; 2^{63} - 1]$ $[-263; 263 - 1]$. В Python реализована так называемая **длинная арифметика**, то есть, по сути, переменная целого типа не имеет ограничений.

Примечание 4. Минус может использоваться для обозначения отрицательных чисел. А вот операции `+`, `/` и `*` всегда стоят между двумя переменными и что-то с ними делают.

```
num1 = -6 # унарный минус
```

```
num2 = 17 - 7 # бинарный минус
```