**Введение**

**Что такое язык программирования ‘python’?**

**Python** в русском языке распространено название *пито́н* [высокоуровневый язык программирования](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. [Синтаксис](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) ядра Python минималистичен. В то же время [стандартная библиотека](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_Python) включает большой объём полезных функций.

Python поддерживает [структурное](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [объектно-ориентированное](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [функциональное](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), [императивное](https://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%BF%D0%B5%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) и [аспектно-ориентированное программирование](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D0%BF%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5). Основные архитектурные черты — [динамическая типизация](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), [автоматическое управление памятью](https://ru.wikipedia.org/wiki/%D0%A1%D0%B1%D0%BE%D1%80%D0%BA%D0%B0_%D0%BC%D1%83%D1%81%D0%BE%D1%80%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), полная [интроспекция](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D1%80%D0%BE%D1%81%D0%BF%D0%B5%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), механизм [обработки исключений](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B0_%D0%B8%D1%81%D0%BA%D0%BB%D1%8E%D1%87%D0%B5%D0%BD%D0%B8%D0%B9), поддержка [многопоточных вычислений](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D0%BF%D0%BE%D1%82%D0%BE%D1%87%D0%BD%D0%BE%D1%81%D1%82%D1%8C), высокоуровневые [структуры данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85). Поддерживается разбиение программ на [модули](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D1%83%D0%BB%D1%8C_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), которые, в свою очередь, могут объединяться в пакеты.

Эталонной реализацией Python является интерпретатор [CPython](https://ru.wikipedia.org/wiki/CPython" \o "CPython), поддерживающий большинство активно используемых платформ. Он распространяется под [свободной лицензией](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D0%BE%D0%B1%D0%BE%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) Python Software Foundation License, позволяющей использовать его без ограничений в любых приложениях, включая [проприетарные](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BF%D1%80%D0%B8%D0%B5%D1%82%D0%B0%D1%80%D0%BD%D0%BE%D0%B5_%D0%9F%D0%9E" \o "Проприетарное ПО). Есть [реализация интерпретатора для JVM](https://ru.wikipedia.org/wiki/Jython) с возможностью [компиляции](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80), [CLR](https://ru.wikipedia.org/wiki/IronPython), [LLVM](https://ru.wikipedia.org/wiki/LLVM), другие независимые реализации. Проект [PyPy](https://ru.wikipedia.org/wiki/PyPy" \o "PyPy) использует [JIT-компиляцию](https://ru.wikipedia.org/wiki/JIT-%D0%BA%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%86%D0%B8%D1%8F), которая значительно увеличивает скорость выполнения Python-программ.

**История:**

Разработка языка Python была начата в конце [1980-х годов](https://ru.wikipedia.org/wiki/1980-%D0%B5) сотрудником голландского института CWI [Гвидо ван Россумом](https://ru.wikipedia.org/wiki/%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). Для распределённой ОС [Amoeba](https://ru.wikipedia.org/wiki/Amoeba_(%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0)" \o "Amoeba (операционная система)) требовался расширяемый [скриптовый язык](https://ru.wikipedia.org/wiki/%D0%A1%D1%86%D0%B5%D0%BD%D0%B0%D1%80%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA), и Гвидо начал писать Python на досуге, позаимствовав некоторые наработки для языка [ABC](https://ru.wikipedia.org/wiki/ABC_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)) (Гвидо участвовал в разработке этого языка, ориентированного на обучение программированию). В феврале [1991 года](https://ru.wikipedia.org/wiki/1991_%D0%B3%D0%BE%D0%B4) Гвидо опубликовал исходный текст в [группе новостей](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D1%83%D0%BF%D0%BF%D0%B0_%D0%BD%D0%BE%D0%B2%D0%BE%D1%81%D1%82%D0%B5%D0%B9) alt.sources.  С самого начала Python проектировался как [объектно-ориентированный язык](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F).

Наличие дружелюбного, отзывчивого сообщества пользователей считается, наряду с дизайнерской интуицией Гвидо, одним из факторов успеха Python. Развитие языка происходит согласно чётко регламентированному процессу создания, обсуждения, отбора и реализации документов PEP ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Python Enhancement Proposal*) — предложений по развитию Python

[3 декабря](https://ru.wikipedia.org/wiki/3_%D0%B4%D0%B5%D0%BA%D0%B0%D0%B1%D1%80%D1%8F) [2008 года](https://ru.wikipedia.org/wiki/2008_%D0%B3%D0%BE%D0%B4), после длительного тестирования, вышла первая версия Python 3000 (или Python 3.0, также используется [сокращение](https://ru.wikipedia.org/wiki/%D0%90%D0%B1%D0%B1%D1%80%D0%B5%D0%B2%D0%B8%D0%B0%D1%82%D1%83%D1%80%D0%B0) Py3k). В Python 3000 устранены многие недостатки архитектуры с максимально возможным (но не полным) сохранением совместимости со старыми версиями Python. На сегодня поддерживаются обе ветви развития (Python 3.x и 2.x), но поддержка Python 2.7 заканчивается в 2020 году.

**Основная часть**

**Начало кода**

Узнав о базовых вещах в python я приступаю к написанию кода но быстро понял что дальше пресловутых выводов в командную строку (командой print) я не уйду, начал искать сторонние библиотеки о которых сейчас пойдет речь.

***Библиотеки языка***

***Python***

Существует несколько типов библиотек (или модулей расширяющих функционал языка) такие как:

***Стандартные библиотеки*** - Богатая стандартная [библиотека](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) является одной из привлекательных сторон Python. Здесь имеются средства для работы со многими [сетевыми протоколами](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB) и форматами [Интернета](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82), например, модули для написания [HTTP-серверов](https://ru.wikipedia.org/wiki/HTTP-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80) и клиентов, для разбора и создания почтовых сообщений, для работы с [XML](https://ru.wikipedia.org/wiki/XML) и т. п. Набор модулей для работы с [операционной системой](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) позволяет писать кросс-платформенные приложения. Существуют модули для работы с [регулярными выражениями](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D0%BE%D0%B5_%D0%B2%D1%8B%D1%80%D0%B0%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5), текстовыми [кодировками](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0), [мультимедийными](https://ru.wikipedia.org/wiki/%D0%9C%D1%83%D0%BB%D1%8C%D1%82%D0%B8%D0%BC%D0%B5%D0%B4%D0%B8%D0%B0) форматами, [криптографическими](https://ru.wikipedia.org/wiki/%D0%9A%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F) протоколами, архивами, [социализации](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B8%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) данных, поддержка [юнит-тестирования](https://ru.wikipedia.org/wiki/%D0%AE%D0%BD%D0%B8%D1%82-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) и др.

### ***Модули расширения и программные интерфейсы*** – Помимо стандартной библиотеки существует множество библиотек, предоставляющих интерфейс ко всем системным вызовам на разных платформах; в частности, на платформе Win32 поддерживаются все вызовы [Win32 API](https://ru.wikipedia.org/wiki/Win32_API), а также [COM](https://ru.wikipedia.org/wiki/Component_Object_Model) в объёме не меньшем, чем у [Visual Basic](https://ru.wikipedia.org/wiki/Visual_Basic" \o "Visual Basic) или [Delphi](https://ru.wikipedia.org/wiki/Delphi_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)" \o "Delphi (язык программирования)). Количество прикладных библиотек для Python в самых разных областях без преувеличения огромно ([веб](https://ru.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D0%BC%D0%B8%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D1%83%D1%82%D0%B8%D0%BD%D0%B0), [базы данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), обработка изображений, обработка текста, [численные методы](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D1%8B), приложения [операционной системы](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0) и т. д.).

Для Python принята спецификация программного интерфейса к базам данных [DB-API 2](https://ru.wikipedia.org/w/index.php?title=DB-API_2&action=edit&redlink=1) и разработаны соответствующие этой спецификации пакеты для доступа к различным [СУБД](https://ru.wikipedia.org/wiki/%D0%A1%D0%A3%D0%91%D0%94): [Oracle](https://ru.wikipedia.org/wiki/Oracle_Database), [MySQL](https://ru.wikipedia.org/wiki/MySQL), [PostgreSQL](https://ru.wikipedia.org/wiki/PostgreSQL), [Sybase](https://ru.wikipedia.org/wiki/Sybase), [Firebird](https://ru.wikipedia.org/wiki/Firebird) ([Interbase](https://ru.wikipedia.org/wiki/Interbase)), [Informix](https://ru.wikipedia.org/wiki/Informix), [Microsoft SQL Server](https://ru.wikipedia.org/wiki/Microsoft_SQL_Server) и [SQLite](https://ru.wikipedia.org/wiki/SQLite" \o "SQLite). На платформе [Windows](https://ru.wikipedia.org/wiki/Windows" \o "Windows) доступ к БД возможен через [ADO](https://ru.wikipedia.org/wiki/ADO) ([ADOdb](https://ru.wikipedia.org/wiki/ADOdb" \o "ADOdb)). Коммерческий пакет mxODBC для доступа к СУБД через [ODBC](https://ru.wikipedia.org/wiki/ODBC) для платформ Windows и [UNIX](https://ru.wikipedia.org/wiki/UNIX) разработан eGenix. Для Python написано много [ORM](https://ru.wikipedia.org/wiki/ORM) ([SQLObject](https://ru.wikipedia.org/w/index.php?title=SQLObject&action=edit&redlink=1" \o "SQLObject (страница отсутствует)), [SQLAlchemy](https://ru.wikipedia.org/wiki/SQLAlchemy" \o "SQLAlchemy), Dejavu, [Django](https://ru.wikipedia.org/wiki/Django" \o "Django)), выполнены программные каркасы для разработки веб-приложений ([Django](https://ru.wikipedia.org/wiki/Django" \o "Django), [Pylons](https://ru.wikipedia.org/wiki/Pylons" \o "Pylons), [Pyramid](https://ru.wikipedia.org/wiki/Pyramid_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D1%8B%D0%B9_%D0%BA%D0%B0%D1%80%D0%BA%D0%B0%D1%81)" \o "Pyramid (программный каркас))).

Библиотека [NumPy](https://ru.wikipedia.org/wiki/NumPy" \o "NumPy) для работы с многомерными [массивами](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%81%D1%81%D0%B8%D0%B2_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) позволяет достичь производительности научных расчётов, сравнимой со специализированными пакетами. [SciPy](https://ru.wikipedia.org/wiki/SciPy" \o "SciPy) использует NumPy и предоставляет доступ к обширному спектру математических алгоритмов (матричная алгебра — [BLAS](https://ru.wikipedia.org/wiki/BLAS) уровней 1—3, [LAPACK](https://ru.wikipedia.org/wiki/LAPACK), [БПФ](https://ru.wikipedia.org/wiki/%D0%91%D1%8B%D1%81%D1%82%D1%80%D0%BE%D0%B5_%D0%BF%D1%80%D0%B5%D0%BE%D0%B1%D1%80%D0%B0%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%A4%D1%83%D1%80%D1%8C%D0%B5)…). Numarray специально разработан для операций с большими объёмами научных данных.

[WSGI](https://ru.wikipedia.org/wiki/WSGI) — интерфейс шлюза с веб-сервером (Python Web Server Gateway Interface).

Python предоставляет простой и удобный программный интерфейс [C API](https://ru.wikipedia.org/w/index.php?title=C_API&action=edit&redlink=1) для написания собственных модулей на языках [Си](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)) и [C++](https://ru.wikipedia.org/wiki/C%2B%2B). Такой инструмент как [SWIG](https://ru.wikipedia.org/wiki/SWIG) позволяет почти автоматически получать привязки для использования C/C++ библиотек в коде на Python. Возможности этого и других инструментов варьируются от автоматической генерации (C/C++/Fortran)-Python интерфейсов по специальным файлам (SWIG, pyste, SIP, pyfort), до предоставления более удобных API (boost::python, CXX, Pyhrol и др.). Инструмент стандартной библиотеки [ctypes](https://ru.wikipedia.org/w/index.php?title=Ctypes&action=edit&redlink=1" \o "Ctypes (страница отсутствует)) позволяет программам Python напрямую обращаться к [динамическим библиотекам](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0)/[DLL](https://ru.wikipedia.org/wiki/DLL), написанным на Си. Существуют модули, позволяющие встраивать код на С/C++ прямо в исходные файлы Python, создавая расширения «на лету» (pyinline, weave).

Другой подход состоит во встраивании интерпретатора Python в приложения. Python легко встраивается в программы на Java, C/C++, [OCaml](https://ru.wikipedia.org/wiki/Ocaml" \o "Ocaml). Взаимодействие Python-приложений с другими системами возможно также с помощью [CORBA](https://ru.wikipedia.org/wiki/CORBA), [XML-RPC](https://ru.wikipedia.org/wiki/XML-RPC), [SOAP](https://ru.wikipedia.org/wiki/SOAP), COM.

С помощью проекта [Cython](https://ru.wikipedia.org/wiki/Cython" \o "Cython) возможна трансляция программ, написанных на языках Python и [Pyrex](https://ru.wikipedia.org/w/index.php?title=Pyrex&action=edit&redlink=1" \o "Pyrex (страница отсутствует)), в код на языке Си с последующей компиляцией в машинный код. Cython используется для упрощения написания Python-библиотек, при его использовании отмечается ускорение кода и уменьшение накладных расходов.

Экспериментальный проект [Shedskin](https://ru.wikipedia.org/wiki/Shedskin" \o "Shedskin) предполагает создание компилятора для трансформации неявно типизированных Python-программ в оптимизированный С++ код. Начиная с версии 0.22 Shedskin позволяет компилировать отдельные функции в модули расширений.

Python и подавляющее большинство библиотек к нему бесплатны и поставляются в исходных кодах. Более того, в отличие от многих открытых систем, лицензия никак не ограничивает использование Python в коммерческих разработках и не налагает никаких обязательств, кроме указания авторских прав.

Одним из каналов распространения и обновления пакетов для Python является [PyPI](https://ru.wikipedia.org/wiki/PyPI" \o "PyPI) ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Python Package Index*).

**Точно начало кода:**

Первым делом необходимо установить и импортировать все необходимые библиотеки при помощи модуля pip install …  
вот фрагмент кода:

C:\Users\imone\PycharmProjects\afina\_first\_project\_test\_1>pip install SpeechRecognition

И импрот в код напроимер тот же speechRecognition:

Import speech\_recognition as sr

Строкой as мы задаем переменную, которая несет в себе длинное слово speecRecognition так просто проще ориентироваться. Далее мы создаем функцию(коробку) в которую кладем библиотеку, которая умеет превращать текст в речь(pyttsx3). После этой манипуляции мы имеем проще говоря коробку, которая превращает текст в речь(вот код)

def speack(what):  
 print (what)  
 engine = pyttsx3.init()  
 engine.say(what)  
 engine.runAndWait()  
 engine.stop()

объясняю мы взяли коробку и назвали ее speack в которой лежит бумажка с кокой либо надписью, например: жюри

мы пишем данное слово в командной строке, затем мы создаем еще одну коробку внутри коробки speack которая ссылается на ранее импортированную библиотеку pyttsx3. Затем ссылаемся на нашу новую коробку engine и заставляем ее говорить тот текст который у нас в бумажки написан, если мы видим и слышим слово жюри значит функция написана верно.

Далее сложнее

def myCommand():  
 r = sr.Recognizer()  
 with sr.Microphone() as source:  
 print("слушаю...")  
 r.pause\_threshold = 1  
 audio = r.listen(source)  
 try:  
 querty = r.recognize\_google(audio, language='ru-RU')  
 print('вы: ' + querty + '\n')

мы должны преобразовать речь в текстиспользуем нашу переменную sr(speechRecognition) и вызываем подзадачу Recognizer(). Затем помещаем это все в коробку с названием r затем надо записать звук с микрофона на бумажку которая называется source. Печатаем в строке для удобства слушаю затем ждем 1 секунду и весь звук нужно отправить в гугл переводчик(другие я не знаю как вызывать) и он уже возвращает текст на бумажкуи и дальше командами if, elif, else(если, или если, иначе) мы проверяем что написано в бумажке например если в бумажке написано стоп

elif "стоп" in querty or 'без звука' in querty:  
 speack("удачи")  
 sys.exit()

компьютер это понимает и выключает данную систему

примечание: python требует чтобы в одной коробке после if стояла или else или elif как вот тут

if 'Открой YouTube' in querty:  
 speack('хорошо, сэр')  
 webbrowser.open('www.youtube.com')

elif "не спишь" in querty:  
 strr = ['всегда к вашим услугам, сэр', 'нет конечно', 'да, задремала']  
 speack(random.choice(strr))

**заключение**

данная работа мне очень понравилась и забрасывать я ее не буду, может быть когда-нибудь она станет конкурентом Jarvis (фильм Железный человек) который и подтолкнул меня и друга к созданию голосового помощника еще в середине лета и по сей день