Введение

1. Вступительные слова

В начале подготовки специализации перед командой разработчиков курса стоял выбор: либо рассказать необходимую математику по ходу изложения курса или же сделать отдельный подготовительный курс. Был принят второй вариант, а именно сперва рассказать основы линейной алгебры, математического анализа и программирования на языке Python. Причем математические концепции в данном курсе приводятся сразу на наглядных практических примерах.

2. История Python

История языка Python начинается в 1980-х годах, когда Гвидо ван Россум, сотрудник центра математики и информатики в Нидерландах, приступил к созданию его первой версии. И до сих пор «Гвидо ван Россум» остается «великодушным пожизненным диктатором»[1].

К данному моменту были выпущены уже три версии языка — Python 1, Python 2, Python 3, причем первые две из них обратно совместимы, а Python 3 и Python 2 — уже нет. Это связано с тем, что те улучшения, которые были сделаны в Python 3, невозможно внести без нарушения обратной совместимости. Официально поддерживаются обе версии: Python 3 — это будущее языка, более современная версия, а Python 2 поддерживается до сих пор из-за огромного количества наработок, которые еще не были перенесены на более свежую версию.

Следует особо отметить, что Python выпускается под свободной лицензией «Python Software Foundation License», разрешающей использовать исходный код проекта не только в открытом, но и в коммерческом программном обеспечении. Python работает почти на всех известных платформах. Существуют версии для MS Windows, Linux, Mac OS, FreeBSD и так далее.

Также привлекательной стороной Python является богатая стандартная библиотека. В нее входят, например, модуль для работы с текстом в различных кодировках и модуль для работы с регулярными выражениями. Кроме стандартной библиотеки, доступны: библиотека `matplotlib` для построения графиков функций, `pandas` — для работы с электронными таблицами и `scipy` — для научных и инженерных расчетов.

Язык Python отличается лаконичным синтаксисом, поэтому на нем и читать чужой код, и писать свой очень просто. Разработчики языка придерживаются философии, называемой «The Zen of Python», текст которой на английском языке выводится интерпретатором по команде `import this`. Ниже представлена часть текста, переведенная на русский язык:

- Красивое лучше, чем уродливое.
- Простое лучше, чем сложное.
- Читаемость имеет значение.
- Ошибки никогда не должны замалчиваться.
- Сейчас лучше, чем никогда.
- Хотя никогда зачастую лучше, чем прямо сейчас.
- Если реализацию сложно объяснить — идея плоха.
- Если реализацию легко объяснить — идея, возможно, хороша.

Python был выбран в качестве основного языка программирования для нашей специализации по многим причинам. Вот некоторые из них:

- Python — свободное ПО.
- Python доступен практически на всех платформах.
- Python прост в изучении.
- Python можно использовать в интерактивном режиме.
- Python имеет большое сообщество пользователей и разработчиков.
- Для Python доступно огромное число библиотек. При этом богатый набор функций имеется в стандартной библиотеке.

[1] Этот шуточный термин используется, чтобы обозначать главу или основателя проекта, который сохраняет за собой право окончательного решения в жизни проекта.

3. Установка Python

Установить Python не сложно, и существует огромное количество способов, как это сделать. Для целей демонстрации, установка Python в составе дистрибутива Anaconda будет произведена на операционной системе Linux.

Сначала с сайта `https://www.continuum.io/downloads` нужно скачать установочный файл для используемой Вами операционной системы и следовать указаниям. Согласно инструкции для Linux, необходимо открыть терминал и выполнить команду `bash Anaconda2-2.4.1-Linux-x86_64.sh` (точное название файла может отличаться). Вам будет предложено ознакомиться с лицензией и указать директорию для установки. Также в процессе установки будет предложено внести изменения в файл `bash.rc`, чтобы выбрать только что установленный Python в качестве используемого по умолчанию. Для того, чтобы эти изменения вступили в силу, необходимо закрыть текущую сессию (окно терминала) и запустить новую.

Запустить интерактивную оболочку IPython можно с помощью команды `ipython notebook`. В результате исполнения этой команды IPython должен открыться в новой вкладке браузера.

4. Знакомство с IPython

После запуска IPython в браузере откроется страница со списком файлов рабочего каталога. С помощью кнопки «new» можно создать новый файл типа «IPython notebook». При создании файл откроется в новой вкладке, где сразу же можно будет указать для него название, а в рабочем каталоге появляется файл с расширением `.ipynb`.

Главное меню программы интуитивно понятно. Особо стоит отметить, что существует возможность экспортировать текущую тетрадь в виде файла .py, результаты расчетов — в виде HTML или PDF. Соответствующие команды расположены в подменю «Download As» меню «File».

Интерфейс тетради IPython основан на работе с ячейками. Их можно создавать, перемещать, редактировать и исполнять находящийся в них код с помощью соответствующих команд на панели инструментов или в главном меню тетради.

Традиционно изучение языков программирования начинается с написания программы «Hello, world!». Функцию `print` можно использовать для вывода на экран переменных любого типа:

```
t = 'Hello, world!'
print t % OUT: Hello, world!
```

В IPython значение переменной или выражения также выводится на экран, даже если опустить `print`. Это, а также богатые математические возможности IPython, позволяет использовать IPython в качестве удобного калькулятора. При этом, если используется ядро Python 2, деление двух целых чисел является целочисленным делением, то есть результатом всегда является целое число. Если же или делимое, или делитель являются числом с плавающей точкой, результат также будет числом с плавающей точкой, а деление будет пониматься в обычном смысле. Например:

```
100 /12           % OUT:  8
100./12           % OUT:  8.333333333333334
round(100./12,3)  % OUT:  8.333
```

Здесь была использована функция `round`, которая используется для округления выражения в её первом аргументе с точностью до числа знаков после запятой, задаваемой её вторым аргументов.

Если требуемая математическая функция не доступна по умолчанию, существует два пути: либо реализовать нужную функцию самостоятельно, либо подключить библиотеку, в которой содержится необходимая функция, с помощью команды `from <library> import <function>`. Функцию `factorial` из библиотеки math можно добавить с помощью следующего кода:

```
from math import factorial
factorial(3) % OUT: 6
```

Кроме ячеек с кодом на Python, в тетрадь можно добавлять ячейки других типов. Для этого необходимо поменять тип ячейки. Это можно сделать в выпадающем списке в панели инструментов. Тип «Markdown» используется для набора текста на одноименном языке разметки:

- Создание заголовков производится путём помещения знака решетки перед текстом заголовка.

- Количество знаков «#» соответствует уровню заголовка.
- Внутри $$ ... $$ можно набирать математические формулы используя LaTeX.
- Более подробное описание доступно на странице википедии https://ru.wikipedia.org/wiki/Markdown.

Если отправить ячейку типа «Markdown» на исполнение, вместо кода разметки появится отформатированный текст.

На Unix-подобных операционных системах код ячейки можно выполнять в интерпретаторе командной строки `bash`. В случае однострочных команд достаточно набрать перед этой командой символ восклицательного знака, а имена переменных IPython начинать с символа доллара:

```
!echo $t
```

Многострочные инструкции командной строки набираются с помощью «bash magic»: на первой строке набирают `%%bash`, а на следующих — желаемый скрипт командной строки. На самом деле IPython предоставляет и другие «магические команды», полный список которых можно получить набрав в ячейке `%lsmagic`.

Среди команд, представленных в `%lsmagic`, есть команда, позволяющая рисовать графики прямо внутри тетради IPython:

```
%pylab inline
% OUT: Populationg the interactive namespace from numpy and matplotlib
```

Простейший пример построения графика:

```
y = range(11) % массив чисел от 0 до 11
y             % OUT: [0,1,2,3,4,5,6,7,8,9,10,11]
```

На этом урок заканчивается. Подробнее про IPython будет рассказано на следующих уроках.