**Министерство просвещения РФ**

**муниципальное казённое общеобразовательное учреждение**

**«Вихоревская средняя общеобразовательная школа № 2»**

**Братского района Иркутской области**

**Индивидуальная проектная деятельность**

**«Язык программирования Python»**

**Выполнил:**

**Олейников Дмитрий,**

**ученик 9-б класса**

**Руководитель:**

**Кудрявцева М. Г.**

**Вихоревка, 2020 г**.

**Содержание**

Введение………………………………………………………………………………………….3

I. О языке программирования Python……………………………………………………..4

* 1. История языка программирования Python……………………………………………...4

1.2. Разработки в области языка программирования Python………………………...…….4

II. Сфера применения языка программирования Python……………………………...….6

III. Недостатки языка Python………………………………………………………………...7

IV. Трудоустройство и средняя зарплата Python-разработчика………………………......8

V. Структуры данных……………………………………………………………………….9

Заключение……………………………………………………………………………………...12

Список используемой литературы…………………………………………………………….13

# **Введение**

Вы когда-нибудь использовали приложение на мобильном телефоне или программу на компьютере? А задумывались ли вы при этом, на каком языке программирования написаны сайты, интернет страницы, скрипты? В наше время знание языков программирования не просто показатель образованности человека, а и необходимость. Если вы хотите идти в ногу со временем, для вас просто обязательно знание хотя бы одного языка.

Языки программирования дают возможность создавать средства для работы, общения и творчества. В мире существуют тысячи языков, позволяющие вести полноценный диалог с компьютером, однако, несмотря на такое разнообразие, число языков, на которых пишет большинство около десятка. Языки программирования вынуждены преодолевать те же трудности, что и многие фирмы, и новые идеи. Зачастую успех языка программирования зависит не только от его характеристик и применяемых технологий, но и от появления в нужном месте, в нужное время. Успех языка может быть столь же непостоянным, сколь непредсказуемыми могут быть причины его провала.

**Актуальность проекта:**

Информационные технологии являются одной наиболее быстро развивающихся областей современной жизни. Они появляются едва не каждый день. И в погоне за прогрессом, подчас полезно остановиться, вспомнить историю и задуматься о будущем. С тем, чтобы со свежими силами вновь окунуться в работу, осваивать новые технологии.

К выбору данной теме меня побудило незнакомое понятие «Мертвые языки программирования». Данный проект позволит приблизиться к истокам программирования. И дать ответ на основной вопрос: почему одни языки программирования остались на стадии зарождения и про них все забыли, а другие легли в основу новых, которые живут и позволяют человечеству развивать информационные технологии?

**Цель:** Изучение основ разработки языка программирования Python и его иллюстрация.

**Задачи:**

1. Познакомится с историей появления, создания и развития языка программирования Python, раскрыть его классификацию

2. Изучить и систематизировать процессы, методы и этапы работы языка программирования Python.

**I. О языке программирования Python**

**1.1. История** **языка программирования Python**

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функция.

Разработчик:[Гвидо ван Россум](https://www.google.com/search?sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605&q=%D0%93%D0%B2%D0%B8%D0%B4%D0%BE+%D0%B2%D0%B0%D0%BD+%D0%A0%D0%BE%D1%81%D1%81%D1%83%D0%BC&stick=H4sIAAAAAAAAAOPgE-LQz9U3MK0yjFfiBLEMMwxMk7UMMsqt9JPzc3JSk0sy8_P0C4ry04sSc3Mz89LjcxLz0ksT01OLrVJSizPT81JTFJIqF7HKXZh8YdOFHRe2XNinAGRsuLBX4cKCC_suNgJh84U9O1gZAUGsoCJtAAAA&sa=X&ved=2ahUKEwi2zoDf1YznAhWS1aYKHfAsCFYQmxMoATAdegQIDRAH&sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605)

Основан в:[20 февраля 1991](https://www.google.com/search?sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605&q=20+%D1%84%D0%B5%D0%B2%D1%80%D0%B0%D0%BB%D1%8F+1991&stick=H4sIAAAAAAAAAONgVhLQL9E3MjWvyIs3j8_KzipOX8QqZmSgcLHlwtYLmy42XNhwYffFfgVDS0tDALv6zOUtAAAA&sa=X&ved=2ahUKEwi2zoDf1YznAhWS1aYKHfAsCFYQmxMoATAeegQIDRAL&sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605)

[Сайт](https://www.google.com/search?sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605&q=python+%D1%81%D0%B0%D0%B9%D1%82&sa=X&ved=2ahUKEwi2zoDf1YznAhWS1aYKHfAsCFYQ6BMoADAfegQIDRAO&sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605)**:**python.org​ (англ.)

[Лицензия](https://www.google.com/search?sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605&q=python+%D0%BB%D0%B8%D1%86%D0%B5%D0%BD%D0%B7%D0%B8%D1%8F&sa=X&ved=2ahUKEwi2zoDf1YznAhWS1aYKHfAsCFYQ6BMoADAgegQIDRAR&sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605):[Python Software Foundation License](https://www.google.com/search?sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605&q=Python+Software+Foundation+License&stick=H4sIAAAAAAAAAONgVuLUz9U3sMwry0lZxKoUUFmSkZ-nEJyfVlKeWJSq4JZfmpeSWJIJFPPJTE7NK04FAKAztwcyAAAA&sa=X&ved=2ahUKEwi2zoDf1YznAhWS1aYKHfAsCFYQmxMoATAgegQIDRAS&sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605)

[ОС](https://www.google.com/search?sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605&q=python+%D0%BE%D1%81&sa=X&ved=2ahUKEwi2zoDf1YznAhWS1aYKHfAsCFYQ6BMoADAhegQIDRAV&sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605): кроссплатформенность

Дата [выпуска](https://www.google.com/search?sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605&q=python+%D0%B2%D1%8B%D0%BF%D1%83%D1%81%D0%BA&sa=X&ved=2ahUKEwi2zoDf1YznAhWS1aYKHfAsCFYQ6BMoADAiegQIDRAY&sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605): [18 декабря 2019](https://www.google.com/search?sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605&q=18+%D0%B4%D0%B5%D0%BA%D0%B0%D0%B1%D1%80%D1%8F+2019&stick=H4sIAAAAAAAAAONgVhLQL9E3SjHMqsjJtUypqiwrXsQqZmihcGHLha0Xdl3YcGHjxYaL_QpGBoaWAK9VPGQtAAAA&sa=X&ved=2ahUKEwi2zoDf1YznAhWS1aYKHfAsCFYQmxMoAzAiegQIDRAb&sxsrf=ACYBGNT1HnkvXgY_wDXIVcG-1oVRPHJq1g:1579334030605);

Python — один из самых популярных языков программирования. Расположение разработчиков он заслужил за простоту изучения и использования. У языка множество сфер применения: на нём можно построить что угодно.

Python — это скриптовый язык программирования. Он универсален, поэтому подходит для решения разнообразных задач и многих платформ, начиная с iOS и Android и заканчивая серверными ОС.

Python - интерпретируемый язык — он не компилируется, то есть до запуска представляет из себя обычный текстовый файл. Программировать можно практически на всех платформах, язык хорошо спроектирован и логичен.

Разработка на нем в разы быстрее, потому что приходится писать меньше кода, чем на Java, С и других языках, — он отлично подходит новичкам.

Если простым языком, компиляция - это "перевод" программного кода в понятный для конкретной операционной системы вид. Т. е. для Windows это будут исполняемые .exe файлы, приложения.

**1.2. Разработки в области языка программирования Python**

1. Веб-разработка:

Чаще всего Python используется в веб-разработке. Для работы с ним используются фреймворки: Pyramid, Flask, CherryPy и самый популярный-это Django.

2. Программы:

Хоть Python и не компилируется, его можно использовать для создания

Многопользовательских приложений. Вот небольшой список того, что было разработано на Python:

* GIMP — визуальный редактор в Операционной Системе Linux;
* Blender — программа для создания 3D-графики.

3. Игры

Многие компьютерные игры были полностью или частично написаны на Python. Существует заблуждение, что этот язык не подходит для серьезных проектов, но на самом деле он использовался в разработке таких хитов, как:

**Battlefield 2**;

**World of Tanks**;

**Civilization IV**;

**EVE Online**.

Несмотря на то что в Python есть возможность реализации пользовательского интерфейса и работы с графикой, чаще всего язык используют для написания скриптов — например, взаимодействия персонажей, запуска сцен, а также обработки событий.

4. .Встроенные системы

На Python часто разрабатывают встроенные системы для различных устройств. Например, его используют в Raspberry P**i** (компьютер размером с карту памяти) и в «Сбербанке» для управления банкоматами.

Также язык применяется во встроенных системах станков с ЧПУ, средствах автоматического регулирования (температуры, расхода жидкостей, давления и так далее) и телекоммуникационном оборудовании.

5. Создание скриптов

Python можно использовать для написания плагинов и скриптов к уже готовым программам. Например, для реализации игровой логики. Также он может использоваться для создания дополнительных модулей.

Часто на Python пишут скрипты, которые встраивают в программы на других языках, чтобы автоматизировать какие-либо задачи.

**II. Сфера применения языка программирования Python**

1. .Системное администрирование

Python часто используется системными администраторами для автоматизации задач.

Он простой, мощный и поддерживает специальные пакеты, которые повышают его эффективность. И, самое главное, он по умолчанию установлен на все серверы с ОС Linux.

Благодаря лаконичности Python можно быстро прочесть код и найти слабые места. Форматирование в языке —это часть синтаксиса.

2. .Научные исследования

В Python есть несколько библиотек, которые можно использовать для проведения исследований и вычислений:

**SciPy** — библиотека с научными инструментами;

**NumPy** — расширение, которое добавляет поддержку матриц и многомерных массивов, а также математические функции для работы с ними;

**Matplotlib** — библиотека для работы с 2D- и 3D-графикой.

Благодаря библиотекам и простоте освоения языка многие ученые выбирают Python — особенно он популярен у математиков и физиков.

3. Data Science

Python — один из самых используемых в Data Science языков. На нем пишут алгоритмы программ с машинным обучением и аналитические приложения. С помощью него обслуживают хранилища данных и облачные сервисы.

Также с его помощью можно парсить (scrapping) данные из интернета. Например, в Google Python применяют для индексации сайтов.

Парсить-автоматически обрабатывать (разбирать) с целью получения нужных данных

**III. Недостатки языка Python**

Несмотря на все достоинства, у языка есть и недостатки. Программы на нем считаются одними из самых медленных. Для сравнения: приложения для iOS на Swift работают в 8,7 раз быстрее, чем приложения на Python.

У Python существует реализация PyPy, которая по скорости близка к Java, но в ней есть не все возможности оригинального языка. Python не подходит для задач, требующих большого объема памяти, — их лучше решать вставками на C или C++.

Другим недостатком является сильная зависимость языка от системных библиотек, из-за чего затрудняется перенос на другие системы. Для этих целей существует инструмент Virtualenv, но и он с недостатками: избыточность полных методов изоляции, костыли, дублирование системных библиотек.

Еще одна проблема — в том, что Global Interpreter Lock (GIL) не позволяет выполнять несколько потоков Python одновременно в реализации CPython. Однако GIL можно отключить на какое-то время, как это сделано в математическом пакете NumPy.

**IV. Трудоустройство и средняя зарплата Python-разработчика**

По данным с hh.ru на начало 2019 года, в России ~4500 вакансий для Python-разработчиков, из них ~2000 в Москве и ~700 в Санкт-Петербурге. Это меньше, чем по запросу «Java» (~5500), но больше, чем по запросу «PHP» (~3600), — можно заметить тенденцию, что Python медленно забирает позиции PHP с рынка веб-разработки. Хотя на PHP все еще написано около 80% всех сайтов в интернете.

Минимальная зарплата по России начинается с 70 000 рублей, а в Москве — с 80 000 рублей. В основном ищут опытных разработчиков, junior-специалисты менее востребованы.

На должность стажера или младшего специалиста можно устроиться только в крупную компанию, а расположены они в больших городах типа Москвы и Санкт-Петербурга. Из-за этого новичкам крайне сложно устроиться в регионах — остается искать заказы на фрилансе.



Во - первых стоит отметить интересную особенность Python. Он не содержит операторных скобок (begin..end в pascal или {..}в Си), вместо этого **блоки выделяются отступами**: пробелами или табуляцией, а вход в блок из операторов осуществляется двоеточием. Однострочные комментарии начинаются со знака фунта «#», многострочные — начинаются и заканчиваются тремя двойными кавычками «"""».
Чтобы присвоить значение пременной используется знак «=», а для сравнения —
«==». Для увеличения значения переменной, или добавления к строке используется оператор «+=», а для уменьшения — «-=». Все эти операции могут взаимодействовать с большинством типов, в том числе со строками. Например

>>> myvar = 3
>>> myvar += 2
>>> myvar -= 1
""«Это многострочный комментарий
Строки заключенные в три двойные кавычки игнорируются»""
>>> mystring = «Hello»
>>> mystring += " world."
>>>**print** mystring
Hello world.
# Следующая строка меняет
значения переменных местами. (Всего одна строка!)
>>> myvar, mystring = mystring, myvar

**V. Структуры данных**

Python содержит такие структуры данных как**списки (lists), кортежи (tuples) и словари (dictionaries**). Списки — похожи на одномерные массивы (но вы можете использовать Список включающий списки — многомерный массив), кортежи — неизменяемые списки, словари — тоже списки, но индексы могут быть любого типа, а не только числовыми. "Массивы" в Python могут содержать данные любого типа, то есть в одном массиве может могут находиться числовые, строковые и другие типы данных. Массивы начинаются с индекса 0, а последний элемент можно получить по индексу -1 Вы можете присваивать переменным функции и использовать их соответственно.

>>> sample = [1, [«another», «list»], («a»,«tuple»)]#Список состоит из целого числа, другого списка и кортежа
>>> mylist = [«List item 1»,2 , 3.14]#Этот список содержит строку, целое и дробное число
>>> mylist[0] = «List item 1 again» #Изменяем первый (нулевой) элемент листа mylist
>>> mylist[-1] = 3.14 #Изменяем последний элемент листа
>>> mydict = {«Key 1»: «Value 1», 2: 3, «pi»: 3.14} #Создаем словарь, с числовыми и целочисленным индексами
>>> mydict[«pi»] = 3.15 #Изменяем элемент словаря под индексом «pi».
>>> mytuple = (1, 2, 3) #Задаем кортеж
>>> myfunction = len #Python позволяет таким образом объявлять синонимы функции
>>> **print** myfunction(list)
3

Можно использовать часть массива, задавая первый и последний индекс через двоеточие «:». В таком случае вы получите часть массива, от первого индекса до второго не включительно. Если не указан первый элемент, то отсчет начинается с начала массива, а если не указан последний — то масив считывается до последнего элемента. Отрицательные значения определяют положение элемента с конца. Например:

>>> mylist = [«List item 1», 2, 3.14]
>>> **print** mylist[:] #Считываются все элементы массива
['List item 1', 2, 3.1400000000000001]
>>> **print** mylist[0:2] #Считываются нулевой и первый элемент массива.
['List item 1', 2]
>>> **print** mylist[-3:-1] #Считываются элементы от нулевого (-3) до второго (-1) (не включительно)
['List item 1', 2]
>>> **print** mylist[1:] #Считываются элементы от первого, до последнего
[2, 3.14]

Строки в Python **обособляются кавычками двойными «"» или одинарными «'»**. Внутри двойных ковычек могут присутствовать одинарные или наоборот. К примеру строка «Он сказал 'привет'!» будет выведена на экран как «Он сказал 'привет'!». Если нужно использовать строку из несколько строчек, то эту строку надо начинать и заканчивать тремя двойными кавычками «"""». Вы можете подставить в шаблон строки элементы из кортежа или словаря. Знак процента «%» между строкой и кортежем, заменяет в строке символы «%s» на элемент кортежа. Словари позволяют вставлять в строку элемент под заданным индексом. Для этого надо использовать в строке конструкцию «%(индекс)s». В этом случае вместо «%(индекс)s» будет подставлено значение словаря под заданным индексом.

>>>**print** «Name: %s\nNumber: %s\nString: %s» % (my**class**.name, 3, 3 \* "-")
Name: Poromenos
Number: 3
String: —
strString = ""«Этот текст расположен
на нескольких строках»""

>>> **print** «This %(verb)s a %(noun)s.» % {«noun»: «test», «verb»: «is»}
This is a test.

Операторы while, **if**, **for** составляют операторы перемещения. Здесь нет аналога оператора select, так что придется обходиться **if**. В операторе **for** происходит сравнение **переменной и списка**. Чтобы получить список цифр до числа <number> — используйте функцию range(<number>). Вот пример использования операторов

rangelist = range(10) #Получаем список из десяти цифр (от 0 до 9)
>>> **print** rangelist
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
**for** number in rangelist: #Пока переменная number (которая каждый раз увеличивается на единицу) входит в список…
# Проверяем входит ли переменная
# numbers в кортеж чисел (3, 4, 7, 9)
**if** number in (3, 4, 7, 9): #Если переменная number входит в кортеж (3, 4, 7, 9)...
# Операция «**break**» обеспечивает
# выход из цикла в любой момент
**break**
**else**:
# «**continue**» осуществляет «прокрутку»
# цикла. Здесь это не требуется, так как после этой операции
# в любом случае программа переходит опять к обработке цикла
**continue**
**else**:
# «**else**» указывать необязательно. Условие выполняется
# если цикл не был прерван при помощи «**break**».
**pass** # Ничего не делать

**if** rangelist[1] == 2:
**print** «The second item (lists are 0-based) is 2»
**elif** rangelist[1] == 3:
**print** «The second item (lists are 0-based) is 3»
**else**:
**print** «Dunno»

**while** rangelist[1] == 1:
**pass**

Для объявления функции служит **ключевое слово «def»**. Аргументы функции задаются в скобках после названия функции. Можно задавать необязательные аргументы, присваивая им значение по умолчанию. Функции могут возвращать кортежи, в таком случае надо писать возвращаемые значения через запятую. Ключевое слово «**lambda**» служит для объявления элементарных функций.

# arg2 и arg3 — необязательые аргументы, принимают значение объявленное по умолчни,
# если не задать им другое значение при вызове функци.
**def** myfunction(arg1, arg2 = 100, arg3 = «test»):
**return** arg3, arg2, arg1
#Функция вызывается со значением первого аргумента — "Argument 1", второго — по умолчанию, и третьего — "Named argument".
>>>ret1, ret2, ret3 = myfunction(«Argument 1», arg3 = «Named argument»)
# ret1, ret2 и ret3 принимают значения "Named argument", 100, "Argument 1" соответственно
>>> **print** ret1, ret2, ret3
Named argument 100 Argument 1

# Следующая запись эквивалентна **def** f(x): **return** x + 1
functionvar = **lambda** x: x + 1
>>> **print** functionvar(1)
2

Язык Python ограничен в множественном наследовании в классах. Внутренние переменные и внутренние методы классов начинаются с двух знаков нижнего подчеркивания «\_\_» (например «\_\_myprivatevar»). Мы можем также присвоить значение переменной класса извне. Пример:

Особенности

* Условия могут комбинироваться.1 < a < 3 выполняется тогда, когда а больше 1, но меньше 3.
* Используйте операцию «**del**» чтобы **очищать переменные или элементы массива**.
* Python предлагает большие возможности для **работы со списками**. Вы можете использовать операторы объявлении структуры списка. Оператор **for** позволяет задавать элементы списка в определенной последовательности, а **if** — позволяет выбирать элементы по условию.

**Заключение**

**Каждый в нашей стране должен научиться программировать, потому что это учит думать.**

**Стив Джобс**

Всем нужно знать программирование. Пусть на самом начальном уровне. И неважно, что вам это не пригодится в вашей работе. Как сказал Стив Джобс, это научит вас думать. XX век стал переломным в вопросе грамотности населения. Люди научились читать, писать, считать и даже начали получать высшее образование. Благодаря этому, сегодня технический прогресс шагает семимильными шагами. У нас есть радио и телевидение, телефоны и компьютеры. Человек даже полетел в космос. В XXI же веке программирование стало новым стандартом грамотности. А для этого нужно знать языки программирования.

 Мы в науке программирования пока что с ЭВМ на ВЫ. и своё развитие в этой области нужно начинать с истории и эволюции языков программирования.

В данной работе я постарался ответить на очень важный вопрос, почему языки программирования «умирают».

**Список используемой литературы:**

1. <https://www.google.com/amp/s/skillbox.ru>

2. <https://pythonworld.ru/>

3. <https://ru.m.wikipedia.org/wiki/Python>